Anonymity and Traceability in Cyberspace

Richard Clayton

University of Cambridge
Computer Laboratory
Darwin College

August 2005

This dissertation is submitted for
the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text.

This dissertation does not exceed the regulation length of 60 000 words, including tables and footnotes, but excluding the bibliography.

Acknowledgements

No programme of PhD work is ever done alone and so I must start by acknowledging the help and assistance of many people.

My supervisor Ross Anderson initially suggested that I might enjoy leaving industry behind and returning, after nearly thirty years, to Cambridge. His help and encouragement has been invaluable throughout. My previous employers at Demon Internet (Thus plc) have also been entirely supportive. Phil Male has kept me on as a consultant; this led directly to some of the work I describe. Andrew Bangs, Simon Edwards, Clive Feather, James Hoddinott, Alex Kiernan, Chris King and many other colleagues have ensured that I had the information I needed and that I remained grounded in addressing real problems.

Elsewhere in industry, I've been assisted by many others of whom I will pick out Andrew Hilborne and Keith Mitchell for their assistance in recalling the start of traceability as an accurate process, but there have been many others whose observations I report in this thesis. Other colleagues from law enforcement and government have assisted my understanding of the issues from their side of the fence, and they provide some of the anecdotes and examples that I present. I specially note Paul Bayer, Tony Hutchings and Simon Watkin, but there are many others as well.

David Greaves, Simon Moore, Sergei Skorobogatov and James Srinivasan have contributed hardware and advice to assist with my Ethernet collision design, and Ben Laurie helped with the DNS needed for the testing of email header reading skills.

My co-authors on academic papers, Ross Anderson, Mike Bond, George Danezis, Markus Kuhn, Ben Laurie, Andrei Serjantov and Ellis Weinberger have of course contributed greatly to my grasp of the subject of Computer Security, the topic into which Traceability seems to fit best. In addition, there have been countless valuable discussions with Jon Crowcroft, Stephen Lewis, Steven Murdoch and all the other members of the Computer Security group; not forgetting the sadly missed Roger Needham and David Wheeler.

I've also been much assisted by my two partners, one business, one life, Chris Hall and Chris Taylor. I would never have finished without their support.

Although I was initially "self-funded", since October 2003 I have been employed on the Cambridge Massachusetts Institute project "The Design and Implementation of Third Generation Peer-to-Peer Systems" and I am grateful for this assistance.

Finally, I owe a great debt to those who have prooofread this thesis and the papers on which some of it is based. These include Ross Anderson, George Danezis, Simson Garfinkel, Tony Hutchings, Stephen Lewis, Tyler Moore and Steven Murdoch. All of the remaining errors are of course my responsibility, but at least they have spotted numerous other howlers before anyone else read it.

Anonymity and Traceability in Cyberspace

Richard Clayton

Summary

Traceability is the ability to map events in cyberspace, particularly on the Internet, back to real-world instigators, often with a view to holding them accountable for their actions. Anonymity is present when traceability fails.

I examine how traceability on the Internet actually works, looking first at a classical approach from the late 1990s that emphasises the rôle of activity logging and reporting on the failures that are known to occur. Failures of traceability, with consequent unintentional anonymity, have continued as the technology has changed. I present an analysis that ascribes these failures to the mechanisms at the edge of the network being inherently inadequate for the burden that traceability places upon them. The underlying reason for this continuing failure is a lack of economic incentives for improvement. The lack of traceability at the edges is further illustrated by a new method of stealing another person's identity on an Ethernet Local Area Network that existing tools and procedures would entirely fail to detect.

Preserving activity logs is seen, especially by Governments, as essential for the traceability of illegal cyberspace activity. I present a new and efficient method of processing email server logs to detect machines sending bulk unsolicited email "spam" or email infected with "viruses". This creates a clear business purpose for creating logs, but the new detector is so effective that the logs can be discarded within days, which may hamper general traceability.

Preventing spam would be far better than tracing its origin or detecting its transmission. Many analyse spam in economic terms, and wish to levy a small charge for sending each email. I consider an oft-proposed approach using computational "proof-of-work" that is elegant and anonymity preserving. I show that, in a world of high profit margins and insecure end-user machines, it is impossible to find a payment level that stops the spam without affecting legitimate usage of email.

Finally, I consider a content-blocking system with a hybrid design that has been deployed by a UK Internet Service Provider to inhibit access to child pornography. I demonstrate that the two-level design can be circumvented at either level, that content providers can use the first level to attack the second, and that the selectivity of the first level can be used as an "oracle" to extract a list of the sites being blocked. Although many of these attacks can be countered, there is an underlying failure that cannot be fixed. The system's database holds details of the traceability of content, as viewed from a single location at a single time. However, a blocking system may be deployed at many sites and must track content as it moves in space and time; functions which traceability, as currently realized, cannot deliver.

Contents

1 In the Beginning **11**

1.1 Thesis outline 12

1.2 Published work 14

1.3 Work done in collaboration 16

2 Traditional Traceability **17**

2.1 Four steps to traceability 17

2.1.1 Step 1: establish an IP address 18

2.1.2 Step 2: establish the owner of the IP address 18

2.1.3 Step 3: establish the user account 19

2.1.4 Step 4: locate the user 22

2.2 Classical problems with traceability 26

2.2.1 Problems with step 1: establishing the IP address 26

2.2.2 Problems with step 2: establishing the address owner 28

2.2.3 Problems with step 3: establishing the user account 28

2.2.4 Problems with step 4: locating the user 30

2.3 Traceability in the wider world 30

2.4 Related work 33

2.4.1 Traceability manuals 33

2.4.2 Tracing DoS and DDoS attacks 35

2.4.3 Survivability 35

2.4.4 The Freiburg privacy diamond 36

2.5 Conclusions 37

3 Traceability Failures **39**

3.1 Dynamic connections . . . 39

3.1.1 Network Address Translation . . . 40

3.1.2 Dynamic Host Configuration Protocol . . . 41

3.1.3 Data retention and validity . . . 41

3.2 New connection technologies . . . 42

3.2.1 Traceability of 802.11 wireless . . . 42

3.2.2 Traceability of ADSL . . . 43

3.2.3 Traceability of cable modems . . . 44

3.3 Is CLI really trustworthy? . . . 46

3.3.1 Generic CLI . . . 46

3.3.2 Forged CLI . . . 47

3.4 Are logs all they're cracked up to be? . . . 49

3.4.1 Picking out the IP address . . . 49

3.4.2 Log folding and other artifices . . . 50

3.5 Confusing responses . . . 51

3.5.1 Hijacking IP address space . . . 51

3.5.2 Incorrect BGP announcements . . . 53

3.5.3 Misleading traceroute . . . 53

3.5.4 "Anonymous" Internet Relay Chat (IRC) . . . 55

3.5.5 Preloading traceability information . . . 56

3.5.6 Establishing the source of email . . . 57

3.5.7 Misleading reverse DNS for email . . . 58

3.6 Conclusions . . . 60

4 Hiding on an Ethernet **63**

4.1 Machine identification on an Ethernet . . . 63

4.1.1 Ethernet frames . . . 64

4.1.2 Address Resolution Protocol . . . 65

4.1.3 Collisions . . . 65

4.2 Previous work on identity spoofing . . . 66

4.3 How to hide on an Ethernet . . . 68

4.4 Experimental setup . . . 69
4.4.1 The Ethernet PHY . . . 69
4.4.2 Implementing a MAC in an FPGA . . . 70
4.4.3 Software . . . 72
4.5 Experimental results . . . 73
4.6 Time limitations . . . 78
4.7 How firewalls can make collisions unnecessary . . . 81
4.8 Conclusions . . . 84

5 Extrusion Detection 87
5.1 Traceability and spam . . . 87
5.2 The general approach . . . 89
5.3 Historical background . . . 90
5.4 Related work . . . 91
5.5 Processing email logs . . . 92
5.5.1 Email log contents . . . 93
5.5.2 Using delivery failures to detect bulk email . . . 93
5.6 Avoiding "false positives" . . . 94
5.6.1 Actual heuristics for detecting spam . . . 95
5.7 Heuristics that detect mass-mailing malware . . . 96
5.7.1 Spotting email loops . . . 97
5.7.2 Leveraging other people's detectors . . . 98
5.8 Results . . . 99
5.9 Conclusions . . . 100

6 Proof-of-Work Doesn't Work 103
6.1 Tackling spam through economics . . . 103
6.2 Proof-of-work systems . . . 104
6.2.1 Hashcash . . . 105
6.2.2 Avoiding a dependence upon processor speed . . . 105
6.3 Quantitative analysis . . . 106
6.3.1 Estimating email volumes . . . 106

6.3.2 The problem of mailing-lists 107

6.4 How much work must be proved? 108

6.4.1 Analysis by considering the economics of spamming 108

6.4.2 Analysis by considering access to insecure machines 111

6.5 Variability in sending rates 112

6.6 Conclusions 114

7 The BT 'CleanFeed' System and the Failings of Traceability 115

7.1 Introduction 115

7.2 Content-blocking systems 117

7.2.1 Packet dropping 117

7.2.2 DNS poisoning 118

7.2.3 Content filtering 119

7.2.4 Existing content-blocking schemes 120

7.3 Design of the CleanFeed system 120

7.4 Circumvention of the CleanFeed system 122

7.4.1 Identifying the IWF and CleanFeed 122

7.4.2 Evading CleanFeed 126

7.4.3 Circumvention by content providers 131

7.4.4 Related work on evasion 133

7.5 Attacking CleanFeed 135

7.5.1 Blocking legitimate content 137

7.6 Real-world experiments 138

7.6.1 Legal issues when experimenting upon CleanFeed 139

7.6.2 Using CleanFeed as an oracle to list illegal websites 140

7.6.3 Countering the oracle attack 143

7.7 Brightview's WebMinder system 144

7.8 The failings of traceability 145

7.9 Conclusions 146

8 Conclusions and Future Work 149

Annotated Bibliography 153

Glossary 183

Chapter 1

In the Beginning

Hey Jude, don't make it bad.
Take a sad song and make it better.
Remember to let her into your heart,
Then you can start to make it better.

— Lennon/McCartney, 1968

It is often the case on the Internet that one wishes to know "who did that" and, more particularly, to locate the person responsible with a view to persuading them not to do "that" again.

The process of mapping from an event occurring in the cyberspace world of the Internet to the everyday world we're familiar with is known as tracing and the property of interest, the ability to do that tracing, has become known as *traceability.*

In contrast, the ability to use the Internet without others being able to determine that you are responsible is now referred to as *anonymity.*

Academics have researched into anonymity for decades, building elaborate systems that intermingle communications in time or space so that it becomes impractical to untangle them. However, there has been little academic interest in traceability, which is often – incorrectly – seen as trivial or obvious.

Meanwhile in the Internet Service Provider (ISP) world, traceability has long been a key issue. In just a dozen years, the Internet has been transformed from an academic research network, with a few thousands of hosts, into an all-pervasive, mass-market commercial entity with hundreds of millions of hosts. Despite this transformation, the ISPs that operate the individual networks that make up the global Internet have remained entirely independent entities, yet to permit their users to inter-work they must co-operate with every other ISP. If users do things that are anti-social, then the ISP that hosts them must act to preserve the reputation of their business. Without traceability, the disruptive user will remain hidden and the ISP risks becoming a pariah that no other networks will communicate with, which is a sure recipe for a commercial disaster.

A small proportion of users are doing things on the Internet that are illegal and that law enforcement officials wish to investigate. These investigators have learnt that they can employ traceability to locate the miscreant although, as we shall see, their requirements are sufficiently different from those of ISPs as to cause substantial practical difficulties.

In these non-academic milieux, of ISPs preserving their reputations and police officers tracking down law breakers, anonymity results directly from any and all failures of traceability. There's no need for fancy academic anonymity systems if you can park in a suburban street, use an open wireless access point to commit an online crime, and drive away before anyone notices.

1.1 Thesis outline

After this introduction, Chapter 2 deals with a traditional view of traceability, reflecting the understanding of the issues that were being considered up to about 1999. It closely follows the classic document on the subject [32], created by the UK ISP industry in 1997–1999 under the auspices of the London Internet Exchange (LINX), although it goes well beyond that document in detailing exactly how the process hangs together.

Chapter 3 presents a far more contemporary examination of the failures of traceability (the presence of anonymity) which result from the processes described in Chapter 2 either being absent, or not working in the way that everyone always assumed. This is an important contribution to understanding how much traceability currently exists in cyberspace. Although many parts of this particular jigsaw have been lying around for some time, this is the first time that they have been collected together, and an examination made of the common themes they exhibit.

In Chapter 4, an entirely new method is described that can be used to achieve anonymity on an Ethernet by running a very precise denial-of-service attack against another node by deliberately colliding with its transmissions. I show that this means that a user can borrow the machine level identity of a co-worker in a hard-to-detect manner. I also identify a previously undescribed problem with "personal firewalls", whereby a system may entirely fail to object when this type of identity theft takes place. It may now be easier to become anonymous by sitting at your own desk than by travelling to the far side of the world.

Traceability, in the policy arena inhabited by the governments and regulators, has become synonymous with the making and retaining of logs of user activity. In the European Union, with its Data Protection regime, this has created a tension with the Data Protection Principles, which insist that since personal data is involved, logging can only be performed for a business purpose and that the logs must be destroyed or anonymised as soon as they are no longer needed. In Chapter 5, I describe a new way of processing email server logs to automatically detect the sending of “spam”. Apart from the significant advantages to ISPs in being able to detect this behaviour and deal with it promptly, this will be good news for the policy makers, because it has now provided a compelling business reason for creating logs of email activity; although they may be less happy to learn that the processing is so effective that there is little reason to retain the logs for more than a few days.

Staying with the theme of spam, and the difficulty of using traceability to locate the spammers, in Chapter 6 I present a detailed analysis of a well-known proposal for dealing with the email spam problem by economic means. It is often suggested that spam has become so prevalent because it is free to send, and the solution is for an artificial cost to be introduced, by requiring all email to carry a “proof-of-work” that a small computational puzzle has been solved. Only genuine senders, it is argued, would bother to solve the puzzles and hence spam would decrease. The scheme can be operated anonymously since the decision to accept an email depends on the presence of the puzzle solution and not upon who sent it. The payment is universal and self-evident and so there can be no defaulting and hence no need validate the sender within a complex identification infrastructure, or to trace the source of email.

In fact, the scheme is so appealing that everyone just assumed it would work, without ever calculating just how complex the puzzle should be. It turns out that making the puzzle complex enough to dissuade the spammers will also make it infeasible for a significant proportion of legitimate senders to maintain their current levels of email activity. Proof-of-work is therefore not an elegant fix for spam. It can at best only provide one facet of a technical fix – and the other facets will involve all manner of accountability and traceability – and so I fully expect such fixes to be too complex, too costly and too inconvenient to roll out in the near future, if ever.

Finally, in Chapter 7, I provide a detailed analysis of another seemingly elegant technical scheme. In this case it is the BT “CleanFeed” system, which aims to prevent access to indecent images of children which have been located by the Internet Watch Foundation (IWF) but that are hosted abroad where the local law enforcement authorities may not promptly remove such content. I give a detailed account of the many ways in which the system might be avoided by users and by content providers; and then outline possible countermeasures for BT and the IWF. I also show how users can exploit the system as an “oracle”, to create lists of blocked sites which they would not otherwise have known about. I view the underlying problem here as being the failure of traceability to deliver the results that would be required for blocking to be effective.

At the end of the thesis there is an annotated bibliography and, since the overwhelming majority of work in this field is available online, URLs are provided that link to the material that has been cited. I have also provided a glossary for those not familiar with the various acronyms and other obscure terms of art which are, necessarily, scattered throughout the text.

1.2 Published work

As part of this work, a number of papers were published (some in collaboration with other researchers) in peer-reviewed academic conferences and workshops:

- Richard Clayton. Stopping Outgoing Spam by Examining Incoming Server Logs. Second Conference on Email and Anti-Spam (CEAS 2005), Stanford CA, USA, July 21–22 2005.
- Andrei Serjantov and Richard Clayton. Modelling Incentives for Email Blocking Strategies. Fourth Annual Workshop on Economics and Information Security, WEIS05, Boston MA, USA, June 2–3 2005.
- Richard Clayton. Failures in a Hybrid Content Blocking System. Fifth Privacy Enhancing Technologies Workshop, PET 2005, Dubrovnik, Croatia, May 30–June 1 2005.
- Richard Clayton. Insecure Real-World Authentication Protocols (or Why Phishing is so Profitable). Thirteenth International Workshop on Security Protocols, Cambridge, UK, April 20–22 2005.
- Richard Clayton. Who'd phish from the summit of Kilimanjaro? Financial Cryptography and Data Security: 9th International Conference FC 2005, Roseau, The Commonwealth of Dominica, February 28–March 3 2005, volume 3570 of LNCS, pages 91–92, Springer Verlag.
- Richard Clayton. Stopping Spam by Extrusion Detection. First Conference on Email and Anti-Spam (CEAS 2004), Mountain View CA, USA, July 30–31 2004.
- Ben Laurie and Richard Clayton. Proof-of-Work Proves Not to Work. Third Annual Workshop on Economics and Information Security, WEIS04, Minneapolis MN, May 13–14 2004.
- Richard Clayton. Improving Onion Notation. In Roger Dingledine, editor, Privacy Enhancing Technologies, Third International Workshop, PET 2003, Dresden, Germany, March 26–28 2003, volume 2760 of LNCS, pages 81–87, Springer Verlag.
- Ellis Weinberger, Richard Clayton and Ross Anderson. A Security Policy for a Digital Repository. National Preservation Office Journal, volume 11, October 2002, pages 12–13.

- Richard Clayton and George Danezis. Chaffinch: Confidentiality in the Face of Legal Threats. In Fabien A. P. Petitcolas, editor, Information Hiding Workshop (IH 2002), Noordwijkerhout, The Netherlands, October 7–9 2002, volume 2578 of LNCS, pages 70–86, Springer Verlag.

- Richard Clayton and Mike Bond. Experience Using a Low-Cost FPGA Design to Crack DES Keys. In Burton S. Kaliski Jr., Çetin K. Koç and Christof Paar, editors, Cryptographic Hardware and Embedded Systems – CHES 2002, Redwood Shores CA, USA, August 13–15 2002, volume 2523 of LNCS, pages 579–592, Springer Verlag.

- Richard Clayton. Workshop Report for IPTPS'02: 1st International Workshop on Peer-to-Peer Systems. In Peter Druschel, Frans Kaashoek and Antony Rowstron, editors, Peer-to-Peer Systems, IPTPS 2002, Cambridge MA, USA, March 7–8 2002, volume 2429 of LNCS, pages 1–21, Springer Verlag.

- Richard Clayton, George Danezis and Markus G. Kuhn. Real World Patterns of Failure in Anonymity Systems. In Ira S. Moskowitz, editor, Information Hiding Workshop (IH 2001), Pittsburgh PA, USA, April 25–27 2001, volume 2137 of LNCS, pages 230–244, Springer Verlag.

The CEAS 2004 paper on Extrusion Detection forms the basis of Chapter 5 and similarly the WEIS04 paper on Proof-of-Work underlies Chapter 6, though in this latter case there has been substantial reworking to address an arithmetic flaw in the original paper, spotted by Ted Wobber of Microsoft. The work on DES cracking reported to CHES 2002 taught me a great deal about modern hardware design using FPGAs and this gave me the confidence to tackle the work described in Chapter 4. Finally, Chapter 7 was cut down considerably, removing most of the detailed explanation of mechanisms, to create the PET 2005 paper.

I have been on the Programme Committees for the PET Workshop in 2003, 2004 and 2005; the CEAS Conference in 2004 and 2005; and am on the PC for the upcoming IFIP SEC 2006. I have also spoken on panels at PET 2003 ("Peer-to-peer, anonymity, and plausible deniability designs"), Symposium on Economic Crime 2003 ("Techno-Risk"), CEAS 2004 ("Payment Systems for Email"), FC 2005 ("Phishing" and a second panel on "Security Economics") and PET 2005 ("Revocable Anonymity"). I have given talks on my work in symposium series within the Computer Laboratory and Economics Department at Cambridge, and also at University College, London; National University of Ireland, Maynooth; Massachusetts Institute of Technology; and the Berkman Center, Harvard Law School. I have also spoken at meetings organised by groups such as the Internet Service Providers Association (ISPA UK), the International Organization of Securities Commissions (IOSCO), the Metropolitan Police, the British Computer Society and at the BA Festival of Science 2002.

In addition to all this academic work, during this period I was the "specialist adviser" to the All Party Internet Group (APIG), an open group for members of the House of Commons and the House of Lords, who consider Internet issues as they affect society, thereby informing Parliamentary debate. Being their specialist adviser meant that I was the author of three reports describing their inquiries. Although the conclusions of these reports are entirely a matter for the Parliamentarians, the discussion and explanation were mine. My academic work helped the MPs and peers understand the issues and the inquiries in turn helped me to get to grips with the policy issues that are never far away when one is considering traceability and anonymity.

- APIG. Revision of the Computer Misuse Act: Report of an Inquiry by the All Party Internet Group, June 2004, 30 pages.
  `http://www.apig.org.uk/archive/activities-2004/computer-misuse-inquiry/CMAReportFinalVersion1.pdf`
- APIG. "Spam": Report of an Inquiry by the All Party Internet Group. October 2003, 36 pages.
  `http://www.apig.org.uk/archive/activities-2003/spam-public-enquiry/spam_report.pdf`
- APIG. Communications Data: Report of an Inquiry by the All Party Internet Group. January 2003, 42 pages.
  `http://www.apig.org.uk/archive/activities-2002/data-retention-inquiry/APIGreport.pdf`

1.3 Work done in collaboration

Chapter 2 is based on the original LINX approach to traceability, for which I edited the Best Practice document, but which was, by its very nature, a collaborative approach. Chapter 3 reports upon a number of instances where traceability has failed. Many of these examples have been garnered from conversations with a wide range of people within the ISP industry. However, their presentation, and the generic lessons I draw are entirely my own responsibility.

Chapter 6 results from a collaboration with Ben Laurie. It was his original idea to do some calculations to specify a simple-minded proof-of-work scheme, and he assisted with the initial outline of the analysis, but apart from that, the resultant chapter here (and indeed the version presented at WEIS04, complete with arithmetic flaw) are entirely my own work.

Chapter 2

Traditional Traceability

Can it be that it was all so simple then?
Or has time rewritten every line?
— Marvin Hamlisch, Alan Bergman & Marilyn Bergman, 1973

This chapter discusses a “traditional” view of traceability, the problem of determining “who did that”. It is based on the situation in the late 1990s, by which time the structure of the modern Internet was in place; with widespread access by consumers and businesses, and a large number of ISPs of various sizes supplying them with connectivity, email and web services.

This type of traceability continues to work very well in many situations. However, from a contemporary standpoint, one might view this chapter as “traceability when everything goes right”, when there are no problems in locating and understanding the necessary information to track down the source of an action. The next chapter will consider a number of circumstances in which, in practice, things can fail to “go right”, addressing issues that have arisen relatively recently, as well as looking more deeply into the accuracy and provenance of some of the data that was traditionally just accepted at face value.

Because this chapter is about the traditional approach, it concentrates on traditional technology such as dial-up access and leased lines. To a large extent, more modern Internet access technologies such as GPRS and GSM mobile telephony, xDSL and cable modem “broadband”, and 802.11 wireless can be traced in analogous ways. However, in practice, some specific issues have arisen with these newer forms of access, discussion of which is deferred to the next chapter.

2.1 Four steps to traceability

Traceability can be expressed in four independent steps. First, one determines the IP address to be traced. Second, one establishes which ISP (or perhaps a university) has been allocated the IP address. Third, the ISP's technical records will indicate which user account was using the IP address at the relevant time. Fourth and finally, the ISP's administrative records will establish the "real-world" identity of the individual who was permitted to operate the account.

2.1.1 Step 1: establish an IP address

The first step is to obtain the IP address of the remote party whose identity you wish to establish. This discussion will focus on IPv4 addresses, but entirely parallel systems exist for IPv6 addresses.

The IP address will be found in log files, in the `Received` header fields of an email or indeed in `tcpdump` traces. For example, in this `ssh` daemon log file extract that documents a password guessing attack on my machine, the IP address to be traced is `200.138.122.4` (an ADSL connected machine, somewhere in Brazil).

```
Mar  7 01:17:06 sshd: Failed password for root from 200.138.122.4 ssh2
Mar  7 01:17:11 sshd: Failed password for root from 200.138.122.4 ssh2
Mar  7 01:17:18 sshd: Failed password for root from 200.138.122.4 ssh2
Mar  7 01:17:38 sshd: Failed password for root from 200.138.122.4 ssh2
Mar  7 01:17:45 sshd: Failed password for root from 200.138.122.4 ssh2
```

In some circumstances only a host name will have been recorded, but this can be translated into an IP address by the simple expedient of consulting the DNS; although, as I shall explain later, it is not always quite as simple as that.

2.1.2 Step 2: establish the owner of the IP address

IP addresses are allocated in a hierarchical manner with blocks of address space being delegated by higher level authorities down to lower level entities for distribution. Consulting the top level master list currently held by IANA [77] will lead to one of five Regional Internet Registries (RIR):

ARIN	American Registry for Internet Numbers
RIPE	Réseaux IP Européens
APNIC	Asia Pacific Network Information Centre
LACNIC	Latin American and Caribbean Internet Addresses Registry recognised by ICANN, October 2002
AfriNIC	The Internet Numbers Registry for Africa operational, April 2005

In turn, their records will document delegation to a National Internet Registry (NIR) or perhaps directly to a Local Internet Registry (LIR). In some cases a registry will document allocations to individual end-users (typically when they use a /24[1] or greater), but allocation at this level of granularity is often performed internally by ISPs and will not be available for public inspection.

Traditionally, access to registry databases was performed with the `whois` protocol, see Figure 2.1, though these days all the RIRs and most NIRs provide the same information via a web interface. A useful practical tip is that when you ask about an IP address that is *not* delegated to them, then they all refer you to the IANA list, with the exception of `http://www.arin.net/` which will refer you directly to a different RIR when that is appropriate.

2.1.3 Step 3: establish the user account

The information held by the RIR, or (for APNIC and LACNIC especially) by the NIR, will give the contact details of an ISP or equivalent entity that is directly responsible for the particular IP address that is of interest. There are then two rather different ways by which the ISP will establish the user details, depending upon whether the IP address usage is “static” or “dynamic”.

Static IP addresses

If the IP address is statically allocated (usually to a permanently connected machine) then the ISP’s routers must contain sufficient information to route packets to the relevant subnet, usually down a particular leased line or xDSL connection. This routing information will have been created from internal databases which can be interrogated (in reverse) to determine the mapping from IP address to machine, and hence to a customer account; the person or organization who purchased the connection from the ISP. They should then know where to locate the machine.

To give an example from the academic world, `128.232.15.208` is the IP address allocated to my laptop when it is connected to the Ethernet in room GE21 of the Computer Laboratory. Packets for this machine will be handed to the JANET network (AS786) which will route them to their “EastNet” which will route them to Cambridge and hence to the Computer Laboratory.

Within the building, the local set-up means that IP addresses are tied to particular MAC addresses (my laptop is `00-B0-D0-BF-8A-1A`) and may be further restricted to just one end-user connection, in this case the structured wiring cable outlet in GE21 labelled `WC0E-45-3`.

[1]A /24 is a network whose IP addresses have a particular 24-bit prefix and any value in the last 8 bits. Hence it contains 256 distinct IP addresses. Similarly, a /20 has a constant 20-bit prefix and 2^{12} (i.e. 4 096) addresses and a /8 has an 8-bit prefix and 2^{24} (i.e. 16 777 216) IP addresses.

```
$whois -host=whois.ripe.net 128.232.15.208

[Querying whois.ripe.net]
[whois.ripe.net]
% This is the RIPE Whois query server #2.
% The objects are in RPSL format.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html

% Information related to '128.232.0.0 - 128.232.255.255'

inetnum:      128.232.0.0 - 128.232.255.255
netname:      CL-CAM-AC-UK
descr:        University of Cambridge Computer Laboratory
descr:        15 J J Thomson Avenue
descr:        Cambridge CB3 0FD
descr:        UNITED KINGDOM
country:      GB
admin-c:      PB219
tech-c:       PB219
status:       ASSIGNED PI
mnt-by:       RIPE-NCC-HM-PI-MNT
mnt-lower:    RIPE-NCC-HM-PI-MNT
mnt-by:       JANET-HOSTMASTER
mnt-by:       CL-CAM-AC-UK-MNT
mnt-routes:   JANET-HOSTMASTER
mnt-routes:   CL-CAM-AC-UK-MNT
mnt-domains:  CL-CAM-AC-UK-MNT
source:       RIPE # Filtered

person:       Piete Brooks
address:      University of Cambridge Computer Laboratory
address:      15 J J Thomson Avenue
address:      Cambridge CB3 0FD
address:      UNITED KINGDOM
address:      GB
phone:        +44 1223 334659
e-mail:       pb@CL.CAM.AC.UK
nic-hdl:      PB219
source:       RIPE # Filtered

% Information related to 'PB219'

route:        128.232.0.0/16
descr:        University of Cambridge Computer Laboratory
origin:       AS786
mnt-by:       JIPS-NOSC
source:       RIPE # Filtered
```

Figure 2.1: Use of the `whois` command

Hence an enquiry into the usage of `128.232.15.208` would eventually end up asking questions of Computer Laboratory sysadmins who would be able to immediately report that it was allocated for use by my laptop in one specific location.

It may be possible to tackle the traceability a different way by consulting the reverse DNS entry for an IP address. This will yield the name of the domain for the machine that is using the IP address, which can also be looked up by `whois`, this time consulting databases maintained by the domain name registrars.

RFC1035 [102] documents the use of the `in-addr.arpa` domain to provide a hierarchical system for mapping from an IP address to an interface name (the usual use for DNS being to map from interface name to IP address). For example, if one looks up `208.15.232.128.in-addr.arpa` (the reversal of the octets permits delegation of the creation of DNS entries) then one can determine the hostname for my laptop, which is `rnc1.al.cl.cam.ac.uk`, as shown in Figure 2.2.

```
$ dig ptr 208.15.232.128-in-addr.arpa

;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5741
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 6, ADDITIONAL: 5

;; QUESTION SECTION:
;208.15.232.128.in-addr.arpa.   IN      PTR

;; ANSWER SECTION:
208.15.232.128.in-addr.arpa. 21600 IN   PTR     rnc1.al.cl.cam.ac.uk.

;; AUTHORITY SECTION:
232.128.in-addr.arpa.   21600   IN      NS      ns.ripe.net.
232.128.in-addr.arpa.   21600   IN      NS      ns2.ic.ac.uk.
232.128.in-addr.arpa.   21600   IN      NS      dns0.cl.cam.ac.uk.
232.128.in-addr.arpa.   21600   IN      NS      dns0.eng.cam.ac.uk.
232.128.in-addr.arpa.   21600   IN      NS      dns1.cl.cam.ac.uk.
232.128.in-addr.arpa.   21600   IN      NS      chimaera.csx.cam.ac.uk.

;; ADDITIONAL SECTION:
ns2.ic.ac.uk.           44207   IN      A       155.198.5.3
dns0.cl.cam.ac.uk.      21600   IN      A       128.232.0.19
dns0.eng.cam.ac.uk.     86400   IN      A       129.169.8.8
dns1.cl.cam.ac.uk.      21600   IN      A       128.232.0.18
chimaera.csx.cam.ac.uk. 86400   IN      A       131.111.8.42

;; Query time: 2 msec
;; SERVER: 128.232.1.3#53(128.232.1.3)
;; WHEN: Tue Aug  9 15:32:31 2005
;; MSG SIZE  rcvd: 293
```

Figure 2.2: Using reverse DNS

RFC2317 [54] extends this scheme for "classless" networks with allocations smaller than a /24. I shall have rather more to say about reverse DNS in Section 3.5.4, since in practice it does not necessarily work as well as I have just described.

Dynamic IP addresses

If the IP address is dynamically allocated (the usual arrangement for dial-up access to the Internet) then the situation is more complex. The ISP will allocate a pool of IP addresses to their Network Access System (NAS) equipment and those addresses will be re-used as phone calls end and new users dial in. Therefore, to determine the user account one must know not only the IP address, but also the time of connection.

Of course static allocations can also change (the details in the previous section for my laptop were accurate from May 2001 until the 18th November 2004 when a new address and hostname were allocated), but with dynamic IP addresses there is an expectation that they will be re-used from minute to minute.

When customers dial into an ISP, a username/password pair is almost invariably used to authenticate their access. This authentication was originally to ensure that the correct customer was charged for access. However, many ISPs now offer "free" services which are financed by the telephone company passing on a proportion of the price charged for the phone call. Nevertheless, even with changing business models, authentication schemes remain, so as to ensure that customers can be held accountable for their actions.

NAS equipment typically uses the RADIUS protocol [117] to determine the validity of authentication credentials. RADIUS servers generate login and logout records that give details of the time of each connection or disconnection, the user account used, the IP address allocated, and perhaps some further usage details. Post-processing of these records will permit the ISP to answer the question, "which user account was assigned this IP address at that time?" It is worth noting that the question can be hard to answer quickly because the details of the logging tend to be messy. For example, it is usual to have to combine flat files from multiple servers before being sure of the account identity.

2.1.4 Step 4: locate the user

Having established which user account has been using a particular IP address, the final stage is to determine who is responsible for that account and contact them in the "real world".

ISPs generally require customers to identify themselves when they sign up for service. However, although consistency checks can be applied to this information ("does the street address match the postcode?") it is usual to accept it at face value and not check it out by, for example, requiring an answer to a letter posted to that address.

Where ISP business models require them to be directly paid for access, they will receive money on a regular basis. The banking and credit card systems provide some traceability, upon receipt of the appropriate legal paperwork to compel disclosure. Banks are also generally prepared to relay important messages to their customers.

Caller Line Identity

The principle linkage back to a dial-up customer will be the phone number of the line used for the call. This number is made available to the terminating NAS equipment by means of "Caller Line Identity" (CLI). ISPs providing "free" access, who have no direct financial linkage to identify their customers, are likely to insist that the CLI is not withheld.

Every UK phone line[2] has an "A" Number that is the "real" number of the line (that appears on the phone bill). A line can have several "presentation" numbers as well, which may be selectable by the user when placing calls. A line may even be flagged to allow callers to present any number they wish; for example in cases where call centres are generating calls on behalf of several clients with different phone numbers. Where the telephone network providers do not generate the number themselves, Ofcom (the UK industry regulator), insists that a contract is made with the subscriber requiring that they only present numbers to which they have a valid entitlement.

The CLI will usually be provided to the phone being called so that the receiver can determine whether or not to answer – or perhaps to allow the called system to present an operator with the details of the caller from a customer support database. However, the CLI can also be withheld by the caller, and every line has a default CLI Presentation Restriction (CLIR) state for an outgoing call which specifies whether the CLI is to be provided or withheld. The CLIR state may be overridden on a per call basis (in the UK, on a BT line, by dialling 141 in front of the called number for a temporary suppression; or 1470 to allow a temporary reveal).

At a protocol level the system is complex.

The two main protocols involved are DSS1 for ISDN (Integrated Services Digital Network) communications between customer equipment and the telco switch and SS7 (Signalling System 7) for communication between the telco switches. These protocols were developed by the Telecommunication Standardization Section of the International Telecommunication Union. DSS1 is described in ITU Recommendation Q.931 [137] and SS7 is described in ITU Recommendation Q.763 [136].

The calling party number and associated state are passed in DSS1 SETUP messages and in SS7 IAM (call setup) packets and the same byte level format is used in each.

[2]The operation of CLI (and an associated American system called ANI, Automatic Number Identification) differs in detail in the USA (and elsewhere) from the UK system described in this chapter. Non-UK readers will need to locate detailed technical descriptions of their own country's mechanisms for themselves.

The detailed rules for handling the various states and user preferences are set out in clauses 3 and 4 of ITU Recommendation Q.951 [138].

If the calling user does not provide a number, as would be the case for non-ISDN "analogue" calls, then the switch will generate a "network provided" number (usually the A number). If the calling user does provide a number then it will be validated and if acceptable it will be passed on. If it fails validation then Q.951 requires the network provided number to be sent (although there is also provision in the packet formats for sending a failed indication). Where there is a "special arrangement" the switch will not validate the number and will set an appropriate state.

The bit settings are recorded in two bits in the calling party number field (ITU Q.763 3.10, ITU Q.931 4.5.10) as described in Table 2.1.

Screening indicator	
0 0	user provided, not verified (National Use Only)
0 1	user provided, verified and passed
1 0	user provided, verified and failed (National Use Only)
1 1	network provided

Table 2.1: The validity of the CLI is mapped to two bits

The calling user's preferences for CLI Presentation Restriction, either a default setting or a per-call override, is recorded into another two bit field in the same byte as the screening indicator, as shown in Table 2.2. The user preference may be discarded and replaced by an indicator that the CLI value is unavailable. This can occur when a telco fails to pass the CLI to systems that they do not trust to operate the same data protection procedures as they do.

Address presentation restricted indicator	
0 0	presentation allowed
0 1	presentation restricted
1 0	address not available (National Use Only)
1 1	reserved for restriction by the network

Table 2.2: The settings for disclosing the CLI are mapped to two bits

The terminating telco, or a subscriber with the "presentation override facility", such as a 999 operator,[3] will always see the CLI value. If this value is marked as "presentation allowed" then a normal subscriber will be able to see the number; if

[3]An operator may also have SS7 level access, which would permit access to the calling party number (the A Number). Investigators, perhaps at an ISP that is also a telco, who have access to SS7 data would also be able to learn this value, in addition to the CLI information.

the marker is “presentation restricted” they will see “withheld”; otherwise they will see “unavailable”.

Note that the number (and its associated flags) can be passed across the Q.931 interface to an ISP’s NAS equipment. However, the telco may well treat an ISP like a normal subscriber and hence they will not see the CLI if the user withholds it, though clearly if the ISP is also a telco then they may be able to access SS7 information as well as the data that was made available to the NAS.

Legal restrictions on access to CLI data

At present, in the UK, the regulations governing the use of CLI [106] permit “Electronic Communications Networks” to use received CLI data for “network/account management purposes” and “in co-operation with the relevant authorities, for emergency calls and the tracing of malicious calls and similar activities”. However, an ISP would also have to meet the test that access to CLI data was “essential to the provision of an Electronic Communications Service”, which would mean arguing that dial-up Internet access could not be offered unless the source of all calls could be traced. Although this might seem hard to argue in the general case, it is more plausible for “free” services where nothing is known about the user and for some time, the regulator has encouraged subscriptionless ISPs to take this approach [146].

Outside the UK the situation is mixed.

In Australia, which has a similar legislative framework in its Privacy Act 1988 [40] and Calling Number Display Code Industry Code [7], a number of telephone call carriers (including Telstra, Optus and Comindico) have been providing CLI to some ISPs since 2002, as requested by the Internet Industry Association of Australia (IIA). Electronic Frontiers Australia argues that this is “overkill” and unlawful [55].

Meanwhile, in New Zealand, an internal memo from Telecom New Zealand [135] states that ISPs will not be given withheld CLI, but that the telco records will have this information if the police require it.

A 2000 working group on Computer Related Crime in Hong Kong concluded [76] that forcing ISPs to record CLI for all calls should be put on hold. They were concerned about cost, likely effectiveness and the inability to deal with calls from abroad.

In India the authorities have taken the view that CLI is a necessity and should not be suppressed by individual users. In their recently finalised (11 May 2004) Licence Agreement for telcos [64] it says at s41.19(iv):

> *“Calling Line Identification (CLI) shall never be tampered as the same is also required for security purposes and any violation of this amounts to breach of security. CLI Restriction should not be normally provided to the customers. Due verification for the reason of demanding the CLIR must be done before*

provision of the facility. It shall be the responsibility of the service provider to work out appropriate guidelines to be followed by their staff members to prevent misuse of this facility. The subscribers having CLIR should be listed in a password protected website with their complete address and details so that authorized Government agencies can view or download for detection and investigation of misuse. However, CLIR must not be provided in case of bulk connections, call centres, telemarketing services."

2.2 Classical problems with traceability

It has always been understood that traceability is fragile and it will often prove impossible to determine who is responsible for particular Internet traffic. In this section I analyse the sort of failures that were "generally known" at the time that traceability was initially being described, explaining them as being failures to complete one of the four steps.

2.2.1 Problems with step 1: establishing the IP address

Spoofing bi-directional TCP traffic

IP packets contain two address fields. The destination address is where the packet is to be delivered and it is – by definition – always valid (although that destination may immediately create a new packet and forward it on elsewhere). Far less validity can be ascribed to the source address in that it is where the packet is said to be from; although honest senders will set it correctly, it can easily be forged or "spoofed".

When TCP is in use communication is bi-directional. Packets to be sent back to the sender are constructed by swapping over the source and destination addresses. Hence, if the source address is invalid, the sender will not receive any of the responses to its transmissions – which may prevent it from conversing in a coherent manner.

A particular issue in TCP is that it is essential to provide valid acknowledgements for the sequence numbers of the incoming data. The starting point for the sequence numbers is uniquely chosen from a 2^{32} space during connection establishment,[4] and hence the valid value will not be immediately apparent. Of course if the real sender can control a machine that is near to the spoofed sender, or that is on the path to the real sender, then it may be possible to intercept or "sniff" the traffic, and so the sequence numbers can be obtained – but it is certainly not usually practical to masquerade as a machine on the other side of the Internet.

However, in 1985 Morris pointed out that it was possible to spoof two way communication with existing systems if their selection of initial sequence number was capable

[4]RFC793 [111], which initially described TCP, requires the initial sequence number to be bound to a, possibly fictitious, clock that increments every 4 milliseconds (cycling round every 4.55 hours).

of being predicted [104]. His concern was not with traceability but with security, since in that era many machines trusted each other on the basis of their identity as determined by their IP address, and so spoofing would permit unauthorised access. He suggested that initial sequence numbers should be randomised, though he was pessimistic as whether values would be sufficiently random to stop the forgers.

The ability to spoof TCP connections has continued to be a weakness in TCP implementations and ten years later in 1995 CERT/CC issued an Advisory on the topic [28]. Problems have continued since then, with reports of the same underlying issue recurring every few months. A recent example from August 2004 is the Thomson (Alcatel) SpeedTouch Home ADSL Modem which eschews a random value in favour of a value that increments by approximately 64 000 per millisecond [120].

Even where random values are used, they have often not, as Morris foresaw, been "random enough". In 1996 Bellovin [12] explained why random values could cause problems for reincarnated connections and proposed a scheme to avoid this problem. Nevertheless, in 2001 a further CERT Advisory [29] was needed. It surveys the history of this security issue, and describes how adding random increments means that, by the central limit theorem, after a while it is possible to guess the cumulative value. The advisory also stresses the importance of "Best Practice", and lists the considerable number of vendors who were still in the process of addressing problems.

Spoofing uni-directional traffic

Where communication is uni-directional the source address cannot be assumed to be valid at all. This was exploited in "Triangle Boy" [119], a proposal for evading a firewall by arranging for incoming traffic to appear to come from an innocuous source rather than from a remote site that the firewall would have blocked. The other two sides of the triangle were used for sending control/acknowledgement traffic.

Because normal traceability cannot be relied upon for uni-directional traffic, this is a common method of mounting a "denial-of-service" (DoS) attack. The attacker sends a large amount of traffic to the destination, but puts invalid information into the source address, so as not to be identified. The usual method of tracing these sort of attacks is to determine where the traffic enters the local network, either by checking for unusually large traffic flows, or by sniffing the traffic on possible links. Having determined the source, the flow can be discarded there, or a network peer contacted to ask them to trace the flow back towards the source.

In a "distributed" denial-of-service attack (DDoS) multiple sources are made to send DoS traffic at the same time. This makes it harder to detect the flow from volume measurements at individual ingress points, harder to implement any blocking and, crucially, will mean that there are far too many individual peers to contact to ask them to co-operate in locating the source of the traffic. Currently, the most effective countermeasure is to arrange for traffic matching specific attack profiles to be filtered out by custom equipment specifically designed for this purpose.

Accuracy of log entries

In practice, one does not establish the IP address of a connection by examining the contents of packets, but by consulting the logs created by a particular application such as an email, web or IRC server. Sometimes these logs do not record the IP address directly but show hostnames, so as to be "human-friendly". The accuracy of these hostnames will vary.

If the application received the hostname first then it is possible to recreate the associated IP address by repeating the DNS lookup that the application will have performed. There is of course no guarantee that a later request will receive the same answer as the application did, especially for one of the "dynamic DNS" services, but in many cases the mapping will be constant for some time. However, even though the genuine mapping remains constant, there will always be doubt as to whether the DNS server had been "spoofed" into using the wrong translation, and hence it is Best Practice to always record the IP address that was actually used.

Alternatively, the application may have started with the IP address and created the hostname to improve the usability of its logs by performing a reverse lookup as described above. Keeping just the hostname and not recording the IP address at all is even more problematic because not only is there doubt as to whether a future DNS access will receive the same answer, but because it is quite common for a forward lookup of the hostname resulting from a reverse lookup to give a completely different IP address than the one that was started with.

There is a great deal more about these issues in the next chapter.

2.2.2 Problems with step 2: establishing the address owner

Classically, the mapping from IP address to owner was not perceived to pose any special difficulties. It was known that there were some errors and omissions in registry entries, so the usual advice for reassuring oneself was to do a `traceroute` to the destination so as to check for consistency with the IP addresses (and router naming) for the last few hops of the trace output.

In practice, a number of problems have arisen with registry entries and with the operation of `traceroute`. However, since these problems are all relatively recent, discussion of them is postponed until the next chapter.

2.2.3 Problems with step 3: establishing the user account

The main practical difficulty with establishing the user account is the practice of re-using IP addresses when Internet access is being provided to dial-up users and IP addresses are allocated dynamically. Although determining which user account is involved should be simple, there are two particular issues that occur again and again.

The first is that the logs that document IP address allocations may be incomplete, and the second is that it is crucial to examine the records for the correct time.

Incomplete logging information

It is typical for RADIUS logging information to be transferred from the NAS equipment to where it is to be recorded by using the `syslog` protocol. This operates over UDP and is therefore "unreliable". Hence, when there is network congestion, some parts of the log may be missing.

It is usual to generate one RADIUS record when a call starts and another when the call ends. When an inquiry arrives at an ISP they will consult the logging to determine which of their users is being traced. If the ISP examines only the start records, then a missing record will result in them falsely identifying an earlier user of an IP address as being responsible for a later user's actions. If they examine only the end records then a missing record will lead them to a later user of the IP address than the one required. If they consult all records then the mismatch of start and end records should be obvious, and if just one record is missing the correct user can still be identified. Nevertheless, in a case when both an end record and subsequent start record are missing, perhaps because there is an outage when no records are recorded at all, then it will become impossible for even a diligent and capable investigator to determine from these logs alone which of two users was responsible for a particular event, if indeed either were.

Incorrect timing data

The mapping from IP address to user account requires the time of usage to be known to a reasonable degree of accuracy. In fact the time must be accurate not only at the remote machine where the IP address is logged, but also at the ISP. Best Practice advice has always been to use Network Time Protocol (NTP) to synchronise local clocks with global standards, but in actual practice many machines have inaccurate settings. It is of course possible to determine the clock's current skew from real time and apply a correction to the historical data. However, this makes assumptions about the linearity of clock drift that may be unreasonable if, for example, the clock is not being updated during some disk or network operations.

A major practical issue with recorded time values is that of timezones. Times to the West of Greenwich are behind (so that 7am in New York is noon in London), which is expressed as `-0500` in email `Received` header fields, but a careless investigator may add five hours instead and determine who used the IP address ten hours away from the time of interest. Further difficulties arise with "daylight-saving" adjustments or "summer time" because times may – or may not – be recorded in "wall clock time" for part of the year, with different systems at the same ISP being configured differently. Time zone abbreviations can also be ambiguous in that `EST` may be

Eastern Standard Time in either the USA (`-0500`) or Australia (`+1000`). Finally, when dealing with ISPs in the Antipodes there is of course a risk of messing up the mental arithmetic completely and getting the wrong day!

Many other problems have arisen with time values, with human error being a significant factor. In one infamous case in 2002, involving a large UK ISP in Leeds, the police were looking for a paedophile. They asked who had been using a dynamic IP address at a particular time, went to the address supplied by the ISP and arrested a man in front of his family at the breakfast table. The subsequent interview at the police station persuaded the officers that some sort of mistake had been made so they reviewed the evidence they had used – which was a printout of an email made by Internet Explorer. They found that instead of using the date and time the email was sent, they had provided the ISP with the date and time of the printout, recorded in the footer text that had been added at the bottom of the A4 sheet of paper. The correct timestamp did indeed lead to the person the police were seeking, but this was scant comfort to the innocent man who had been falsely accused.

2.2.4 Problems with step 4: locating the user

As was indicated above, ISPs may not be in a position to verify information that is provided to them about the owner of an account. I explained the rôle played by the banking system and the phone company's CLI in establishing their identity.

However, it may not be the user who is operating the account. ISPs invariably permit accounts to be used by anyone who is in possession of the password for the account. This password may become known to others for a wide range of different reasons. It is remarkably common for users with problems to post their passwords to Usenet as part of the debugging info they are providing to others, in plaintext [149] or in a weakly encrypted form [83]. Alternatively, passwords become known to others in the same workplace, school, or home – because the use of "yellow sticky" Post-it Notes to write them down is so widespread.

2.3 Traceability in the wider world

In the previous section we encountered a very serious problem with classical traceability: it does not actually provide a method of tracing users, but only determines which particular user account was involved. The stolen-password discussion made this clear, although the same underlying issue occurs repeatedly when one examines the binding, such as it is, between an account and who might be using it.

This failure to proceed beyond identifying the responsible account is not accidental, but a direct result of the structure of the Internet business.

ISPs are commercial entities who wish to remain in business. As such, they are interested in having as many profitable customers as possible and in avoiding financial loss. An ISP's most fundamental product offering is "connectivity" – access to the rest of the Internet. That connectivity depends upon other parts of the Internet believing that it is better to swap packets with the ISP than not to do so.

Traceability is therefore of interest to the "rest of the Internet" in determining the source of a particular annoyance – which can then be cut off. It is of interest to the ISP to determine which of their customers is responsible and to get in contact with them or, if that fails to solve the problem, to disable their account lest the "rest of the Internet" lose patience with them and cease to handle their traffic.

The key sanction that an ISP has is to close an account, but they must also ensure that the user does not immediately open another. This is why "pay" ISPs check for customers opening new accounts with credit cards that were used to operate banned accounts. It is also why the "free" ISPs insist upon CLI from their callers. Of course miscreants may have access to several credit cards or several phone lines, but the number is strictly limited and so the abuse will eventually stop.

ISPs are not especially interested in knowing "whose fingers were on the keyboard" or the formal identity of the owner of an account. Although having customers calling themselves `Mr.M.Mouse, EPCOT, Florida` might be sign of impending trouble (as well an inconvenience to the marketing department, who would find it difficult to send brochures describing new products), an ISP might well be prepared to take the money in the short term. They would do this safe in the knowledge that if "Mickey" makes a nuisance of themselves then they can be easily "TOSsed" – a jargon word meaning to disconnect for infringement of Terms of Service (TOS).

This laid-back approach to formal identity does not suit Law Enforcement.[5] They are specifically interested in correct identification so that in the event of a crime, an appropriate individual can be identified, prosecuted and tossed into gaol.[6] Once Law Enforcement had understood "cyberspace" in the early 1990s and developed the philosophy of "what is illegal offline is illegal online" they started to concentrate on two aspects of traceability in particular: the mapping from dynamic IP addresses to accounts; and the mapping of account usage to particular telephone lines by the use of CLI. Both of these procedures depend on ISP logs, and these logs are seldom retained for extended periods – just long enough to deal with reports of abuse, which usually arrive within a few days.

Unfortunately, Law Enforcement works to different timescales. Until quite recently,

[5] "Law Enforcement" is used in this thesis as a generic term to cover the police, secret services, customs agencies, tax collectors and even officials employed by local authorities to enforce consumer and environmental regulations.

[6] Ross Anderson points out that this is a Western view of the way in which the judicial process should operate. In ancient China magistrates might not establish guilt with finer granularity than clan membership and a verdict might, for example, just require the Tang clan to submit one person for execution – leaving it to the clan leadership to decide who to offer up.

investigators lacked training about the Internet and would arrive weeks or months after events, when specialist officers had become involved in investigations and could point out Internet-related lines of enquiry. Other officers, dealing with serious crimes and serious criminals, are interested in how illegal acts, robberies and terrorist spectaculars have been planned. They wish to examine patterns of communication, which now includes communication over the Internet, from years earlier.

Governments have come to understand that Internet traceability depends on logs (though in practice it depends on much more). They have seen the choice as being between "data retention" where data is retained by the ISPs for policing needs, and "data preservation" where ISPs fail to destroy data upon the receipt of a specific request to keep it safe for some envisaged access. For example, UK ISPs and telephone companies were all asked to preserve data for about four months after the events of September 11th 2001 [79], and twice more in 2005 after the London bombings of the 7th and 21st July [116].

However, data preservation does not really address the needs of the investigators who are backtracking to the origins of a serious crime, because no-one would have known to preserve the data so long before the event. Since it is precisely these investigators who know most about the Internet, they have had the most influence in formulating the policies that Law Enforcement requested, resulting in consistent lobbying for a data retention scheme. They have not only been concerned with the relatively limited period of data retention at ISPs, but also with the changes occurring within traditional fixed-line and mobile telephone companies. The telcos always used to retain data for many years, but it is expensive to archive data,[7] and competition has made them rethink the economic justification of their policies. At the same time, the telco legal teams have become aware of what the business has been doing, and the wider policy debate has led them to conclude that their existing practices are unlikely to be lawful.

The UK Government has never favoured data preservation, but so far it has stopped short of imposing a compulsory data retention scheme because of the likely cost to industry. At present the UK has a voluntary data retention scheme set up under Part 11 of the Anti-Terrorism Crime and Security Act 2001. Officially this is successful, although the details of which companies have volunteered have not been disclosed on the basis that this should remain commercially confidential. At present the UK is "policy washing" their latest thinking by attempting to persuade the European Union (EU) to adopt a "Council Framework Decision" containing a scheme for compulsory retention of particular classes of information, which would then be implemented throughout Europe and of course in the UK.

A sensible policy would be to require particular types of traceability to be achievable if attempted within a given period. However, the policy makers seem to have a limited understanding of how traceability works, or the ongoing structural changes to

[7]There is more to an archiving system than nipping down to the shop for some 300GB drives!

telco and Internet access, and are likely to settle for mechanistic requirements to keep particular types of logs – whether they will be useful or not. At the time of writing (mid-2005), the policy washing process has run into procedural issues relating to which EU "pillar" the requested action falls under and hence what procedure can be used to introduce it.[8] The present "third pillar" activity is widely held to be without a sound legal basis and a "first pillar" Directive is believed to be the only correct procedure. It seems likely that this will cause a considerable delay before a pan-European policy can be agreed.

To conclude this policy section: traceability works well to the user account level, because that's exactly what ISPs need. Once the account is identified it can be disabled and access will cease. Traceability works badly with respect to user identity, because it would cost ISPs money to improve this and the only relevance to them is to prevent serial abuse, and in practice that is adequately controlled by checking for the re-use of CLI, credit cards or even just an (alleged) name and address.

Even if the ISPs were obliged to implement some sort of "know your customer" scheme then it is very likely that this would only mutate the problem into determining whether or not the banking system was capable of locating the abodes of all of their customers; and success in tracking down the perpetrator of an action would be entirely predicated on the somewhat implausible assumption that all ISP customers will handle access credentials in a secure and safe manner.

It is difficult to precisely identify who is operating remote equipment. It is not possible to have physical sight of a national identity card before every surfing session, and so there will always be a gap between knowing who owns an account, knowing where it was operated from, and being sure about who was actually using it.

Meanwhile, the contribution from Governments is increasingly towards imposing mechanistic requirements for data retention that will extend the period during which existing traceability mechanisms can be exercised, whilst doing nothing to improve the accuracy of the process.

2.4 Related work

2.4.1 Traceability manuals

The mechanics of traceability have long been known to people familiar with TCP/IP and SMTP, the email transfer protocol. However it was only with the advent of sig-

[8] A "third pillar" action would relate to "Police and Judicial Co-operation in Criminal Matters" and would be decided by qualified majority voting in the Council of Ministers, with the European Parliament consulted, but not having any real power. A "first pillar" action would see data retention as being a market harmonisation issue and would involve the bringing forward of a Directive and, if – as is usual – it used the co-decision procedure, then the European Parliament would have the right of veto.

nificant quantities of email spam (in about 1995) that there was any attempt to create documents that attempted to describe the process for the benefit of others. One of the earliest such documents was “Figuring out fake E-Mail & Posts” [61] first published on Usenet in December 1995. Some of the information it contains is incorrect[9] but the basics of using `Received` header fields, `whois`, `dig` and `traceroute` are essentially correct.

In April 1997, the London Internet Exchange (LINX) started a project to document how traceability worked. This was at the instigation of Keith Mitchell, LINX’s CEO, who had become concerned that some ISPs were failing to keep track of who their users might be, just when there was considerable concern about child pornography and the spam issue was beginning to become really serious. He believed that a simple “tool kit”, documenting the “Best Practice” employed by leading ISPs, would benefit the entire industry and clearly demonstrate a responsible attitude. This latter issue became especially significant with the creation of the “Association of Chief Police Officers, Internet Service Providers and Government Forum” (later renamed the Internet Crime Forum): a discussion group for the police, Home Office, Department of Trade and Industry (DTI) and the ISP industry. The Forum became extremely keen for the work to be completed, seeing it as an essential way for Law Enforcement to learn how traceability worked and to raise general standards within the ISP industry so that, as the introduction put it:

> *“Anonymity should be explicitly supported by relevant tools, rather than being present as a blanket status quo, open to use and misuse.”*

The document was initially edited by Andrew Hilborne. I became involved with the project in 1998 and volunteered as the new document editor, reworking the limited amount of existing material into a manageable form and recording a great deal of extra information. The document was formally adopted by LINX as “Best Current Practice” (BCP) in May 1999 [32]. It has been very widely circulated since then and is often quoted in international forums, such as those established by the G8, as the definitive document on the topic.

Since 1999, there has been a profusion of documents produced on traceability and especially on how to read email headers. Many of the “anti-spam” websites have their own material, a typical example of which is the “Reading Email Headers” webpage on `www.stopspam.org` [96].

An excellent modern treatment of the rôle of log files in assisting with traceability is the June 2004 JANET Guidance Note on “Logging” [41]. After addressing the legal issues involved in keeping logs, a number of worked examples show how logs can be used to determine responsibility for end-user actions – and indeed how they can be used to expose forged information that could otherwise implicate the innocent.

[9]Message IDs are trivially forged, so examining them is little more than a heuristic.

2.4.2 Tracing DoS and DDoS attacks

Because tracing Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) attacks is so hard, a number of proposals have been made to assist with this:

- In July 2001, Bellovin *et al.* proposed "ICMP traceback" whereby routers send valid info about their identity and the nature of the flow along with 1 in 20 000 of the packets they handle [13].

  A site suffering from a DoS or DDoS attack would therefore receive machine-processable information about where the packets are coming from. The idea has some difficulties; it only works well if lots of routers implement it, and it will be ineffective if the attacker can spoof traceback packets (a trust model that permits the signing of reports would be hard).

  The IETF created a working group called "iTrace" to consider the topic, and various further improvements were proposed, but ISPs realised that the results would not be useful, and work has ceased.

- Also in 2001, Snoeren *et al.* proposed a "hash based ICMP traceback" where routers hashed the contents of packets and used the resultant digest value to set a single bit in a Bloom filter [123].

  With several independent hashes it is possible to get good differentiation between packets. When a DoS stream occurs, the router can be interrogated to determine if a particular packet (modulo the expected variations in contents), passed through it. However, it is unclear whether the large amount of fast RAM needed would be economic. Also, organizations will not wish to let others interrogate their equipment; and hence traceback would still rely on co-operation from upstream network operators.

Note that in each case, the effect is to identify a subset of IP address space from which the packet has emanated. It will not on its own identify the true source. Hence the main use of these techniques, if employed, would be in improving the "time to block" for an attack rather than for traceability.

2.4.3 Survivability

Lipson produced a CERT/CC white paper in 2002 [92] that examines the problems of traceability on the Internet, in specific relation to cyber-attacks. He identifies the key technical issues as being a failure to design for tracing, failing to provide for highly untrustworthy users, the growth of the Internet far beyond the research network originally envisaged, and more traffic being hidden within protocol "tunnels". On the human side, he sees system administration skills declining on average, cross-border multi-jurisdictional attacks hampering investigations, and the bad guys using automated tools that include explicit measures to destroy audit trails. As technology trends, he picks out an increasing use of dynamic linkages between user and IP address and the growth of anonymising services. The view he takes is that since defensive measures are unlikely to be effective, it is crucial to have deterrence through tracking and tracing.

Having surveyed various tracking and tracing methods from the literature at that time, the later sections of the report discuss future work. He points out, almost in passing, that one is often more interested in tracking an individual rather than the IP address they used. He suggests that solutions to some problems will lie in establishing trust in identities (whilst preserving privacy), others can be addressed by being vigilant about resource usage (for example, he cites "SYN cookies" [17] as an elegant way to avoid resource exhaustion).

His overall view is that one should be looking at "survivability" as the key characteristic of future systems, bending in the face of attacks, rather than "security" which is too often a fragile all-or-nothing property.

2.4.4 The Freiburg privacy diamond

In [152] Zugenmaier *et al.* introduce the 'Freiburg privacy diamond' as an attacker model for evaluating anonymity systems for mobile computing. They consider possible linkages between entities as in Figure 2.3.

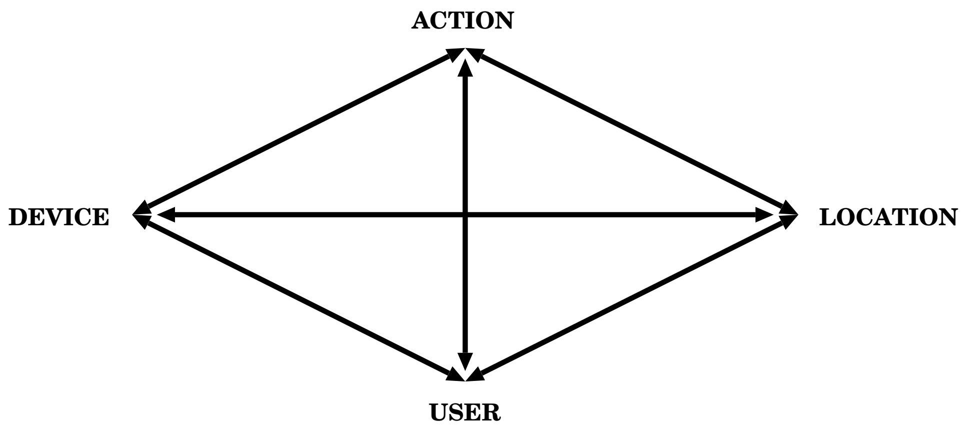


Figure 2.3: The 'Freiburg privacy diamond'

The idea is that you can link a user with an action either directly, as shown by the vertical arrow, or perhaps by linking the user with a particular device and that device with the action, following two arrows to make the linkage. The idea is taken forward in Zugenmaier & Hohl [153] by expressing the effect of anonymity mechanisms to be the obscuring of particular sets of connections. For example, if a user is known to be at a particular location they can only be anonymous if it is impossible to link action with location, device with location and user with device.

Applying this model in the current context we can see that Step 1 (establishing an IP address) is forging the link from action to an abstract notion of a device. Steps 2 and 3 (establishing the owner of the IP address and the user account) attempt a binding between the abstract notion and the actual device, blurring this with an abstract notion of the user. Step 4 (locating the user) involves the links from device to location and location to user.

2.5 Conclusions

In this chapter I have set out the four basic steps that are involved in classic traceability, determining "who did that". I have then explained the problems encountered when attempting to follow these steps, but only considering the issues that were known and well-understood up to about 1999, the time of the LINX BCP document.

There is, of course, a fundamental limitation to traceability in that sometimes the tracing process will identify a machine that is just an innocent "stepping stone", relaying traffic on behalf of someone else. If this relaying is legitimate, then there may be some logs of where the traffic came from and the traceability process can continue with the new IP address. If the relaying is not legitimate, then there may be no logs – leastways none that can be trusted – and unless there is ongoing traffic to observe, the trail will run cold.

This limitation on what traceability can achieve has meant that much recent work on such topics as tracking down "botnet" operators has concentrated on how the groups of compromised machines are grown and managed. At present botnets typically use dynamic DNS for rendezvous and IRC servers for command and control. The Honeynet Project has published an overview of how botnets operate [72], but much current anti-botnet work remains secretive and unpublished, albeit with successful botnet takedowns occasionally appearing in the press [142]. Traceability does of course play its part in this work, but it is not being used end to end.

Also, traceability can continue to be used if the trail, broken at one place, can be picked up again elsewhere. Traffic confirmation is an extremely powerful way of showing that traffic passing over one link is linked to traffic flowing on another. For example, Zhang and Paxson [150] showed that this linking could be made reliable with some simple time-based algorithms, even when some flows were using secure encrypted protocols such as SSH (secure shell).

However, even when the immediate source of the packet is exactly what is being sought, and there are no stepping stones to hop across, a key difficulty with traceability has already become apparent in this chapter. The problem is that the traceability is implemented by the ISPs for the benefit of ISPs. It works moderately well down to the account level, where an ISP can determine which of their customers needs to be educated, or disconnected. However, other players who wish to use traceability are generally interested in determining which particular individual they should interview or imprison. Identifying individuals needs traceability to function right to the edge of the network, and in many cases beyond.

A failure to understand this disjunction of requirements led to considerable difficulties in the 1990s. Law Enforcement felt that the ISPs had magical powers to trace people that they were failing to share. Equally, ISPs resisted proposals to add extra complexity to systems that only barely functioned, because the tracing capabilities being proposed were of no real commercial interest to them.

Nowadays, although everyone involved in traceability on a day-to-day basis has an appreciation of the range of needs and the multiple steps needed to establish the source of an action, the notion that "traceability is all about logs" persists at intergovernmental levels.

The consequent concentration of policy initiatives on data retention, to the exclusion of many other issues, is especially disturbing because in the next chapter we shall see that traceability can be rather more fragile than anyone back in 1999 ever really suspected.

Chapter 3

Traceability Failures

If I listened long enough to you
I'd find a way to believe that it's all true
Knowing that you lied straight faced while I cried
Still I look to find a reason to believe

— Tim Hardin, 1966

This chapter examines failures of traceability on the Internet, concentrating on technologies and scenarios that either postdate the LINX traceability work or were missed altogether when that document was created. This collection of information is, to my knowledge, unique and some of the specific issues relating to "broadband" are being recorded in a public document for the first time.

Although many failures come about because deployed systems are unable to keep sufficient logging information, others result from insufficient thought about traceability in the initial system design. The later part of the chapter describes a number of failures that ought not to occur, but demonstrate how invalid, albeit completely understandable, assumptions about the consistency of information can lead to inaccurate conclusions.

3.1 Dynamic connections

Traceability has long had to deal with dynamic access to the Internet, since dial-up links receive a new IP address for each new phone call. Crucially, this variable linkage from IP address to identity was directly under the ISP's control. Therefore, provided some constant identifiers (account name, CLI information) were recorded, the ISP was able to identify who was responsible for an event – at least sufficiently well to revoke the account credentials being used.

However, the rising popularity of Network Address Translation (NAT) and Dynamic Host Configuration Protocol (DHCP) has transferred the ability to identify which machine performed some action from the ISP to the customer. This is not usually a problem for the ISP, who can still implement sanctions at the account level, but it is a significant issue for others who care about traceability, such as Law Enforcement, with their goal of precise identification.

3.1.1 Network Address Translation

Network Address Translation (NAT) was originally invented [141] as a way of tackling IP address depletion and the associated problem of the growth of the global routing table. It was described in RFC1631 [53] and has become a popular scheme for connecting households and small businesses to the Internet. This is because it allows several machines to share an account with a single IP address, which is usually all that is supported by the cheapest tariffs. Pressure on ISPs to restrict IP address usage has meant extra paperwork for anything other than the most trivial of subnet allocations – so there is always a wish to charge more for extra IP addresses. Furthermore, since the initial demand for multiple machine installations was mainly from businesses, ISPs were motivated to charge a premium for blocks of IP addresses; and they have generally continued this practice.

In its most common form, a NAT device handles a single Internet-facing IP address and an internal network of anything from one or two machines to several dozen. It keeps a table of all connections made by the internal machines, and rewrites IP headers as they pass through the device so as to fix up the source address and port number in a consistent and appropriate manner. Extra complexity arises with protocols such as FTP that embed IP addresses within the data portion of packets; but as modern protocols are supposedly designed with NAT in mind, this complexity ought to be limited (though the problems that SIP, the relatively recently developed protocol used for Voice-over-IP, has with NAT implementations shows that significant problems are continuing to occur [140]).

Looked at from the Internet side, a NAT device with a single IP address appears to be a single machine (although careful examination of the data streams does permit an observer to detect individual hosts behind it [14, 87]). This means that the device acts as a form of firewall – incoming connections will be refused unless it has been configured to pass them on to a specific internal machine.

Low end (consumer-oriented) NAT devices often have no logging facilities at all, though more expensive devices (which will also handle a range of IP addresses and not just one) are usually capable of sending connection information to a `syslog` server. However, the volume of logs created will be substantial, so unless a company is using the logs for monitoring employee activity, or feeding them into an Intrusion Detection System (IDS), it is unlikely they are being recorded at all. In a consumer setting this is not much of an issue to Law Enforcement – it would be standard practice to seize all computers in a house and forensic examination would show which one was involved in some particular illegal activity. However, in a business with large numbers of mission-critical machines, where no NAT logs were available, such an approach might be seen as an unacceptable imposition.

3.1.2 Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) is intended to be a zero-configuration method of setting up hosts on a local area (Ethernet) network [48]. Hosts broadcast a request packet that is responded to by a DHCP server (where possible, using Ethernet level addressing to reach the requester). The server allocates an IP address and passes this back along with other configuration information, such as the subnet mask, gateway address, local domain and so forth.

The IP address can be statically allocated, in which case DHCP merely avoids the need to expressly configure the host, or it can be dynamically allocated from a pool, which means that visiting machines can easily be accommodated. DHCP servers allocate an IP address for a "lease time" and may extend the lease before the time expires[1] or may allocate a new IP address.

Clearly, traceability within a DHCP system ought to be straightforward since logging will show the historical correspondence between IP address and host (identified by Ethernet MAC address). Most host-based DHCP servers are capable of generating logs, and professional sysadmins will usually ensure the logging is enabled and that they have records of the MAC addresses for the machines they permit to access the network. However, the DHCP service is often provided by a firewall device, an ADSL or cable modem, or by an Ethernet switch. Where DHCP is merely an extra feature added on to another device, there may be no logging capability at all, or the systems will only be capable of emitting information using `syslog`; but it would be extremely common for no `syslog` server to be present. Hence, accurate last-hop traceability is seldom available when these simple systems are used. That said, they often use very long default lease times and clients tend not to release addresses as they are switched off overnight, so it is common for hosts to use the same IP address for days at a time. Thus, in practice, if a machine is currently using an IP address then this is a strong indication that it will have been using it in the recent past; although there will seldom be evidence of this "beyond reasonable doubt".

3.1.3 Data retention and validity

As mentioned in the previous chapter, Governments have become enamoured of the idea that retention of logging data will fix cyberspace traceability issues. However, NAT and DHCP show the limitations of this perspective.

[1]The DHCP protocol actually has several timers and a complex state machine, but the details need not concern us here.

Unlike logs of dial-up connections or customer address records, the relevant logs are not held by a small number of telcos and ISPs, but are in the hands of a vast number of individual sysadmins at end-user sites. Logging may not be possible, logs may not be recorded, logs may not be retained and logs certainly cannot be trusted. This last point is especially significant – an investigation that accesses ISP logs will give little thought to the possibility that the ISP sysadmin might be a participant in the wrongdoing. Even at the largest of companies, the odds of involvement by someone who has access to the logging records are greatly increased.

3.2 New connection technologies

This section looks at how traceability works with some relatively new ways of connecting to the Internet, 802.11 wireless, and two "broadband" technologies, ADSL and cable modems.

3.2.1 Traceability of 802.11 wireless

There are many ways of setting up wireless networks, but an extremely common arrangement is a wireless base station, permanently connected to the Internet, that provides service for roaming clients in the near vicinity. In this arrangement the base station almost invariably provides both DHCP and NAT – with all the problems for traceability that have just been described.

Even if the NAT and DHCP issues have been addressed, wireless systems have a further problem in that the roaming clients may not be entirely under the administrative control of base station owner. Where systems are operated in public spaces such as airports, hotels or coffee-shops, there is usually some traceability because credit card details will be used to pay for the access. However, when systems are operated by consumers in their homes, access may be possible from nearby houses and streets. Sometimes people deliberately operate open access policies.[2] More usually, consumers simply fail to understand why they might wish to secure their networks. When they do wish to prevent unauthorised access, they often fail – sometimes unknowingly – because the configuration instructions are incomplete, opaque, or changes in the software mean that they are now just plain wrong.

Furthermore, even where security has been successfully put in place, there are significant technical problems with the current generation's confidentiality scheme [59] and keys can be recovered by eavesdroppers within a few minutes or hours [108, 132]. Although many wireless access points offer filtering based on MAC address, and may even log which MAC addresses are used, these MAC addresses can be sniffed

[2]For example the CONSUME project (`www.consume.net`) promotes the self-provision of a broadband Internet structure by means of neighbourhood level co-operation.

from the airwaves and then trivially spoofed when the associated hardware is not operating.

What this all means is that untraceable wireless access to the Internet is widely available and is already being used for criminal purposes. For example, in Canada in November 2003, Toronto police stopped a car that was travelling the wrong way down a one-way street. The occupant was naked from the waist down and was viewing obscene images of children on a laptop computer [44]. The police concluded that the material had been downloaded over a wireless link from insecure access points in local houses.

In another recent case, which I have been told about privately, UK police raided a flat looking for the sender of "phishing" emails that were being used as part of a fraudulent scheme to steal money from bank accounts. The flat's occupant was a bank employee, which had persuaded the police that they had the right man, although he seemed to be using a different name for the purposes of the scam. As it turned out, their suspect had a wireless access point with no security enabled, and it rapidly became apparent that this was likely to be the source of the emails and the bank employee was innocent. Officers remained on the scene, waiting for carpenters to arrive to fix the smashed-down front door. Whilst they were still there, by a lucky chance, someone walked down the street and – to avoid walking upstairs – called up to attract the attention of his mate, who lived in another flat opposite... since the shouted name matched the one the police were seeking, they were able to arrest the right man after all!

3.2.2 Traceability of ADSL

In the UK almost all ADSL infrastructure provision is by BT, the incumbent telco. In the standard consumer-grade "DataStream" product, signals travel from the ADSL modem to a Digital Subscriber Line Access Multiplexer (DSLAM) in the local exchange. The DSLAM sets up a Permanent Virtual Circuit (PVC) over BT's ATM network to a "Home Gateway" machine, which is handed authentication credentials by the ADSL modem. These credentials specify the ISP providing service and so they are passed to a RADIUS system at that ISP. The ISP can then check that the user knows the account password, and allocates an IP address to be used.

The traceability in this system starts from the use of a particular IP address at a particular time. The ISP's RADIUS logs can then be consulted to establish the credentials that were used, i.e. this yields the user account involved, which will have contact information in the same way as a leased line does.

However, on careful inspection, it can be seen that the system does not involve any binding between the credentials and the circuit information available to the DSLAM. What this means is that stolen credentials can be used with another ADSL modem and, provided that its local DSLAM connects to the same Home Gateway, then all of the traceability will point at the owner of the credentials rather than the actual user. Of course this subterfuge should come to light if the traceability is pursued in a different way – viz: the logs at the Home Gateway machine are consulted to determine which PVC offered up the credentials, so the circuit can be traced to the DSLAM and thence over the permanently connected wire to the originating premises. However, I am told (off-the-record), that in 2003 when the police first tried this alternative method of tracing (having determined that the credentials' owner had a cast-iron alibi), they found that the Home Gateway had not kept the necessary logs. I have approached BT to confirm that this problem has been fixed, but they refuse, on principle, to comment on the detailed operation of their system.

It is interesting to analyse this failure from an economic perspective. When stolen credentials are used for ADSL access the ISP will not lose out – their services tend to be flat rate and they need only ensure that credentials are not simultaneously in use for two or more sessions, which they enforce on their RADIUS system. If there is abuse, then the ISP has the option of closing the account (or issuing new, uncompromised, credentials). Equally, BT does not lose out because they charge the ISPs for ADSL connections and are indifferent as to who uses them. If there is abuse, then BT is merely a carrier and will not be involved.[3] Hence there was no revenue-protection reason, even at second hand, for the Home Gateways to log associations between credentials and circuits and from a strictly economic standpoint it is unnecessary to do so. The major requirement for traceability right to the edge of the network comes from Law Enforcement – so the costs to BT of fixing the loophole are almost solely incurred to assist the police.

3.2.3 Traceability of cable modems

Cable modems using the DOCSIS standard [26] generally have two MAC addresses, one for the modem itself and one copied from the CPE (customer premises equipment) – the router or computer that is attached to the modem. After power-up, the modem will scan for a downstream frequency on which to receive data. It will then await an Upstream Channel Descriptor (UCD) that specifies the frequency/width/modulation details for an upstream (return) channel. It then needs to see a Bandwidth Allocation Map (MAP) packet which indicates the time slots it can use. It will then use "ranging requests" to establish appropriate power levels for its transmissions and to obtain fine tuning information on frequency and timing. Finally, once the transport medium can be used appropriately, it will use DHCP to obtain an IP address for itself and will fetch a configuration file using TFTP. This file gives details of the service that has been paid for and will, for example, specify limits (caps) on transfer rates.

[3]They have substantial statutory protection as a "mere conduit" under s17 of Statutory Instrument 2002 No. 2013 "The Electronic Commerce (EC Directive) Regulations 2002".

In order to prevent theft of service the cable company will usually insist that the modem MAC address is pre-registered.[4] Security considerations in the DOCSIS standards have concentrated on requiring that the MAC address must not be alterable, that the configuration file must not be accessible by the customer and in arranging that some (mainly TV) content is distributed encrypted. Attacks initially involved cloning modems, so the cable company will scan for the same MAC being re-used on different cable segments and blacklist it if there is duplication.

In the past few years, theft of service efforts have focused on "uncapping" (changing the configuration file to permit higher speed data transfers). On some modems it has been found to be possible to provide an up-rated configuration file via the CPE interface at boot time (this appears to be a development procedure that was not removed from shipped systems). The TFTP client finishes its activity before the modem has had time to deal with the physical layer configuration for the cable, and so the official file is never fetched. The cable companies now monitor for this lack of TFTP traffic – so you can now obtain programs that will spoof the requests to fool them... this type of "hack" has led on to the full-scale re-engineering of cable modems, and is now a small industry in its own right, operating from websites such as `www.tcniso.net`.

Once the physical layer is operating, the CPE facing section of the modem uses DHCP to obtain another IP address for the CPE to use. It then bridges Ethernet frames across to the customer equipment itself. Once again, the cable company will usually require that the MAC address of the customer's equipment is registered. Since some companies prohibit the use of routers that permit multiple machines to be connected (and they might identify the equipment manufacturer from the MAC address) and indeed just because it can be a nuisance to re-register, it is now common for routers to be able to "clone" an existing piece of equipment's MAC address onto its own hardware and supply that as its identity.

Traceability for cable modems is achieved by taking the IP address being sought, consulting the DHCP logs to establish the MAC address and then determining which customer registered this MAC address. Hence theft of credentials will mean that you can register and use a new MAC address in an apparently untraceable manner. Of course the customer whose credentials were stolen will not have service (their MAC address will have been cancelled) so the problem will be detected, unless the thief quickly resets the MAC address back to the original value. However, even if the theft is discovered, there is still a problem in locating the machine that has been using the MAC address, because there is no binding between the (untraceable) MAC address used for the CPE and the (traceable) MAC address used for the cable modem itself.

However, this traceability failure can be addressed by taking a holistic approach to

[4]NTL, a UK cable company, avoids the need for pre-registration by providing a limited function configuration file to unknown modems, which allows access to a special registration website which asks for customer credentials and then registers the modem to that customer.

the problem and cutting across several different levels of abstraction in the protocol stack. NTL have an interesting scheme whereby changes of MAC address are permitted and automatically handled, provided that they are not too frequent.[5] This means that they must have a workable scheme to bind the DHCP request to the modem that is making it. They have not documented how they do this, but one possible way would be to examine the "upstream bandwidth allocation" subsystem and establish in which "mini-slot" the DHCP request was transmitted. This can then be tied to a "service ID" which will have been allocated to a particular modem (details are in the DOCSIS Radio Frequency Interface Specification [125]).

Other traceability issues arise when modems are stolen (or cloned) in that although they can be blacklisted, there may be significant issues pinning down their physical location if they are used before that occurs.

However, overall, cable modems have fewer problems with traceability than ADSL does. The reason is almost entirely economic, in that the security features that have been added to prevent theft of service put significant obstacles in the way of being untraceable. Nevertheless, once again, there is a need for specific investment in traceability features to avoid falsely accusing customers whose authentication credentials have become compromised.

3.3 Is CLI really trustworthy?

Although broadband access to the Internet continues to grow, dial-up access remains widely used. As discussed in the previous chapter, traceability works well to establish which account was responsible for an action – but if it is desired to locate the individual involved, then this is done by using CLI. However, there are a number of practical problems with CLI, which will now be considered.

3.3.1 Generic CLI

Many systems supply a generic CLI rather than an actual CLI. For example, all calls made from the University of Cambridge (many thousands of phones) have the same CLI, provided by the central switch. Similarly, some cut-price calling card systems (where you enter a card number before the number you wish to reach) do not relay the CLI from the originating phone. As will be discussed in more detail below, Voice-over-IP systems, a fast-growing sector, usually offer a customised CLI for calls that break-out from the IP world into the traditional phone system, but generic CLI will be provided for systems that offer universal SIP access to "800" numbers (the phone calls are free to the person operating the gateway, so this isn't uncommon).

[5]No more than four per three hour period [147].

Of course, in all these scenarios, the system that generates the generic CLI will have some logs and some traceability of its own. However, the recipient of the call cannot easily judge the quality of this traceability. The system may accept calls where the CLI is withheld, or the logs may be kept for a relatively short period. I have personal experience from my time working at an ISP of investigating an account that had been created solely for spamming Usenet, and finding that the CLI was generic (a calling card system marketed to students). So this is not only an actual problem, but also one that is not immediately obvious to an ISP.[6] Only when a particular CLI is investigated will it become clear that the ISP's security policy has been subverted by another system which operates different criteria for admission control.

3.3.2 Forged CLI

As was explained in the previous chapter, the industry regulator, Ofcom, makes it a requirement that subscribers must be contractually bound to provide only CLI they are permitted to use. However, there is no general requirement to configure telco switches to reject incorrect CLI. This is currently formulated in an Ofcom Network Interoperability Consultative Committee Specification [105] as a requirement on the telco to validate the CLI (and fix up incorrect values) unless there is a contractual "special arrangement" in which case the customer promises to behave. A customer who breaks the contract would lay themselves open to civil action, but at present in the UK (and USA), they do not commit a criminal offence for the deception itself.

In practice, very complex arrangements are often in place, which make it hard for the phone company to validate the CLI values they are presented with. A large corporation may wish to present a standard "reception" number no matter which of dozens of individual sites made a call. Another company may route outgoing calls from dozens of sites, with many disparate CLI values, across their own infrastructure and deliver calls to the public network wherever it is cheapest to do so. In both cases, almost every phone company switch would need to validate every CLI value offered against a remote database, which is not currently practicable.

What this means is that the CLI value provided by a PABX (private automatic branch exchange, i.e. the customer's own phone system) will often be trusted to be correct – and this in turn means that anyone who can reprogram the PABX will be able to provide any CLI they wish, with the changes they make unlikely to be logged anywhere. In the UK, in practice, ISDN connections that might be purchased by individuals will restrict CLI provision to a very small number range; however the freedom to set a wide range of values is available on the more high-end products.

It should be noted that the reprogramming of a PABX to supply forged CLI may be done by unauthorised insiders, or by external intruders who have access to a

[6] In this particular case, it may not have been obvious to the caller either. Without being able to interrogate them, it is hard to say whether their anonymity occurred by design or through chance.

control interface – commonly left enabled by phone system installers to allow them to correct faults remotely, and which may well still have the manufacturer's default password set. Much of PABX related fraud relates to inadequate controls on DISA (Dial In System Access or Direct Inward System Access), i.e. the ability to dial in to a PABX and make an outgoing call at the company's expense. However, some fraud is done by means of remote access interfaces and it would be naïve to believe that people with this level of access would always refrain from altering the CLI on the fraudulent calls that they made.

Recently, intentional forging of CLI has become available in the USA and Canada as an openly advertised service. In September 2004, a company called Star38 announced a commercial service for spoofing CLI which they were targeting at Law Enforcement, debt collectors, private investigators and similar "good guys". The mechanism appeared to be their system calling both ends of the conversation, with any CLI of the caller's choosing given to the target to mislead them. The system was withdrawn within days, with the entrepreneur claiming to have been harassed and delivered a death threat [16]. A similar system continues to be offered by many other companies such as Camophone (`camophone.com`), PI Phone (`www.piphone.com`), Spooftel (`www.spooftel.com`) and Telespoof (`www.telespoof.com`), and in practice the CLI can be forged by almost anyone with a copy of `asterisk`[7] and a telephone operator that does not police the values being set.

Voice-over-IP (VoIP) telephony (i.e. the transmission of voice calls over TCP/IP protocols on the open Internet) is often associated with services that permit such calls to break out to the traditional telephone network. In some cases these outgoing calls are assigned a CLI of a specific geographical number, which the operator will accept incoming calls on, routing them across the Internet to the VoIP system. The user is expected to choose a number prefix which either reflects their location or that minimises the cost of other parties calling them.

In other cases, the VoIP call will be given a CLI number of the user's choice, as is the case, for example, with 1899's VoIP service (`http://www.1899.com/voip.php`). In the UK, the Ofcom rules mean that that a user must promise 1899 that the phone number provided is theirs, but anecdotal evidence is that the checks made are merely limited to ensuring that the number exists. This is essentially the same problem just discussed with PABXs, except that one no longer needs any expensive hardware but can just sign up for a service on a website. Although Internet access is not available via 1899 VoIP, any system that relied on CLI as an authenticator for signing up for, or reconfiguring, a service could be misled into believing that the request was traceable.

In all of these cases, although the CLI for the call is bogus, there may well be traceability information at the SS7 level (in the telco switches) and so, in principle,

[7]`asterisk` (`www.asterisk.org`) is an open source program that provides Linux users with a full-featured PABX. With appropriate hardware for interfacing to ISDN it will set appropriate CLI information on an outgoing call.

the call can be traced back to where it entered the public telephone network. That location could, again in principle, hold details of the true caller so they would be traceable. However, if the phone number held by a CLI spoofing company was a VoIP geographical number, and that was used from an IP address where the call had been made with spoofed CLI... then clearly at some point the traceability is going to break and some other method of tracing, such as examining the credit card details used to pay for the service is more likely to lead to the person being sought.

3.4 Are logs all they're cracked up to be?

The starting point for the traceability process outlined in the previous chapter is that you have a log of some kind and that log contains an IP address. This section examines how the data held in logs can be somewhat misleading.

3.4.1 Picking out the IP address

In the SMTP protocol (RFC2821 [86]), clients connect to servers, say `HELO` (or `EHLO`) to identify themselves, and then transfer their email. The server appends to the start of each incoming email a `Received` header field that records information about the client. For example, a "standard" install of Exim [69] will record:[8]

```
Received: from nidd.cl.cam.ac.uk
        ([128.232.8.175] helo=cl.cam.ac.uk ident=[abcdefg])
        by mta1.cl.cam.ac.uk with esmtp (Exim 3.092 #1)
        id 1DI4Ya-0007tf-00; Sun, 03 Apr 2005 13:50:24 +0100
```

which indicates the email arrived from `128.232.8.175` and a reverse lookup indicated that this was called `nidd.cl.cam.ac.uk` and an IDENT (RFC1413 [131]) protocol request elicited the response `abcdefg`. During the protocol exchange, the machine actually said "`HELO cl.cam.ac.uk`" (which is a violation of RFC2821 because there's no machine called that, but Exim tolerates this minor infraction).

However, suppose that the machine had set the string returned by a reverse DNS lookup to be `10.0.0.1` and, improperly, commenced the SMTP protocol with a "`HELO 192.168.0.1`" command. Then the `Received` header field would be:

```
Received: from 10.0.0.1
        ([128.232.8.175] helo=192.168.0.1 ident=[abcdefg])
        by mta1.cl.cam.ac.uk with esmtp (Exim 3.092 #1)
        id 1DI4Ya-0007tf-00; Sun, 03 Apr 2005 13:50:24 +0100
```

[8]Modern versions of Exim will suppress parts of this information if there is duplication, and there is rather more to Exim's handling of reverse DNS than indicated here. See Section 3.5.7 for more on this topic.

which might mislead the naïve into tracing the origin of email as if it had come from `10.0.0.1` or even `192.168.0.1`. This is a real practical problem because `Received` header fields often fail to conform to the (rather widely drawn) standards. So, although it is possible to determine how Exim behaves, if `Received` header fields generated by an unknown email server are being examined, then it may be a matter of luck whether the correct element is considered.

Caramés and Martínez [27] point out a related problem in logging lines generated by the Apache web server. By default, Apache generates logging lines in Common Log Format (CLF). For example (wrapped to fit):

```
192.0.2.17 - frank [10/Oct/2000:13:55:36 -0700]
             "GET /apache_pb.gif HTTP/1.0" 200 2326
```

where `192.0.2.17` is the IP address of the client that issued the HTTP `GET` command. If `HostnameLookups` is turned on then this IP address is replaced by the result of a reverse DNS lookup – but if this is a text string that resembles an IP address, such as `10.1.23.45`, then this string will be placed into the CLF line and it will look as if a completely different IP address issued the command and the reverse DNS lookup failed:

```
10.1.23.45 - frank [10/Oct/2000:13:55:36 -0700]
             "GET /apache_pb.gif HTTP/1.0" 200 2326
```

It should be noted in passing that the Apache documentation for `HostnameLookups` does not mention the way that it replaces valid information with flawed data in an irreversible manner. However, it does support `HostnameLookups=Double` which will do a forward lookup on the value obtained from the reverse lookup in order to check for consistency; the comparison is made with the `tcpwrappers` setting of `PARANOID` which does at least hint their might be some security aspect to this setting.

3.4.2 Log folding and other artifices

Log files may be very large and so it is unusual to inspect them directly. Instead, it is standard practice to extract the lines which are needed using a tool such as `grep` and then process only the subset of lines that were picked out. This generally works well for *ad hoc* requests, though where data will be needed on a regular basis there is a strong case for putting the logs into a proper database management system.

The difficulty with `grep` is that it knows nothing of the structure of the data it is processing; it merely looks for a particular string within logging lines. This means that if you can arrange for your identifier to be placed on a different logging line than your actions, then `grep` will generally fail to associate the two. The usual way to do this would be to identify text that is logged verbatim and insert an end-of-line character into it. Clearly, a well-written program will escape any unusual characters (into `\n` text strings, for example) before recording them into the log files, but many systems are not this robust.

A more general example of the problem was demonstrated by Caramés and Martínez (in the same paper [27] cited above). They observed that many systems fetched the reverse DNS for a host and then put that string into the logging file verbatim. This allowed them to play a series of tricks on systems that viewed log files within HTML frameworks. For example, arranging for the hostname returned to be:

```
<script>alert('message text')</script>.infohacking.com
```

caused log analysers to display alert boxes and a program called 'IPlanet 6' could be fooled into suppressing logging lines altogether by arranging for hostnames to start with the magic string "`format=`".

3.5 Confusing responses

In this section I consider the way in which traceability can be made more difficult if the person being traced takes some active steps to hide themselves. These steps will be, ultimately, fruitless, but may be sufficient to fool the naïve or those without access to fancier tools.

3.5.1 Hijacking IP address space

One of the underlying assumptions of traceability is that looking up an IP address in an RIR's `whois` system will yield the contact details for the owner of the relevant network. It turns out that the databases involved have limited authentication and there have been numerous examples of theft of IP space (or "hijacking" as it has come to be called).

A typical *modus operandi* is to identify a "dot-com boom" company that is no longer trading but that is named in a registry database as having been allocated some IP address space. A leased line is then purchased from an ISP and the ISP is requested to arrange for the line to use the stolen address space. In the past, the ISP may well have just gone ahead and set up the routing, but a careful ISP will now check that you are able to respond to email sent to one of the contact addresses in the RIR database. This may be easy to arrange because the dot-com company domain name may have lapsed, or the contact email details may have given a `@hotmail.com` or `@yahoo.com` address that can be acquired because it is currently unused.[9] If the

[9]A friend found that he was unable to transfer his `.net` domain name to a new hosting provider because he had provided a Hotmail address when he set it up, and had long since stopped using Hotmail. For security reasons, the `.net` registrar required him to respond to a verification email sent to the old contact address. It turned out that the simplest way to fix his problem was to re-register with Hotmail and re-acquire the email address. It was then easy to change the contact details for the domain to something more appropriate.

email address cannot be acquired then forging a letter from the defunct company to the registry has been effective in getting them to change the contact details.

Once the address space has been routed, standard traceability information will lead to the defunct company – where complaints will go unanswered. The standard operating procedure when complaints are unanswered, for any reason, is to redirect the complaints to the hosting ISP (the "upstream" entity), which can be identified either from the registry entries, or by executing a `traceroute` to the offending addresses and examining the ownership of the networks traversed just before the destination is reached. Of course it may take some time before complaints are made, or it is realised that they are not being responded to. During this period, the hijacked space can be used with impunity.

The two classic cases of address space hijacking in the public domain [112] involve Trafalgar House and the County of Los Angeles.

In the former case, six /16 blocks were hijacked from a large UK conglomerate that had been taken over by a Norwegian firm. The first anyone knew of this was when Richard Cox, who provided technical support for Trafalgar House, was telephoned by someone complaining about spam. He investigated and found that one of the blocks had been appropriated by a Dutch spammer and the others were now being controlled by "FedFinancial Corp" who were using a brand-new email domain with fax, phone and street address details belonging to a company that set up corporations for a fee. He contacted the ISPs announcing the hijacked space and this led to the discovery of more hijacked space and that ARIN had been misled into believing the FedFinancial had contracted with Trafalgar House (which didn't exist any more) to provide network management services. It took some time, and release of the details to NANOG – an organization for network operators with a widely read mailing-list – before the hijacking was stopped.

In the Los Angeles County case, a /16 that was used for a private network, not connected to the Internet, was appropriated by a small hosting company in Northern California. When one of these machines was used for "hacking", a phone call was made to County officials to complain – at which point the theft became apparent.

This type of fraud was fairly popular during 2003 and 2004 because IP address space is worth money. For example, ARIN charges \$4 500 a year for each /16 owned, APNIC \$5 000 for each /16 allocated and RIPE charges a slightly lower annual membership fee (based on total usage rather than each individual allocation). Hence, people expect to pay for address space blocks, so obtaining space for free, and splitting a larger block into multiple smaller allocations and selling them on, could be quite profitable.

His experiences in the Trafalgar House case led Cox to set up a mailing-list for the professionals involved to swap information about hijacked address space. About a hundred separate thefts have been identified so far, but there has been little recent (2005) activity – because there are fewer obvious targets left to be hijacked, because the registries have tightened up their procedures, and because it is now known that hijacking is being actively monitored for by experts.

3.5.2 Incorrect BGP announcements

BGP, the Internet's interdomain routing protocol, is not especially secure [25]. In particular, "prefix hijacking" permits the unauthorised use of address space. This activity requires the collusion of an ISP,[10] whereby it is claimed that a route is available to a particular block of IP addresses. If the route is shorter than the alternative (counting the number of ASs traversed) or if the route is "more specific" (it is for a smaller subnet), then routers will send packets to the better route. The IP addresses within the stolen subnet can then be used for nefarious purposes in a hard to trace manner – or, worse, other machines can be impersonated.

There are a number of groups monitoring BGP announcements in an attempt to detect prefix hijacking, looking particularly for announcements that are made and then quickly withdrawn. There is particular concern about the key elements of the network infrastructure; for example, 'Team Cymru' (`www.cymru.com`) pay specific attention to any changes in the announcements for the top level DNS servers. There are currently no reliable figures for how much deliberate hijacking is occurring because innocent configuration mistakes are regularly made and it is hard to distinguish these from "enemy action"; Mahajan *et al.* [99] found that errors led to problems with between 200 and 1 200 prefixes per day. A key factor they identified was user interface design problems within router management software.

3.5.3 Misleading traceroute

Traceroute works by sending out IP packets (usually ICMP "echo request" or UDP packets) with monotonically increasing time-to-live values. The assumption is that each of the packets will be routed the same way, so the sender will receive back a sequence of ICMP "time exceeded" packets from intervening routers and an ICMP "echo" or ICMP "port unreachable" packet from the destination.

Dallachiesa Michele has developed a daemon called `hopfake`[11] which swallows an incoming traceroute packet (or whatever type) and forges extra "time exceeded" responses for this packet and a number of those that follow. The idea predates `hopfake`, but Michele's program is by far the best known implementation of it. The subterfuge means that a host can disappear from where it should actually

[10]BGP conversations can be secured against third party attack using an MD5 digest with a shared secret as an integrity check. But, even if this is not used, the protocol is complex and inside information is needed to know where to attack, so at present (although there is considerable lamentation about the lack of security) few if any attacks are occurring on BGP *per se.*

[11]Available from `http://www.acidlife.com/~xenion/`

appear in a `traceroute` log and pretend that it is connected at some other place on the Internet. For example, this `traceroute` output purports to show my laptop connected to NASA via a rather overloaded (and totally imaginary) trans-Atlantic link (from hop 15 onwards):

```
$ traceroute rnc1.al.cl.cam.ac.uk
traceroute to rnc1.al.cl.cam.ac.uk (128.232.15.208), 30 hops max
[...]
10 po0-0.lond-scr.ja.net (146.97.33.33) 3.136 ms 2.869 ms 3.348 ms
11 po0-0.cambridge-bar.ja.net (146.97.35.10) 6.934 ms 5.986 ms 5.747 ms
12 route-enet-3.cam.ac.uk (146.97.40.50) 6.030 ms 6.600 ms 6.643 ms
13 192.153.213.194 (192.153.213.194) 6.390 ms 6.085 ms 6.233 ms
14 gatwick-n.net.cl.cam.ac.uk (131.111.2.98) 6.770 ms 6.721 ms 6.254 ms
15 spider.ncts.navy.mil (138.147.50.5) 70.159 ms 25.330 ms 130.036 ms
16 www.army.mil (140.183.234.10) 25.420 ms 46.783 ms 63.860 ms
17 darpademo1.darpa.mil (192.5.18.104) 25.847 ms 240.187 ms 49.609 ms
18 iso.darpa.mil (192.5.18.105) 120.384 ms 47.911 ms 172.156 ms
19 demosparc.darpa.mil (192.5.18.106) 103.113 ms 126.097 ms 50.091 ms
20 border.hcn.hq.nasa.gov (198.116.142.1) 103.670 ms 97.419 ms 139.894
21 foundation.hq.nasa.gov (198.116.142.34) 139.574 ms 129.631 ms 119.58
22 rnc1.al.cl.cam.ac.uk (128.232.15.208) 91.816 ms 156.515 ms 146.091 ms
```

The traceability process outlined in the previous chapter does not strictly require the use of `traceroute`, but it is often used to resolve ambiguities as in 3.5.1 above.

The proper technique for establishing how a network is connected to the rest of the Internet is to consult the BGP routing table. For example the “looking glass” at `route-server.exodus.net` will respond:

```
route-server.savvis.net>sh ip bgp 128.232.15.208
BGP routing table entry for 128.232.0.0/16, version 47700
Paths: (2 available, best #1)
  Not advertised to any peer
  3356 786
    208.172.146.29 from 208.172.146.29 (206.24.210.26)
      Origin IGP, localpref 100, valid, internal, best
      Originator: 206.24.210.26, Cluster list: 208.172.146.29
  3356 786
    208.172.146.30 from 208.172.146.30 (206.24.210.26)
      Origin IGP, localpref 100, valid, internal
      Originator: 206.24.210.26, Cluster list: 208.172.146.29
```

This shows that `128.232.15.208` (my laptop’s IP address) is presently being advertised by AS786 (JANET) who peers with AS3356 (Level3) who then peers with Savvis (AS3561). Hence any questions about the advertising of this IP address should be addressed to the appropriate people at JANET, no matter what the RIPE entry for `128.232.15.208/32` might say (though of course for this example, there is no inconsistency).

However, this is not really the end of the story. AS paths may themselves be bogus and have extra entries added to attempt to show that an announcement is being relayed rather than originated within a particular network. Because routing decisions are generally made on path length, this is sometimes done intentionally as a crude form of traffic engineering. The only way to be absolutely certain about traffic origin is to interview all the relevant network engineers and then assess any inconsistencies that come to light.

3.5.4 "Anonymous" Internet Relay Chat (IRC)

In 2004 I was asked by the UK police to comment upon a traceability problem they had encountered. They had an IRC log of illegal activity which gave the miscreants identity (anonymised of course) as `fred@host-1-2-0-192.isp.co.uk`. They had approached `isp.co.uk` with details of the incident, but the ISP had told them that no-one was using `192.0.2.1` at the given time (even when everyone checked the clocks for daylight-saving adjustments and all the other standard problems that can occur when tracking users down).

What had actually happened was that `fred` had logged on to an IRC server. The server had done a reverse DNS lookup on `fred`'s IP address and recorded the result. If there had been no response to the reverse DNS request, then the IP address would have been used instead. The server would then have checked for an IDENT daemon on `fred`'s machine.[12] If there was an IDENT response, then the result would have been used, otherwise the IRC login name is used, preceded by a tilde (`~`). Finally, the IRC user has a nickname (`BADBOY`, say), and so participants in the IRC channel would see:

```
BADBOY [~fred@host-1-2-0-192.isp.co.uk]
```

When the police mounted an investigation using the `BADBOY` information they had seen in their IRC session, the ISP resolved the hostname and checked who was using the particular IP address at the particular time. However, incorrect reverse DNS had been supplied and – because one of the LINX Best Practice principles (record the actual IP address) had been ignored – this broke the traceability. The police were slightly unlucky in that some IRC servers, for example those on DALnet, do check the reverse DNS by immediately determining whether you get the original IP address by resolving the string supplied (and if this check fails then the server operator is informed and the IP address is used instead).

It occurred to me that a likely explanation for the traceability failure was that there was a simple typo in the ISP's reverse DNS configuration files, so I wrote a program to check for inconsistencies between the forward and reverse DNS. The program was only single-threaded and hence took several days to run (the ISP owned a /11 and

[12]An IDENT service [131], is seldom provided – except on Unix machines, where it can be used to distinguish between logged in users.

a number of other large blocks) but it eventually listed 1 953 527 addresses where there was a mismatch between forward and reverse DNS.[13]

Unfortunately, none of the inaccurate reverse DNS values matched the string the police were looking for. This suggests that some other piece of address space, not owned by the ISP, had been set up to respond with the ISP's string, thereby misleading all concerned into thinking that someone would be easy to trace whereas in fact they were almost anonymous. Clearly a full-scale sweep of reverse DNS values would unmask the IP address being used, but this would be a considerable effort and, if the dissembling DNS server supplied random values (perhaps based on the identity of the requestor), then it would not necessarily be easy to spot amongst all the other invalid settings.

A final warning about reverse DNS is appropriate. Much software assumes that there will only ever be one PTR resource record for a particular `...in-addr.arpa` enquiry. However, multiple records ought to be returned, and sometimes are, by machines that operate under multiple identities. Nevertheless, in very many cases, only the first of a set of multiple responses will ever be processed.

3.5.5 Preloading traceability information

Usenet articles are distributed using a flood-fill algorithm and as they pass from server to server their `Path` header field is extended to indicate which machines have seen them. Servers are meant to avoid offering articles to peers that have already handled them. People who wish to hide the injection point of articles will "preload" the `Path` header field with a misleading list of servers.

Hence all one can say with confidence about a dubious article is that the injection point will have been one of the servers along the `Path` and very much the same issues of provenance arise as with the preloading of AS paths discussed in Section 3.5.2. However, if multiple copies of the article can be obtained then these, along with some knowledge of the usual paths followed by articles, are often sufficient to identify a likely injection point – which will often turn out to be an insecurely configured server and usable by anyone.

Emails collect `Received` header fields as they pass from host to host and so by reading them in order, one can reconstruct the original source of the email. In just the same way as with Usenet `Path` header fields and BGP AS paths, these email header fields can be preloaded and spammers often do this to try and obscure the source.

[13]The bulk of the mismatches clearly related to IP addresses that were currently unused, and the majority of the rest seemed to be caused by having two different hierarchical naming schemes and failing to ensure they were used consistently. It was fairly obvious that, although in the distant past there might have been full consistency, there were no mechanisms for ensuring that this was maintained as the network evolved and new subnets were added or the rôle of existing subnets was changed.

Rather an old example of this is:

```
Received: from pd9e320ce.dip.t-dialin.net ([217.227.32.206])
    by punt-2.mail.demon.net  id aa2109796; 29 May 2003 3:35 GMT
Received: from kennettd.freeserve.co.uk (26475 [39.240.199.101])
    by  arnet.com.ar (8.12.1/8.12.1) with ESMTP id 15687
    for <richard@happyday.demon.co.uk>; Wed, 28 May 2003 20:39:33 -0700
Received: from gmx.de ([125.223.22.213])
    by comcast.net (8.9.3/8.9.3) with SMTP id 11434
    for <richard@happyday.demon.co.uk>; Wed, 28 May 2003 20:39:28 -0700
Message-ID: <1421811705ulfkdugCkdss|gd|1ghprq1fr1xn@Writeme.com>
From: "Tatoo" <lyb13483oe29278m@yahoo.com>
```

where the true source was `217.227.32.206` (a German dial-up host) but the header fields had been forged to look as if before that the email came from `39.240.199.101` which is within an address block that IANA have yet to issue to an RIR.

The ill-informed would look at the later `Received` header fields and might ask Freeserve or Arnet (an Argentinian ISP who own the `arnet.com.ar` domain) about the spam. Although the forgery is a little crude, in that there's an obvious discontinuity in the server names, the really nice touch is the faked five second delay shown from the (alleged) previous hop (`125.223.22.213`, which was unallocated in 2003, although 125/8 was issued to APNIC in January 2005).

Interestingly, exactly the same pattern (using unallocated space and a five second delay) showed up in a May 2003 email that a foreign police force were examining in relation to a harassment case. Instead of `arnet.com.ar`, a UK company was named and the foreign police had asked their colleagues at the UK's National Hi-Tech Crime Unit (NHTCU) to assist them. The NHTCU people rapidly identified the correct source (an insecure Californian ADSL connection) but asked me for a second opinion (I concurred) because they had not encountered faked header fields of this type before.

3.5.6 Establishing the source of email

Reading email header fields to establish the source of email requires detailed local knowledge. The failsafe method is to read downwards (backwards though time) until one encounters a transfer from a machine that it is not trusted by you (with your local knowledge of local machines). The last trusted `Received` header field then tells you the IP address from whence the email arrived. You then talk to the owner of that machine and use their local knowledge (of the machines local to them) to establish where the email came from before that. Clearly this is time-consuming, so there is a tendency to trust remote systems if they appear to be run by reputable companies.

Considerable extra complication occurs when email has been delivered to one system and then forwarded again to another. This breaks what linkage there may be (possibly none) between the way that the email appears to be addressed (in the To header field, etc.) and the actual addressing within the SMTP protocol ("the SMTP envelope"). Some progress can be made in assessing authenticity by examining the DNS entries ("MX records") that identify the email servers for the domain, but these are really just hints rather than firm evidence that header fields are genuine and have not been forged.

Goodman considers the same problem from a different angle [63]. He was interested in being able to apply per-user spam filtering rules on individual workstations. Rules that check upon the actual source of email (asking if it came from a well-known open server) are very effective. However, it is complex to apply such rules on a workstation within a large company. This is because email may have been received at any one of a number of gateway machines and then forwarded within the company, possibly several times via different division and department servers. The workstation has a header field reading problem: without global knowledge of all of the company's email servers, it will not be able to positively identify the gateway machine when carefully forged Received header fields are present. Goodman puts forward an algorithm which will either yield a result or a "don't know", and suggests various fixes, including an internal DNS service that dispenses global knowledge, and cryptographic header field signing (though this latter scheme would involve a key distribution scheme which is probably harder than the original server identity dissemination problem).

3.5.7 Misleading reverse DNS for email

I decided to combine several of the ideas from this section into a practical test to determine how effective they would be in fooling people who regularly determined the source of email. With the kind assistance of Ben Laurie of 'The Bunker' I arranged for a Unix machine to be configured with an extra interface whose IP address was 213.129.75.58 but whose reverse DNS was, completely improperly, set to be bay15-f8.bay15.hotmail.com.

An email was then sent from a Hotmail account to establish the header fields added by Hotmail. These include an X-Originating header field to indicate the sending IP address used to access the Hotmail web interface. A similar email, with these header fields preloaded (and the datestamps changed appropriately) was created. This email was forged to implicate 163.195.192.74, a South African government machine.

The email was sent via the new .58 interface so that upon arrival at its destination it might be expected to gain a Received header field:

```
Received: from bay15-f8.bay15.hotmail.com [213.129.75.58] (helo=hotmail.com)
    by mail.highwayman.com with smtp id 1DZYGo-000N36-62
    for test@highwayman.com; Sat, 21 May 2005 19:00:18 +0100
```

but the remainder of the header fields would be the plausible, but entirely forged:

```
Received: from mail pickup service by hotmail.com with Microsoft SMTPSVC;
     Sat, 21 May 2005 11:00:15 -0700
Message-ID: <BAY15-F8AA02948F4F41C0615D64D3090@phx.gbl>
Received: from 163.195.192.74 by by15fd.bay15.hotmail.msn.com with HTTP;
    Sat, 21 May 2005 18:00:15 GMT
X-Originating-IP: [163.195.192.74]
X-Originating-Email: [r65536@hotmail.co.uk]
X-Sender: r65536@hotmail.co.uk
From: "Richard Clayton" <r65536@hotmail.co.uk>
To: "Test Account" <test@highwayman.com>
Bcc:
Subject: Email Traceability Test #1 from Richard Clayton
Date: Sat, 21 May 2005 18:00:15 +0000
Mime-Version: 1.0
Content-Type: text/plain; format=flowed
X-OriginalArrivalTime: 21 May 2005 18:00:15.8667 (UTC)
       FILETIME=[F13BE0F8:01C55E2E]
```

The correct method of tracing the origin of this particular email should have started by examining the `213.129.75.58` address and concluded it was sent from 'The Bunker'. Their next step should then be to contact a sysadmin at 'The Bunker' to seek further assistance. However, my intention was for the reverse DNS setting of `bay15-f8.bay15.hotmail.com` to fool readers into skipping this address by deciding it was obvious that Hotmail had handled the email, so that they would then consult the other `Received` header fields and conclude that the original source was some six thousand miles further south of the actual origin in rural Kent.

In practice, more than half the people (several research students, an ISP abuse team manager, a senior hi-tech-crime police officer) correctly identified the source, however, three (a security manager, another ISP abuse team manager and a police officer from their training school) were fooled into believing in the South African origin, and it could well fool others who were less familiar with the procedure of working downwards though the header fields assessing their probity, rather than starting from the (apparently) easy to understand header fields at the (apparent) end of the chain.

It is of course difficult from a simple *ad hoc* experiment such as this to know how close anyone else came to being fooled, or how much care and attention was given by extremely busy people to an unsolicited puzzle. Therefore it would be unwise to read too much into the results or to draw any conclusions about the likelihood of errors being made when the results really mattered.

However, the experiment highlighted one important issue that I had not foreseen. In practice, the `Received` header field data seen by the email recipients was not as misleading as set out above, but instead it looked rather more like this:

```
Received: from [213.129.75.58] (helo=hotmail.com)
    by mail.highwayman.com with smtp id 1DZYGo-000N36-62
    for test@highwayman.com; Sat, 21 May 2005 19:00:18 +0100
```

The reason for the lack of the reverse DNS string is that the email server has not only done a reverse lookup on the IP address, but has also checked that doing a forward lookup on that will yield the original IP address. Because the `hotmail.com` domain only includes legitimate entries, the check will fail, and the misleading string is omitted from the email header field. This belt-and-braces validity checking now appears to be widespread and all of the email servers encountered in the experiment used it. For example, it was introduced in Sendmail 8.8.6 in June 1997; and in Exim it was long thought to be present – because it was believed that `gethostbyaddr()` made this check automatically – but it turned out that this was not the case on all operating systems, so explicit code was added in v4.30 (Dec 2003).

3.6 Conclusions

The previous chapter concluded that there was more to traceability than just logs, and highlighted the disjunction between a system operated by ISPs that was good at locating accounts and the needs of Law Enforcement to locate people.

This chapter has emphasised this message, with modern access methods exhibiting the same features – the ISP is in a position to identify and disable an account, but there are significant problems in locating the actual perpetrator. Pressure by Law Enforcement has led to some improvements in traceability where centralised fixes are possible; but the profusion of unintentionally open wireless access points is unlikely to change until the manufacturers conclude that security will be a better selling feature than the ease of initial set-up.

A key difference in this chapter's examination of traceability compared with the previous one is the emphasis on the fallibility of various mechanisms and tools. It was always just assumed that CLI identified a phone line, `traceroute` identified a route, the correct IP address was extracted from email headers, `grep` displayed all relevant logging information, reverse DNS was valid and ISPs did not promulgate fake AS information over BGP. Fortunately, most of the time, things work just fine, but that's no guarantee of performance in any particular case.

For Law Enforcement, traceability is usually used as "intelligence", it leads you to the right front door, you kick that down and seize the people and machines inside. If it starts to become seen as "evidence", a vital part of a court cases, then it is reasonably likely to meet the civil test of "on the balance of probabilities" but considerable care will be needed to ensure that it meets the criminal standard of "beyond reasonable doubt".

However, traceability evidence would not only have to be accurate but would also have to be presented in the adversarial setting of a court room. This would require considerable attention to detail and the ruling out of some of the more unusual, albeit quite plausible, scenarios I've discussed above. It may well be possible to examine the logs of BGP announcements and see that there is no prefix hijacking announcement of a "more specific" route to the IP address in question, and hence the RIR database can be relied upon to demonstrate ownership. However, if it was just assumed that the routing was normal and no checks were made, then a clued-up defence barrister could make a prosecution expert look very foolish indeed.[14]

A final point to note is that Law Enforcement does not approach traceability as being just a technical matter. They also ask themselves how plausible the results are and use that to inform their actions. In the anecdote about wireless access points, the police chose to break down the door because the occupation of the inhabitant was consistent with the crime being investigated. In the test email tracing example, the NHTCU officer made a point of explaining that he was happy with his identification of the source because there was nothing to connect me with a South African origin for the email, but much to link me with Ben Laurie and 'The Bunker'.

Even when the correct source of a communication is identified the police still need to trace to an individual, and they have "out-of-band" methods for this. In the February 2000 'MafiaBoy' case (where a 14-year-old boy ran highly effective denial-of-service attacks against Yahoo!, CNN, E-Trade and others), the Royal Canadian Mounted Police rapidly located the house where the attacker lived, but had to operate a telephone wiretap for many weeks in order to determine which member of the family (father or three brothers) was responsible [144].

In the next chapter, we shall see that still more complexity can arise when people set out to deliberately "frame" someone nearby.

[14]In [126] Sommer discusses the 1994 "DataStream Cowboy" case (the US Air Force Rome Labs intrusion). Although a large amount of forensic evidence was gathered, Sommer shows that significant portions of it could have been excluded by the Judge, or had been handled in inappropriate ways that would have cast significant doubt on its authenticity. This problem is now much better understood than a decade ago; but that does not make it any easier to solve.

Chapter 4

Hiding on an Ethernet

I had to interrupt and stop this conversation
Your voice across the line gives me a strange sensation
I'd like to talk when I can show you my affection
Oh I can't control myself
Oh I can't control myself
Oh I can't control myself
Don't leave me hanging on the telephone

— Jack Lee, 1976

This chapter considers a new method of impersonating a nearby machine in such a way that traceability will lead an investigator to conclude that another machine is responsible for some specific activity. The method, which involves a short-lived denial-of-service at the bits-on-the-wire level, is demonstrated for a 10BASE-T 10 Mbit/s Ethernet system, but the same scheme would work with other technologies as well, in particular with 802.11 wireless systems.

The high-level result demonstrated yet one more time – that traceability can break down catastrophically at the very edge of the network – is of significant importance if one is trying to use traceability to "prove" that a particular machine was used to commit an action, rather than just as a way of narrowing down a list of suspects.

4.1 Machine identification on an Ethernet

There are a number of different types of Ethernet, running at different speeds from 2 Mbit/s to 10 Gbit/s and upwards. They can operate in full-duplex mode, where access to the transmission medium is guaranteed, or in a half-duplex mode, where access is mediated by Carrier Sense Multiple Access with Collision Detection (CSMA/CD). Ethernets may be extended via repeaters, where there is a single collision domain spanning multiple segments, or switched, where an active component arranges for packets to be propagated only on a minimal set of segments and collisions cannot occur between different segments.

In this chapter I shall be specifically considering a 10BASE-T 10 Mbit/s half-duplex Ethernet as defined in IEEE Standard 802.3 [74], but the mechanisms described would work within a single collision domain for any other type of half-duplex Ethernet. The scheme will not work on a switched Ethernet but the basic idea is applicable to 802.11 wireless systems.

4.1.1 Ethernet frames

Ethernets share a common logical structure for a frame, as shown in Figure 4.1:

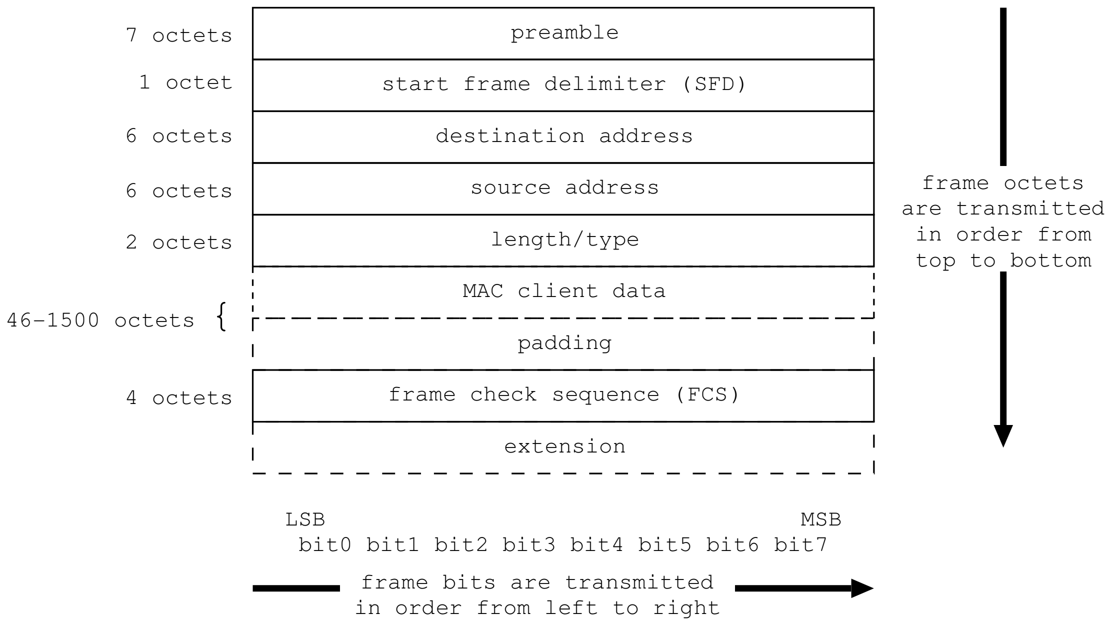

7 octets
preamble
1 octet
start frame delimiter (SFD)
6 octets
destination address
6 octets
source address
2 octets
length/type
46–1500 octets
MAC client data
padding
4 octets
frame check sequence (FCS)
extension
frame octets are transmitted in order from top to bottom
LSB
MSB
bit0 bit1 bit2 bit3 bit4 bit5 bit6 bit7
frame bits are transmitted in order from left to right

Figure 4.1: Structure of an Ethernet frame

The *preamble* is a seven octet sequence of alternating `1` and `0` bits intended to allow the receiver to synchronise its receiver clock with the transmitted signal. It is followed by a *start frame delimiter* (SFD) transmitted as the value `10101011`, i.e. the receiver will see a sequence of `1-0` pairs (the exact number depending on how fast its clock synchronises) finishing with a `1-1` pair.

The *destination address* is the 48-bit address of the destination station(s) and the *source address* is the 48-bit address of the station that is the source of the frame.

The addresses are usually called "MAC addresses" and are intended to be globally unique. A frame can be addressed to an individual station, whose MAC address will start with a zero, or to multiple stations, in which case the address will start with a one on the wire (so the first octet of the written form has an odd value). The destination of "all ones" is the broadcast address and indicates that a packet is to be received by every station on the LAN.

The *length/type* field is a 16-bit value either giving a length for the frame or (as is more usual) if its value is #0600 or greater then it specifies which protocol is encapsulated in the frame; #0800 is used for IP, #0806 for ARP etc.[1]

The *data* field holds an appropriate payload for the particular protocol and, if necessary, *padding* is added to bring it, for 10 Mbit/s systems, up to at least 46 octets.

Finally, a *frame check sequence* (FCS), in fact a Cyclic Redundancy Check (CRC), field is sent, which is calculated over all octets after the SFD. It serves to provide an error detection mechanism. Rather obscurely, the FCS is transmitted in the opposite order to every other field.

The *extension* field is used at speeds over 100 Mbit/s as extra padding to ensure that frames are long enough for collisions to be correctly detected.

4.1.2 Address Resolution Protocol

To send an IP packet over an Ethernet, one places the IP information into the data field of an Ethernet packet, sets the appropriate type field value, and transmits it. Before this can be done, the correct destination address for the frame will need to be determined. This is obtained by employing the Address Resolution Protocol (ARP), first standardised in RFC826 [109].

The initial ARP message is a broadcast packet sent by a station with IP address X that requests "who is Y tell X". A station on the LAN which has been configured to have the IP address Y will respond directly to the sender, saying "Y is" and reporting its MAC address. Because the initial message specifies its source address it is not necessary to use a broadcast response. ARP information is cached with an aggressive timeout which is usually of the order of 20 minutes. To assist with mobility, systems which have just rebooted or been reconfigured send out a "gratuitous ARP", which is a request for their own IP address. This ensures that other machines discard old information that may be present in their caches – and also causes another machine with the same IP address to respond, immediately highlighting a configuration error.

4.1.3 Collisions

Because an Ethernet (in half-duplex operation) is a shared broadcast medium, it is possible for two stations to decide to transmit at the same time. When this happens, both stations are supposed to detect that this has occurred. They then both issue

[1]The # notation indicates a value expressed in base-16.

a "jamming" signal to ensure that the other station definitely detects the collision and then they wait a short while and retransmit the frame.

The scheduling of the retransmission is determined by a controlled randomisation process called "truncated binary exponential backoff". The delay is an integer multiple of the transmission time for 512 bits (for 10 Mbit/s operation), about 1/20 000 of a second; with the integer randomly chosen from the range $[0, 2^n - 1]$ for retry n, except that from retry number 10 onwards the choice is always from $[0, 1023]$. Retries are supposed to continue for at least 16 attempts before an error is reported.

The intent behind the randomisation is to prevent a second collision between the same two stations. The intent of the backoff is to attempt to reduce demand for access to the medium during busy periods.

Because an Ethernet can be constructed from up to five cables separated by repeater units, it is possible for stations to transmit for some time before their data reaches every other station and any collision can be detected. The limitation is caused by the transmission time along the cable (which can be up to 1 500 metres in total) plus the delay time through up to four repeaters. The 802.3 standard allows for this and expects collisions to occur until 512 bits into a frame (at 10 Mbit/s), hence the requirement to pad the data to 46 bytes so that the total frame size, once clocks are synchronised, will exceed this. Collisions that occur after this point are called *late collisions*; 10 Mbit/s systems may retransmit after a late collision, but are not required to do so.

4.2 Previous work on identity spoofing

There are a number of ways of spoofing identity on an Ethernet, which are discussed below. They are seldom described in the academic literature, but are reported, more or less competently, in the informal articles that emerge from the "hacking" community.

There has been rather more formally published work on wireless LAN (802.11) problems and there is a good overview of wireless attacks, some of which involve impersonation, in Bellardo and Savage's USENIX paper [10]. They credit Lough's PhD thesis [94] as the earliest work on these issues; Lough was developing a comprehensive taxonomy of attacks on computer systems and protocols, applying a standard analysis method to the 802.11 protocol and showing that it had similar problems to earlier systems.

It is possible for a user with limited skills to borrow someone else's IP address merely by reconfiguring their own machine to use the other person's setting. However, the "gratuitous ARP" on booting will cause a failure if the machine which truly owns the IP address is currently running. Thus a simple impersonation would require the owning machine to be switched off, or to have been disabled by a denial-of-service attack that keeps it busy, or that causes it to “blue-screen” (lock up), or reboot.

More subtle attacks involve subversion of the ARP protocol. This contains no security provisions at all, and so it is easy to attack.

For example, in “ARP poisoning” the attacker will arrange for spoof ARP responses to be sent to the two stations that are talking to each other. They will then overwrite their cache entries with the MAC address of the attacker and send all their traffic via this “man-in-the-middle” who can record – or alter – the data that is being exchanged. This attack can be made to work on a switched network as well as within a single collision domain.

ARP poisoning can be detected by monitoring programs such as `arpwatch` which keep track of IP address assignments within a network. It can also be prevented by an active component, such as a switch that discards spoofed ARP packets. For example, Cisco’s DAI (Dynamic ARP Inspection) uses a combination of administratively set bindings and snooping of DHCP (Dynamic Host Configuration Protocol) packets to establish the validity of ARP traffic.

On a wireless network there is an extra protocol layer that can also be attacked. Wireless clients and infrastructure access points agree their relationships with an authentication protocol. However, de-authentication messages can be spoofed so as to cause genuine traffic to be ignored. 802.11 networks do not use CSMA/CD in the same way as 802.3 networks because senders may not hear each other, although a receiver might hear both. Therefore a scheme called CSMA/CA (Carrier Sense Multiple Access Collision Avoidance) is available (albeit seldom used) which involves a short request-to-send (RTS) packet to which a short clear-to-send (CTS) response is given. At the end of a successful reception an acknowledgement (ACK) is used. Stations should hear either the RTS or CTS and will avoid transmitting during an appropriate period given by message length information included in these packets. Lough pointed out how an RTS “flood” would lead to a denial-of-service attack, however, Bellardo and Savage found that fielded implementations contained bugs, which meant that the channel reservations were often being ignored.

Lough also suggested a low-packet-count attack on the binary exponential backoff algorithm that is used to share access to the wireless medium. One of the aspects of his taxonomy was “improper randomness”. He pointed out that the usual implementation of this backoff was to use a pseudorandom number generator such as a linear congruential generator. If the seed for this system became known, then an adversary could arrange to transmit at exactly the right time to jam a transmission.

What Lough does not discuss, nor does it seem to have previously been suggested, is that one could wait until one learnt the source address of a packet and then transmit in such a way as to cause an intentional collision with traffic from a specific station. This new idea will now be examined.

4.3 How to hide on an Ethernet

The method of identity spoofing on an Ethernet that is presented in this chapter is for one machine, M say, to "borrow" the IP address and MAC address of another, A say, and impersonate them to send data. The recipient of the frames, B, will think that the proper owner, A, is transmitting and deal with the data accordingly.

Such a simple scheme does not work, because when B responds to the spoofed traffic then it will be received not only by M (who can listen promiscuously for A's traffic as well as its own), but also by A. If it is TCP traffic then A will respond with an RST packet and B will close the connection.

The novel idea is for M to arrange to deliberately collide – on the Ethernet medium itself – with A's transmission so that the RST cannot be transmitted. Once M has finished spoofing then collisions need no longer occur, A can be allowed to send the queued packet, and B will ignore it because by then it will not correspond to any open connection. A will experience delay, but not a complete failure, so provided no timeouts are reached, it will not have a fault to record or report to the user.

The deliberate collision is performed by the simple expedient of examining frames as they are transmitted. If the frame's source address is found to be A's MAC address then a signal is transmitted (`101010101`... would be sufficient) so that A detects a collision, jams for a moment and then dutifully backs off for a period before attempting to resend.

The method leaves very limited traces in any system logs. All stations will see a higher than usual number of collisions, but the Ethernet Management Information Base (MIB) data structures that record this will seldom be examined. Also, A and B will record unusual events in their TCP MIBs, but again these will seldom be consulted. Even if they are inspected, the MIBs are just counters, so there will be no specific temporal link with M's activity.

The method has considerable advantages over traditional identity spoofing in that it can be done whilst machine A is in active use. Indeed it is advantageous to do this since there will almost certainly be other logs that show activity by A and this will reinforce the faith being placed on logs of the spoofed activity. Conversely, if suspicion is to be successfully cast upon A's user then it would be best if they did not turn out to have a cast-iron alibi that they were somewhere else at the time.

Since the method does not employ ARP spoofing, standard tools such as `arpwatch` will not detect it. However, because it relies on colliding with frames, it will not work on a switched network where the hardware prevents collisions between packets on different physical segments. It may also be of limited effectiveness on very large Ethernets if an attempt is made to jam a station that is a long way away; the extra delay of waiting for the source address may just be sufficient to permit A to finish sending a short frame before the collision occurs on its part of the medium.

4.4 Experimental setup

Although the collision scheme is conceptually simple, it proved to be far from simple to demonstrate it working in practice. Some custom hardware was needed, and this was constructed within an FPGA device on one of the department's "teaching boards" – an FPGA development system with a range of peripherals and memories on a standard PCB. The FPGA had an embedded ARM processor and this was used to run a pre-existing port of Windows CE. The port was enhanced to handle interrupts rather more flexibly, and a Windows driver was developed for the custom hardware. This section will now cover this all in more detail.

4.4.1 The Ethernet PHY

The 802.3 Ethernet standard divides receiver design into a number of different sublayers, as shown diagrammatically in Figure 4.2. It gives detailed accounts of the

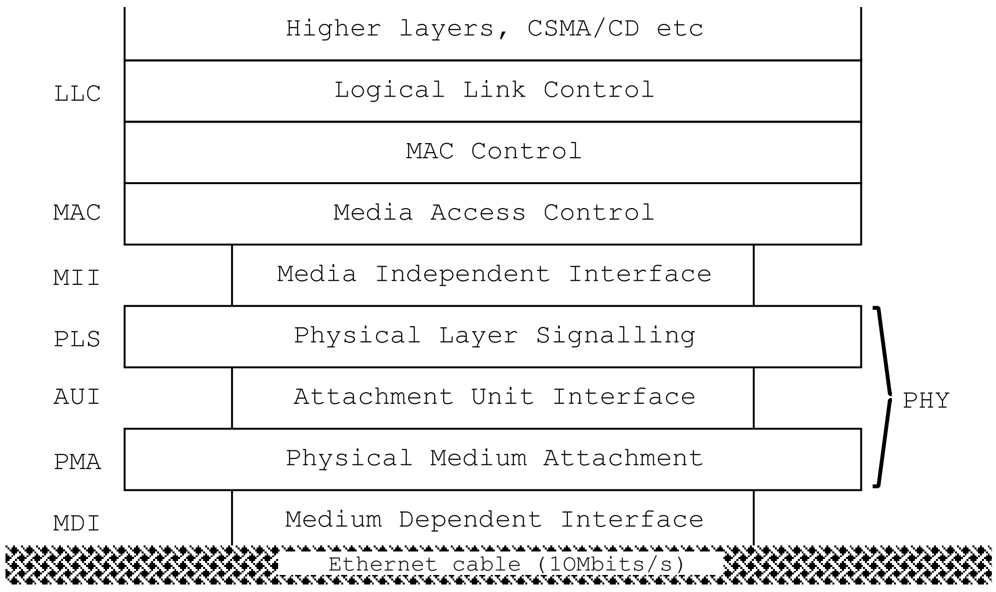

Higher layers, CSMA/CD etc
LLC
Logical Link Control
MAC Control
MAC
Media Access Control
MII
Media Independent Interface
PLS
Physical Layer Signalling
AUI
Attachment Unit Interface
PMA
Physical Medium Attachment
PHY
MDI
Medium Dependent Interface
Ethernet cable (10Mbits/s)

Figure 4.2: Logical arrangement of Ethernet receiver (sub)layers

functions to be performed by each of the sublayers, and the interface between them is precisely specified. From this it can be established that the non-standard functionality required, a deliberate collision with another station's transmission, involves improper behaviour within the Media Access Control (MAC) layer.

Inspecting the documentation for various devices that implemented MACs showed that the decision as to whether or not to transmit was invariably a hardware function. Although some MAC subsystems implement considerable functionality in software, the real-time aspects of handling collisions are invariably placed into hardware. It proved impossible to find a device which could be subverted into performing the required intentional transgression of the Ethernet standards.

In practice, the logical arrangement shown in Figure 4.2 is mapped into three actual component parts: “magnetics” for extracting the signal from the Ethernet cable at the MDI layer, a “PHY” that deals with the Manchester encoding of the signal and then a “MAC” that deals with transferring bytes of data to and from the Ethernet. With higher levels of integration, the MAC and PHY are now commonly found in a single integrated circuit “chip”, although it used to be the case that they would be implemented as a closely-coupled two-chip set.

I was able to borrow a 1996 vintage PHY chip which had been mounted, with the appropriate magnetics, onto a small PCB. It had been developed for the ‘ATM Warren’ project [65] where it was used as part of an interface between the ATM system and 10BASE-T Ethernet. Figure 4.3 is a photograph showing the PCB and the PHY chip – an Intel LXT901 Universal 10BASE-T and AUI Transceiver [75].

4.4.2 Implementing a MAC in an FPGA

The customised MAC with the required non-standard functionality was implemented in Verilog (a high-level hardware design language) and compiled into an Altera EPXA1F484C1 [5], a 4 160 logic element (100 000 gate) FPGA device which can be clocked at up to 166 MHz. The device contains 16 KBytes of dual-port RAM – used for buffering input and output frames – and an embedded ARM922T processor. A standard Excalibur EPXA1 board [4] was used for development purposes, to which a locally designed daughter board had been added (see photo in Figure 4.4). This was then connected via a ribbon cable to the PHY board.

The hardware functionality provided by the MAC design is split into a receiver module, a transmitter module and a main controlling module. The receiver detects Ethernet frames and places them into a circular buffer of seven 2K buffers. For simplicity, no attempt is made in the hardware to determine whether these frames are addressed to this station (or the station to be spoofed) or not. The transmitter works with a single buffer, since performance was not an issue, and, at an appropriate time, taking into account the 802.3 standard, it outputs the buffer onto the Ethernet.

The controlling module deals with interrupts, resets and other housekeeping issues. When the receiver detects an incoming frame that is to be deliberately collided with, it arranges for the transmitter to send a simple `01010101`... signal so that the sender will back off and transmit again later.

The design is relatively straightforward, occupying about a third of the FPGA, the main implementation difficulty being that there are multiple clock domains involved: the fast ARM clock, the 10 MHz transmitter clock generated by the PHY device and a different 10 MHz receiver clock that is regenerated from Ethernet packets created by other stations. This caused considerable difficulty until a rigorous approach was

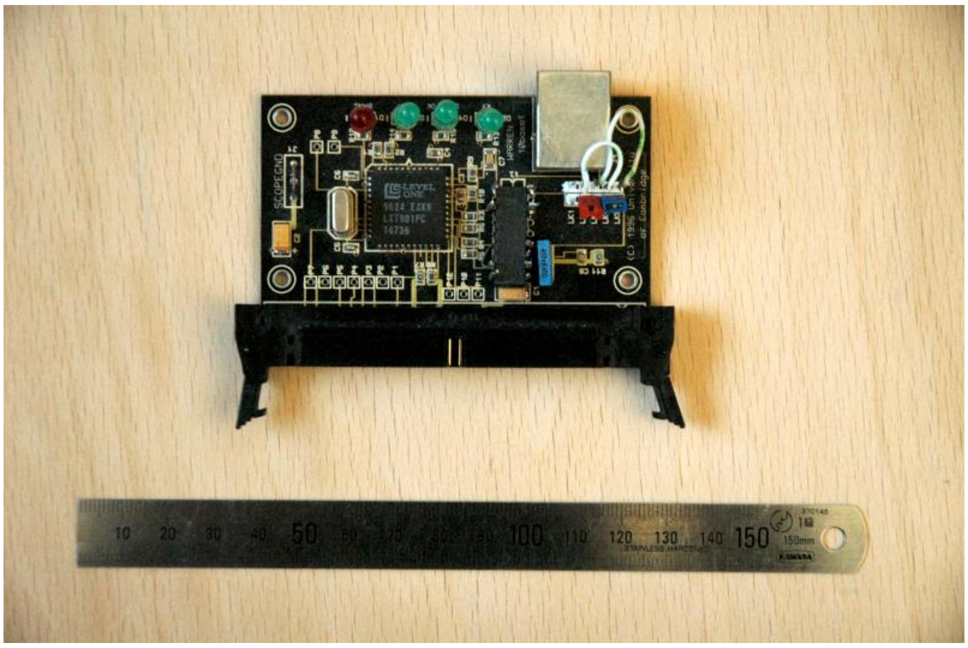


Figure 4.3: Warren PCB showing Intel LXT901

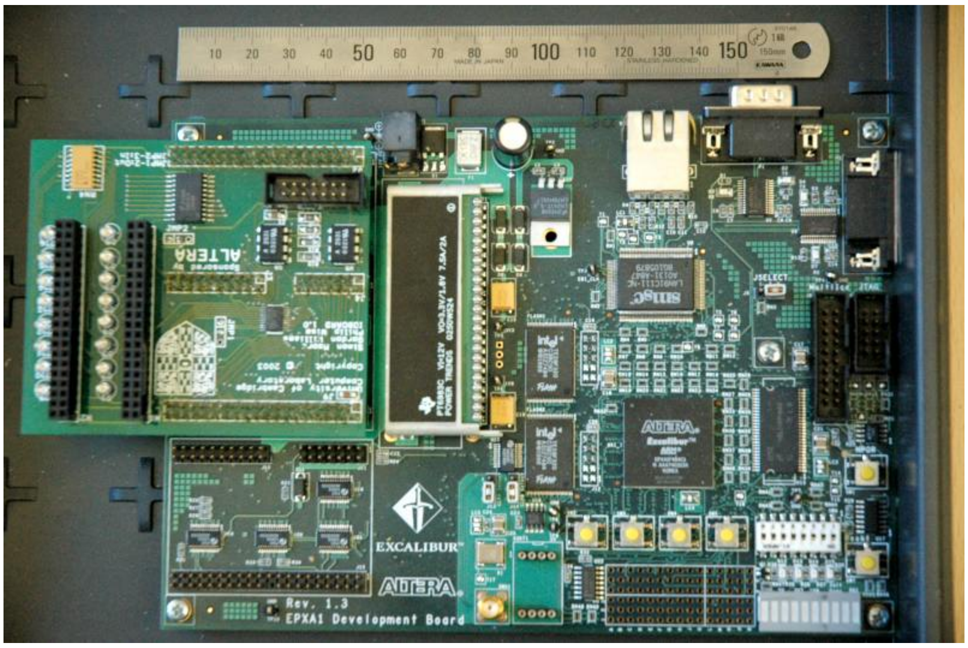


Figure 4.4: EPXA1 development board

taken to ensuring that signals were transferred from one clock domain to another in a proper, glitch-free manner.

4.4.3 Software

Windows CE v4.2 had previously been ported onto the EPXA1 development board by a Cambridge group [103], and this was used as an operating system to support a TCP/IP stack that could be used to demonstrate the spoofing.

The existing Windows CE code contained only minimal interrupt handling, so this part of the kernel was enhanced by adding support for the six interrupt sources coming from the programmable (FPGA) portion of the device. The main portion of the work was developing a connectionless miniport driver for the Ethernet MAC, writing a user level program to provide a demonstrator and creating a “device” for handling an IOCTL interface between these two programs. Figure 4.5 shows how the various components fit into the Windows CE architecture.

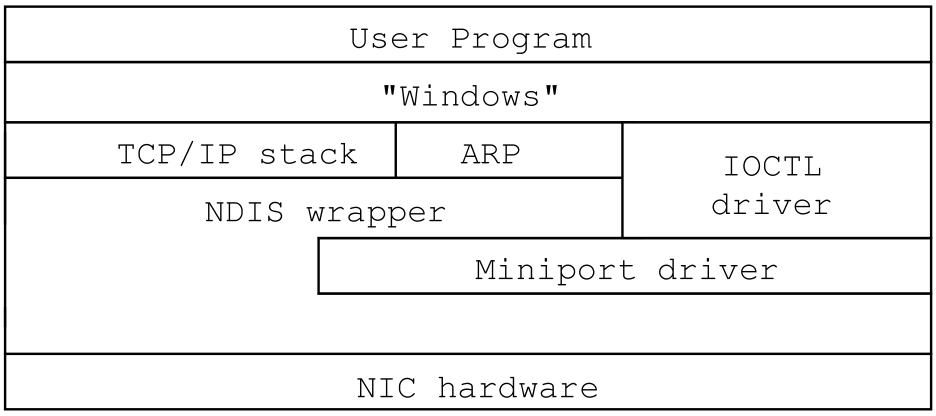


Figure 4.5: Windows CE software component architecture

The key function provided by the IOCTL interface is a request to “spoof” a particular identity. When this request has been made the miniport driver alters the contents of the IP packets it is handling in order to:

- Detect outgoing SMTP packets, replace the source IP and MAC address (currently set to “this machine”) to the spoofed machine. Then fix up the IP and TCP checksums accordingly.
- Detect incoming SMTP packets, replace the destination IP and MAC addresses (of the spoofed machine) with the values for “this machine”. Then fix up the IP and TCP checksums accordingly.

A complete implementation would also deal with ARP packets that the spoofed machine should be handling, and would short-circuit its own ARP requests if the spoofed machine should handle them. However, this adds complexity and is not necessary in demonstration software that can just assume that no relevant ARP traffic will occur during the short period that the spoofing is done.

4.5 Experimental results

A simple experiment was performed to demonstrate that the spoofing worked as predicted. This involved the sending of an email over an SMTP connection, purporting to come from another machine, whilst that machine was running normally.

The experimental configuration is shown in Figure 4.6. The laptop at the bottom left, running Windows 2000, was the one to be spoofed. The FPGA board, top centre, running Windows CE was used to run the SMTP client and do the spoofing. The second laptop, bottom centre, running Windows XP, was used for three purposes; it was used to load Windows CE and configure the FPGA, it provided an SMTP server to receive the spoofed email and it also ran a screen emulator for the copy of Windows CE running on the FPGA because this did not have a video display of its own. The blue box at the top left is a eight-port 10 Mbit hub; the oscilloscope at the right was used for hardware development and to capture traces; and, finally, the keyboard at the bottom right was used to keep notes.

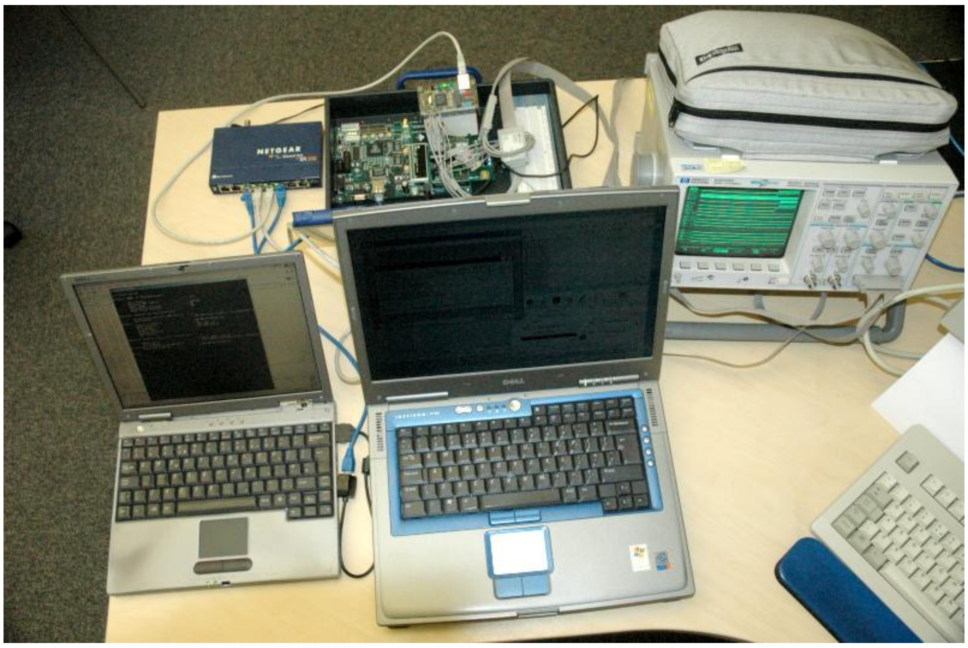


Figure 4.6: Experimental setup

The experiment consisted of a spoofing program running on the Windows CE on the FPGA sending an email to the SMTP server, but using the identity (the MAC address and IP address) of the Windows 2000 laptop. The FPGA hardware was configured to know the MAC address of the Windows 2000 laptop, so that it could choose to collide with traffic emanating from that machine.

With the collision mechanism disabled then the following Ethernet packet trace was recorded when an attempt was made to send an email using a spoofed identity. The trace was captured using `ethereal`, a packet-sniffing program:

```
Time      Source         Destination   Proto

0.000000  192.168.1.2    128.232.110.14 TCP
            1028 > smtp [SYN]      Seq=0 Ack=0 Win=32768 Len=0 MSS=1460
0.000040  128.232.110.14 192.168.1.2    TCP
            smtp > 1028 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
0.000367  192.168.1.2    128.232.110.14 TCP
            1028 > smtp [RST]      Seq=1 Ack=4087568586 Win=0 Len=0
0.033839  192.168.1.2    128.232.110.14 TCP
            1028 > smtp [ACK]      Seq=1 Ack=1 Win=32768 Len=0
0.033874  128.232.110.14 192.168.1.2    TCP
            smtp > 1028 [RST]      Seq=1 Ack=207398712 Win=0 Len=0
```

The first packet is spoofed, and is responded to with a SYN/ACK from the email server (`128.232.110.14`, my laptop's new IP address), which believes the connection is genuine. The true owner of `192.168.1.2` then responds with an RST packet to the unexpected (to it) SYN/ACK packet and the server drops the connection. However, the spoofing machine is unaware of the RST and sends the ACK to finish the three-way handshake – to which the server responds RST because the connection has been closed. The spoofer closes down, and the spoofed machine follows the rules and does not respond to the, still unexpected, RST. Hence no email can be sent.

With the collision functionality enabled then there is no interference from the machine being spoofed, and the SMTP conversation can proceed unhindered:

```
Time      Source         Destination   Proto

0.179326  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [SYN] Seq=0 Ack=0 Win=32768 Len=0 MSS=1460
0.179368  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
0.185828  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [ACK] Seq=1 Ack=1 Win=32768 Len=0
0.191688  128.232.110.14 192.168.1.2    SMTP
            Response: 220 happyday.al.cl.cam.ac.uk
                      Turnpike ESMTP server ready
3.150447  128.232.110.14 192.168.1.2    SMTP
            [TCP Retransmission]
            Response: 220 happyday.al.cl.cam.ac.uk
                      Turnpike ESMTP server ready
3.365744  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [ACK] Seq=1 Ack=59 Win=32710 Len=0
3.650833  192.168.1.2    128.232.110.14 SMTP
            Command: HELO stolen.name
```

```
3.651239  128.232.110.14 192.168.1.2    SMTP
            Response: 250 OK, happyday.al.cl.cam.ac.uk, how may I be of
                      service to stolen.name?
3.762137  192.168.1.2    128.232.110.14 SMTP
            Command: MAIL FROM:forged@stolen.domain
3.762394  128.232.110.14 192.168.1.2    SMTP
            Response: 250 2.1.0 OK, MAIL
3.869111  192.168.1.2    128.232.110.14 SMTP
            Command: RCPT TO:rnc1@cl.cam.ac.uk
3.869502  128.232.110.14 192.168.1.2    SMTP
            Response: 250 2.1.5 OK, RCPT
3.955785  192.168.1.2    128.232.110.14 SMTP
            Command: DATA
3.956101  128.232.110.14 192.168.1.2    SMTP
            Response: 354 Let's have the data now
4.132051  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [ACK] Seq=84 Ack=203 Win=32566 Len=0
4.389926  192.168.1.2    128.232.110.14 SMTP
            Message Body
4.558684  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=203 Ack=117 Win=17404 Len=0
4.564883  192.168.1.2    128.232.110.14 SMTP
            Message Body
4.759835  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=203 Ack=301 Win=17220 Len=0
4.765209  192.168.1.2    128.232.110.14 SMTP
            Message Body
4.961011  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=203 Ack=548 Win=16973 Len=0
5.097897  192.168.1.2    128.232.110.14 SMTP
            Message Body
5.262763  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=203 Ack=565 Win=16956 Len=0
5.269099  192.168.1.2    128.232.110.14 SMTP
            Message Body
5.463946  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=203 Ack=648 Win=16873 Len=0
6.301303  128.232.110.14 192.168.1.2    SMTP
            Response: 250 2.6.0 OK, message received,
                      id <tqRzmTABiDxCBA16@happyday.al.cl.cam.ac.uk>
6.557020  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [ACK] Seq=648 Ack=283 Win=32486 Len=0
6.631513  192.168.1.2    128.232.110.14 SMTP
            Command: QUIT
6.631739  128.232.110.14 192.168.1.2    SMTP
            Response: 221 2.0.0 happyday.al.cl.cam.ac.uk
                      Turnpike ESMTP server closing transmission channel
6.632635  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [FIN, ACK] Seq=370 Ack=654 Win=16867 Len=0
6.639601  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [ACK] Seq=654 Ack=371 Win=32399 Len=0
7.077423  192.168.1.2    128.232.110.14 TCP
            1029 > smtp [FIN, ACK] Seq=654 Ack=371 Win=32399 Len=0
7.077441  128.232.110.14 192.168.1.2    TCP
            smtp > 1029 [ACK] Seq=371 Ack=655 Win=16867 Len=0
```

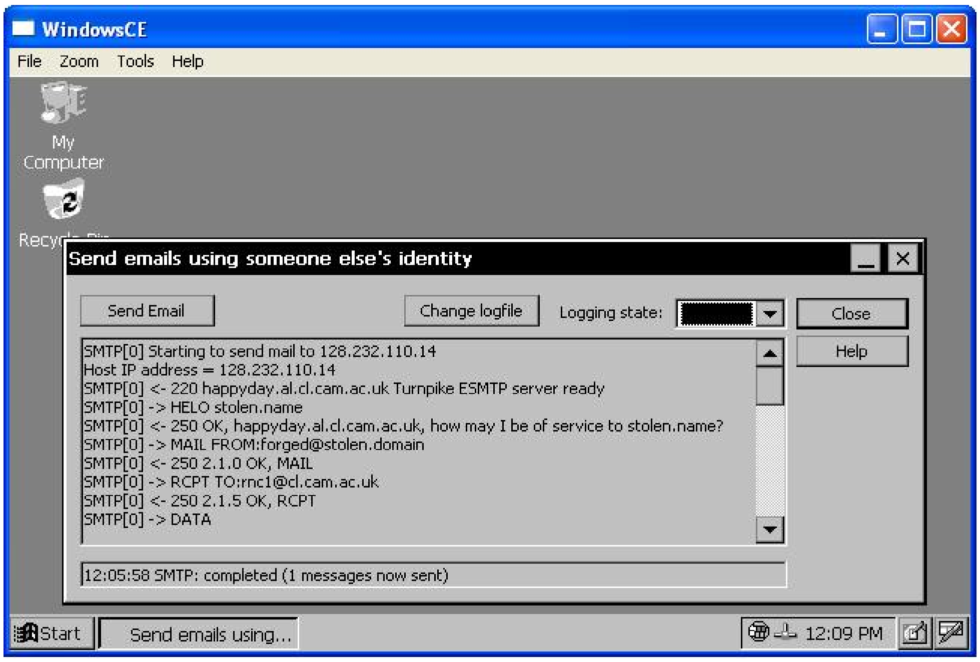
The same trace looks very different when it is examined from the point of view of a copy of `ethereal` running on the machine whose identity is being usurped. This machine tries to send a great many RST packets,[2] which are recorded into the logs, however, because of the collisions, none are actually transmitted. For example, with the RSTs picked out for clarity:

```
Time        Source          Destination  Proto

1.931109   192.168.1.2    128.232.110.14 TCP
             1029 > smtp [SYN] Seq=0 Ack=0 Win=32768 Len=0 MSS=1460
1.931241   128.232.110.14 192.168.1.2    TCP
             smtp > 1029 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
1.931333   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=1 Ack=2970176204 Win=0 Len=0
1.937609   192.168.1.2    128.232.110.14 TCP
             1029 > smtp [ACK] Seq=1 Ack=1 Win=32768 Len=0
1.943610   128.232.110.14 192.168.1.2    SMTP
             Response: 220 happyday.al.cl.cam.ac.uk
                       Turnpike ESMTP server ready
1.943650   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=1 Ack=2970176204 Win=0 Len=0
4.902340   128.232.110.14 192.168.1.2    SMTP
             [TCP Retransmission]
             Response: 220 happyday.al.cl.cam.ac.uk
                       Turnpike ESMTP server ready
4.902457   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=1 Ack=2970176204 Win=0 Len=0
5.117451   192.168.1.2    128.232.110.14 TCP
             1029 > smtp [ACK] Seq=1 Ack=59 Win=32710 Len=0
5.402559   192.168.1.2    128.232.110.14 SMTP
             Command: HELO stolen.name
5.403117   128.232.110.14 192.168.1.2    SMTP
             Response: 250 OK, happyday.al.cl.cam.ac.uk,
                       how may I be of service to stolen.name?
5.403211   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=19 Ack=2970176222 Win=0 Len=0
5.513850   192.168.1.2    128.232.110.14 SMTP
             Command: MAIL FROM:forged@stolen.domain
5.514221   128.232.110.14 192.168.1.2    SMTP
             Response: 250 2.1.0 OK, MAIL
5.514267   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=51 Ack=2970176254 Win=0 Len=0
5.620805   192.168.1.2    128.232.110.14 SMTP
             Command: RCPT TO:rnc1@cl.cam.ac.uk
5.621330   128.232.110.14 192.168.1.2    SMTP
             Response: 250 2.1.5 OK, RCPT
5.621418   192.168.1.2    128.232.110.14 TCP
             1029 > smtp  [RST] Seq=78 Ack=2970176281 Win=0 Len=0
5.707500   192.168.1.2    128.232.110.14 SMTP
             Command: DATA
  etc. etc...
```

[2]The strange ACK value results from setting the acknowledge number field to the same as the sequence number field; where it is displayed as an offset from the other direction's initial value.

Figure 4.7 shows the sending program when it has just sent the email:



```
SMTP[0] Starting to send mail to 128.232.110.14
Host IP address = 128.232.110.14
SMTP[0] <- 220 happyday.al.cl.cam.ac.uk Turnpike ESMTP server ready
SMTP[0] -> HELO stolen.name
SMTP[0] <- 250 OK, happyday.al.cl.cam.ac.uk, how may I be of service to stolen.name?
SMTP[0] -> MAIL FROM:forged@stolen.domain
SMTP[0] <- 250 2.1.0 OK, MAIL
SMTP[0] -> RCPT TO:rnc1@cl.cam.ac.uk
SMTP[0] <- 250 2.1.5 OK, RCPT
SMTP[0] -> DATA

12:05:58 SMTP: completed (1 messages now sent)
```

Figure 4.7: Program for sending email with a stolen identity

and the email that was received read like this:

```
Return-Path: <forged@stolen.domain>
Received: from stolen.name ([192.168.1.2]) by happyday.al.cl.cam.ac.uk
   with SMTP id <tqRzmTABiDxCBA16@happyday.al.cl.cam.ac.uk>
   for <rnc1@cl.cam.ac.uk> ; Thu, 30 Jun 2005 19:22:57 +0100
Message-ID: <demo1@stolen.domain>
Date: Thu, 30 Jun 2005 19:22:02 +0100
From: Impersonated User <forged@stolen.domain>
To: Richard Clayton <rnc1@cl.cam.ac.uk>
Subject: Demonstration email #1
MIME-Version: 1.0

This email actually came from 192.168.1.4

However, not only has it been forged to appear to have come from
<forged@stolen.domain> but also the traceability information in the
Received header field has been recorded by the (honest) recipient
to be 192.168.1.2

This would mislead an investigator into examining the wrong machine....
```

where `192.168.1.2`, the apparent source of the email, is the IP address of the laptop running Windows 2000; i.e. all the traceability information points at that, totally innocent, machine rather than at `192.168.1.4`, the FPGA device running Windows CE that actually sent the email.

A typical collision from the experiment is shown in Figure 4.8. The received data (RXD) trace starts normally with the `101010..1011` pattern, followed by the destination MAC address and then the source MAC address. As soon as the source MAC address is matched, the deliberate interference of transmitting data (TXD) can be seen – and the legitimate transmission then stops and becomes a jamming pattern. The other signals displayed are transmit enable (TEN), the transmit and receive clocks (TCLK, RCLK), carrier detect (CD) and collision (COL).

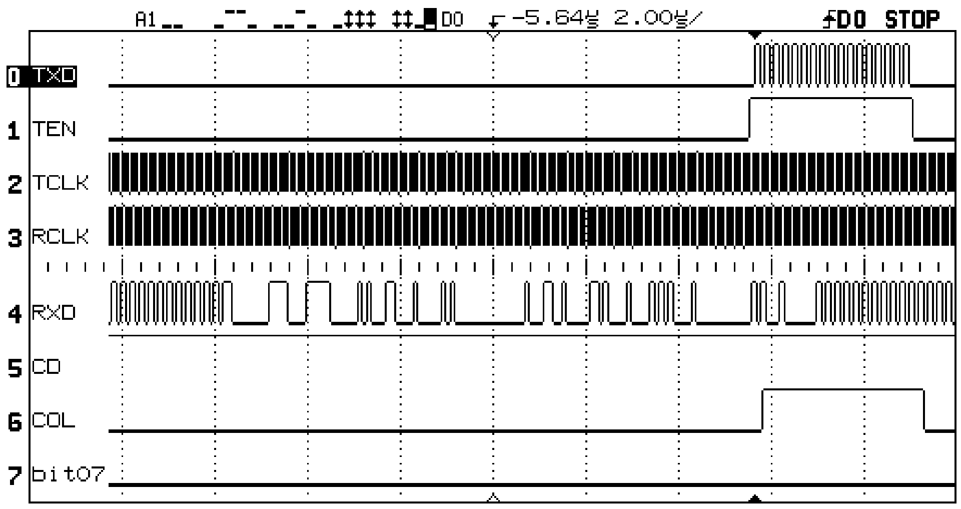


Figure 4.8: An intentional collision on the Ethernet

4.6 Time limitations

In order to assess how important this spoofing attack will be in practice, it is necessary to establish the duration for which collisions can occur without the user of the machine becoming aware of it.

If the Windows 2000 laptop was idle then experiments showed that its identity could be stolen for an indefinite period without any obvious indication being given to the user that RST packets were failing to be transmitted.

If it was not idle then it entirely depended on the program being run as to how quickly the spoofing was detected. Of course the user was not told "someone is impersonating you", but the network did not seem to work properly – something that they might recall if they were eventually accused of the activity that was performed in their name.

For example, a `ping` program sending a packet every second would report data loss within a few seconds. Since data loss is not unexpected, `ping` will continue reporting the loss until stopped – leaving it to the user to decide if this was an error or not.

If the laptop was running a file transfer program such as `scp` then this displays regular progress updates and after five seconds without any data transfer it will report “stalled”. However, the underlying protocol stack will not report the link to have failed, thereby causing `scp` to abort, for at least 20 seconds and sometimes not for 60 seconds or more.

To understand why the detection depends on the program, it is necessary to see what actually happens when collisions occur. The Ethernet specification prescribes retransmission with increasing backoff, and that can be seen in practice, as Figure 4.9 illustrates:

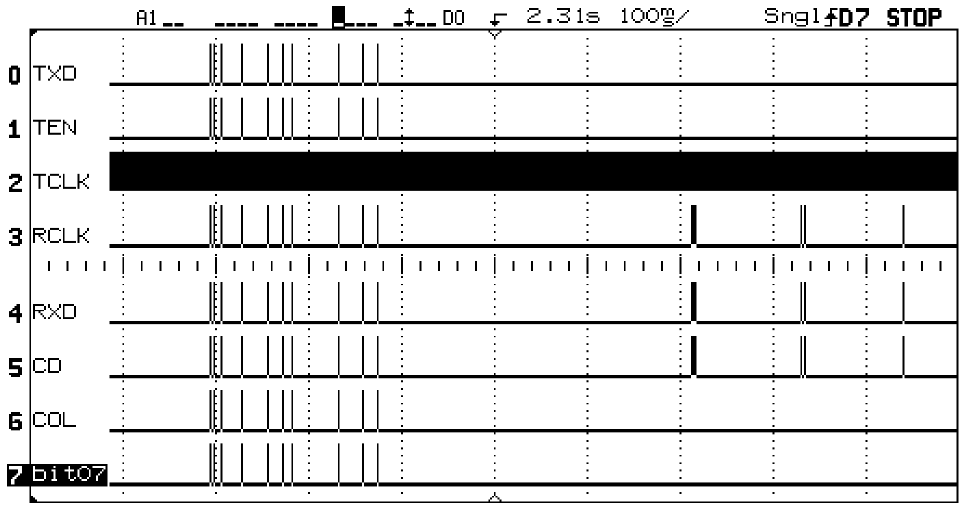


Figure 4.9: Retransmission after a collision

In this trace, with 100 ms per division, the signals displayed have the same identity as those in Figure 4.8 (which was a 2 ms per division), except that `bit07` is now the internal signal that requests a collision. The retransmission delay starts off very short but then lengthens. However, after about 200 ms the Ethernet card (in this experiment an “Intel DC21143 PCI Fast Ethernet Adaptor”) gives up and reports a failure.

In practice, the TCP/IP stack will take no special notice of the Ethernet card reporting a failure, but will treat this event the same as any other transmission problem. This means that it will retransmit the packet at a time dictated by the TCP protocol – which will depend on the round-trip time it has been able to calculate for the connection.

However, in the case of a RST being sent in response to an unexpected incoming packet a retransmission is not required. Hence, a typical trace for this case is shown in Figure 4.10. The signals have the same identity as in Figure 4.9, but this time there are 200 ms per division. When there is activity in sending the email (when RXD is active) then an RST will be generated and, as can be seen, the collision request signal (`bit07`) is asserted to ensure transmission of this RST packet fails.

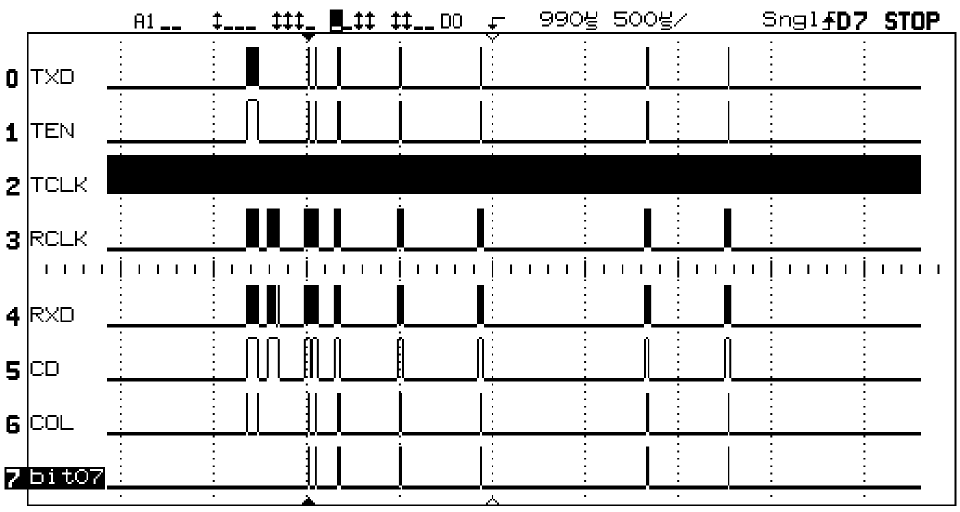


Figure 4.10: RST packets that encounter an intentional collision

This suggests that an improved design would delay the decision on whether or not to collide with a particular packet until its exact nature has been determined.

The idea of this improvement would be to avoid delaying traffic, and thereby risking an effect at the user interface level, unless the packet being sent could be identified to be the RST packet that must be suppressed. Being more precise in identifying the packet would mean that the collision would not occur until many more octets had gone by (the exact number depending on whether it was decided to match the destination port and IP address as well as detecting the RST setting in the TCP control bits field) and this would limit the application of the method to shorter cable segments.

However, an RST is generated for every packet exchanged with the remote server, and there are a great many of these. Hence the normal condition would be that there was almost always an RST packet that had to be blocked, so this extra frill was not considered worth implementing – although, by considerably slowing down the packet sending rate of the spoofing session, it could probably be arranged that collisions were infrequent enough that valid traffic was not significantly impacted.

It should also be noted that the ARM processor within the FPGA device was somewhat underpowered for the tasks it was given – running Windows CE, updating the remote display and handling all incoming Ethernet packets. This led to some overruns (where incoming Ethernet packets were not processed quickly enough and the seven-slot buffer was filled). This in turn led to delays awaiting TCP retransmissions (one of which occurs in the packet traces above). This all meant that it took 7 seconds or more to send an email, which is slow – for example, the Windows XP laptop used in the experiment typically takes less than a second to send an email to a nearby server. So although the experiment showed that the scheme was practicable, it also demonstrated that a more powerful processor would be needed to improve its chances of being undetected if anything more than very small amounts of data were to be transferred.

4.7 How firewalls can make collisions unnecessary

In 2001 I developed an FPGA design to brute-force DES (Data Encryption Standard) cryptographic keys. In a CHES 2002 paper [34] Mike Bond and I described how we had been led to a number of new insights by this practical instantiation of his theoretical attack on the IBM CCA 4758 system (used in retail banking to protect parts of the cash machine infrastructure). It was this expectation of once again learning from the experience of "doing it for real" that led me to build the system described in this chapter to see how it would come out in practice, rather than settling for a theoretical description of how it might be expected to work. I was not disappointed, because it has provided an unexpected result.

Whilst preparing the `ethereal` traces presented above, I found that the predicted RST packets were absent. The system was complex, with custom hardware and several custom software components, so locating the problem took some time. However, it eventually became clear that there was a very simple explanation. The Windows 2000 laptop was running a "personal firewall" program called ZoneAlarm.[3] This program runs on the machine it is protecting and filters incoming IP packets with the aim of making your system "more secure". ZoneAlarm was discarding the packets containing the responses from the email server because they did not correspond to any known outgoing connection. Because the packets were never passed to the TCP/IP stack they were never found to be unexpected and so they were never responded to with an RST packet.

This behaviour is not particular to ZoneAlarm, but also applies to the Microsoft Windows XP firewall, and doubtless many other software firewall programs from other vendors have similar designs.

What software firewalls try to do is to prevent unauthorised access to servers running on your machine. Although a simple system such as Windows 95 had just a few servers running (mainly to handle local area networking) there are now, on later systems, a myriad of these components, operating not only protocols such as HTTP (`tcp/80`) but ISAKMP, a key exchange protocol (`udp/4500`), SSDP (Simple Service Discovery Protocol, used by Windows Messenger) (`udp/1900`), Local Security Authority Service Server (`tcp/445`) etc., etc. – a total of ten servers on my own Windows XP laptop, and I've not installed half the things that I might.

Since it is hard to keep track of all these servers, let alone ensure they are all correctly configured, software firewalls have been seen as an easy way to ensure that only a small number of hosts, perhaps just those on the local subnet, can access them. It

[3] `http://www.zonelabs.com/`

is now standard security advice to enable these firewalls – though of course their protection is only against external threats. Malware that gets onto the machine will disable or reconfigure the firewall component to ensure that it does not interfere with any of its (unauthorised) traffic.

Quite obviously, in order to prevent servers being exploited by specially crafted packets, it makes a lot of sense to refuse all unexpected UDP packets, but it is only necessary to take special steps to refuse the initial packet (the SYN packet) of a TCP connection. If a TCP connection is not created, further packets will be automatically rejected at the kernel level with an RST and the (possibly fragile) server code will be unaware that the traffic arrived.

However, a myth has arisen[4] that it is safer to run in "stealth mode", where you discard all incoming packets, than to respond to unwanted TCP traffic with a connection refused (RST) or even an ICMP packet such as "destination unreachable".

The notion behind this myth appears to be that attackers start by "port scanning" to determine whether a machine exists before deciding to attack it, perhaps even checking one machine at a time so that a failure to respond will delay the scanning. Although clearly there is a certain amount of port scanning going on, a lot of current attacks – especially automated malware attacks – merely send their exploit traffic to every address without checking whether a machine exists first, and of course the delay waiting for a response is irrelevant if the attack program is designed to handle multiple probes simultaneously.

So, if a machine is running a software firewall configured in stealth mode, you will be able to borrow its identity and communicate with servers with complete impunity. The software firewall will discard all the incoming traffic and will never issue an RST. Since there is no need for special hardware to collide with RSTs, as outlined in this chapter, this is an extremely practical attack to mount, requiring relatively low levels of skill and no special equipment such as FPGA development boards.

Of course software firewalls will usually offer the option of logging unusual events, and discarding packets from the server is very likely to be seen as unusual. For example, the Windows XP firewall will record the sending of an email by another machine that it using the same IP address and MAC address as follows:

```
14:41:11 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 48  SA 3388313642 29096970 17520 - - - RECEIVE
14:41:11 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 98  AP 3388313643 29096970 17520 - - - RECEIVE
14:41:11 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 115 AP 3388313701 29096988 17502 - - - RECEIVE
14:41:12 DROP TCP 192.168.1.2 128.232.110.14 25
```

[4]Steve Gibson, `http://www.grc.com/faq-shieldsup.htm`, claims to have invented the term "stealth". His "ShieldsUP!" web pages for testing whether machines respond to incoming traffic are widely known and their promotion of "stealth" as the most desirable state for TCP ports has clearly had a significant influence on consumer expectations for software firewalls.

```
         1026 60  AP 3388313776 29097020 17470 - - - RECEIVE
14:41:12 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 60  AP 3388313796 29097047 17443 - - - RECEIVE
14:41:12 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 69  AP 3388313816 29097053 17437 - - - RECEIVE
14:41:12 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 40  A  3388313845 29097086 17404 - - - RECEIVE
14:41:12 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 40  A  3388313845 29097268 17222 - - - RECEIVE
14:41:13 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 40  A  3388313845 29097287 17203 - - - RECEIVE
14:41:16 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 120 AP 3388313845 29097287 17203 - - - RECEIVE
14:41:16 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 127 AP 3388313925 29097293 17197 - - - RECEIVE
14:41:16 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 40  FA 3388314012 29097293 17197 - - - RECEIVE
14:41:16 DROP TCP 192.168.1.2 128.232.110.14 25
         1026 40  A  3388314013 29097294 17197 - - - RECEIVE
```

However, in Windows XP this logging is disabled by default, so there will be no record unless the machine's user has enabled it within the dialog reached from the "Advanced" page of the firewall property sheet (which is quite unlikely). Even if a log has been recorded, its default length is only 4 MBytes. This is sufficient for about a week of past traffic on a machine such as mine – that is seldom connected to the Internet without being protected, to some extent at least, by hardware firewalls. The log would fill the allocated space and discard the older entries far faster on machines which were connected to the Internet in a more exposed manner.

The best method of impersonating a machine running a stealth mode software firewall would be to rewrite the Ethernet device driver on the attacker's machine so that it alters outgoing packets to contain the impersonated MAC address and IP address. The packets returned from the server must be captured in a promiscuous mode, so this attack only works when the return traffic is visible, i.e. it will still not work on a switched network. There is no need to do anything special for ARP traffic because the machine being impersonated is able to transmit at will, so it can be allowed to respond normally.

An alternative approach would be to use a device driver that already supports multiple MAC addresses, but it would still need some alteration to prevent ARP traffic (especially gratuitous ARPs) from causing problems.

Relaxing the "threat model" underlying this chapter, that there is monitoring to ensure that IP addresses and MAC addresses always match correctly, means that the attack can be performed at a higher level in the stack, perhaps even at user level. If all that is needed is to send packets with a different source IP address and then listen promiscuously for responses, then kernel privileges are unnecessary. However, it is considerably simpler to add an extra component to the stack to alter packet addressing, rather than re-implementing the TCP protocol in the spoofing program.

4.8 Conclusions

My initial intention in writing this chapter was to show that the binding between IP address and a machine or a user was extremely weak. In fact it is so weak that the only manner in which it is enforced is for ARP traffic to be continually validated, and for the current owner of an address to object with a RST packet if they detect their identity being used by another device.

I then demonstrated that this objection to "identity-theft" was half-hearted at best, and could be blotted out by an 802.3 layer protocol violation – explicitly colliding with transmissions so that the sender will eventually give up and discard the packet.

Experiments showed that the Ethernet hardware layer gave up very quickly, within a second or so, and would then move on to sending real traffic whose failure to get through would matter. This would severely restrict the volume of wickedness that can be perpetrated in someone else's name. However, higher levels of the protocol stack have their own timeouts, so that in practice, without the complexity of colliding only with RST packets, an alternate identity can be assumed for ten to twenty second periods with little likelihood of detection.

The attack only works when collisions are possible and traffic to other machines can be sniffed. This means that on a switched Ethernet,[5] which is fast becoming the standard method of deployment, the attack cannot be mounted. However, 802.11 wireless systems, which are increasingly common, are also vulnerable, despite some minor differences in the way that they operate.

Implementing the attack "for real", rather than relying on a theoretical description of it, meant that I came across, almost by chance, a far simpler method of achieving the same ends without any of the timing limitations. It is now becoming common for personal computers, Windows machines in particular, to be running software firewalls in "stealth mode". This means that they do not send RST packets to shut down connections that have been made by someone impersonating them. The chapter title is "Hiding on an Ethernet" – but there's no need to hide if the person you're impersonating skulks in the undergrowth and refuses to come out and denounce you!

On 802.11 wireless networks the impact of these new firewall attacks may be mainly economic. If connectivity must be paid for by the hour, in a coffee-shop or an airport, then it will be possible to get a "free ride" by impersonating a Windows XP machine whose software firewall is almost bound to be enabled. The wireless system operator can take some steps to detect the cloning, for example by learning from the cable companies and detecting the same MAC address simultaneously communicating with widely separated access points; but wireless propagates so unpredictably that this

[5]Switched networks cannot, in theory, be sniffed. However, it may be possible in practice using techniques such as "MAC flooding", "MAC duplication" or just exploiting insecure management interfaces. Hewes [71] provides an overview of the methods, albeit without stressing that modern switches can be (and sometimes are) configured to defeat these attacks.

may be ineffective. They may also be able to detect anomalous use of port numbers – because an operating system usually allocates client ports sequentially. They might even examine the IP packet ID field values, since the problem of detecting clones is similar to that of estimating how many hosts lie behind a NAT device [14].

Clearly the wireless system operators would like to see software firewall behaviour modified, but they have very limited leverage. The user running the firewall may get slightly less bandwidth and may be scared of being impersonated by someone wicked – but neither threat is likely to persuade them to switch off a firewall in what will seem to be an overtly dangerous environment.[6] The software vendor has no direct relationship with the wireless operator and is unlikely to wish to make their system appear "less secure" in the marketplace, merely to prop up another company's business model.

These attacks, in whatever form, are especially pernicious when they are used to do bad things with someone else's identity – rather than just to steal wireless service whilst awaiting a plane. This is because they can be implemented when the machine being impersonated is switched on and its user is actively using it. This makes it unlikely that the user will have an alibi that would cast any doubt on the apparently impeccable traceability process that has led to their machine.

Fortunately, there is some opportunity for sysadmins (or police officers mounting surveillance before swooping on their suspect) to detect the type of attacks I have described. The hardware system produces an unusual number of collisions on the network; software firewall logging can be enabled and distinctive patterns picked out; or the suspect machine can be monitored, or forensically examined after its seizure – it would be hard to blame the user for an abusive IRC session if there was no sign of an IRC client; or a surreptitiously introduced keystroke monitor didn't record any of the phrases used.

In practice of course, prosecutions are usually mounted on the basis of what is discovered on a particular machine or at the user's home, rather than relying on server logs and traceability to form the basis of a conviction. So the type of impersonation I have discussed is probably of most interest to a defence barrister who is trying to cast doubt into the minds of a jury. Nevertheless, police officers and sysadmins who are working their way through voluminous logs and planning how to arrest the user of workstation #17, or implement a disciplinary process for inappropriate use of the web on company time, should bear in mind that it may be someone else that they seeking – who is hiding on the Ethernet and framing an innocent person.

[6] In practice, the wireless network – although seemingly a very public environment – may be better protected from threats from the wider Internet than the user's own system at home. The real risks, such as sending passwords in the clear, have little to do with software firewall packet admission policies.

Chapter 5

Extrusion Detection

And the pattern still remains, on the wall where darkness fell,
And it's fitting that it should, for in darkness I must dwell.
Like the color of my skin, or the day that I grow old,
My life is made of patterns, that can scarcely be controlled.
— Paul Simon, 1965

This chapter discusses an innovative system I created for Demon Internet, a large UK-based ISP. It was a consulting engagement approved by my supervisor. The system processes email server logs and promptly detects unusual patterns of activity. The reports generated by the log processing system mean that Demon's staff can proactively deal with customers who are inadvertently sending spam.

5.1 Traceability and spam

Traceability permits the recipients of unsolicited bulk email ("spam") to determine where it has arrived from. The email will contain `Received` header fields that accurately record the source IP address – and the process outlined in Chapter 2 will establish which ISP is responsible for the network from which the sender is operating. The ISP itself will then be able to establish which user account was used to send the spam email, and can then act to prevent further abuse.

These traceability mechanisms work extremely well, precisely because ISPs find it essential for their reputation and commercial success to be able to prevent excessive amounts of spam coming from their networks. When the mechanisms break, or the ISP just gets behind in handling incoming abuse reports, then the ISP can find all of its outgoing email being blacklisted. The inconvenience this causes can significantly affect the ISP's ability to retain, and profit from, their legitimate customers.

These days the source of spam email is seldom the spammer's own machines, but an innocent end-user, whose machine is insecure, through which the email has been relayed. Even though it is not the true origin, the community expects this intermediary machine to be secured, and the pressure on the ISP to ensure a prompt fix of the problem makes few allowances for lack of any deliberate intention by the ISP or the end-user to be a source of spam.

Although it might be possible to determine where the email was arriving from before it was relayed, there is seldom any interest in doing this. The community does not expect the ISP to attempt to locate the real source, and that is mainly because it is not felt to be a tractable problem.

If further tracing was to be attempted then the relaying machine would need to be logging incoming traffic – which is unlikely to be the case. Even if logs can be made available, the trail may be further obscured by the use of additional "stepping stones", as discussed in Section 2.5. Also, it is often the case that there is no ongoing traffic flow to follow, but the compromised machine is fetching lists of destinations and email bodies from intermediate web sites, perhaps set up on yet more compromised machines. The existence of these websites can be deduced by the impact of their failures, which leads to the occasional arrival of spam that is entirely blank, or that contains "`404`" error reports for missing content!

Yet more practical complexity arises because the various components of a spam sending system may well be in multiple jurisdictions, ISPs cannot compel anyone to assist them and Law Enforcement is seldom interested in spam sufficiently to use their powers of compulsion locally, let alone invoking Mutual Legal Assistance from other countries. Thus, almost no-one traces the sources of spam email, and senders are generally tracked down – to the limited extent that they are – using out-of-band methods, such as tracking the supply chain for the products they promote.

The success of traceability in identifying the immediate source of spam, and the resultant pressures on ISPs to deal with incoming reports, means that the "abuse team" that addresses the problem is a major expense for an ISP. The system described in this chapter is used to help the abuse team by processing Demon Internet's email server logs to identify any customers who are sources of spam. This permits action to be promptly taken, often before any reports arrive from the destinations of the email, and usually before any blacklisting of the mail servers occurs.

Of course, once the report is created, traceability is still being used to identify which of Demon's customers has sent the email to the server, but the email server may only be accessed by Demon customers, and not by third parties, and so the whole of the identification of the sender – down to the account level – is self-contained and relies only upon Demon's own records.

Traceability works very well within this scheme, without any of the problems outlined earlier in this thesis, because there is no need to identify some responsible person (even supposing the insecurity of the machine could be said to be one person's responsibility). It is only necessary for Demon to be able to prevent the account from sending email, or disable it altogether, and this prevents further abuse and meets Demon's needs and that of the wider community.

Interestingly, the log processing system provides a clear "business case" for creating logs of email activity (as often wished for by Government and Law Enforcement). However, since these logs are processed on a daily basis, it also makes it very difficult to justify keeping them for much longer than a day (which is a far shorter time than Law Enforcement would like).

5.2 The general approach

Many Internet Service Providers (ISPs) provide email "smarthost" servers for their customers. The customer uses a very simple email client to transfer outgoing email to the smarthost, which will then deal with all the complexity of arranging for delivery to remote sites. Some ISPs go further and redirect all outgoing `tcp/25` (the well-known port for the SMTP [86] email protocol) traffic to the smarthost, so as to make its use compulsory.

In this chapter I show how automated processing of the smarthost's email server logs can be extremely effective in enabling an ISP to detect customer machines that are being used, usually without the customer's knowledge, to send out spam. Since this log processing is closely related to the network monitoring systems that are used for intrusion detection, I have dubbed this monitoring of outgoing events "extrusion detection".

There is no generally agreed definition of "spam"; but the UK ISP industry describes it as "unsolicited bulk email". Not surprisingly, there is also no specific measurable attribute that is capable of unambiguously characterising outgoing spam – which would permit it to be immediately blocked during the SMTP transfer.

Systems for processing incoming email, such as filters, could be used to examine the content of outgoing email to form a view as to whether it was likely to be spam, but this is expensive and prone to error. It is especially difficult to assess the status of *solicited* bulk email such as companies send to their customers or individuals send to their friends with invitations to weddings or birthdays. Accessing the content of email also gives rise to complex regulatory and contractual issues. However, it turns out to be practical, cheap and less intrusive to process the summary "traffic data" in the email server logs and form an accurate view, by the application of suitable heuristics, as to whether there is evidence of the sending of spam.

One might expect to detect bulk unsolicited email solely by its volume, but the most effective measure is not its "bulk" nature – which it shares with legitimate traffic such as opt-in mailing-lists – but in monitoring failures to deliver. Destinations of "unsolicited" material are often invented, out-of-date, or the destination simply refuses to accept the message. Simple measurements of failure rates can catch other types of legitimate activity, so further processing attempts to distinguish them from spamming. The most effective heuristics have been in detecting the chameleon nature of what is being sent. The senders of spam wish it to be hard to categorise at the remote site and the variability, especially of sender identity, that they use for this purpose provides a very conspicuous indication of the presence of outgoing spam at the sending site.

Clearly, when heuristics are used, there are two possible failure modes. Some abuse may fail to be detected (a "false negative"), and some entirely proper activity may be characterised as spam (a "false positive"). Experience has shown that the two main areas of difficulty in reducing both types of failure have been the operation of legitimate, but poorly maintained, mailing-lists and the inability of some email client programmers to create standards compliant software.

5.3 Historical background

The sending of bulk unsolicited email ("spam") has been a significant problem for Internet users for several years. The senders ("spammers") originally relayed their material through remote servers ("open relays") that would accept email from anyone and then deliver it to its destination. These open relays would accept long lists of destinations for each individual email so that a spammer using a low bandwidth "throw-away" dial-up account could have their message "amplified" by the high bandwidth relay and thereby delivered in bulk.

By 1996 or so, most major ISPs had reconfigured their email servers to prevent relaying except by their own customers. The spammers then switched their traffic to email servers at end-user sites because many of these were still configured to allow relaying, albeit at lower bandwidth levels. The battle against spammers in the later 1990s concentrated on ensuring that end-user email servers were secured against unauthorised relaying and in creating support documentation in local languages so that servers outside the English-speaking world could be correctly configured.

As servers became more secure – the major impact having been made by changing system defaults to be secure "out of the box" – the spammers changed target again and started to exploit end-user machines directly, even when they were not running an email server. I use the general term "open server" for unauthorised use of end-user machines, with the techniques employed ranging from the exploitation of misconfigured SOCKS proxies through to the deliberate installation of "trojan" software, the latter sometimes being distributed by means of mass-mailing viruses. Most recently, since late 2003, the spammers have returned to those customers who are running email servers and have utilised default accounts and brute-forced weak passwords, so that they can remotely authorise themselves to relay.

Although the main method of sending has changed several times, the reaction at the receiving end has remained constant throughout: hostility and a desire to receive no more junk. Since it is reasonably straightforward to inspect the header fields and determine where email arrived from, the receivers started to complain. Initially they would write to `postmaster@` addresses to reach the technical staff operating the open servers. However, the end users operating these systems did not always realise the significance of the complaints or even read `postmaster@` email at all. Complaints therefore started to go to the ISP where they began to be dealt with by "abuse teams" operating behind an `abuse@` address, first formally documented in RFC2142 [43].

Meanwhile, the desire to receive no more junk has led to a growth in "blacklists" recording the IP addresses from which spam is sent, so that others can choose not to receive further email from such locations, on the basis that it too is likely to be spam. Besides their obvious rôle in documenting the current set of spam sources, these blacklists have a significant secondary effect in that they put pressure on the sources of spam to fix insecurities, because otherwise no-one will accept their genuine email. This pressure becomes intense when an end-user is compromised, spam is sent via the ISP smarthost, and the smarthost itself is blacklisted. Until the listing is removed no customer of that ISP can send email to sites that are using the blacklist. Hence, ISP abuse teams spend a considerable proportion of their time identifying the source of any spam that went via their smarthost, disconnecting customers and educating them on how to correctly configure their software and then, finally, negotiating with blacklist owners to be de-listed. Any scheme that reduces the time-to-detect and the ultimate time-to-fix for compromised customer systems is of direct benefit to an ISP.

5.4 Related work

The above description of abuse team activity is one of reacting to incoming reports. Although some proactive work has been done in scanning customer machines for "open servers", the number of possible vulnerabilities makes this increasingly ineffective. In particular, it is generally considered unacceptable behaviour for ISPs to attempt to brute-force customer email server passwords to check for weakness.

The reactive nature of most anti-spam activity can also be seen in the current emphasis on automatic detection by processing incoming email contents, for which there is a plethora of commercial tools, and automatic reporting of the source, perhaps using the same tools or maybe via a stand-alone website such as SpamCop [129]. Although passing outgoing email through a spam filtering system is a possible option for ISPs, the cost of the tools and the need for significant extra computing power make filtering systems expensive to operate, and legal issues mean that customer contracts may need to be re-negotiated. Hence, filtering is not currently widely employed on outgoing email.

Email servers generally produce summary statistics (top 50 users, top 50 destinations, etc.) but these statistics were mainly intended for use in load balancing and no attempt is made to distinguish good traffic from bad. However, these statistics are widely used for abuse detection because the same customers tend to regularly appear on them and therefore "new faces" are likely to be being newly exploited by spammers and will be worth investigating further.

The notion of detecting improper activity in an outgoing direction is clearly related to the analysis of incoming traffic or "intrusion detection", which was introduced by Anderson [6] and Denning [45] and has continued to develop into an active field of research. However, there has been limited interest in investigating outgoing traffic and little work on email handling *per se*.

Bhattacharyya *et al.* [19] report a Malicious Email Tracking (MET) system which was later generalised by Stolfo *et al.* [130] into an Email Mining Toolkit (EMT). Their approach was to gather detailed statistics on email flows to and from individual accounts and aggregate statistical information on groups of accounts. Abnormal behaviour is then highlighted, whether caused by spam, malicious attachments or virus infections. Since they consider the content of email they drew attention to the privacy issues raised by implementing their system at an ISP. However they do not address the statistical difficulties that arise when several disparate individuals (perhaps in a family or a small business) appear as a single customer from the ISP's perspective.

The EMT system also, by implication, considers legitimate mass-mailing to be outside their system whereas the ISP will see this as occasional large flows from an otherwise low-volume user. This problem of "false positives" is less of an issue when only the top 50 customers are being considered. But when data for the whole customer base is processed then false positives can waste considerable time and, perhaps more importantly in practice, by swamping the abuse team with dross, lead to a loss of faith in the detection tools, so that they are ignored – even when some of their reports are accurate.

5.5 Processing email logs

I have developed a system to process the email logs generated by an ISP smarthost, so that I can detect any customers that are a source of outgoing spam. The system currently works with Exim logs [69], but very similar information can be found in the logs generated by many other email servers. Since the system condenses log information down into a fairly generic working format, it would be simple to use the same back-end programs, algorithms and heuristics, but with a different initial processing stage, to cope with an alternate log format as input.

5.5.1 Email log contents

Exim records the arrival of email on the server, noting SMTP protocol level details such as the `HELO` (or `EHLO`) command that is supposed to identify the sending host and the `MAIL FROM` details that are supposed to identify the email's sender. Exim also records the actual IP address of the machine that sent the email and this can be used to determine which customer account was used. The logs do *not* contain any of the content of the email, or potentially helpful spam-detection details such as whether the email contained "attachments" or what the `Subject` header field may have been.

Incoming email is queued for delivery to its destinations (it may be multiply addressed). Each delivery attempt will result in a further logging line indicating whether delivery was successful. After a number of delivery attempts to a particular destination have failed, it is standard practice to pass the email to a secondary "fall-back" system which will continue trying. This prevents the build-up of long queues on the main email server and, in the current context, conveniently ensures that a "complete" record of deliverability is available within a few hours.

The logs are collated every 24 hours and processed through a fairly straightforward Perl program [148]. This creates a single record per email which records date and time, size, who the email was allegedly from, who it was actually from, who it was to be delivered to, and whether a successful delivery was promptly made.

5.5.2 Using delivery failures to detect bulk email

An obvious characteristic of spam is that it is sent to a large number of addresses. Many of these addresses come from ancient lists and no longer work. Also, many spammers guess addresses, either working methodically through possibilities in a "dictionary attack", or employing a "Rumplestiltskin attack" – using local parts (left of the `@`) that belong to a user at one domain to see if they will reach a real user at another domain. Thus a key characteristic of spam is that much of it fails to be delivered. For example, this is part of a typical spam run where all of the deliveries failed (needless to say, all the `hinet.net` information was entirely forged):

```
2004-03-01 05:07:25 2jj88@ms32.hinet.net    -> jj88@yahoo.com.tw
2004-03-01 05:07:25 9rwgb@ms29.hinet.net    -> rwgb@sinamail.com
2004-03-01 05:07:26 8sjk642@ms38.hinet.net  -> sjk642@hotmail.com
2004-03-01 05:07:27 6seasons@ms26.hinet.net -> seasons@url.com.tw
2004-03-01 05:07:28 5story@ms35.hinet.net   -> story@ms51.url.com.tw
2004-03-01 05:07:28 7pq4492@ms47.hinet.net  -> pq4492@gigigaga.com
```

Therefore one might expect that processing the logs to detect when lots of outgoing email fails to be delivered would provide an effective method of spotting outgoing spam – and indeed it does. Unfortunately, when an ISP has a large number of customers, then besides detecting three or four new spam sources per day, it also throws up many hundreds of false positives. So further refinement is needed.

5.6 Avoiding "false positives"

Many systems generate delivery failure reports for the spam that they receive; they accept it and later create a "bounce message". The rejection may be because they are running more sophisticated spam detectors on second-line machines, or because the destination domain is valid but their system later determines that the particular local part is invalid. These bounce messages should, according to the email standards [86], be sent with a "null" return path, viz: a `MAIL FROM: <>` command should be used in the SMTP protocol. Unfortunately, many end-users are running systems (perhaps home-built) that do not conform to the standards and generate bounces with a non-null return path. For example, these (anonymised) messages are clearly rejections for incoming spam, but the return path is not null:

```
2004-03-01 00:54:33 user@example.com -> NfkGIS@foobarr.com
2004-03-01 01:11:57 user@example.com -> ct4j2L0n7ZJ@samevalues.com
2004-03-01 01:28:36 user@example.com -> ViewCatcher@spicy.ws
2004-03-01 01:48:39 user@example.com -> getyourcard@energizing.ws
2004-03-01 03:47:55 user@example.com -> web.off@ms1.ofmx3.com
2004-03-01 04:30:30 user@example.com -> P39S7RJXldA@eoffersdirect.com
```

Since the source address for incoming spam has often been forged, these rejections will fail to be delivered, generating a rejection report that is returned to the customer, where they probably think the spam load is twice what it actually is! Similar problems arise if `vacation` (or similar) messages are generated for non-existent addresses. The overall effect is that the customer is sending a lot of outgoing email to a large number of addresses that fail – which (as just discussed) is a good heuristic for spotting spam – and hence there is a "false positive".

Some systems set the bounce return path the same as the forward path (perhaps hoping to increase the load at the other end – more likely through ignorance), but the majority use a constant setting for their `MAIL FROM` commands, sometimes a user name (especially for a `vacation` program) but commonly an identity like `mailer-daemon@` followed by their domain. However, spammers believe that their email would be easy to filter out if they used fixed values for `MAIL FROM`, so they use constantly changing values, often random addresses at large, well-known, ISPs such as `aol.com` or `msn.com`. This means that the heuristics can assess failures per sending address, and thereby distinguish spammers from users with rejection systems that are not standards-compliant.

The next problem is that many end-user sites are not the final destination for incoming email. It is common to see email being forwarded (redirected) to members of staff working at remote sites, or to family members who are away from home, perhaps at college or university. These deliveries may then fail, perhaps because the remote site has a spam detector of its own, or perhaps just because the destination mailbox is full. It is conventional to preserve the original `MAIL FROM` when forwarding in this way, and this means that the sending site is apparently (indeed actually) a source of spam. For example, some of these emails will be genuine and some will be forwarded spam:

```
2004-03-01 02:45:58 menhongbo@sina.com -> remote.user@example.com
2004-03-01 08:08:18 Dporosity@AOL.COM -> remote.user@example.com
2004-03-01 15:02:44 BA@BritishAirways.com -> remote.user@example.com
2004-03-01 16:18:28 staff@c2m22.com -> remote.user@example.com
2004-03-01 15:54:11 herdoxxvhghhv@yyhmail.com -> remote.user@example.com
2004-03-01 21:06:54 nshffmg@yahoo.com -> remote.user@example.com
2004-03-01 23:52:17 postmaster@thx-trade.com -> remote.user@example.com
```

However, spammers tend to be greedy and send large amounts of email, aimed at multiple destinations, through compromised machines. Therefore, it is possible to distinguish these cases: a small number of destinations will be redirected traffic; a large number of destinations is most likely spam and should be investigated.

5.6.1 Actual heuristics for detecting spam

The intermediate format records derived from the email logs, as described in Section 5.5.1, are processed on a daily basis by a second large and fairly complex Perl program that generates reports about apparent spamming activity. All settings are configurable, the current algorithms and trigger values applied to each individual customer in turn are:

```
Consider emails from each particular source address:
  Destination address is identical to source   => assume rejection message
  if failure to deliver >5, OK delivery <= 100 => assume rejection daemon
  if failure to deliver >5, OK delivery  > 100 => assume mailing-list
    BUT if >1 mailing-list, then do not treat specially
    BUT if >2 rejection daemons, then do not treat specially

Consider destination addresses:
  if >4 to same address => assume to be forwarded email

Now consider all email that is not a rejection, mailing-list or forwarded:
  Report if total score is >100 when summing the values:
    +10 if receiver rejects with distinctive 'this is spam' message
    +10 if receiver delays acceptance with a 4xx return code
     +1 if receiver responds 'please try later' after RCPT TO
     +3 if >3 destinations for the email, and all fail
  Report if >40 emails where
    1, 2 or 3 destinations and all fail
    or if >3 destinations and >25% of them cannot be delivered to
```

5.7 Heuristics that detect mass-mailing malware

Mass-mailing malware (sometimes called "worms" or "viruses") is becoming a significant problem. A rogue program emails copies of itself to remote sites and tries to infect that site, either by exploiting a security hole in the email software or by "social engineering" to persuade the user that they should execute the program. In the second half of 2003 a particularly successful program was `Swen` which masquerades as a Microsoft-issued security patch.[1] The malware will locate destination addresses from user address books, the contents of webpages in the browser cache, Usenet news server meta-data, etc. Since these addresses are often out-of-date, invalid or malformed, the outgoing email will have many delivery failures and it is therefore spotted by the spam detection heuristics.

Malware, naturally, wishes to avoid being detected or trivially filtered when it is received. The practice has therefore taken hold, doubtless copied by one author from another, of obfuscating the identity of the sending system in the argument field of the `HELO` command that is used at the start of the SMTP protocol transfer (and which is then recorded into the email `Received` header field). This is a wrinkle that spammers have seldom bothered with, possibly because they have always thought in terms of many machines sending to each destination, rather than one specific machine attempting to infect others.

It can be difficult to determine the correct `HELO` argument field value – indeed a great deal of software gets it entirely wrong. I found that 62% of Demon's customers used a single word value when connecting to the smarthost, whereas the SMTP standard prescribes a fully qualified domain literal or an IP address [86]. Malware often avoids all of the difficulty in setting the correct `HELO` argument[2] by using random text and, crucially for detection purposes, changing that text from one email to the next. For example, with the names anonymised, but the original `HELO` arguments:

```
HOST = example.com, HELO = Hhspn
2004-03-01 22:25:56 postmaster@example.com -> user1@findanamateur.com
HOST = example.com, HELO = Zfi
2004-03-01 22:26:42 postmaster@example.com -> user2@uk-personals.net
HOST = example.com, HELO = Xgqdgueg
2004-03-01 22:27:11 postmaster@example.com -> user3@photosound.co.uk
HOST = example.com, HELO = Osrummj
2004-03-01 22:27:50 postmaster@example.com -> user4@photosound.co.uk
```

Hence the most effective heuristic is to spot any customer that has employed multiple `HELO` argument field values, never re-using the same text. This does yield "false

[1]So successful is `W32/Swen.a@MM` that copies of the initial variant of this malware were still circulating in July 2005.

[2]Consider a host using NAT to access the Internet. The correct argument for the `HELO` command will be the Internet-facing IP address or corresponding hostname, but this is unavailable to malware (which cannot ask a sysadmin to configure it) unless it makes an outgoing connection to a machine that can report back the IP address being used. However, when multiple IP addresses are available to the NAT system, it will be impossible to predict which would be used by the next connection!

positives" from companies where multiple employees on different machines, but using NAT so that they share a single IP address, each send a single email per day. But, as the results in Section 5.8 illustrate, this is not a major difficulty.

In early 2004, after the log processing program had been deployed for some time, the `MyDoom` program spread very rapidly across the Internet. This malware forges sender domains, but uses a correctly matching `HELO` argument. My system was very effective at detecting `MyDoom` infected customers (it found just over 1 000 in total, 293 on the first day alone). However, it incorrectly identified them as open servers; partly because the heuristics in use at that time failed to give an appropriate score for the presence of "dots" in the `HELO` values, and partly because the high rate of redistribution meant that many systems forged the same domain more than once.

The heuristics now being used for malware detection (again, the values are all tuneable) are:

```
If more than 10 HELO arguments are only used once (in 24 hour period)
  then report unless less than the number of HELOs used >1 times
Count how many different HELOs matched the message sender
  report if more than 3 different ones were seen

The report is of a virus UNLESS the majority of HELOs contain dots AND the
average message length < 18K bytes, which is reported as an open server
```

5.7.1 Spotting email loops

Having successfully detected customers that were sending spam and those with virus/worm infections, I attempted to validate my results by checking that I was managing to detect all of the customer sites that appeared in the Exim statistics reports of the "top 50" users. This showed that I was failing to detect a completely different type of problem, which turned out to be "email loops".

Loops arise for a number of reasons and different heuristics are needed for each. In the simplest form an email system does not realise that it should be accepting email for a particular domain. It therefore forwards incoming messages to the smarthost, and that returns the email to the true destination, which is the same machine that just failed to accept it. After a few trips around this loop the email will have recorded so many `Received` header fields that the smarthost will give up trying to deliver with a "too many hops" report and record this event in the server logs. This is obviously simple to detect and the customer can be contacted to suggest that they correct their problem.

If the email system at the customer end of the loop is the first to deem the hop count to be exceeded, then it will generate the failure. This can still be detected by spotting that the same message identifier is being processed more than once. False positives arise where there is no loop but just a single outgoing message which has already been split into individual emails to each of its multiple destinations – so it is necessary to check if email destinations match as well as their message identifiers.

Some systems manage to avoid increasing the count of `Received` header fields and so the loop never terminates. This usually arises because some sort of poorly configured `vacation` program is in operation. This may not be spotted by repeated message identifiers, because a new one is generated each time around the loop. However, the destination will remain constant and the size of the email will remain fixed or grow by a small constant value.

Finally, there are loops caused by mishandling message delivery failure reports. Standards-compliant email software gives delivery failure reports ("bounce messages") a null return path. This should ensure that if they cannot be delivered, then no further report will be generated. Unfortunately, a poorly-written `vacation` program may respond to the failure report and generate extra traffic. This may not be a problem because the bounces usually come from a "user" such as `mailer-daemon@` that discards all incoming email. However, if two poorly configured systems manage to talk to each other then an email loop, often circulating very fast indeed, will result. Such a loop can be detected by using the heuristics already mentioned, but "speaking to robots", i.e. sending email to one of the dozen or so conventional names like `mailer-daemon@`, is treated as evidence of an impending problem and a request is made that it be dealt with as a customer support issue.

5.7.2 Leveraging other people's detectors

Many ISPs and individual users run complex scanners on their incoming email, trying to detect malware and/or spam. Most systems reject email that is misaddressed, but some will distinguish between accounts that never existed and those that have been deactivated. This third-party information can be used to make more informed decisions about the type of delivery failure being experienced, and hence the nature of what is being sent through the smarthost. Also, although my program does not process the content of the email, it can weigh the evidence of what some remote sites report it to contain. The heuristics in Section 5.6.1 show how this third-party information contributes to the scoring function.

If the remote site detects a virus, then this adds weight to the hypothesis suggested by the `HELO` argument values, and the message may give the specific identity of the virus or worm. If the remote site reports that non-existent addresses are being targeted, then this adds weight to a hypothesis that this is a spammer trying a dictionary attack, rather than a customer operating a mailing-list and failing to remove old addresses. The lazy mailing-list operator is clearly wasting resources and should change their ways, but their activity is essentially legitimate (and long standing), so there is no need for urgent disconnection, as there would be with a high volume spammer.

Recently, some large sites have tried to defend themselves against dictionary attacks by accepting all email. Their view is that when they accept some email and reject the rest they disclose which email addresses are valid. Unfortunately, if an entire spam run is sent to such sites, then all outgoing email is accepted, no errors occur, and the heuristics described here are ineffective. Fortunately, such sites remain the exception at present.

5.8 Results

The system described in this chapter has been deployed at Demon Internet, a large British ISP with several hundred thousand customers, operating a mixture of dial-up access, ADSL connections and leased lines. The system was originally developed in June 2003 and has been running in continuous "live" service – with daily reports fed to the ISP abuse team – since August 2003.

Initially, there were large numbers of customers being detected as sources of spam, but effective action by the ISP abuse team has reduced the problem significantly and, in particular, reduced almost to zero the number of "repeat offenders" where customers continue to run insecure systems.

For this evaluation, I collected detailed statistics for the four-week period starting on the 1st March 2004. There were no public holidays during this period, so the business users were operating five days each week. Many customers send email directly to its destination, but during the measurement period some 84 562 distinct customers used the ISP smarthost, sending a total of 33 393 384 emails to 51 801 043 destinations.[3]

The results of running the extrusion detection program on the logs for this period are summarised in Table 5.1. As can be seen, some particularly good results were obtained in detecting customers infected by mass-mailing email malware (viruses) and those who were causing substantial email loops. In the latter case, only those sending 10 000 emails or more in a day are counted, the 867 customers with smaller loops were disregarded.

Abuse Type	**Correct**	**False +ve**	**False –ve**
Open servers	56	69	10
Virus infection	29	6	4
Email loops	14	3	0

Table 5.1: Effectiveness of extrusion detection: 1 Mar 2004 to 28 Mar 2004 (28 days)

As can also be seen, the open server detection is reasonably effective, with a little under half the reports being correct and the others (just two or three a day) being

[3]These figures ignore usage of the smarthost by internal systems.

trivial to reject by manual inspection. The figures for this category reflect both a wish to avoid false negatives, and also some difficulties in automatically ensuring that unusual, but legitimate, behaviour is not flagged up.

The false negative figures were obtained by comparing the results with a very much more sensitively tuned version of the program – which generated an unacceptably high number of “false positives” and so it was completely unsuitable for production use. A check was also made to ensure that reports from other sources did not indicate that any activity was being missed.

The false negatives for “open servers” were inflated (from 3 to 10) by one particular spammer who, most unusually, relayed email with a fixed sender string to destinations that mainly accepted the email – indistinguishable activity from a legitimate mailing-list. These false negatives only came to light when a summary of the active mailing-list source addresses was examined and it was realised that several different customers were apparently operating the same mailing-list, and it then became clear what was really going on.

The specific reasons for the “false positives” are summarised in Table 5.2. Obviously, altering some heuristic values would have decreased these values, but in each case some genuine spam relaying would have been missed.

Count	**Reason**
36	Customer operating mailing-list(s) with many failing destinations
22	More than 40 genuine delivery failures
4	Customer was forwarding spam to outworkers
4	Varying `HELO` messages were in fact genuine
2	Misdiagnosed virus infection as spam
1	Misdiagnosed email loop as spam

Table 5.2: Reasons for “false positive” open server detection

5.9 Conclusions

I have shown that a relatively simple email server log processing program can be remarkably effective at detecting ISP customers whose systems are being used for sending bulk unsolicited email (spam). My system has been able to detect the patterns of activity the spammers create, and I have been very successful in distinguishing these patterns from those made by legitimate senders. As a bonus, I was also able to detect customers who are infected by virus or worm malware and to spot resource-wasting email loops. The ISP abuse team has been able to act promptly on the system’s reports, often before any complaints have arrived.

I was fortunate that the ISP I was working with was initially using static IP addresses for all customers including dial-up and ADSL users, which simplified the system because there was a one-to-one mapping between IP address and customer account. When dynamic addresses are in use, there are several ways of proceeding. Clearly, the user identities can be established for each record, which makes analysis the same as in the static case, but this is a database-intensive procedure. Alternatively, an analysis of "sessions" is possible, spotting periods of continuous activity that can be ascribed with a high degree of probability to an individual customer.

Demon Internet is now using dynamic IP addresses for a small number of customers' systems and it has turned out to be unnecessary to take any special steps to deal with them. It is quite effective to treat these dynamic addresses as if they were static. When a large amount of email is being sent the customer machine usually stays online, using the same IP address throughout, so the legitimacy of their total activity can be assessed. Conversely, when machines are online for only short periods, although activity from two or more customers may be combined, this seldom triggers the spam-spotting heuristics.

The system is subject to ongoing improvement, with a key aim being to reduce the false positives further, whilst continuing to keep false negatives at a low rate. Demon Internet is also considering real-time analysis of delivery failures and, where suspicious activity is found, slowing down the rate at which the smarthost accepts email. This will ensure that the spam output is reduced without a catastrophic impact on customer relations when "false positives" occur.

Further developments, not reported in this thesis, but that were presented at the recent CEAS 2005 Conference [37], are to process the logs for *incoming* email. It turns out to be possible to detect further problems, usually email virus infections that are being sent direct, bypassing the outgoing smarthost. The small number of items that are sent to other customers show up, again in very distinctive patterns, within the incoming server logs.

However, one should not become too enthusiastic about this new approach to controlling spam, since it is quite possible that the spammers and virus writers will adjust their behaviour and adapt. They could, for example, restrict their destinations to those they knew to be valid. Since the senders of spam often work for multiple firms at the same time, they could send many different advertisements to one destination rather than the same material to many destinations – this would be hard to distinguish from legitimate email forwarding. They could try and duplicate the sender names that were normally in use, or just reduce the variation in the identities being assumed – but the latter might affect detection rates at the remote sites. They could be rather less greedy – since lower volumes of transfer would blend in more with other activity – and they might, overall, get extra traffic through before they are detected. Finally, they could disguise their junk as bounces where there is little discernible pattern in genuine traffic – but of course, such messages would be trivial to filter at the destination.

I am cautiously optimistic, since I believe that it will continue to be hard to disguise one type of activity as another, so “extrusion detection” will continue to be successful for some time to come. However future work in this area cannot be expected to employ quite such simple heuristics.

Chapter 6

Proof-of-Work Doesn't Work

You know I work all day to get you money to buy you things
And it's worth it just to hear you say you're going to give me everything
So why on earth should I moan,
'cause when I get you alone
You know I feel OK

— Lennon/McCartney, 1964

Traceability permits the immediate source of unsolicited bulk email ("spam") to be identified. Innovative systems such as that described in the previous chapter can pick out distinctive sending patterns so as to improve the efficiency of ISPs in identifying users who are operating compromised machines. However, this is still a reactive approach, dealing with the problem once it is occurring and, as already discussed, traceability does not provide an effective way of finding spammers and dealing with them directly.

Since traceability is ineffective in locating and hence deterring spammers, it would be extremely useful to deploy a "survivability" scheme – in the sense used by Lipson [92], as was discussed in Section 2.4.3. This chapter considers an elegant scheme that has been widely promoted and that, if it was to work as claimed, would make the traceability of spammers almost irrelevant. Unfortunately, as we shall now see, in present conditions it *doesn't* work; leaving traceability to achieve what it can.

6.1 Tackling spam through economics

It is often suggested that unsolicited bulk email ("spam") is such a problem on the Internet because the current economic framework for email handling does little to discourage it. If only, it is suggested, the senders of email could be made to pay for their traffic! Spammers would then cease their indiscriminate distribution of messages, and email volumes would reduce as the senders targeted more carefully or just gave up altogether.

Nevertheless, almost no-one (other than those hoping for a handling fee) thinks that using actual money is a good way to achieve this economic utopia, and even the holders of patents for "eMoney" systems have failed to generate any significant enthusiasm for their wares.

However, there is an alternative to real-world money, which was first proposed by Dwork and Naor in 1992 [50]. Their idea was to have the sender of an email perform a complex computation as evidence that they believe that an email is worth receiving. The sender then proves to the recipient that this processing work has been completed and the email will then be accepted. The processing time is "free", so there should be a minimal burden upon legitimate senders, but it is a finite resource, so that the spammers will not have unlimited amounts of processing time at their disposal and so cannot continue to send in bulk.

In this chapter, I briefly review "proof-of-work" systems in Section 6.2 and then in Section 6.3, I present data on current email sending activity. I consider an uncomplicated scheme for reducing spam within which every email carries a proof-of-work result, and develop a realistic quantitative statement of the goals it must achieve.

The crucial question is "how much work must be proved?", so in Section 6.4, I analyse the problem first from an economic perspective, "how can we stop it being cost-effective to send spam?"; and then from a security perspective, "spammers can access insecure end-user machines and will steal processing cycles to solve the puzzles". These two analyses lead to broadly similar conclusions as to the puzzle difficulty required to discourage spam, and show that puzzle solving is not prohibitively expensive for an average email sender.

However senders are not all average, and in Section 6.5 I examine some real-world data to assess the impact of variability in actual usage of email. This data demonstrates that a "universal" proof-of-work system cannot effectively discourage spammers without having an unacceptable impact on a significant proportion of the legitimate senders of email. I conclude in Section 6.6 that schemes incorporating proof-of-work will require significant extra complexity to ensure that the legitimate senders are excused some of the proof-of-work effort. If hybrid schemes are developed, they will inevitably be more complex and more fragile than the universal alternative, which would have been preferable – if only it was actually viable.

6.2 Proof-of-work systems

Proof-of-work puzzles have not been restricted to email, but have also been proposed for metering visits to websites [60], providing incentives in peer-to-peer systems [121], mitigating distributed denial-of-service attacks [101] and rate limiting TCP connections [84]. Jakobsson and Juels' paper [82] has an extensive survey that considerably extend this list of potential usages.

Wherever these systems are using proof-of-work as a way of limiting the abilities of attackers, the type of analysis provided in this chapter will be relevant.

6.2.1 Hashcash

Dwork and Naor's 1992 scheme proposed a central authority that would hold a secret key that permitted authorised bulk senders to shortcut the computational effort, but decentralised systems would be far more practical on today's Internet.

The most widely known proof-of-work system is Back's independently invented "hashcash" [8] which requires the sender to produce a string whose cryptographic hash starts with a certain number of zeroes.[1] An important property of the hashcash puzzle (and all proof-of-work puzzles) is that they are very expensive to solve, but it is comparatively cheap to verify the solution.

In order to tie the hashcash puzzle to a particular email, the crucial parts of the string that must be produced are the email recipient address, a timestamp and a unique value or "nonce", which is repeatedly varied until the required number of zeroes is found in the cryptographic hash value. The preset values have been chosen so that a puzzle solution will only be valid for a particular destination, and also so that the destination can check the timestamp is recent and hence limit the size of the database it maintains to ensure that the solution has not been presented before.

The hashcash scheme is immune to attacks whereby an innocent user is fooled into computing values for the benefit of another, although – since the puzzle solution is not bound to the actual contents of the email – anyone who can intercept incoming email can extract the hashcash strings and deliver their own content in its place.

6.2.2 Avoiding a dependence upon processor speed

It has always been an acknowledged problem with proof-of-work schemes that the amount of processing power available to particular users can vary enormously. Work that might take 10 seconds on a 3 GHz Pentium could take half an hour or more on a 33 MHz Palm Pilot.

To address this problem, Abadi *et al.* [1] proposed puzzles that rely on accessing large amounts of random access memory (RAM). Because memory speeds do not, at present, vary nearly as much as CPU speeds, these have a far more constant

[1] In fact, the original version called for two strings that when hashed would yield the same initial bit-string – the requirement to provide a single string whose hash value starts with a number of zeroes is a recent improvement.

performance – and Dwork*et al.* [49] have developed puzzles of this style with just a factor of four in performance between the slowest and fastest machines.

Finally, it should be noted that whilst most of the proposed proof-of-work schemes do not perform a useful calculation, Jakobsson and Juels [82] suggest the term "Bread Pudding Protocol" for any system within which the results can be re-used for another purpose. Where this is occurring, this might make environmentalists feel slightly happier about the amount of power that is being consumed for no directly valuable purpose.

6.3 Quantitative analysis

My basic assumption in this chapter is that that the goal of introducing a proof-of-work system is to reduce the amount of spam the average person receives to below some fraction, S, of their legitimate email.

Independent "consumer" testing of the best commercial spam-filtering solutions shows that they currently achieve an S of 0.06 [124], but if one is going to re-build the email infrastructure to incorporate proof-of-work it is obviously necessary to construct a system that will do better than this.

Therefore one might set the, arbitrary but clearly desirable, goal of reducing S to 0.01 (1% of email is spam) or, given the significant effort and disruption involved in replacing everyone's email system, 0.001 (just one spam email in a thousand).

6.3.1 Estimating email volumes

First of all,[2] it is necessary to know how much legitimate email each person receives. Radicati estimated [114] that, as of mid-2004, 7.24×10^{10} emails are sent per day and quote a figure of 5.78×10^8 email users on the Internet. These numbers continue to grow and Radicati estimate there will be 7.62×10^8 users by the end of 2008. Clearly these numbers are only estimates, but they are in line with those quoted by various other research organizations.[3]

[2]The original CEAS 2004 paper [88], on which this chapter is based, uses a consistent set of estimated figures from November 2003. The initial revision for this thesis updated these to the latest available set, which were from June 2004. Since these were in much the same ratios to each other (and so the calculated results hardly differed) no further attempt has been made to update the numbers before submission. The important thing, given that there is significant growth going on, is to ensure that a consistent set of values is used, all from pretty much the same month.

[3]Where there *is* considerable variation in figures, it is in relation to the cost of spam, where companies usually restrict themselves to calculating productivity loss (because that is apparently easy). However, some use unrealistic assumptions (because actual measurement is hard) and employ linear models (despite it not being 20 times as hard to delete 20 times as much junk).

At the same time, June 2004, Brightmail were estimating that 65% of all email was spam [23] (other filtering companies give other figures, but in the same general range). This means that the daily total was 4.7×10^{10} spam and 2.5×10^{10} legitimate emails. Dividing by the population figure shows that each email user received, on average, about 45 legitimate and about 80 spam emails per day.

However, for the purposes of this analysis, it is of more interest to consider computers rather than people, since it is computers that actually perform the work that is to be proved. The Internet Domain Survey [80] counts how many systems have DNS entries and hence it can be estimated that approximately 2.6×10^8 hosts existed in June 2004. This means, given the figures above, that each machine is used by 2.2 people (and hence, on average, sends about 100 real emails). This figure of 100 is probably an overestimate, because I have ignored machines that are not directly connected to the Internet. It would be possible to address this lack of precision by considering statistics on sales of PCs, firewalls or even Ethernet cards, but since the calculations will depend on other values which are also very hard to accurately estimate, there is limited value in pinning this particular figure down any further.

6.3.2 The problem of mailing-lists

So far, I have assumed that the sending of legitimate email is equally distributed amongst all the Internet's users. This is the "best-case scenario" where the effort of providing proof-of-work falls equally upon everyone. However there is an obvious exception to this which cannot be ignored. A great deal of email comes from "mailing-lists", viz: one email is sent to a "list exploder", which then distributes the email to many list subscribers.

Since it is clearly impractical to calculate an individual proof-of-work for each recipient of a mailing-list, I assume that the original sender does a proof-of-work for delivery to the exploder and the recipients delegate the checking and accept everything they are sent (or, better, check that the destination embedded into the puzzle solution string matches the details of the system that has forwarded the email to them). Although the need to do this delegation may be obvious for community lists where everyone can contribute their email opinions, where the information flow is entirely one way (for example, a cinema's weekly "What's On" summary), it may be less obvious to recipients that delegation is necessary. In this latter case, distributed trust systems such as IronPort's "Bonded Sender" [9], may have a rôle to play.

There is almost no published data on mailing-list volumes, for any type of list. The "How Much Information" project at Berkeley [97] estimates mailing-list email at 1.0×10^8 items per day, but my experience suggests that this is a significant underestimate. This may be attributed to the Berkeley team considering only a single major mailing-list hosting site.

Therefore, I examined data for incoming email at Demon Internet,[4] and counted (after a spam filtering stage) the volume attributable to senders who sent email to more than ten customers. From the resultant data I estimate that the proportion of non-spam email that is some kind of mailing-list traffic is currently about 40% of the total (i.e. 1.0×10^{10} items per day). Hence, assuming that the rest of the email is evenly distributed, this leaves us with a final average of about 60 legitimate non-list emails being sent by each host per day.

6.4 How much work must be proved?

Collecting together the information from the previous section, it is now possible to set out the objective of a universal proof-of-work scheme a little more precisely.

I wish to set a cost C for each proof-of-work, such that it is possible to send an average of 60 emails per day per host, whilst limiting the total amount of spam to $S \times 2.5 \times 10^{10}$ per day, for a value of S of, say, 0.01 and preferably much less.

What is really important is that I wish to avoid inconveniencing legitimate activity, and therefore there must be a substantial difference between the cost of normal email sending and the cost to spammers of continued activity.

The task of setting C is made more complex because there must be factors built in for the variable speed of hosts in solving puzzles; adjustments must be made for the amount of time that hosts are switched on (not necessarily online) per day; and, since the 60 is only an average, there must be allowance made for variations in the amount of legitimate email being sent.

6.4.1 Analysis by considering the economics of spamming

An obvious method of estimating how big C should be is to express it in money terms and compare it with the profit for each spam email. If C exceeds the profit, then a rational (economically motivated) spammer will cease activity.

If a standard spam-capable PC unit (no monitor being needed) is assumed to cost around \$500 and will last for a thousand days (almost 3 years), then this works out to 50 cents per day if secondary factors such as interest payments are disregarded. There will be another 25 cents per day for electricity (120 watts, 8.5 cents/kWh). xDSL links at upload speeds of 256 kbit/s can be shared between about ten machines, assuming that each machine sends only a few tens of thousands of small (2–5 KByte) emails per day. At current prices, daily connectivity will therefore cost less than 25 cents per machine.

[4]Demon Internet is the large (200 000 customer) ISP in the United Kingdom that is using the "extrusion detection" scheme described in Chapter 5.

Hence, with a total cost of no more than 100 cents per machine per day, a spammer would break even by sending 20 000 emails per machine per day and charging 0.005 cents for each. To prevent as many as 20 000 emails being sent, C must be set so that each calculation takes at least 4.3 seconds. Naturally, using special-purpose hardware (e.g. FPGAs) instead of standard PCs could well reduce the cost per email of creating the proof-of-work, so sending fewer emails would be cost effective and C would need to be set higher.

Although sending out the email can be highly automated, there are many other aspects to running a spamming business and those who choose this method of making a living will wish to be recompensed for their activities. Even if spamming is a sideline for them, we might assume they might be looking for at least \$100/day (c. \$30 000/annum). Hence, if they operated 100 machines (a \$50 000 investment) they'd be looking to clear 100 cents/day per machine over and above the running costs. So they'd be looking to send 40 000 emails per 24 hours per machine at 0.005 cents each, (i.e. 4 million emails per day – a quite plausible number for a small scale operation: in April 2004, Scott Richter is reported to have sent 50–250 million messages a day, charging 0.020 cents for each [20]). To keep the volume below 40 000 emails per day, C need be set to only 2.2 seconds.

Calculating the price per email

It will be noted that I have considered a price per email of 0.005 cents, without any explanation of where this figure came from. This value is crucial in determining the volume at which spam becomes profitable, so it deserves close examination. Spam is generally sent by spammers on behalf of a third party, so I can determine what the market price appears to be, and, since proof-of-work would disturb the market equilibrium, I can also calculate what it could rise to by determining how much those who use spam for advertising can afford to pay.

There is limited information available on spam sending prices, and "just asking" can only be expected to provide data on the "list price" or "introductory offer", neither of which is likely to be entirely useful. However, Goodman and Rounthwaite [62] provide a survey of how much professional spammers are actually charging for sending emails. They give examples that range from 0.030 down to 0.001 cents per email, and one might deduce from their information that rates below 0.005 cents are only marginally profitable at present. According to their survey, spammers used to charge as much as 0.100 cents per email. Note that if the result of deploying proof-of-work systems were to be that the price returned to this amount then, by the economic argument set out above, it would be necessary to restrict spammers to 2 000 emails per machine per day.

The price that spammers can charge advertisers will depend on how cost-effective spam is, that is, how many sales are made as a result of advertising this way. This is notoriously hard to measure, but a good indicator is the "response rate" of how many sales are made which can be directly attributed to a mailshot. Figures for response rates to legitimate "opt-in" email show click-through responses to sales promotions averaging around 0.7% [134]. Actual figures for response rates to spam emails are extremely rare, although there is a lot of "what if" speculation.

In November 2002 the Wall Street Journal [100] gave real-world examples of spam response rates of 0.0130% and 0.0023%. In June 2003 the New York Times [68] ran an article on the marketing of "Iraqi Most Wanted" decks of cards. Response rates here were 0.0127%, and were "four times" the response rates for products such as printer ink. Thus, the current response rate to spam can be tentatively identified at around 0.003%.

At the current market rate of 0.005 cents per email, then at a 0.003% response rate it is cost-effective to advertise goods with a profit margin of just $1.67. At this same response rate, if the rate per email returned to the historical high of 0.100 cents, then the advertisers would need to be selling goods with a profit margin of at least $33.33. This is not implausible: mortgage leads are worth $50, cellphone sales about $85 and there are examples of companies selling fake medicines for $59.95 that cost $2.50 to make [38]. Goods with a lower profit margin would also become viable to advertise if spammers improved their response rates by learning lessons in presentation from legitimate marketeers.

Hence, by noting that a lot of spam advertises high-profit-margin goods and could be made more attractive to consumers, I would suggest that it is entirely realistic to assume that a price of 0.100 cents per email could return. Thus, for proof-of-work to be an effective discouragement, spammers must be restricted to 2 000 emails per day per machine, i.e. one must set C to 43.2 seconds.

Since, as calculated above, the average machine only needs to send about 60 emails per day, there is a fair amount of "headroom" for variations in legitimate activity before the limit of 2 000 is reached. However, it must be assumed that spammers will purchase equipment that solves puzzles quickly, whereas legitimate senders will use what they already own. Hence one should incorporate the factor of 4 in solving speed that is the best that Dwork*et al.* [49] can achieve, and so the headroom is not a factor of 33 but instead a rather less impressive 8.

Unfortunately, this method of estimating C does not give much insight into the value of S (how much spam will continue to arrive). The difficulty is that the effect will be to suppress the least profitable forms of spam, and neither I, nor anyone else, has convincing data as to what proportion of the total that this might be. Also, if spam is more convincingly presented, or the spammer can make a profit not only from sending email but also from owning other parts of the supply chain, then it would be necessary to raise C even more to make spamming uneconomic. However, the headroom of just 8 is so low (it is not the factor of many thousands that one would wish for) that there is limited scope for raising C without starting to affect legitimate email.

6.4.2 Analysis by considering access to insecure machines

A more realistic analysis is to see that spammers will not necessarily be purchasing their own hardware. Because email traffic from their own machines is now widely blocked, they relay a great deal of their spam via poorly administered third-party machines. In the past, insecure email servers were exploited. Today, it is far more likely that an incorrectly configured HTTP or SOCKS proxy, operated by an otherwise unexceptional customer, will be used to access an email server that trusts the insecure machine. On a typical day in June 2004, the SORBS DNS Blocklist [128] listed 1 200 000 open HTTP proxies and 1 400 000 open SOCKS proxies. Many could be used to relay email.

A more recent development has been for spammers to take over (or *0wn* in hacker parlance) the machines. It has become common for viruses to carry a payload that would allow a remote person to control the infected machine. ICSA Labs gives a 2003 figure for virus infections in medium to large US companies of 108 machines per thousand per year [21], but virus protection differs markedly in other sectors. Estimates of global virus infection rates seem to be little more than guesses,[5] but in late January 2004, `MyDoom`[6] was reported by F-Secure to have infected a million machines with NAI putting it at half that figure. At Demon Internet, `MyDoom` infected 1 customer in 85 – logs of their outgoing email tripped the pattern recognition heuristics I described in the previous chapter – and this rate would scale to three million machines across the whole Internet.

At present, infections by mass-mailing viruses such as `MyDoom` are relatively easy for third parties to detect. The systems send out many copies of the virus, which will be reported by recipients or may be detected by ISP systems. A virus-borne attack where the overwhelming majority of the infected machines just kept quiet, and calculated "proof-of-work" results for others, would be effectively invisible to third parties. It would also be invisible to the machine owner if the appropriate steps were taken to keep the proof-of-work processing at a lower operating system priority than other tasks, the approach used for background calculations by systems such as SETI@home [122]. Hence, undetectability will allow a large number of machines to be *0wned* for long periods.

If it is assumed that spammers switched all of their email to these *0wned* machines (with each controlling a suitable proportion of them) then, to maintain June 2004 sending rates, a pool of a million machines[7] would have to send 47 000 emails each

[5]Anti-virus company estimates for infection by the August 2003 `MSBlast` virus were around the half million mark. However, figures from Microsoft's Windows Update service, reported in April 2004, suggest that the true rate of infection was somewhere between 8 and 16 million [89].

[6]This was the initial version, `W32/MyDoom.a@MM`.

[7]Estimates vary wildly as to the number of compromised machines on the Internet, but botnets of a hundred thousand or more machines are now being regularly reported. A million machines can be seen as quite a conservative total for the whole spammer community to be controlling.

per day. This value is plausible, being the same order of magnitude as the current average number of emails sent by exploited customer machines (as detected at Demon Internet) of 21 000 per day.

If, for example, I require S to be 0.01 (1% of email is spam), then I need to reduce the amount of spam sent to 250 emails per *0wned* machine per day, with a value for C of 346 seconds. Of course, if the spammers manage to *0wn* more than a million machines, or I require S to be 0.001, then C must be increased accordingly.

This analysis shows that there is only a "headroom" of about 4 between legitimate activity and that which I wish to prevent, and this headroom is of the same order of magnitude as the factor of 8 that the economic analysis provided. In this second method of analysing the issue it has not been necessary to apply any adjustment for puzzle solving speed, because the spammers will *0wn* many different types of machine. However, if for any reason the spammers could selectively take control of "fast" machines, then the headroom should be reduced still further.

6.5 Variability in sending rates

I examined logging data from Demon Internet for those customers who used the ISP's email server as a "smarthost" for their outgoing email. I had data for about 50 000 customers on this particular weekday, which I believe to be typical. I excluded one customer being actively exploited by a spammer and the four infected by a virus; and also the 145 with configuration problems that caused email to loop continuously. Counts were made of the number of emails sent with particular SMTP `MAIL FROM` settings (which can, to a first approximation, be assumed to map to different individual senders using different individual machines). The cumulative frequency of total emails sent is plotted in Figure 6.1.

As can be seen, although 93.5% of machines sent less than the global average of 60 emails per day, the distribution has a very long tail. If I consider how many machines sent more than 4 or 8 times this amount (the two values of headroom calculated above) then it can be seen that a proof-of-work scheme would prevent legitimate activity by 1.56% and 0.62% of customers respectively. Some of these will be operating mailing-lists and may not need to provide proof-of-work, but other eBusiness systems will surely be significantly affected.

The position is considerably worse if hourly sending rates are considered, as presented in Figure 6.2 which shows how many emails each customer sent during any hour that they were active. It must be assumed that the spammers will be running 24 hours a day, so it is necessary to restrict them, by the calculations above, to 83 or 11 emails per hour. However, many users will only switch on their machines shortly before they actually send their email. As can be seen from the graph, the proof-of-work scheme – for the two headroom assumptions – would now inconvenience 13% or 5% of legitimate customers.

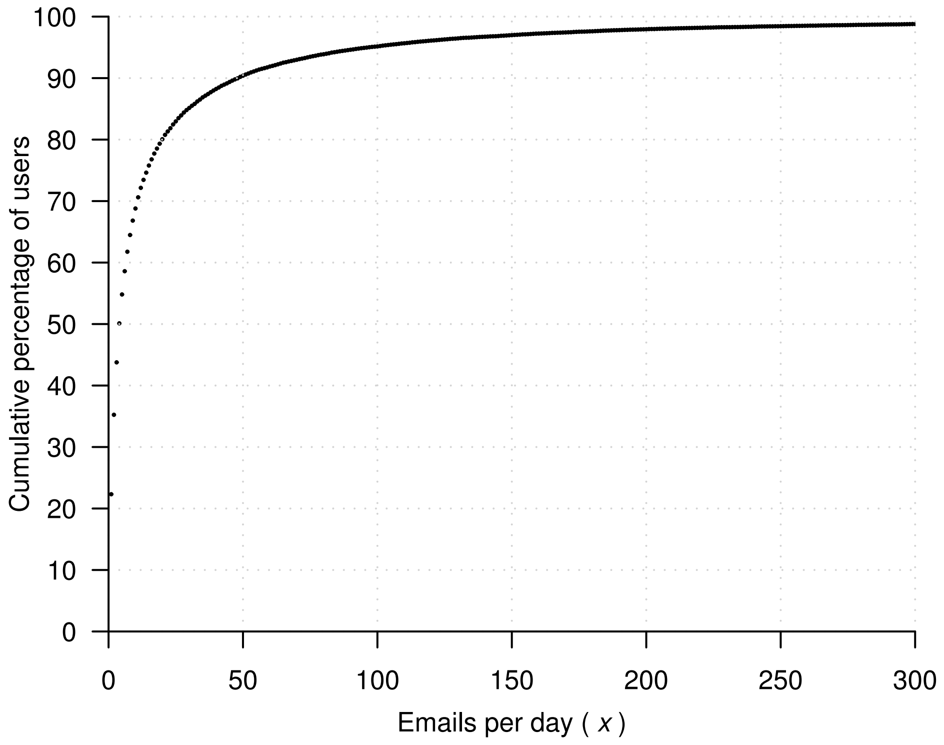


Figure 6.1: Percentage of users sending no more than x emails per day

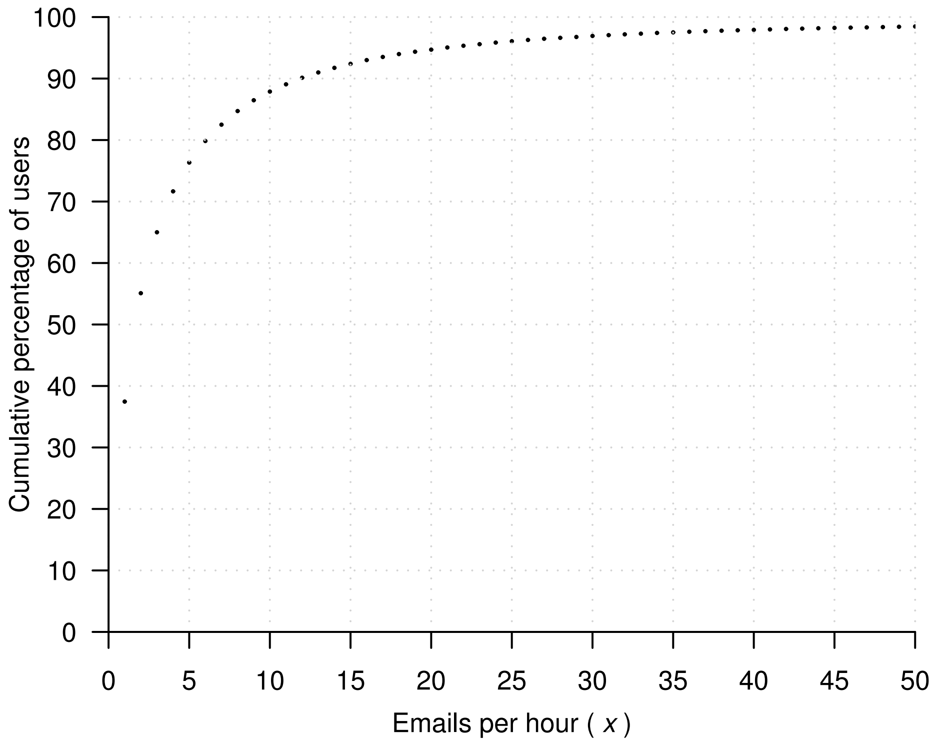


Figure 6.2: Percentage of users sending no more than x emails per hour

6.6 Conclusions

In this chapter, I have presented two different methods of calculating the limits to be placed on the sending of spam. If it is to be uneconomic to send spam, then senders must be restricted to 2 000 messages a day; even then, it would still make economic sense to send spam about highly profitable products.

Alternatively, if we want to reduce spam to 1% of normal email, yet we assume, (very realistically) that the spammers, between them, can steal the efforts of a million compromised machines – then the limit must be set to 250 messages a day.

Unfortunately, the limits one would wish to set are close to the global averages for email usage. Not surprisingly therefore, examining the variations of email sending by some real users showed that significant numbers of them would be unable to provide "proof-of-work" for all the legitimate email that they currently send, making the system infeasible to deploy. For proof-of-work schemes to be plausible, one would be looking for many orders of magnitude between the work that must be performed by the "good guys" and that achievable by the "bad guys".

Deploying a universal proof-of-work scheme seems an attractively simple way of defeating spam, not least because it would operate in a completely anonymous manner without any necessity to establish the identity of senders or trace the origin of incoming email. However, in order to make it viable at all, I have had to assume that there will be special arrangements for mailing-list email, and even then I have demonstrated that the scheme fails to be effective.

Naturally, one can imagine hybrid schemes where proof-of-work is merely one part of the whole and one can create "whitelists", validate correspondents cryptographically, solve human interactive puzzles or "CAPTCHAs" [2], deploy single-use email addresses, consult directories of trusted senders, and so on and so forth. The aim of these extra mechanisms will be to relieve the "good guys" from having to provide a proof-of-work with every email, while still insisting that the "bad guys" always have to make an effort. Such schemes, if they ever exist, will lack anonymity properties, will be very complex and, I believe, will be very fragile. A universal scheme would be far more likely to be robust, but what this chapter shows is that it cannot be made effective unless the spammers' cost base can be significantly increased and the number of insecure end-user machines significantly reduced.

Finally, I note that the simple application of real-world measurements and values has enabled me to debunk the magic properties that some have ascribed to "proof-of-work" systems in fighting spam. I am concerned that some of the other situations where proof-of-work has been proposed may also not have been properly analysed. It is important to consider how much money adversaries can spend, and how many resources they might steal. I commend this type of analysis to anyone considering using "proof-of-work" as a fairy dust that can be sprinkled on to a system design to enable it to survive attacks by determined opponents.

Chapter 7

The BT 'CleanFeed' System and the Failings of Traceability

I see the bad moon arising.
I see trouble on the way.
I see earthquakes and lightnin'.
I see bad times today.

— Creedence Clearwater Revival, 1969

There is considerable interest in some quarters in "censoring" Internet content so that particular types of material cannot be obtained. Edelman, in his work on website blocking [52], sums this up as follows:

> *"In the United States, controversial content often consists of sexually-explicit images. In Europe, hate speech is often of greatest concern. In parts of Asia, political speech is sometimes targeted."*

In this chapter I analyse a real-world system that has been deployed in the UK by BT. I consider how its approach can be subverted not only by "information hiding" approaches that make it overlook requests that it should block, but also by methods that undermine the assumptions that it makes about traceability. It is also possible to use the system to "reverse engineer" the list of sites being blocked which, since they host child pornography, is of considerable public policy significance.

7.1 Introduction

There are a number of mechanisms for blocking Internet access to content. Barring particular IP addresses makes entire sites unavailable, but this can cause significant collateral damage when other websites share the same address. It is also possible to subvert the DNS so that websites cannot be located. Barring access to particular URLs is a more precise technology in that it can make specific parts of sites unavailable. However, it is much more expensive, requiring stateful inspection of packet contents within a firewall, or the use of web proxies that interpose themselves between the requestor and the remote content.

In Britain, notwithstanding Edelman's stereotyping, there has been most interest in blocking indecent images of children (so-called "child pornography"). It has been illegal to "take" these images since 1978, illegal to "possess" them since 1988 and illegal to "make" them since 1994 [70]. In particular, the meaning of "make" has been defined in case law to include saving images onto a computer disk [42] and voluntary browsing of images so that they appeared, even momentarily, on the screen [3]. More detailed information about the legal background can be found in Sommer's book chapter [127] and in a position paper I wrote for the Foundation for Information Policy Research (FIPR) on reform of the law on "making" [35].

In 1996 a consensus was forged in the UK between the Government, police forces and the ISP industry to tackle the problem of indecent images of children [78]. At that time the problem was seen to mainly consist of the presence of these illegal images on Usenet, and a body called The Internet Watch Foundation (IWF) was established to operate a "hotline" for reports by members of the public. The experts at the IWF establish whether the report has identified an illegal image of a child (indecent images of adults can be entirely legal). If it is illegal then all UK ISPs are sent a report and the article is promptly removed from all of their news servers, thereby preventing these ISPs from being prosecuted for "possession".

Over time, the focus of the IWF's work has moved from Usenet (which by 2003 comprised just 2.4% of all reports [81]) to content hosted on websites. The IWF accepts reports of web content no matter where in the world it is hosted. UK material is reported directly to the UK police, whereas material hosted offshore is reported to the National Criminal Intelligence Service (NCIS) who pass the report via Interpol to the relevant national authority where the website is hosted. To avoid duplication of effort, the IWF maintains a database of URLs that have been inspected and keep a record of when they led to illegal material. In particular, it became apparent to the IWF that although some countries took down illegal content promptly, some websites remained accessible for a considerable time.

BT is one of the largest UK ISPs, operating under brand names such as "BT Openworld", "BT Yahoo!", "BT Click" etc. In late 2003 they decided to create an innovative blocking system, internally dubbed "CleanFeed".[1] Their aim was to prevent their Internet customers from accessing, either by accident or design, any of the illegal images of children listed in the IWF database. The existence of the system was leaked to the press [22] shortly before it became live in June 2004. The CleanFeed system is a hybrid design, incorporating both redirection of traffic and

[1]The official name for the project is the BT Anti-Child-Abuse Initiative.

the use of web proxies. It is intended to be extremely precise in what it blocks, but at the same time to be low-cost to build and operate.

In this chapter, I start with an account of content-blocking mechanisms, along with details of their worldwide deployment and previous studies of their effectiveness. The BT system is then described and its efficacy is considered. I then discuss how it can, rather unfortunately, be used as an *oracle* to reveal which sites it is blocking and give experimental results that show that this is a significant problem.

The work described in this chapter shows that the initial location of content is only one aspect of traceability. A blocking system needs ongoing knowledge as to where the content is currently located and this can be somewhat elusive when viewers and content providers both have strong incentives to mount active attacks.

7.2 Content-blocking systems

There are three basic methods of blocking content available to ISPs and network operators. These are packet dropping (which operates at OSI layer 3), content filtering (operating at higher protocol layers), and DNS poisoning (to prevent any connection to the site being made at all).

7.2.1 Packet dropping

Packet dropping systems are conceptually very simple. A list is created of the IP addresses of the websites to be blocked. Packets destined for these IP addresses are discarded and hence no connection can be made to the servers. The discarding mechanism can take note of the type of traffic, for example, it could just discard HTTP (`tcp/80`) packets and leave email alone.

The actual implementation can be done in several ways. For example, firewalls can be deployed on all end-user connections and these can then be programmed with the relevant list of addresses. Alternatively, existing network routing protocols can be employed to redirect traffic for the relevant addresses to a "black hole" that discards the packets. This latter scheme, sometimes called *IP null routing* when a router is not told a destination, is often used by ISPs to deal with short-term denial-of-service attacks where overwhelming amounts of traffic are targeted at their customers. The chief advantage of using the routing protocols is that when the blocking list changes this information is automatically (and very promptly) propagated to all the systems that need to be aware of it.

The main problem with packet dropping is the collateral damage that it causes because *all* of the web content on the particular IP address will become inaccessible. This can be very significant. Edelman [52] obtained a list of all the `.org`, `.com` and `.net` domains and tried to resolve the conventional website address for each of them by prefixing the domain name with `www` and looking this up in the DNS. His paper shows that 87.3% of such sites share IP addresses with one or more other sites and 69.8% with 50 or more other sites. There is no reason to presuppose that content that might be suppressed is hosted in any special way, so his conclusion was that there is a significant risk of "overblocking" with schemes that suppress content by methods based solely on IP addresses.

7.2.2 DNS poisoning

DNS poisoning systems work by arranging that DNS lookups for the hostnames of blocked sites will fail to return the correct IP address. This can be (naïvely) achieved in BIND, a well-known DNS server program, by creating a fake zone file such as

```
@     IN    SOA   localhost. root.localhost. (
                  2004010100 86400 3600 604800 3600 )
@     IN    NS    localhost.
@     IN    A     127.0.0.1
*     IN    A     127.0.0.1
```

and then adding extra lines to `named.conf` as follows:

```
zone "block1.com" in { type master ; file "fakezonefile" ; } ;
zone "block2.net" in { type master ; file "fakezonefile" ; } ;

   etc. etc...
```

The effect of this configuration is that BIND becomes authoritative for the blocked domains (`block1.com`, `block2.com`, etc.) and resolves hosts within those domains to the local (to the requestor) address of `127.0.0.1`. This makes all content hosted within those domains inaccessible. Similar arrangements, such as adding `127.0.0.1` entries to `hosts` files, can be used to block banner adverts on the web by arranging for failures to occur when accessing domains such as `doubleclick.com`.

This solution also suffers from overblocking in that no content within the blocked domain remains available. Thus it would not be an appropriate solution for blocking content hosted somewhere like `geocities.com`; blocking one site would also block about three million others. However, the overblocking differs from that identified by Edelman in that it does not extend to blocking other domains that are hosted on the same machine. There is also some "underblocking" in that a URL containing an IP address, rather than a hostname, would not be affected; because a browser would simply use the IP address and would not consult the DNS at all.

DNS poisoning also affects other services, such as email. The blocking of right-wing and Nazi material mandated by the regional government in North-Rhine-Westphalia in Germany has been studied by Dornseif [47]. He found that the majority of local providers had opted for DNS poisoning but had made significant implementation errors by using simple-minded BIND configurations, such as the one given above. Although `www.stormfront.org` was (correctly) blocked by all of the ISPs he checked, only 15 of 27 ISPs (56%) also blocked `stormfront.org` as they should have done, and he believes that all but 4 of them only blocked it accidentally. Further, just 12 of 27 ISPs (44%) permitted access to `kids.stormfront.org`, which was not subject to a blocking order. Email should not have been blocked at all, but nevertheless 16 of 27 ISPs (59%) caused it to fail for some domains; and in the case of `postmaster@www.stormfront.org`, every one of the ISPs he studied were (incorrectly) blocking email.

7.2.3 Content filtering

Content filtering systems will not only block entire websites but can also be used to block very specific items, such as a particular web page or even a single image. They determine that the URL being accessed is one of those to be blocked and then ensure that the corresponding content is not made available. This type of system is extremely accurate in blocking exactly what is on the list of URLs, no more, no less, and hence there should be no overblocking – provided, of course, that the list of URLs was correct in the first place.

The two main ways of providing content filtering are by means of firewalls and by using web proxies. A web proxy mediates incoming HTTP requests and will serve content from local copies (if it is a caching proxy) or make a request to a remote site on behalf of the actual requestor. It is straightforward, in principle, to arrange for a proxy to check each URL against a list of URLs to be blocked and if there is a match then provide an error message, or alternative content. A firewall implements content filtering by means of "stateful packet inspection", viz: examining the content of packets for application-level activity. Hence this solution is logically the same as the web proxy approach. A physical difference is one of architecture, in that the firewall machines will be placed on all relevant links, whereas a web proxy deployment will be at a small number of central sites. A more practical difference is that the web proxy is able to return an application level failure (such as "`404`") whereas a firewall will typically refuse to forward the packets and return a TCP level failure (viz: reset the connection).

Quite clearly, web proxies are ineffective at blocking content if their usage is optional. Hence it must be arranged that all customer traffic passes through the proxy, leading to considerable expense in providing equipment that can handle the load. Also, to prevent a single point of failure, the equipment must be replicated, which considerably increases the cost. Despite these economic issues, many ISPs used to fit "transparent web proxies" to their systems – not for content-blocking, but to serve web pages from local copies. They found that this was cost-effective because bandwidth used to be extremely expensive. However, the rapidly falling cost of moving IP packets around has made the business case for retaining, let alone deploying, such systems problematic, and they are becoming increasingly rare. The bottom line for most ISPs considering blocking systems is that although content filtering is the most precise method, it is also far more expensive than the alternatives.

7.2.4 Existing content-blocking schemes

A number of content-blocking schemes are known to have been deployed in various countries [151]. In China the current method appears to be a firewall scheme that resets connections [107]. Saudi Arabia operates a web proxy system with a generic list of banned sites, from a filtering software provider, augmented by citizen reported URLs submitted via a web form. This form is apparently very popular because "hundreds of requests are received daily" [85]. In Norway, the child pornography blocking system introduced in October 2004 by Telenor and KRIPOS, the Norwegian National Criminal Investigation Service, serves up a special replacement web page "containing information about the filter, as well as a link to KRIPOS" [139].

In Pennsylvania USA, a state statute requiring the blocking of sites adjudged to contain child pornography was struck down as unconstitutional in September 2004. The evidence presented to the court was that ISPs had, for cost reasons, been implementing blocking by means of packet dropping and DNS poisoning. Careful reading of the court's decision [143] shows that the resulting overblocking was by no means the only relevant factor; no evidence had been presented to the court that the blocking had "reduced child exploitation or abuse"; and procedural mechanisms for requesting blocking amounted to "prior restraint", which is forbidden under the First Amendment to the US Constitution. However, the mechanisms actually deployed were significant, since the court determined that it was also "prior restraint" that future content at a website would in practice be suppressed, even though the abusive images of children had been removed.

7.3 Design of the CleanFeed system

The exact design of the BT CleanFeed system has not been published. This description is based on several separate accounts and although it is believed to be substantially correct, it may be inaccurate in some minor details.

The scheme is a hybrid, involving a first stage mechanism that resembles packet dropping, except that the packets are not discarded but are instead routed to a second stage content filtering system.

The system is shown diagrammatically in Figure 7.1. The first stage examines all traffic flowing from customers (along the path labelled *a* in the figure). If the traffic is innocuous, then it is sent along path *b* to its destination in the normal way. If the traffic is for a suspect site, parts of which may be blocked, then it is redirected along path *c* to the second stage filter. This first stage selection of

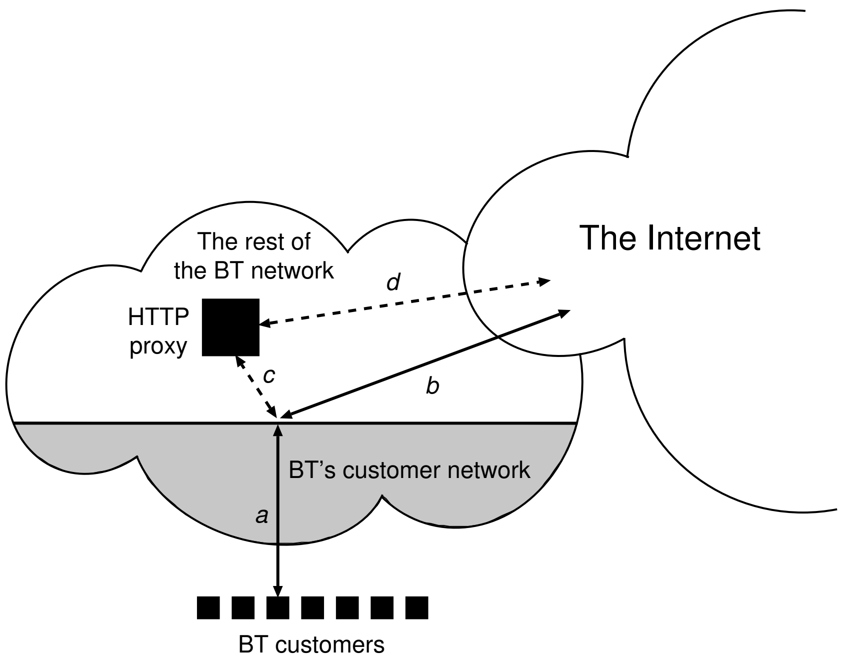


Figure 7.1: The BT CleanFeed system

traffic is based on the examination of the destination port number and IP address within the packets. The second stage filtering is implemented as a web proxy that understands HTTP requests. When the request is for an item in the IWF database a `404` (page unavailable) response is returned, but all other, innocuous, requests are relayed to the remote site along path *d* and the material returned to the customer in the reverse direction.

The IP addresses used by the first stage are obtained by working through all the entries in the IWF database and translating the hostname into an IP address in the normal way by making a DNS query. The results are amalgamated and used to modify the normal packet routing (controlled by BGP) within the customer-facing portion of the BT network (shaded in the diagram) so that the HTTP packets for these addresses will be routed to the web cache.

The second stage web proxy uses the URLs from the IWF database. Because there are concerns about keeping a human-readable form of the list on the server, it is held in what journalists have called an “encrypted form” (presumably as cryptographic hashes). The request is also “encrypted” (hashed) and a check for a match is then made. When there is no match, the proxy issues a request to the remote site in the usual way and then presents the response to the requestor. It is unclear, and not especially relevant to this discussion, whether the proxy also acts a cache, serving local versions of recently accessed material.

When compared with the generic solutions outlined in Section 7.2 and the systems deployed elsewhere in the world discussed in Section 7.2.4, the CleanFeed system has some significant advantages. Although its first stage uses the same approach as “packet dropping”, it does not suffer from overblocking because the second stage web proxy can be as selective as necessary. However, the second stage can use low-cost equipment because it only needs to handle a small proportion of overall traffic. By avoiding DNS poisoning, the designers can be sure that only web traffic will be affected and not other protocols such as email.

Therefore CleanFeed is, at first sight, an effective and precise method of blocking unacceptable content. However, there are a number of detailed implementation issues to address as soon as one assumes that content providers or content consumers might start to make serious attempts to get around it.

7.4 Circumvention of the CleanFeed system

The CleanFeed system operates in two stages and circumventing either stage will permit access to the content. In this section I discuss the various strategies available for circumvention. Before starting on that discussion, since it is an important building block that makes some of the other strategies possible, I first explain how accesses by the IWF and/or the CleanFeed system components might be detected by the content provider – so that these accesses can be treated in some special way.

7.4.1 Identifying the IWF and CleanFeed

If a content provider can identify that an access is being made by the IWF then they can provide information that is specific to that access. For example, they might provide innocuous images to the IWF and illegal images to everyone else. If they can do this successfully, the IWF will never blacklist their site and so it will not be blocked by CleanFeed. It is possible that some content providers already take this approach, since the IWF report that only 33% of hotline reports are substantiated as potentially illegal [81]. Identifying accesses by CleanFeed itself (in particular, components that do DNS lookups) also gives the content provider the opportunity for denial-of-service attacks as outlined in Section 7.5 below.

A summary of how a content provider might detect IWF/CleanFeed activity is presented in Table 7.1, along with possible countermeasures. Most of the table entries should be self-explanatory, but the more interesting are expanded upon below. Note that the access detected by the content provider does not necessarily have to be for a web page – the same information may be leaked by DNS queries.

The IWF’s IP address allocation:

Identifying the /26 network allocated to the IWF in the RIPE registry is straightforward. One looks up the IP address of the IWF mail server (specified by the MX record in the DNS) and then determines who that IP address has been allocated to.

Content Provider Strategy	**Countermeasure**
Recognise the accessing IP address.	Access via web proxies.
Recognise source of DNS requests.	Use DNS proxies for name resolution. Ensure that CleanFeed access is random rather than a regular occurrence.
Anonymously report a sacrificial website to the IWF. Anyone who arrives to look at it must be the IWF or, later on, the police. Bar similar access in future.	Choose proxies and anonymous access systems likely to be used by genuine customers so that content provider will bar them as well.
Serve active content to run on viewer's machine and reveal their identity; for example by including it in request URLs or by making requests directly to the server.	Disable Java, JavaScript, etc.
Serve cookies (from a central site) to tie visits to disparate sites together.	Refuse to return cookies (and/or clear cookie cache before browsing a new site).
Serve content with a request it be cached. Failing to refetch indicates a repeat visit.	Clear cache before browsing a new site.
Ensure unique URLs are used in advertising spam. A second access from a new IP address will be the IWF acting upon a report from the public.	Discard any obvious tracking information appended to URLs. Avoid starting visits in the "middle" of a website, but follow the links from the front page.

Table 7.1: Detecting content access by the IWF or the CleanFeed system

It might not have been possible to find the entry in RIPE (and the organization name is misleadingly set to "Internet Foundation" rather than IWF, probably through human error rather than an attempt at security). If so, then the simplest method of determining the IWF's network allocation would be to find some pretext for exchanging email with an IWF employee because the sending IP address will then be recorded in a `Received` header field.

Experimental checking of access:

Let us suppose that the IWF wished to cease accessing suspect websites directly from their own network. This will occur either as a countermeasure against being immediately recognised, or possibly even because their own ISP has implemented a blocking system (an issue that, I am told, caught out some in the UK police when the BT CleanFeed system was introduced). However, the content provider can still determine that a particular access is from the IWF.

If the content provider creates a website and fails to publicise it to anyone, then no-one will access it (except for various instantly recognisable search-engine “spiders”, and of course “worms” and other malware). If the website is then self-reported to the IWF hotline, any access that does occur must be IWF analysts in the first instance and police forces thereafter. As a bonus, it would be possible for the content provider to assess the speed of reaction to reports and the delay before their local enforcement agency becomes involved. Assuming that the content provider’s identity is not unmasked by this process (and hence they are put out of business), then they will have valuable information to use in the future.

The IWF’s countermeasures for this type of unmasking are twofold. First they can ensure that they have a sufficiently diverse set of access addresses that having some compromised is immaterial. They might do this by inviting members of the public, or business organizations, to run proxy server software on their behalf. However, there would be significant legal and procedural difficulties to overcome to ensure that the “good” IWF accesses to remote sites could not be mistaken for “bad” accesses by users at the donating sites. Second, the IWF could arrange for it to be against the best interests of the content providers to treat their addresses specially. The easiest way to achieve this would be to channel IWF activity via web proxy caches, either at ISPs or third party systems. Treating these IP addresses specially would be counterproductive for the content provider because large numbers of potential viewers would also be handled as if they were the IWF. Third, the IWF could use anonymous browsing systems such as Tor [46] or JAP [18] where their traffic is mixed with that of others and their identity cannot be determined. Once again, the content provider may not wish to prevent (potentially lucrative) access from such systems.

Use of active content:

The content providers have further tricks available to them, in that they can serve active content, such as Java or ActiveX, in order to determine the IP address (and hence the identity) of the accessing machine, even if it is behind a web proxy.

The basic approach is to have the code ascertain the viewer’s IP address and to then access a URL which includes that IP address as a parameter. Since the contact is with the server that sent the web page, this access is permitted within the Java (and JavaScript) security models. As a countermeasure, active content can be disabled within the browser (in practice it will be necessary to disable all of Java, JavaScript and ActiveX) but this may significantly disrupt the viewing experience for sites that are heavily dependent upon these technologies. The alternative is to use tools that permit access to “innocuous” content, but filter out constructs that could reveal the viewer’s identity. This type of filtering is extremely complex, as I demonstrated, in conjunction with others, in a 2001 paper on breaking the anonymity of users of a “lonely hearts” website [33].

Multiple accesses with the same identifier:

If the content provider site uses cookies, which are stored on the requesting machine, then it can tie together accesses from multiple locations to the same originating machine. If it is known that the first request is from the IWF then further accesses that return the same cookie are also from the IWF. Cookies are only returnable to the serving site, but the content provider can use the trick developed by companies such as DoubleClick Inc., which is to associate the cookie with a banner advert served from a single central server which can track visits to all participating sites [56].

A related approach is to spot a tell-tale failure to access some content. If the website marks some of its content (again, for example, a banner ad) to be cached, then if a visitor to another site fails to ask for that item then it must be that they have a local copy and hence, as with the cookies, the two sessions can be tied together.

These schemes are fairly simple for the IWF to counter. Modern browsers have numerous settings for handling cookies and so the IWF can ensure that they are cleared away before moving on to examine a new site. Similarly, they should be clearing out the local caches on their systems before switching to a new task.

A further way of tracking IWF access is available to those who advertise their content via unsolicited bulk email (spam). It is a simple matter to personalise these emails to track any resulting access, whether intentional by clicking on a link, or unintentional through exploiting email reader features such as the automatic downloading of images. If the recipient of the email reports the content to the IWF (or indeed to any other agency) then there will be a second access, using the same unique identifier, to the content. However, this second access will be from a markedly different IP address than the first, almost certainly in a different AS (different ISP address space). The website can then deal with this second access by serving innocuous content, recording it as identifying the IWF, or by taking such other actions as it may see fit.

This method does carry a risk of "false positives", where individuals who are interested in the content return to an email in the privacy of their own home. However, if the IWF are assumed to work UK office hours, then a reasonable attempt can be made to distinguish the IWF from a hot prospect as a customer.

The IWF's main countermeasure to this approach is to identify that it is occurring and to try and avoid using the personalised identifier when accessing the suspect site. That is, before accessing the site they should edit the URL to make it as generic as possible. The "office hours" problem could be fixed by arranging to use proxies and third party relays on the other side of the world. An 11am access from the IWF offices in Oakington doesn't look like a potential customer for illegal content, whereas if the access came from Sydney (where it would be 9pm), it might look quite attractive to the content provider to display their true wares and not dissemble.

7.4.2 Evading CleanFeed

There are generic ways a content requestor (a customer) can avoid content-blocking, such as tunnelling traffic via proxies that can access the content directly without any intervention. Dornseif [47] discusses this and a number of other techniques and the OpenNet Initiative reported on various ways to evade China's partial blocking of Google [107]. As I show in Section 7.4.4 below, this topic is related to an existing body of work on Intrusion Detection Systems that must handle obfuscated traffic. However, no-one has yet written a complete treatment of how to evade blocking systems, perhaps because it would fill a complete thesis on its own.

For the hybrid CleanFeed system it is obviously effective to evade either of the two stages. However, CleanFeed's design does have some advantages because countermeasures can involve the first stage being less precise about traffic selection because the second stage will provide accurate blocking. Table 7.2 summarises possible evasion strategies, some of which will now be discussed in more detail.

Content Requestor Strategy	**Countermeasure**
Use a tunnelling technique, or a proxy system, such as Tor, JAP, etc.	Also block the tunnels and the proxies (this is unlikely to be scaleable).
Use IP source routing to send traffic via routes that will evade the blocking.	Discard all source routed packets (often Best Practice anyway).
Encode requests (perhaps by using %xx escapes) so that they are not recognised.	Ensure URLs are put into a canonical form before they are checked.
Add specious characters to URLs (such as leading zeroes) to avoid recognition.	Ensure URLs are put into a canonical form before they are checked.
Send specious HTTP/1.1 `Host:` details for an HTTP/1.0 site.	Check whether remote site acts upon `Host:` information.

Table 7.2: Evading the CleanFeed system

Using a web proxy:

Browsers can be configured to use a web proxy that will connect to a website on their behalf. Provided that the proxy is not within BT's customer network, it will be able to access the blocked content without interference and return it to the customer. In principle, CleanFeed could examine any requests that are sent to external proxies. However, this would increase the number of IP addresses routed via the CleanFeed proxy by many orders of magnitude, which would be entirely impractical.

The main reason for the very large number of possible proxies is very many end-user systems are insecurely configured and inadvertently provide service to anyone who accesses them. Websites such as `http://tools.rosinstrument.com/proxy` provide substantial online databases of publicly accessible proxies. Lists of open proxies are also published by anti-spam organizations such as SORBS [128] because email spammers can exploit proxies by issuing commands through them such as `CONNECT emailserver.example.com:25 HTTP/1.0`. Since they are connecting via a local host, the spammers can exploit an email server that would otherwise bar them as strangers. Although the intent of the SORBS list is to "name and shame" proxy operators – and put pressure on them to fix their configuration – in practice the lists are growing rather than shrinking and now number in the millions.

Proxies are also available as paid-for services, with the best known service being `www.anonymizer.com`, but there are dozens of others. Systems such as `findnot.com` provide not only a proxy but also a virtual private network (VPN) solution whereby all traffic, not just web traffic, is sent over an encrypted tunnel to their servers and only then routed to and from its destination in the usual way. The encryption means that selective blocking is impossible and it would have to be all or nothing.

In passing, one can note that third party services would also provide a way of circumventing DNS poisoning. CleanFeed does not use DNS poisoning, so there is no reason not to use BT's own DNS servers, although it would be easy to avoid them if it was necessary. Despite experts advising that recursive lookups on DNS servers should be disabled [73], the low profile of attacks on DNS servers leads to this advice being widely ignored. However, since there are currently few uses for "open" DNS servers, there do not appear to be any databases of "insecure" systems available at the present time.

Obfuscating web accesses at the HTTP level:

The CleanFeed system does not actually compare a requested URL directly with the IWF blocking list, but instead compares their cryptographic hashes. This design feature is apparently to avoid the security issues that would arise if the IWF list was present *en claire* on the web cache machine. Provided that a cryptographically sound hash function (such as SHA-1) is used, there will not be any lack of precision in what is blocked. However, if the URLs in the IWF list are processed naïvely, this can make the system trivial to circumvent, and the use of hash values means that it will not be possible to adjust the matching code to make it more "fuzzy".

The difficulty initially arises because many different URLs can specify the same content. For example, many web servers (especially those running on a Microsoft Windows platform) do not treat URLs as being case sensitive, so these URLs may well all access the same page:

```
http://www.example.com/Webpage.html
http://www.example.com/webpage.html
http://WWW.EXAMPLE.COM/WEBPAGE.HTML
http://www.Example.com/WebPage.Html
```

Requests can also be encoded in hexadecimal without changing the underlying request, for example:

```
http://www.example.com/%57%65%62%70%61%67%65%2E%68%74%6D%6C
http://www.example.com/W%65bpage.html
http://www.example.com/W%%36%35bpage.html
```

though note that the third example exploits a double-parsing process performed by Microsoft's IIS web server but not by other web servers such as Apache.

Servers that only host one website are likely to have been configured to respond to requests that give the site's IP address rather its name. Hence the blocking system needs to take account of this access mechanism as well. However, there are a number of different ways of encoding the IP address. For example, if `www.example.com` resolves to `10.11.12.13` then these URLs all lead to the same location:

```
http://10.11.12.13/Webpage.html
http://012.013.014.015/Webpage.html
http://0012.000013.0000014.00000015/Webpage.html
http://168496141/Webpage.html
```

where in the second example the components of the IP address are expressed in octal, in the third there are extraneous leading zeroes (and the value remains in octal), and in the fourth the address has been expressed as a single decimal value.

In the past, a number of other, rather unlikely, variations have been found to be handled by common browsers [93] and doubtless in the future the wide range of semantically identical but syntactically differing IPv6 address formats will give rise to even more variations.

In order to formulate the precise de-obfuscation requirements for the CleanFeed system it is necessary to understand the exact mechanisms employed when intentionally obscured URLs, such as those in these examples, are accessed.

Whenever a user types a URL into their browser three components are generated; the scheme name, the hostname and the path:

- The *scheme name* is `http`, `ftp`, `telnet` etc. and specifies the protocol to be used for accessing the resource.
- The *hostname* specifies the remote host and is dealt with in two ways. Firstly it is looked up in the DNS to provide an IP address to the protocol handler and secondly, in HTTP/1.1, it is encapsulated within the protocol itself so that the remote site can run multiple servers on a single IP address [58].
- The *path* will not be processed within the browser at all, but is sent to the remote site for it to deal with.

Thus, for example, the user level request:

```
        http://0012.000013.0000014.00000015/W%65bpage.html
```

will result in a browser making an HTTP (`tcp/80`) connection to the IP address `10.11.12.13` (because the values are interpreted by the browser to be octal) and sending a `GET` request containing the text strings `0012.000013.0000014.00000015` and `W%65bpage.html` to the server.

Microsoft's Internet Explorer v6 web browser will send:

```
GET /W%65bpage.html HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
    application/vnd.ms-excel, application/msword,
    application/vnd.ms-powerpoint, application/x-shockwave-flash,.*/*
Accept-Language: en-gb
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows.NT.5.0; Q312461;
    .NET CLR.1.0.3705; .NET.CLR.1.1.4322)
Host: 0012.000013.0000014.00000015
```

In particular, it should be noted that the obfuscated forms of both the path and the hostname are being sent to the remote site, so it too will need to process these and decide what content is being requested.

Hence the amount of obfuscation that still permits access to the content is mainly determined by the remote site, but is also partly determined by what the browser is prepared to handle. However, a specially crafted anti-blocking program running locally on the user machine would be able to take normal requests and specially obfuscate the `GET` request information so that the user could use the normal form of URLs, and browser imposed restrictions on syntax would be irrelevant.

Quite obviously, the fix that CleanFeed must apply to prevent all of these problems is to convert the URLs in the database into a canonical format prior to creating the cryptographic hash that is to be compared. Although one should start with the published standards for URL syntax, it will also be necessary to determine what syntax web-servers actually accept in order to handle all possible forms of obfuscation that will work in practice. As an example of how complex this will be, it is reported [93] that some servers will map an IP address of `266.267.268.269` onto `10.11.12.13` because arithmetic overflows on each octet value are ignored.

The alternative to making the canonicalisation code ever more complex will be to take a pedantic approach and block any request that contains syntax that is anything other than straightforward to handle. However, this runs the risk of disenfranchising customers using poorly-written browsers that emit non-standard requests.

All of the above analysis may be summarised by saying that the CleanFeed system must be supplied with the cryptographic hash of the canonical form of the path, which is to be compared with the hashed canonical form of the value from the GET request. The paths are only candidates for blocking if the canonical form of the `Host` supplied in the request matches either the canonical form of the site's name in the IWF supplied URL *or* the canonical form of any valid IP address for the site, as fetched from the DNS.

The final set of HTTP obfuscation issues relates to actually locating the relevant protocol components within a request. For example, if there are two `Host` entries then CleanFeed may well clean this up and send only one entry to the remote website. However, the system would have been evaded if the `Host` entry that was emitted failed to be the one that was checked.[2]

More subtly, the `Host` entry may be entirely misleading. It was introduced in HTTP version 1.1 to permit servers to provide "virtual hosts" on a single IP address [58]. The server uses the `Host` value to know which of the many websites it hosts is actually being accessed and, as already discussed, the CleanFeed system also checks for a match on the host details before deciding to block. If the blocked site was an HTTP/1.0 system, which has a one-to-one mapping between IP address and server, then it will take no notice of the `Host` setting. Hence, the user could arrange to supply a misleading `Host` value in the `GET` request and CleanFeed will not block the request, yet the remote site will respond normally. The CleanFeed system could detect that this was happening by examining the version number in the HTTP responses and dealing specially with servers that did not respond `HTTP/1.1`, but this is likely to involve significant redesign since the system will currently be making its blocking decisions when a request arrives from the user and it would have to wait until a response arrived from the remote website.

Source routing:

Source routing permits the sender of a packet to select some aspects of the route it will take to its destination. The feature is selected by an IP option setting that is as old as the IP protocol itself, and is described in RFC791 [110]. Selecting a route that avoided the first stage redirection mechanism would be a way of evading CleanFeed.

"Loose Source Routing" is IP option 3, but is encoded with a "transmit" flag to give the value 131. The option specifies a series of IP addresses to which the packet must be sent before it is finally routed to its destination. Alternatively, "Strict Source Routing" (IP option 9, encoded as 137) can be specified. This works the same way as Loose Source Routing, except that intermediate nodes must pass the packet directly to the next specified address.

Source Routing is seldom needed for legitimate traffic, the only common exceptions being its use in debugging network problems or routing traffic over specific,

[2]Linhart *et al.* [91] have recently described a similar scheme for circumventing proxies which involves the use of multiple `Content-Length` values within a single HTTP request.

uncongested, links in special situations, for example the use of `traceroute -g` or `telnet @hostname1:hostname2`. When Source Routed packets are seen, they are often associated with "wicked" activities, such as evading firewalls, so it is not unusual to see such packets dropped, and my own tests found that BT are currently discarding source-routed packets; hence this evasion method does not currently work with CleanFeed.

7.4.3 Circumvention by content providers

This section looks at techniques that can be used by content providers to ensure their sites remain available, even to people whose access is supposed to be prevented by CleanFeed. These passive measures are summarised in Table 7.3 and details are given below. I leave to section 7.5 more proactive measures that are, in essence, attacks upon the CleanFeed system itself, which could be used to make it significantly more expensive to operate.

Content Provider Strategy	**Countermeasure**
Move site to another IP address.	Regular updates of IP addresses that are to be blocked.
Change port used for access (harder to track than address change, but may disrupt users as well as the blocking system).	Redirect other ports (paying careful attention if ports such as `tcp/53` (DNS) need to be intercepted).
Provide site on multiple URLs.	Generate generic rules for blocking URLs.
Accept unusually formatted requests.	Extend canonicalisation to reflect what server accepts (which may be hard to establish with certainty).

Table 7.3: Content provider strategies to negate CleanFeed blocking

Changing IP address:

The most straightforward tactic for a content provider is to move their content from a blocked location to an unblocked location and the simplest way of changing location is to use a different IP address. If CleanFeed is blocking `10.0.0.1` the content is moved to `10.0.0.2`. The cost and practicality of such a move will vary depending upon the hosting arrangements that the content provider has made. Certainly, where the infrastructure is directly owned by the content provider (perhaps they own, or can control, a small company with a /24 or a hosting site with a /21) then they will be in a position to change IP address very easily. Indeed, they can prearrange to respond on a number of IP addresses without using any extra equipment (a technique a Linux user would call *IP aliasing*).

Assuming each move of IP address is accompanied by a change to the DNS, then users will still be able to locate the website. Clearly, the countermeasure is to track IP address changes, through regular inspection of the DNS.

However, there is a limit to the speed at which the CleanFeed system can follow these address changes because of its use of BGP in the first stage of the system. If changes are made too often, then the routers will conclude that a "route flap" is occurring and will "damp" further changes to prevent them propagating. The fix for this is to avoid withdrawing routes and just add the extra addresses, perhaps amalgamating the various individual addresses (/32 routes) into larger blocks. This will of course send all of the traffic for the content provider's network through the second stage, which – because it is a self-contained system – can track the changes in real time without ill effects. The extra traffic in the second stage system will lead to increased load, but not of course to any extraneous blocking.

The CleanFeed system can easily tell that this type of location movement is occurring merely by checking logs of DNS activity. It can then set the checking rate to be "fast enough". Note that it is not sensible to rely on the announced time-to-live values in assessing how often to check for changes, because the site may be using IP aliasing and arranging to cycle through its set of addresses rather faster than the declared parameters would suggest.

Changing port number:

Another way in which content providers can avoid blocking is by providing their content on the same address, but on a different port than the conventional HTTP port 80. This is far less attractive for the content provider, because there is no automated discovery mechanism for these new port numbers, such as DNS provides for an IP address change, and so returning customers would be unable to determine which port was now being used. This could be fixed by accessing the real website through an intermediate page containing only links (or automatic redirection). As well as permitting port numbers to be altered, the rest of the URL could be changed as well. However, despite there being nothing illegal on the page itself, the IWF may decide to add the intermediate page to their database, on the basis that it leads directly to the images they wish to block.

As it happens, at present, changing port completely avoids content suppression, since CleanFeed only redirects port 80. However, there is no significant technical reason why CleanFeed should not redirect a range of ports. If content providers moved to a port which was also used for large volumes of other traffic (peer-to-peer file sharing protocols perhaps) then issues of load might arise; viz: the second stage might be swamped by the traffic. Also, the web cache will not be transparent to the non-HTTP traffic, which could cause problems if the move was to a specialist port such as `dns` (53).

Changing URL:

As discussed above in Section 7.4.2, blocking HTTP/1.0 servers is difficult because the `Host` setting in the `GET` request will be ignored by the web server, but could be used to fool the CleanFeed system into permitting an access that ought to be blocked. The countermeasure discussed was to detect that the server was using v1.0 and change the blocking strategy accordingly.

However, if the server was to run v1.1 but intentionally ignore the `Host` setting then the content would remain accessible. This strategy would only work if a provider was prepared to promulgate a large number of hostnames, but it would be easy to send out advertising material (spam!) that told one potential customer that the host was `http://www.aaaa1.example.com`, the next that it was `http://www.aaa2.example.com`, etc. The CleanFeed countermeasure would be to detect that this was happening and generate blocking rules based on wildcards (or regular expressions), but the success of this would depend entirely upon the facilities available in the proxy cache software. Note that adding new subdomains (below `example.com`) has no financial cost for the content provider whereas as changing to a new domain (such as `example1.com`) would involve paying a registrar for the new domain.

Accept generic requests:

Clearly, all the techniques that allow end-users to circumvent CleanFeed will work even better if the content provider is actively assisting, for example by accepting many forms of escape character within URLs. This can be taken to its ultimate by a provider that distributes a program to intercept requests and scramble them so that CleanFeed sees impenetrable ciphertext in the `Host` and path values. Alternatively, it may be found that placing an innocuous value into `Host` and passing the real host information in a non-standard field, such as `Ghost`, will evade the blocking. Unfortunately, in these days when end users are suspicious of trojan programs there might be resistance to running an executable file, but most of these attacks can be programmed in JavaScript – which is less likely to be seen as threatening; indeed it is commonplace for "shopping cart" systems to "encrypt" links so as to prevent end users from circumventing the need to visit all the payment pages.

7.4.4 Related work on evasion

An intrusion detection system (IDS) monitors the activity of a system with a view to detecting unauthorised actions. There are two main approaches, that of "anomaly detection" where deviations from the normal patterns of behaviour are noted and "misuse detection" where explicit patterns are identified. In this community the term "evasion" is used for attacks that fail to be detected by an IDS. Quite clearly, the CleanFeed system has very great similarities with a misuse detection system and must be robust against evasion.

The problem of obfuscating requests has long been recognised in the IDS literature and as far back as 1997 Cohen was humorously listing "50 Ways to Defeat your Intrusion Detection System" [39]. Although many of the 50 ways are irrelevant, several of the ideas he proposes are directly applicable to HTTP requests.

Modern academic misuse detection systems now tackle some of Cohen's issues by using very abstract models of misbehaviour in domain-specific languages such as STATL [51] or GrIDS [31]. Nevertheless, the problem of parsing the raw data remains an *ad hoc* process and problems with HTTP have not been addressed by the academic community (although careful work has been done on similar problems at the IP level [67]). What published work there has been on HTTP has been either by those who have produced evasion tools such as Rain Forest Puppy's `whisker` [115] or by practitioners, such as Roelker [118], who as recently as September 2004 was still finding it worthwhile to explain more than a dozen different ways of manipulating URLs and the HTTP protocol to evade IDS systems.

Back in the mainstream of IDS research, there is a continuing concern that although IDS systems may be effective against one form of an attack, the "signatures" may not be general enough. Recent work by Vigna *et al.* shows how it is possible to develop quantitative measures of IDS performance in the face of "mutant exploits" [145] that are created by obfuscating activity at several different protocol levels.

As an interesting aside, URL manipulation is not always done to attack a system, but can have beneficial uses. For example, Hacker (his surname, not his *modus operandi*) suggests [66] setting resource strings to have an unusual mixture of upper and lower case characters (such as `~/Cgi-Bin/`) so as to detect scripted intrusion attempts (which are likely to access a standardly named `~/cgi-bin/` directory).

Also worthy of note is Feather's 2001 proposal, put forward as an IETF Internet Draft [57]. This attempted to specify how to create standard hash values from URLs, exactly the issue dealt with above. He wished to be able to specify a standard way of distributing lists of websites for purposes such as incorporating them into end-user content-blocking systems and, like BT, he wished to prevent the list from being converted back into a list of websites.

Feather realised that it was necessary to canonicalise the URLs before processing them through the cryptographic hash function, and pointed out the need to fix up the case of the hostnames and remove extraneous material such as trailing fragment identifiers (introduced by `#`, and providing access to positions within a document). His proposal went through several iterations over the next two years, without anyone who reviewed it addressing the need for much more substantial canonicalisation rules as discussed here and, in particular, the need to cope with the *de facto* syntax recognised by web servers as well as the *de jure* syntax described in the RFCs.

However, the Internet Draft has not been revised for some time, and the IETF now views it as entirely lapsed, so it is unlikely to become formalised as an RFC.

I have already mentioned the potential problems that the CleanFeed system faces when two `Host` values are provided in a single `GET` request. This is a special form of a very difficult problem that is faced by IDS systems, and which can be a very effective attack on firewalls doing "stateful packet inspection".

If a requester splits their request across many packets, either at the IP or TCP level, then these segments can be made to "overlap" each other [113]. If different text is placed into the overlapping sections then the IDS system has the problem of second-guessing how the remote system will handle these overlaps – will it use the first version or the last? Handley *et al.* examine these questions in great detail [67], and their methodology ought to be applicable to higher level protocols such as HTTP. However, since the CleanFeed system is not snooping on the traffic, but terminating the connection from the user and creating new requests to the content provider, it is most unlikely to do partial parsing of the request, but will only examine the input when it is complete, so there is no advantage in creating inconsistent intermediate states, but only value in having a final state that contains duplication or inconsistencies.

7.5 Attacking CleanFeed

Content providers could also actively attack the CleanFeed system, with a view to having it closed down because of the collateral damage occurring when other websites are accessed. Example strategies and countermeasures are summarised in Table 7.4 below. Some build upon being able to identify CleanFeed system accesses and provide bogus information to them (see Table 7.1 above).

Once again, the more interesting topics will now be dealt with in more detail.

Overloading the web caches:

A key aspect of the CleanFeed design is that the filtering effect of the first stage permits the use of (relatively) inexpensive devices in the second stage. The web caches do not need to be of a very high specification because they only need to handle the traffic for the handful of sites that are co-located with the blocked sites.

If the content provider can arrange for large amounts of traffic to be selected by the first stage, then the second stage proxy may fail to function correctly. This could either be by failing "open" (i.e. not filtering anything) or, more likely, failing "closed" (i.e. causing collateral damage by preventing, or substantially delaying, access to legitimate sites).

Content Provider Strategy	**Countermeasure**
Change location (IP address) of content very rapidly and often so that first stage routing "flaps".	Add addresses quickly but remove slowly. No harm in sending extra traffic to the second stage unless it is overloaded.
Return specious DNS results referring to high traffic third-party websites. Hope to overwhelm the second stage web cache.	Avoid automating changes of IP address, and run sanity checks on returned values.
Return specious DNS results implicating BT customer or service machines, hoping thereby to create traffic loops.	Discard all results for external sites that claim to be inside the BT network.
Overload system by creating large numbers of addresses to block (e.g. by distributing content, perhaps by hijacking innocent machines to host the material).	Unlikely to hit any limits in second stage web cache. In first stage, stop considering single addresses but redirect entire subnets. Provided cache can cope with traffic volume, no faults will be visible.

Table 7.4: Attacking the CleanFeed system

For the type of material that the CleanFeed system is blocking, it might be difficult for a content provider to arrange to host their site on the same machine as a highly popular website with large amounts of traffic; although this might be arranged for political speech, such as a state like China might wish to block. However, since the content provider controls the DNS for their site they could, spuriously, add in the IP addresses of popular sites. These extra IP addresses could either be provided solely to the CleanFeed system, or be a general feature such that at low probability any viewer would be given the wrong address. From the viewer's point of view the result will be a browser failure (`www.popular-site.com` will refuse requests for `www.nasty-site.com`) but from CleanFeed's point of view the result is a potential disaster – with all access to the popular site going via the second stage web proxy machines. Nothing will be blocked incorrectly, but much will be slowed down.

Creating loops:

There will be difficulties with traffic levels if requests can be made to flow around in a loop. TCP/IP level loops would, eventually, be dealt with by the IP time-to-live counter, but HTTP level loops could potentially flow round forever because the web cache machine creates a new outgoing request for each incoming request it is prepared to honour.

Loops can arise in two ways, by the content provider falsely claiming to be situated within the BT customer network,[3] or falsely claiming to be sited on the web cache machine itself. Manual checking of IP addresses should be sufficient, but it would be simpler to mandate that addresses that are within BT's own network can never be added to the CleanFeed database.

Swamping the system:

Swamping the CleanFeed system by requiring it to block "too many" URLs does not seem practical. Unless the web proxy is especially poorly designed, it will be able to handle almost indefinite numbers of URLs. If these URLs moved around a lot, then checking out the DNS entries (as explained to be necessary above) could cost significant manual effort, but the second stage itself is almost indefinitely scalable – and hence not worth attacking.

The first stage of the CleanFeed system does have scaling issues. It works by the addition, via BGP, of extra routes to the tables used by the routers. There will be an upper limit to the rate of addition (computational resource issues could cause routing "flaps") and the routing equipment may run out of memory for recording the extra data. Without detailed knowledge of BT's equipment, it is impossible to estimate the storage limit, but an educated guess would place it between 10 000 and 50 000 routes. Hence, any content provider (or cabal thereof) that can appear to be located at 50 000 or more unique IP addresses cannot be completely suppressed.

Of course, if the unique addresses are adjacent to each other then it will be possible to describe them as, say, /24 routes rather than many individual /32 addresses. This does not cause any problems, even if some /24 subnets contain many legitimate machines – the second-stage filtering will ensure that no collateral damage occurs. However, the more traffic that is routed to the second stage, the more likely it is to fail to cope with the workload.

The simplest way of providing content on large numbers of IP addresses is by means of a proxy network. In October 2003 Wired reported [98] that a Polish group was advertising "invisible bulletproof hosting" by exploiting a network of 450 000 end-user machines on which they had surreptitiously planted their own software. As discussed in Chapter 6, obtaining the services of a million machines is quite conceivable, so the Polish claim is not entirely outrageous, although without independent verification it cannot be seen as entirely trustworthy.

7.5.1 Blocking legitimate content

In a handful of special cases, the content provider can arrange for legitimate content to be blocked. Besides the inconvenience to those requiring access to this content,

[3]If the content was *actually* within the BT customer network then one can assume that there would be absolutely no difficulty in having it promptly removed – there would be no need to add the site to the list of sites to suppress.

the effect is to bring the blocking system into disrepute and it must therefore be seen as an important attack.

Systems sometimes provide links based on IP addresses rather than by hostnames. This often occurs when linking to back-end database systems that serve query results or extracts from atlases.

For example, a link to the Google cache of "the snapshot that we took of the page as we crawled the web" might be of the form `http://66.102.9.104/search?q=cache:FFKHU5mkjdEJ:www.cl.cam.ac.uk/users/rnc1/`. If so, then by ensuring that an illegal image at `http://www.example.com/search` was blocked by CleanFeed (using an anonymous hotline report) then the owner of the DNS for `www.example.com` can arrange for Google's cache to become inaccessible by serving `66.102.9.104` as a possible IP address for `www.example.com`.

Similarly, 'etc venues', a prestigious provider of venues for conferences attended by the Great And The Good, has web pages to describe their sites such as `http://www.etclimited.co.uk/venues/park.html`. In late 2004 the online directions were being linked to as `http://195.224.53.128:8080/directions/etc/parkstreet` and hence if the IWF database acquired a URL to be blocked of the form `/directions/etc/parkstreet` then the same attack could be performed.

At present, such blocking would merely be unfortunate, with a `404` being returned. However, if the CleanFeed system was modified in future to display information about the type of page it believed it was blocking (as the Telenor/KRIPOS system does in Norway), then there could be grounds for an action for defamation.

The countermeasure is to ensure that whenever a DNS lookup yields new IP addresses they are checked for accuracy. However, if many IP address changes are being made by content providers in the hope of avoiding CleanFeed blocking altogether, it will become too expensive to manually check every change. Automated processes will be required for the testing that determines whether the content is the same but accessed via a different address. Unfortunately, an automated process cannot be relied upon to distinguish between an illegal website changing both content and IP address, and a spuriously supplied IP address. The system tuning is likely to be set to avoid illegal material becoming available. Hence, automation could lead to CleanFeed's users finding some legitimate sites blocked and this in turn would lead to a devaluing of the system's reputation.

7.6 Real-world experiments

I have extensively discussed the attacks that might be made on the effectiveness or integrity of the CleanFeed system – and then explained how they might be countered. It would clearly be useful to determine which of the attacks are effective in practice and which are defeated, either because the CleanFeed system already contains a countermeasure or because of some other aspect of the design that is not immediately apparent. However this is not possible, as will now be explained.

7.6.1 Legal issues when experimenting upon CleanFeed

Most experiments upon the CleanFeed system would require an attempt to access the sites containing the illegal material that the system is intended to block. If an evasion method was attempted and was successful in evading the blocking, then, under UK law, a serious criminal offence would be committed by fetching indecent images of children. Although there are statutory defences to inadvertent access, these could not apply to an explicit access attempt. There are "research exemptions" in some laws, such as those applying to copyright protection mechanisms or the possession of "hacking tools", but these do not apply when considering access to child pornography.

Experimenting with the techniques available to a content provider would involve working with the providers of illegal content, which would be ethically questionable, even if it was not directly a criminal offence. Even demonstrating that the IWF's access was easy to distinguish (a pre-requisite for some of the attack techniques) would involve submitting a false report and thereby wasting some of their analysts' time by causing them to examine websites unnecessarily, which is undesirable.

There is a method by which experimentation could be done, without these legal and ethical problems. If a test site, containing completely legal images, was added to the IWF database then it would be possible to perform all the necessary experiments on user and content provider strategies – and, as countermeasures were added, assess their effectiveness. Permission to add such a site was sought, but has been refused, so the only people running experiments will be the consumers or providers of illegal content and they are unlikely to report their results.

It would also be possible to perform experiments on the system (albeit wasting some of the IWF's time) by arranging to serve illegal material from a website long enough to get it listed, but then removing it. This could be lawfully achieved by a US researcher setting up a site containing "virtual child pornography". In April 2002 in *Ashcroft v Free Speech Coalition* the US Supreme Court struck down some parts of the Child Pornography Prevention Act, particularly those relating to visual depictions of minors engaged in sexually explicit conduct, but where no actual child was involved – for example if the images are computer generated [133]. Such images are illegal in the UK, and the IWF would add a site that hosted them into their database, but no offence would be committed in the US by the US researcher hosting the site. Once the website had been placed in the IWF database and blocked by the CleanFeed system, the images could be replaced by innocuous material and experiments performed by UK-based researchers without committing an offence. However, there are obvious risks to the reputation of researchers who experimented in this way and currently no assistance from the US looks likely to be forthcoming.

Nevertheless, it was possible to experimentally demonstrate – in an entirely lawful manner – that a user can exploit CleanFeed to construct lists of illegal websites. This is undesirable and unexpected, and should be taken into account when discussing the public policy issue of whether to encourage this method of content-blocking.

7.6.2 Using CleanFeed as an oracle to list illegal websites

The CleanFeed system redirects traffic for particular IP addresses to a web proxy that then determines whether a particular URL should be blocked. It is possible to detect the first stage action, construct a list of redirected IP addresses, and to then determine which websites are located at those IP addresses – and hence use the system as an *oracle*[4] for locating illegal images.

The list of redirected IP addresses is created by a special scanning program. This sends out multiple TCP packets, each to a different address, with the destination port set to `80` and a TTL (time-to-live) value that is sufficient to reach the CleanFeed web proxy, but insufficient to reach the destination IP address (thus it will not unnecessarily trip intrusion detection systems at a remote site). If the IP address is not being redirected then the TTL will be decremented to zero by an intermediate router that will report this event via ICMP. If the IP address is being redirected then the packet will reach the web proxy. If the outgoing packet is sent with a SYN flag then the web proxy will respond with a packet containing SYN/ACK (the second stage of the TCP three-way handshake) and forging the IP address of the destination site. If the IP address is sent without a SYN flag then the proxy should respond with a packet with the RST flag set (because there is no valid connection).

The program was instructed to scan a /24 subnet (256 addresses) of a Russian web-hosting company (of ill-repute), with the results shown in Figure 7.2. Note that for this scan the SYN bit was set in the outgoing packets; when the SYN bit was absent the same pattern was visible but the RST packet was discarded by a local firewall!

These results (the high order octets of the IP addresses have been intentionally suppressed) show responses of either an ICMP packet (for TTL expired) from one of BT's routers in their 166.49.168/24 subnet, or a SYN/ACK packet, apparently from the remote site, but in reality from the CleanFeed web cache machine. The results clearly show that the CleanFeed system is intercepting traffic to a number of websites hosted at the Russian supplier. The full results show a total of seventeen IP addresses being redirected to the web cache.

Of course, knowing the IP address of a website does not allow one to view the content (unless it is using HTTP/1.0 or the server selects one main site to serve when just the IP address is present). However, reverse lookup directories exist that provide a mapping from IP address to web server name (they are constructed by

[4] *oracle* is being used in the sense of Lowe [95] as a system that will accurately answer any number of questions posed to it without regard to the consequences.

```
17:54:27  Starting scan of [~~~.~~~.191.0] to [~~~.~~~.191.255] (TTL 8)
17:54:27  Scan: To [~~~.~~~.191.0]  : [166.49.168.13], ICMP
17:54:27  Scan: To [~~~.~~~.191.1]  : [166.49.168.5],  ICMP
17:54:27  Scan: To [~~~.~~~.191.2]  : [166.49.168.5],  ICMP
17:54:27  Scan: To [~~~.~~~.191.3]  : [166.49.168.5],  ICMP
17:54:27  Scan: To [~~~.~~~.191.4]  : [166.49.168.9],  ICMP
17:54:27  Scan: To [~~~.~~~.191.5]  : [166.49.168.9],  ICMP
17:54:27  Scan: To [~~~.~~~.191.6]  : [166.49.168.13], ICMP
17:54:27  Scan: To [~~~.~~~.191.7]  : [166.49.168.13], ICMP
17:54:27  Scan: To [~~~.~~~.191.8]  : [166.49.168.13], ICMP
17:54:27  Scan: To [~~~.~~~.191.9]  : [166.49.168.5],  ICMP
17:54:27  Scan: To [~~~.~~~.191.10] : [166.49.168.9],  ICMP

... and similar responses until

17:54:28  Scan: To [~~~.~~~.191.37] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.38] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.39] : [166.49.168.1],  ICMP
17:54:28  Scan: To [~~~.~~~.191.40] : [~~~.~~~.191.40], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.41] : [166.49.168.13], ICMP
17:54:28  Scan: To [~~~.~~~.191.42] : [~~~.~~~.191.42], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.43] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.44] : [166.49.168.5],  ICMP
17:54:28  Scan: To [~~~.~~~.191.45] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.46] : [166.49.168.13], ICMP
17:54:28  Scan: To [~~~.~~~.191.47] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.48] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.49] : [~~~.~~~.191.49], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.50] : [~~~.~~~.191.50], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.51] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.52] : [166.49.168.5],  ICMP
17:54:28  Scan: To [~~~.~~~.191.53] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.54] : [166.49.168.5],  ICMP
17:54:28  Scan: To [~~~.~~~.191.55] : [~~~.~~~.191.55], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.56] : [166.49.168.1],  ICMP
17:54:28  Scan: To [~~~.~~~.191.57] : [166.49.168.5],  ICMP
17:54:28  Scan: To [~~~.~~~.191.58] : [166.49.168.1],  ICMP
17:54:28  Scan: To [~~~.~~~.191.59] : [166.49.168.1],  ICMP
17:54:28  Scan: To [~~~.~~~.191.60] : [166.49.168.13], ICMP
17:54:28  Scan: To [~~~.~~~.191.61] : [166.49.168.1],  ICMP
17:54:28  Scan: To [~~~.~~~.191.62] : [~~~.~~~.191.62], SYN/ACK
17:54:28  Scan: To [~~~.~~~.191.63] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.64] : [166.49.168.5],  ICMP
17:54:28  Scan: To [~~~.~~~.191.65] : [166.49.168.9],  ICMP
17:54:28  Scan: To [~~~.~~~.191.66] : [~~~.~~~.191.66], SYN/ACK
17:54:29  Scan: To [~~~.~~~.191.67] : [166.49.168.13], ICMP
17:54:29  Scan: To [~~~.~~~.191.68] : [166.49.168.13], ICMP
17:54:29  Scan: To [~~~.~~~.191.69] : [166.49.168.1],  ICMP
17:54:29  Scan: To [~~~.~~~.191.70] : [166.49.168.9],  ICMP
17:54:29  Scan: To [~~~.~~~.191.71] : [166.49.168.13], ICMP
... etc.
```

Figure 7.2: Results of scanning for IP addresses redirected by CleanFeed

resolving entries from the list of top level domain names). One such directory is sited at `whois.webhosting.info` and this was used to check out the IP addresses that CleanFeed was blocking.

Typical results (again there has been some intentional obfuscation) were:

```
~~~.~~~.191.40   lolitaportal.****
~~~.~~~.191.42   no websites recorded in the database
~~~.~~~.191.49   samayhamed.****
~~~.~~~.191.50   amateurs-world.****
                 anime-worlds.****
                 boys-top.****
                 cute-virgins.****
                 cyber-lolita.****
                 egoldeasy.****
                 elite-sex.****
                 ... and 26 more sites with similar names
```

and in total there were 91 websites on 9 of the 17 IP addresses. No websites were reported as using the other 8 IP addresses that were being blocked. This may be because the content has moved and the IWF have yet to update their information, or it may be because they were sites hosted in other top level domains, such as `.ru`, that the reverse lookup database does not currently record.

Checking the other IP addresses, not blocked by CleanFeed, showed a higher proportion of nil returns, but similar looking names. It is not possible to say whether these sites are innocuous or just not known to the IWF at present.

For the reasons explained above, *none* of these sites have been examined to determine what sort of content they actually contain, but it is fairly clear that if one was deliberately setting out to view illegal material then the CleanFeed system provides a mechanism that permits one to substantially reduce the effort of locating it. Further, since domain names can misrepresent the content (purveyors of pornography do not follow a truth-in-advertising code) it permits such a viewer to weed out superficially alluring website names and only select the ones that the IWF has already determined will contain illegal material.

Experiments showed that scans could be conducted at rates of up to 98 addresses per second using a simple dial-up connection. With this level of performance (and a broadband user could scan far faster) it would take 500 days to examine the entire 2^{32} address space – or, more realistically, 160 days to scan the 32% of the address space currently routable.[5] To scan just Russian IP addresses (and the IWF claim that 25% of all the websites they know of are located in Russia) then this is approximately 8.3 million addresses, which would take just under 24 hours. A suitable "BT Yahoo!" dial-up account that is filtered by CleanFeed is available for free and the phone call will cost less than £15.

[5]source: `http://www.completewhois.com/statistics/index.htm`

7.6.3 Countering the oracle attack

The oracle attack described in the previous section works by determining the path the packets take towards their destination. It is hard to counter in practice. The packets being sent by the end user can be made indistinguishable from normal TCP traffic – so they cannot just be discarded by a simple packet filtering system. The responses are either ICMP packets or SYN/ACK packets; again the latter must be permitted to pass, so discarding the former would not do anything especially useful.

If a web proxy is deployed in the network before the first stage at which a routing decision is made (which currently seems to be the case with the "BT Click" pay-as-you-go connectivity product) then the oracle attack fails (the web proxy treats all the packets the same, whether or not they will be candidates for redirection). However, this is an expensive fix, and BT have been removing compulsory (transparent) web caches from their products for marketing reasons.

The scanning attack is defeated if the first stage proxy does not redirect the packets to the web proxy unless their TTL setting is sufficient to reach the remote site. However, this would be complex to configure and would require specialised hardware, rather than standard routers running standard implementations of BGP. Even with this fix, it would almost certainly still be possible to distinguish web cache responses by examining the detail of what was returned.[6]

An alternative approach is to make the scan less accurate. If the CleanFeed system redirected traffic destined for more IP addresses than the minimum necessary, then the scan results would contain even more innocuous websites than at present. It may be entirely practical to redirect /24 subnets rather than individual /32 addresses, the only question being whether or not there would be a substantial increase in traffic to the web caches.

Another way of reducing accuracy would be to make the first stage redirection less predictable by introducing a statistical element. If sites were sometimes blocked and sometimes not, then the scan would take longer to be sure of its results. However, this might not be a viable option with existing equipment and it is rather perverse to defend a blocking system against attack by arranging that sometimes it fails to operate.

The easiest way of dealing with the oracle attack would be to detect it occurring, most simply by examining logs at the web proxy, and then treating it as "abuse" and disconnecting the customer. It would probably take an attacker some time (and a number of terminated accounts) to determine how to reduce the activity sufficiently to avoid being detected.

[6]The text of this paragraph is the original version that I wrote in early 2005 and which was presented at the PET workshop. See Section 7.7 below for more about how the first stage might deal with packets with unusual TTL values.

7.7 Brightview's WebMinder system

This chapter is an extended form of a paper presented at the 2005 PET Workshop [36]. This paper came to the attention of Brightview (a subsidiary of Invox plc) who announced [24] shortly afterwards that the oracle attack it describes was also effective against "WebMinder", their own two-stage content filtering system, used by the UK ISPs that they operate. I had not been aware of this system until I saw their announcement. Their design is architecturally similar to that of CleanFeed, but they are employing Cisco's proprietary Web Cache Communication Protocol version 2 (WCCPv2) to redirect suspect traffic to a number of patched `squid` proxy servers.

In their announcement, Brightview also claimed that although their system had been vulnerable, they had now made the oracle attack "no longer effective". What they had done was to change stage one of the system to discard all packets with a TTL of less than 24. This means that the scanning program has to use higher TTLs; and hence both the web proxy and remote sites will receive the packets and return SYN/ACK responses – and, it was claimed, that would prevent the two sites from being distinguished.

It is true that the exact method of attack described above is defeated (and was achieved with just a one line change to the WCCPv2 configuration). It is also true that the fix is rather more elegant than just described in Section 7.6.3 which was envisaged to involve using different TTL limits for every possible destination. Nevertheless, as I had predicted, it remains straightforward to distinguish the web proxy from the remote site whose content it is filtering. The simplest technique I have found so far is to send the scans with a high TTL (such as 128)[7], to evade the countermeasure, and then examine the TTL in the returned packets.

Consider this subset of the results from scanning the /24 subnet to which the Russian sites listed above have now moved (with some other internal renumbering):

```
Scan: To [~~~.~~~.234.51] : [~~~.~~~.234.51], TTL=49 RST
Scan: To [~~~.~~~.234.52] : [~~~.~~~.234.52], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.53] : [~~~.~~~.234.53], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.54] : [~~~.~~~.234.54], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.55] : [~~~.~~~.234.55], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.56] : [~~~.~~~.234.56], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.57] : [~~~.~~~.234.57], TTL=59 SYN/ACK
Scan: To [~~~.~~~.234.58] : [~~~.~~~.234.58], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.59] : [~~~.~~~.234.59], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.60] : [~~~.~~~.234.60], TTL=49 RST
Scan: To [~~~.~~~.234.61] : [~~~.~~~.234.61], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.62] : [~~~.~~~.234.62], TTL=49 RST
Scan: To [~~~.~~~.234.63] : [~~~.~~~.234.63], TTL=59 SYN/ACK
Scan: To [~~~.~~~.234.68] : [~~~.~~~.234.68], TTL=49 RST
```

[7]Setting a high TTL means that the packets will reach the hosting sites, which may detect a "port scan"; hence this attack is more "visible" than the original version.

```
Scan: To [~~~.~~~.234.69] : [~~~.~~~.234.69], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.70] : [~~~.~~~.234.70], TTL=59 SYN/ACK
Scan: To [~~~.~~~.234.71] : [~~~.~~~.234.71], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.72] : [~~~.~~~.234.72], TTL=49 RST
Scan: To [~~~.~~~.234.73] : [~~~.~~~.234.73], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.74] : [~~~.~~~.234.74], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.75] : [~~~.~~~.234.75], TTL=49 SYN/ACK
Scan: To [~~~.~~~.234.78] : [~~~.~~~.234.78], TTL=49 RST
Scan: To [~~~.~~~.234.79] : [~~~.~~~.234.79], TTL=59 SYN/ACK
```

The results show RSTs from machines that are not running web servers (and there is no response where the IP address is unused). All the other IP addresses respond with SYN/ACK, but the TTL is 59 $(64 - 5)$ for the nearby WebMinder web proxy and 49 $(64 - 15)$ for the Russian sites which were ten hops further away. In practice the Russian sites returned a range of TTL values such as 45, 46, 47 (reflecting minor network connection differences) and 113, 238 (reflecting alternative operating system choices for the initial TTL values), but the web proxy value was constant and very different from any value returned by a real site.

Clearly, there are steps that Brightview could now take to obfuscate this latest hint, and an arms race could result as ever more complex methods are used to distinguish a server running `squid` in a UK service centre from machines running many different types of web server in other countries. However, it is a general principle that, in situations like this, hiding your true nature is impossible. So the best that can be hoped for by Brightview (and indeed by BT, should they tackle this problem) is to make the oracle attack arbitrarily difficult rather than defeating it altogether.

7.8 The failings of traceability

The title of this chapter is "The BT 'CleanFeed' System and the Failings of Traceability" and this relates to fundamental problems that arise with the use of traceability by blocking systems.

CleanFeed, and other blocking systems, are designed on the premise that whenever illegal content is found on the Internet then its location can be obtained by applying tracing techniques; and thereafter access to that location can be blocked. Unfortunately, traceability does not really work like that. It is indeed possible to trace the immediate source of material at a specific time when accessed by a particular machine, but all three of the source, the time and the viewer are merely specific instances of all possible accesses. Generalising the traceability from the specific to create universal blocking is liable to failure.

The actual material can be hosted almost anywhere so that – as in the example of the "bulletproof webhosting" – knowing where it appears to be served from can be of very limited value.

The material may be at different locations at different times and so tracking its movements becomes a full-time job. This cannot be fully mechanized so that the blocking system is automatically reconfigured. There must always be an element of human judgement involved, because there is always risk that an attempt will be made to cause the system to block legitimate content or to swamp it with large amounts of legitimate traffic. The IWF's task in finding and reporting the material is already very labour intensive. They are unlikely to welcome the news that keeping tabs on the material, in the face of attacks, also involves expensive manual tasks.

Furthermore, the site may not have the same content when viewed by the blockers as by everyone else. It is usually assumed that the same content is being supplied by a website to everybody, but it is quite common for this not to occur. High-volume websites are hosted in multiple locations by companies such as Akamai, with traffic directed to a nearby, lightly-loaded, site – whose contents may be slightly out-of-date with respect to the master site. Google runs different ads for different users (viewers at MIT (`18.0.0.0/8`) see a front-page link to job ads whereas others will see a link to Google advertising programmes). Hence, having sites serve up misleading content to the IWF or Law Enforcement should be seen as relatively normal behaviour and not some exotic scheme of only theoretical interest.

Systems like CleanFeed can have some success in preventing inadvertent access to sites that are not bothered that they are being blocked. However, it is very important to understand that traceability is a complex series of deductions from the available information. It is particularly necessary to be suspicious of all of the information when, of necessity, a great deal is being provided by the people you are trying to track down. They may not only be trying to hide, but may also be trying to bring the blocking system into disrepute by causing it to become overloaded or inaccurate.

7.9 Conclusions

BT's CleanFeed was designed to be a low-cost, but highly accurate, system for blocking Internet content. At first sight it is significant improvement upon existing schemes. However, CleanFeed derives its advantages from employing two separate stages, and this hybrid system is thereby made more fragile because circumvention of either stage, whether by the end user or by the content provider, will cause the blocking to fail.

This chapter has described attacks on both stages of the CleanFeed system and set out various countermeasures to address them. Some attacks concern the minutiae of comparing URLs, while others address fundamentals of the system architecture. In particular, the CleanFeed system relies on data returned by the content provider, especially when doing DNS lookups. It also relies on the content provider returning the same data to everyone. All of this reliance upon the content providers' probity could well be entirely misplaced.

The CleanFeed design is intended to be extremely precise in what it blocks, but to keep costs under control this has been achieved by treating some traffic specially. This special treatment can be detected by end users and this means that the system can be used as an oracle to efficiently locate illegal websites. This runs counter to its high-level policy objectives.

Although legal and ethical issues prevent most experimentation at present, the attacks are extremely practical and would be straightforward to implement. If CleanFeed is used in the future to block other material, which may be distasteful but is legal to view, then there will be no bar to anyone assessing its effectiveness. It must be expected that knowledge of how to circumvent the system (for all material) will then become widely known and countermeasures will become essential.

An important general conclusion to draw from the need for a manual element in many of the countermeasures is that the effectiveness of any blocking system, and the true cost of ensuring it continues to provide accurate results, cannot be properly assessed until it comes under serious assault. Thinking of these systems as "fit-and-forget" arrangements will be a guarantee of their long-term failure.

Chapter 8

Conclusions and Future Work

In the dime stores and bus stations,
People talk of situations,
Read books, repeat quotations,
Draw conclusions on the wall.

— Bob Dylan, 1965

Traceability used to be seen as very simple. One started with an IP address, checked which ISP owned it, then asked the ISP to ascertain the user and tell that user to stop whatever it was that had drawn attention to them in the first place. If you were a lawyer or a police officer then you might ask for the user's home address so you could turn up at their front door with a writ, or perhaps a sledgehammer.

Traceability also used to be seen as a secret and magical rite by the lawyers and police, who were locating far fewer front doors than they wished, and felt that knowledge of some specialised sorcery was being withheld.

To assist them, the main UK ISPs, through LINX, wrote down the incantations being used so as to demystify the process. However, when closely examined, the data that provided the traceability was only sufficient to identify activity to the level of an account, and didn't identify the users. This suited the ISPs just fine because if the user failed to stop being troublesome then the account could be disabled. However, the lawyers, and especially the police, remained dissatisfied with the results they achieved and, at the policy level, started to concentrate on logging – perceiving that without logs there was no traceability, but often failing to see that even with logs, traceability could sometimes fail.

In Chapters 2 and 3 of this thesis I provided many examples and analysis of specific instances that underpin this view of traceability as an inexact process that regularly fails at the edge of the network. One should not be surprised that systems maintained by ISPs to provide traceability to ISP accounts become less precise once one is no longer using them for ISP purposes and start trying to trace back to actual people. Although the earlier examples I presented are not new, several of the cases in Chapter 3 are being documented for the first time, and the analysis of their meaning is one of the important contributions of this thesis.

To further emphasise that anonymity is a direct consequence of the lack of the traceability, Chapter 4 presented a novel method of stealing identity on an Ethernet. A nearby user's identity is stolen and then a highly directed denial-of-service attack, intentionally colliding with their packets, prevents their system from complaining. Hardware was constructed to demonstrate that the attack was more than a mere theoretical possibility, and email was sent that was a perfect forgery of what another machine would send. The attack is especially pernicious because it works best when the nearby user is actually present to be "framed". If they were elsewhere, with a solid alibi, then the subterfuge might be detected. Although few Ethernets today run as single collision domains, and so the attack is of limited interest on wired LANs, it is also expected to be effective on 802.11 wireless systems – which are becoming very widely deployed.

As a direct result of implementing my attack "for real", rather than relying upon a theoretical description of how it might be expected to work, I came across a previously undescribed problem with "personal firewalls". The type of identity theft I was doing should not succeed without special hardware because a machine should object when it receives packets for a connection which it knows nothing about. However, in pursuit of security-through-invisibility, the personal firewall builders have arranged to discard the unexpected packets so that no such objection is made. This is a useful data point for the argument that security measures driven more by hype than by analysis may be making matters worse.

ISPs do not, at least currently, create logs of activity on their system for the benefit of the police or lawyers. These logs have specific business purposes, a key one of which is to provide firm evidence when customers are suspected of abuse. In Chapter 5, I described a system that I developed for a UK ISP that analyses the logs from the server they provide for customers' outgoing email. Some relatively simple heuristics picked out the customers whose machines had been hijacked and were being used to send spam. This works because spammers hope to avoid detection at the destination and therefore vary the pattern of what they send in a manner that no legitimate sender would ever do. As a bonus, the system also detects email virus infections, including new outbreaks, and resource-sapping loops where email is speciously sent round in circles. Although the system provides a compelling reason for creating email server logs, the efficiency of its abuse detection means that there is no need to preserve the logs for long periods just in case third-party reports arrive later on. So the overall impact may be to reduce traceability for third parties unless they are very prompt in making their requests for information.

Tracing the origin of a particular spam email is relatively straightforward and most ISPs will deal with the reports that they receive. However, spam now comes from so many individual sources that this is not an efficient or scaleable way to suppress the volume, and traceability does not provide good solutions for finding the spammers themselves. Many people have analysed the spam problem in economic terms and come to the conclusion that making a charge for email will abolish, or at least severely limit, the sending of spam. A particularly elegant method of doing this, that works in an anonymous way so there's no need to trace any identities, is the use of "proof-of-work" systems that demonstrate the solution of computational puzzles. In Chapter 6 I analysed exactly how much work needs to be proved and showed that there is insufficient headroom between a price that will discourage spam and that which will prevent significant quantities of legitimate email from being sent.

This is an important result for the promoters of "proof-of-work" schemes to digest. It means that a spam solution based on economic principles is going to have to provide for only a subset of email carrying proof-of-work. This means in turn that, to prevent abuse, such schemes will have to permit whitelisting based upon complex identity-based mechanisms – or perhaps the chimera of "Trusted Computing" will be able to assist by attesting to the provenance of particular messages. In either case, solid cryptographic binding to content will be necessary, along with an effective traceability regime to counter stolen credentials. The complexity of applying the correct local tests to incoming email, along with the difficulty of checking for system-wide evidence of misuse, leads me to have significant doubts as to the likely success of hybrid systems.

Finally, in Chapter 7, I examined CleanFeed, a content-blocking system that has been deployed by BT in the UK. The system cunningly combines two existing approaches so as to produce a high level of accuracy while avoiding excessive expense. However, any circumvention scheme for either part of the two-stage system is sufficient to avoid the blocking. It is also possible for content providers to attack the second stage by persuading the first stage to send it extraneous traffic. In addition to all of these problems, it is possible to use the system as an "oracle" to determine which sites are being blocked. These are significant criticisms to make of a blocking system, since if it can be evaded it is a waste of money, if it can be attacked then it damages other activities, and if it can be used to construct lists of sites hosting child pornography then it has arguably made things worse, rather than better.

Many of the attacks on CleanFeed can be fixed, though some could prove to be extremely challenging. What cannot be addressed within the current ideas of traceability is that content does not necessarily have a single location nor does the provider have to give everyone the same meta-data about the location. In fact the provider does not have to provide the same content to everyone, which is a significant problem for those creating the lists of content to block; suddenly it is they who have to avoid being traced.

So what of the future?

Quite clearly there will be many more examples of traceability problems to understand, explain and perhaps even fix. Far too many of the systems that underpin traceability are far less accurate than anyone really imagines. From the RIR databases to the validity of CLI to the assumption that Ethernet MAC addresses are unique, nothing quite works in the simple and solid way that we often assume. There is a real risk of significant miscarriages of justice should traceability start being seen as "evidence" rather than "intelligence" and it is vital to educate Law Enforcement and Governments on the limitations of traceability – whilst still accepting that a great deal of the time it is perfectly possible to work out "who did that" and smash down the right door.

Quite clearly as well, there is much more that can be done to process logs, spotting spammers and hijacked machines. But we're slowly getting spam under control, so the challenge here will be to apply the same key idea – that it is really hard to make one type of traffic look like another – to new problems. At present the obvious candidate to tackle this way is the growing scourge of criminally motivated DoS and DDoS attacks. I believe that a heuristic based approach to detecting inappropriate patterns of traffic in the pre-attack phase is entirely realistic, even though the volumes of data to be processed are immensely larger than within email server logs.

My final vision of the future is the immense importance to be attached to accepting and indeed welcoming the presence of imperfection. The LINX's approach, from almost a decade ago, was that anonymity should not be accidental. However, this is a principle that is cheap to adopt when intentional anonymity is widely available. That may not always be the case.

In Code [90], Lessig argues that cyberspace does not have an inherent nature. It is what it is because the code, in software and also in hardware, gives it a particular nature. Up to now, that nature has been of freedom and limited regulation. As cyberspace takes a more central place in our society, that nature is being changed as the underlying code is changed. Ultimately, Lessig suggests, we may make choices so that cyberspace becomes even more regulated than the real world. The code could produce a panopticon within which all behaviour would be acceptable or immediately obvious and punished. If that comes to pass, then there will be no intentional anonymity and I do not find that an attractive prospect.

However, I hope that this thesis has brought something new to the debate about the properties of "code" and that is a notion of fallibility – even if the code is intended to provide traceability, even if your workstation demands your identity card each morning, even if the logging is universal and retained forever, all the evidence we have so far is that the implementation of cyberspace is bound to be significantly flawed. There will always be some anonymity where the traceability fails.

Annotated Bibliography

You've been with the professors and they've all liked your looks
With great lawyers you have discussed lepers and crooks
You've been through all of F. Scott Fitzgerald's books
You're very well read it's well known.
But something is happening here and you don't know what it is
Do you, Mister Jones?

— Bob Dylan, 1965

This bibliography has been annotated, partly because it has assisted me in remembering what was in each of the works I have cited, but mainly because it has enabled me to place relevant, but trivial, detail here – rather than breaking the flow of earlier chapters, or filling the pages with excessive numbers of footnotes.

Since almost everything is on the web these days, I have included URLs to assist readers in locating material that they might like to consult for themselves. In only a handful of cases are the documents not online, usually because the organization involved wishes to charge for access. Unfortunately, URLs rot away and lead to abandoned web servers, bankrupt companies or sometimes to shiny new content that omits the interesting features I have noted within the older documents that I viewed. If this happens to you when a URL fails to function, then a search engine will probably locate the document's new home; or maybe preservation systems like archive.org will still be functional so that you can look at the web as it used to be when I validated these URLs were all working – in August, 2005.

[1] M. Abadi, M. Burrows, M. Manasse and T. Wobber: Moderately Hard, Memory-bound Functions. In Proceedings of the 10th Annual Network and Distributed System Security Symposium (NDSS), San Diego, Feb 2003.

Proof-of-work schemes, the solving of computational puzzles, are designed to introduce a primitive form of economics into transactions. Solving the puzzle indicates that some amount of effort has been expended. The problem with traditional puzzles has been that they are CPU intensive so that purchasing a faster CPU allows one to solve more puzzles. This paper introduces puzzles that are memory access intensive – because there is far less disparity in memory speeds on current systems than in CPU clock speed. Hence these puzzles are "fairer" than the traditional form.

`http://research.microsoft.com/research/sv/sv-pubs/memory-final-ndss.pdf`

[2] L. von Ahn, M. Blum, N. J. Hopper, J. Langford: Using Hard AI Problems for Security. In E. Biham (Ed.): Advances in Cryptology – EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, LNCS 2656, Springer-Verlag, 2003, pp. 294–311.

A "CAPTCHA" is an automated test that humans can pass, but current computer programs cannot because the tests are based on, currently, unsolved Artificial Intelligence (AI) problems. The authors use CAPTCHAs for steganographic communication, arguing that either they work – and hence they have a useful communications system – or they do not, and they have advanced the AI field. The CAPTCHAs proposed in the paper are the reading of words whose letters have been distorted or the identification of images, perhaps of common animals, which are also slightly distorted. In practice, these authors claim, humans have little problem with the distortion, but computers fail miserably; although recent work [30] suggests they may have been over-optimistic.

`http://www-2.cs.cmu.edu/~jcl/papers/captcha_crypt/captcha_crypt.pdf`

[3] Y. Akdeniz: Court of Appeal Clarifies the Law on Downloading Pornography From the Web. Computer Law and Security Report, 18(6), 2002, pp. 433–435.

This article discusses the decisions of the Court of Appeal in R v Graham Westgarth Smith and Mike Jayson *(CA. [2002] EWCA Crim 683), (No 2001/00251/Y1), 7 Mar 2002. The court held that "the act of voluntarily downloading an indecent image from a web page on to a computer screen is an act of making a photograph or pseudo-photograph", viz: that it is a serious criminal offence.*

`http://www.cyber-rights.org/documents/smith_jayson.pdf`

[4] Altera Corporation: EPXA1 Development Board Hardware Reference Manual version 1.1. MNL-EPXA1DEVBD-1.1, Altera Inc., Sep 2002, 50pp.

Manual for the Excalibur FPGA development board used in Chapter 4.

`http://www.altera.com/literature/manual/mnl_epxa1_devbd.pdf`

[5] Altera Corporation: Excalibur Devices Hardware Reference Manual version 3.1. MNL-EPXA10HRM-3.1, Altera Inc., Nov 2002, 224pp.

Hardware Manual for the Altera Excalibur FPGA devices as used in [4].

`http://www.altera.com/literature/manual/mnl_arm_hardware_ref.pdf`

[6] J. P. Anderson: Computer security threat modelling and surveillance. Technical Report, James P Anderson Co., Fort Washington, Pa, 1980.

This is a very early report upon the potential of intrusion detection systems. It characterises different types of unauthorised use and then explains how audit logs can be processed, using statistical techniques, to detect these.

`http://csrc.nist.gov/publications/history/ande80.pdf`

[7] Australian Communications Industry Forum: Industry Code – Calling Number Display. ACIF C522, Feb 2003.

The Australian statutory code of practice for the handling of CLI and the associated user preferences for suppressing its display.

`http://www.acma.gov.au/acmainterwr/telcomm/industry_codes/codes/c522c.pdf`

[8] A. Back: Hashcash. hashcash.org, 1997.

Hashcash is a countermeasure to denial-of-service. A hashcash stamp is evidence that the creator has performed a parameterizable amount of computational work – by creating a partial hash collision. Back invented this concept independently of Dwork and Naor [50] and has been very successful at popularising it, so many people now use the term "hashcash" as a generic description of "proof-of-work" systems. The focus of the hashcash website is on providing working tools to use hashcash as a spam prevention measure.

`http://www.hashcash.org/`

[9] S. Banister: Bonded Sender Program Overview. White Paper, IronPort Inc., Jul 2002.

The 'Bonded Sender Program' is designed to ensure that legitimate commercial emails are not inadvertently caught in spam filters. Senders assert the legitimacy of their email by posting a financial bond. If an end user complains that a message was unsolicited, a charge is made against the bond. This market-based mechanism is intended to ensure that only legitimate email senders can afford to participate in the programme.

`http://www.bondedsender.com/media/bsp_overview.pdf`

[10] J. Bellardo, S. Savage: 802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions. Proceedings 12th USENIX Security Symposium, Washington DC, USA, 4–8 Aug 2003, USENIX Association, 2003, pp. 15–28.

This paper provides an experimental analysis of attacks on wireless LAN (802.11) availability. It proposes some low overhead mechanisms to mitigate the attacks without recourse to cryptographic authentication.

`http://www.usenix.org/publications/library/proceedings/sec03/tech/`
`full_papers/bellardo/bellardo.pdf`

[11] S. M. Bellovin: Security Problems in the TCP/IP Protocol Suite. Computer Communications Review, 19(2), Apr 1989, pp. 32–48.

This paper provides a more extensive discussion of the initial sequence number guessing attack first described by Morris [104]. It also describes source routing attacks and subversion of routing protocols at several different layers, viz: ICMP redirect, RIP and EGP. There is also a discussion of vulnerabilities in a number of well-known protocols, and an explanation of how authentication and encryption are generic types of defence. Fifteen years later, Bellovin published a commentary on this paper at ACSAC 2004 [15].

`http://www.cs.columbia.edu/~smb/papers/ipext.pdf`

[12] S. Bellovin: Defending Against Sequence Number Attacks. RFC1948, IETF, May 1996.

This informational RFC revisits Morris's sequence number guessing attack [104], eschewing the randomness proposed in [11], opting instead for a scheme based on the original ideas of stepping the sequence number every 4ms, but using a hash function to ensure that the base values for every host/port are unlinkable. The advantage of this scheme is that existing mechanisms for dealing with stale packets from the previous instantiation of a connection are correctly dealt with by a reincarnated version of the connection.

`http://www.ietf.org/rfc/rfc1948.txt`

[13] S. Bellovin, M. Leech, T. Taylor: ICMP Traceback Messages. Internet Engineering Task Force (IETF) Draft, Jul 2001.

Internet Drafts are "works in progress" and it is usually unwise to reference them. However, in this case the best available description of this traceback scheme for locating the source of DoS attacks is within this document. The original version was revised a few times and version -04 (February 2003) is archived at `http://www.ietf.org/proceedings/03mar/I-D/draft-ietf-itrace-04.txt` *but this draft has now expired and has not been replaced.*

`http://www.ietf.org/proceedings/01aug/I-D/draft-ietf-itrace-00.txt`

[14] S. M. Bellovin: A Technique for Counting NATted hosts. Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement, Marseille, France, 2002 (IMW'02). ACM Press 2002, pp. 267–272.

This paper proposes a method of detecting how many hosts are behind a NAT device by examining the ID field in IP headers and using the observation that most operating systems just increment this from one packet to the next. This will not work so well today because many systems now scramble the ID field to prevent "idle scanning" and so more complex analysis would be necessary.

`http://www.cs.columbia.edu/~smb/papers/fnat.pdf`

[15] S. M. Bellovin: A Look Back at "Security Problems in the TCP/IP Protocol Suite". 20th Annual Computer Security Applications Conference (ACSAC), Tucson AZ, USA, Dec 2004.

This is a commentary, written 15 years later, on the author's classic 1989 paper [11]. He gives a further discussion of Morris's attack [104] and indicates the shortcoming of his own 1996 proposal [12] in that it involves significant work for machines, such as web servers that must handle a great many short-lived connections.

`http://www.cs.columbia.edu/~smb/papers/acsac-ipext.pdf`

[16] K. Belson: Citing Threats, Entrepreneur Wants to Quit Caller ID Venture. New York Times, 4 Sep 2004.

Press article giving Jason Jepson's account of the threats made against him for offering a service that would allow spoofing of caller ID. His service, provided through a start-up called Star38, has been suspended.

`http://www.nytimes.com/2004/09/04/technology/04caller.html?ex=1252123200&en=68bab740982a4cb1&ei=5088`

[17] D. Bernstein: Re: SYN Flooding [info]. Usenet Message-ID: `<1996Sep1600.47.44.4875@koobera.math.uic.edu>`, 16 Sep 1996.

SYN cookies store an encrypted version of server state in the ACK value of a SYN/ACK response. Servers can thereby handle floods of TCP initiation packets without committing resources to SYN packets that may not be genuine, but merely part of a DoS attack.

`http://cr.yp.to/syncookies.html`

[18] O. Berthold, H. Federrath, S. Köpsell: Web MIXes: A system for anonymous and unobservable Internet access. In H. Federrath (Ed.): International workshop on designing privacy enhancing technologies: design issues in anonymity and unobservability, LNCS 2009, Springer-Verlag, 2001, pp: 115–129.

This paper describes a MIX-based system for anonymous real-time Internet access.

`http://www.dbis.informatik.hu-berlin.de/~berthold/papers/webmixes.pdf`

The system has been implemented as the Java Anon Proxy (JAP), whereby a local proxy on the browsing machine connects to one of the MIX cascades and thereby achieves (a measurable amount of) anonymity. Full details of JAP, and downloadable code, are available.

`http://anon.inf.tu-dresden.de/`

[19] M. Bhattacharyya, S. Hershkop, E. Eskin: MET: An Experimental System for Malicious Email Tracking, in Proc. 2002 Workshop on New Security Paradigms (NSPW-2002). Virginia Beach, Va, ACM Press, 2002, pp. 3–10.

The Malicious Email Tracking system (MET) checks for malicious email passing in or out of a network by comparing current activity with a database of previous behaviour.

`http://www1.cs.columbia.edu/ids/publications/met-nspw02.pdf`

[20] P. Boutin: Interview with a spammer. InfoWorld, 16 Apr 2004.

A journalist interviews Scott Richter, a well known sender of bulk email (spam). It provides useful figures on how many messages Richter sends per day and what he charges for his services, which helps to validate the calculations performed in Chapter 6 of this thesis.

`http://www.infoworld.com/article/04/04/16/16FEfuturerichter_1.html`

[21] L. Bridwell: ICSA Labs 9th Annual Computer Virus Prevalence Survey. ICSA Labs, 2004.

ICSA Labs' annual Virus Prevalence Survey gathers data to measure the prevalence of computer viruses and malware in medium to large companies (more than 500 PCs, two or more LANs and at least two remote connections). The report describes the existing computer virus problem and attempts to interpret trends of virus propagation and infection vectors, and explores possible risk mitigation methods.

`http://www.trusecure.com/cgi-bin/download.cgi?ESCD=w0169&file=`
`wp_vps2003_report.pdf`

[22] M. Bright: BT puts block on child porn sites. Observer, 6 Jun 2004.

This story broke the news of the BT CleanFeed system, saying "the decision by Britain's largest high-speed internet provider will lead to the first mass censorship of the web attempted in a Western democracy". It also observes, "Blocking websites is highly controversial and until now has been associated only with oppressive regimes such as Saudi Arabia and China, which have censored sites associated with dissidents. But many in the field of child protection believe that the explosion of paedophile sites justifies the crackdown."

`http://observer.guardian.co.uk/uk_news/story/0,6903,1232422,00.html`

[23] Brightmail Inc.: Spam Percentages and Spam Categories, Jun 2004.

Brightmail Inc. operate an email filtering service. They operate a large number of dummy mailboxes and these receive unsolicited email from spammers who are send it out indiscriminately. They then construct generic filtering rules and use these to protect their customers' mailboxes. Detailed monthly statistics were published for several years at `http://www.brightmail.com/spamstats.html`*, though since the company was purchased by Symantec in June 2004 these are no longer directly available. However, the highlights of each month's figures was reported in press releases and these were widely reported in the specialist computer press and so the data has not been entirely lost to posterity. The URL provided below is just such a report, which conveniently reproduces all the tables of data from the original source.*

`http://www.clickz.com/stats/sectors/software/article.php/3379701`

[24] Brightview Internet Services Ltd: WebMinder, a configuration for restricting access to obscene sites identified by the Internet Watch Foundation. 9 Jun 2005, 21pp.

This note describes the design of WebMinder, a hybrid content blocking system. Appendix C of this note describes how the oracle attack described in my PET workshop paper [36] was defeated. Section 7.7 of this thesis explains how another attack, based on returning TTL values, still works.

`Not available online.`

[25] K. Butler, T. Farley, P. McDaniel, J. Rexford: A Survey of BGP Security Issues and Solutions. Technical Report TD-5UGJ33, AT&T Labs – Research, Florham Park, NJ, Feb 2004 (revised Jun 2004).

This paper considers the vulnerabilities of existing interdomain routing, and surveys work on BGP security. The limitations and advantages and implications of proposed solutions are explored.

`http://www.patrickmcdaniel.org/pubs/td-5ugj33.pdf`

[26] Cable Television Laboratories Inc.: Cable Modem/DOCSIS – Project Primer. CableLabs, 13 Dec 2004.

This web page is a brief introduction to DOCSIS (Data-Over-Cable Service Interface Specification) standards. Links lead to the various standards documents for multiple generations of this system for providing data access over Cable TV networks.

`http://www.cablemodem.com/primer/`

[27] H. V. Caramés, T. C. Martínez: ILLC – Inverse Lookup Log Corruption. infohacking.com Advisory, 4 Mar 2003.

This security research Advisory describes the problems that arise when reverse DNS lookup results are not sanitised before being processed by log processing systems. Five different products were broken in similar ways when reverse DNS yielded text that included HTML scripting commands.

`http://www.infohacking.com/INFOHACKING_RESEARCH/Our_Advisories/`
`ILLC/ILLC_english_provisional.doc`

[28] CERT/CC: IP Spoofing Attacks and Hijacked Terminal Connections. CERT Advisory CA-1995-01, 23 Jan 1995.

This CERT Advisory discusses attacks seen "in the wild" that used TCP connection spoofing (exploiting weaknesses in the random generation of initial sequence numbers) following by a SUN OS 4.1.x specific session hijacking attack. The anti-spoofing advice in this Advisory is superseded by CA-1996-21. It is noteworthy that it was still necessary to publish this Advisory ten years after the 1985 Morris paper [104] and six years after the more extensive analysis by Bellovin [11] in 1989.

`http://www.cert.org/advisories/CA-1995-01.html`

[29] CERT/CC: Statistical Weaknesses in TCP/IP Initial Sequence Numbers. CERT Advisory CA-2001-09, 1 May 2001.

This CERT Advisory describes the attacks that can be made if it is possible to guess the initial sequence number of a TCP connection. It then describes how a superficially plausible approach of adding random increments gives rise to the possibility of "statistical guessing" because there is insufficient variance in the summed values.

`http://www.cert.org/advisories/CA-2001-09.html`

[30] K. Chellapilla, K. Larson, P. Simard, M. Czerwinski: Computers Beat Humans at Single Character Recognition in Reading based Human Interactive Proofs (HIPs). Second Conference on Email and Anti-Spam (CEAS 2005), Stanford CA, USA, 21–22 Jul 2005.

The message is in the title! HIPs (or CAPTCHAs [2]) based on distorted characters do not give computers as much difficulty as expected – albeit this work relates to reading individual characters; the segmentation problem, of splitting a HIP image into the separate characters that need to be recognised, remains difficult at present.

`http://www.ceas.cc/papers-2005/160.pdf`

[31] S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, J. Rowe, S. Staniford-Chen, R. Yip, D. Zerkle: The Design of GrIDS: A Graph-Based Intrusion Detection System. Technical Report CSE-99-2, U.C. Davis Computer Science Department, Jan 1999, 49pp.

GrIDS is a prototype intrusion detection system that was designed to explore the issues involved in doing large scale aggregation of traffic patterns. It features a hierarchical decomposition of the protected organization and its networks. GrIDS marshals reports of incidents and network traffic, places them into graphs, and is able to aggregate those graphs into simpler forms at higher levels of the hierarchy

`http://seclab.cs.ucdavis.edu/projects/arpa/grids/grids.pdf`

[32] R. Clayton (Ed.): LINX Best Current Practice – Traceability. London Internet Exchange (LINX), Version 1.0, 18 May 1999.

This is the classic description of traceability. LINX is the premier UK Internet Exchange Point and became involved with regulatory issues almost as soon as it was created in 1994. This document was produced to explain to Law Enforcement, and to the UK government, the technical methods by which one might establish "who did that" on the Internet.

`http://www.linx.net/noncore/bcp/traceability-bcp.html`

[33] R. Clayton, G. Danezis, M. G. Kuhn: Real World Patterns of Failure in Anonymity Systems. In I. Moskowitz (Ed.): Information Hiding, 4th International Workshop, IH 2001, LNCS 2137, Springer, 2001, pp. 230–244.

This paper describes attacks on the HushMail encrypted email system and on a web-based "lonely hearts" dating service. In particular it discusses some generic attacks on the anonymity of web-based systems that try to hide the identity of their users. These are relevant to the IWF's problem of trying to access suspect websites without revealing their identity.

http://www.cl.cam.ac.uk/~rnc1/Patterns_of_Failure.pdf

[34] R. Clayton, M. Bond: Experience Using a Low-Cost FPGA Design to Crack DES Keys. In B. S. Kaliski Jr., Ç. K. Koç, C. Paar (Ed.): Cryptographic Hardware and Embedded Systems – CHES 2002, Redwood Shores CA, USA, 13–15 Aug 2002, LNCS 2523, Springer Verlag, pp. 579–592.

In 2001 I implemented a hardware 'DES cracker' on a low-cost FPGA board and used it to brute-force DES key values. This paper describes a number of insights which were gained from this practical instantiation of Mike Bond's theoretical attack on the IBM 4758 CCA.

http://www.cl.cam.ac.uk/~rnc1/descrack/DEScracker.pdf

[35] R. Clayton: Clause 53 of the Sexual Offences Bill: The problem of "making". Foundation for Information Policy Research (FIPR), 23 Mar 2003.

This briefing paper discusses the background to the state of UK law on "making" indecent images of children and proposes a way of dealing with the problems faced by sysadmins and ISP abuse teams – because they commit offences when handling reports of illegal material. In the event, the Government decided to amend the Sexual Offences Bill so as to provide a statutory defence for reporting and investigative activity.

http://www.cl.cam.ac.uk/~rnc1/SexualOffencesBill.pdf

[36] R. Clayton: Failures in a Hybrid Content Blocking System. Fifth Privacy Enhancing Technologies Workshop, PET 2005, Dubrovnik, Croatia, 30 May–1 Jun 2005.

Workshop paper based on Chapter 7 of this thesis; the main difference being that the detailed explanations of the entries in Tables 7.1, 7.2 and 7.3 are omitted.

http://www.cl.cam.ac.uk/~rnc1/cleanfeed.pdf

[37] R. Clayton: Stopping Outgoing Spam by Examining Incoming Server Logs. Second Conference on Email and Anti-Spam (CEAS 2005), Stanford CA, USA, 21–22 Jul 2005.

This paper describes how processing the logs of incoming email at an ISP's email server can detect the small number of items of spam (or email viruses) being sent by one ISP customer to another. It also provides an estimate of how effective the method is by assessing the worldwide prevalence of such email. It is a more recent extension of the work presented in Chapter 5 of this thesis on examining the logs for outgoing email, using very much the same type of heuristics.

http://www.cl.cam.ac.uk/~rnc1/incoming.pdf

[38] S. Cobb: The Economics of Spam. ePrivacy Group, Feb 2003.

This is a discussion of the "parasitic economics of spam", viz: that the sender of spam is paying less to send it than everyone else in the delivery chain is paying to handle it. The document very helpfully collects together some figures on the actual profit margins for some items that are regularly advertised by spam, which provided useful figures for the calculations in Chapter 6 of this thesis.

`http://www.spamhelp.org/articles/economics_of_spam.pdf`

[39] F. Cohen: 50 Ways to Defeat Your Intrusion Detection System. Fred Cohen Associates, Dec 1997.

Cohen lists rather more than fifty ways that an Intrusion Detection System will fail to spot attacks on networked machines. The main theme is that these systems fail to map actual attacks into a canonical version of what is occurring and therefore fail to match the patterns within their databases.

`http://all.net/journal/netsec/1997-12.html`

[40] Commonwealth of Australia: Privacy Act 1988.

Australian legislation on privacy. The URL links to the current version of the Act, including all subsequent amendments.

`http://scaleplus.law.gov.au/html/pasteact/0/157/pdf/Privacy1988.pdf`

[41] A. Cormack: Logfiles. JANET guidance note GD/NOTE/008, Jun 2004, 21pp.

A relatively recent description of the rôle of logging information in tracking down abuse. It contains a number of worked examples.

`http://www.ja.net/services/publications/technical-guides/gn-logfiles.pdf`

[42] Court of Appeal: R v Jonathan Bowden, [2001] Q.B. 88; [2000] 2 W.L.R. 1083; [2000] 2 All E.R. 418; [2000] 1 Cr. App. R. 438; [2000] 2 Cr. App. R. (S.) 26; [2000] Crim. L.R. 381. Judgment date, 10 Nov 1999.

In the Bowden case, indecent images of children had been downloaded from the Internet, stored on a computer's hard disk and also printed out. The defendant submitted that he was not guilty of "making" but only of "possessing" the material, a much less serious offence. The trial judge, held that a "making" offence, within the meaning of s.1(1)(a) of the Protection of Children Act 1978, had been committed and Bowden then pleaded guilty. The original judgment was appealed, but the Appeal Court agreed that the original decision was correct. However, the higher court did quash Bowden's four month sentence as being excessive.

`http://www.geocities.com/pca_1978/reference/bowden2000.html`

[43] D. Crocker: Mailbox Names for Common Services, Roles and Functions. RFC2142, IETF, May 1997.

This standards-track RFC (probably better perceived of as a BCP) collected together the various "well-known" mailbox names which are used to reach operational "rôles" within an organization. In particular, it first documented the de facto *standard of using* `abuse@` *to reach the personnel who dealt with abuse issues.*

`http://www.ietf.org/rfc/rfc2142.txt`

[44] CTV: Police warn of Wi-Fi theft by porn downloaders. 23 Nov 2003.

This is the CTV.ca report on a Toronto police press conference announcing the arrest of a paedophile who was caught whilst out "war-driving" to steal Internet access to illegal material via insecure consumer wireless access points.

`http://www.ctv.ca/servlet/ArticleNews/story/CTVNews/1069439746264_64848946`

[45] D. E. Denning: An Intrusion-Detection Model. IEEE Trans. on Software Engineering, SE-13(2), Feb 1987, pp. 222–232.

This paper presents a model of a real-time intrusion detection expert system capable of detecting break-ins, penetrations, and other forms of computer abuse. The model is based on the hypothesis that security violations can be detected by monitoring a system's audit records for abnormal patterns of system usage.

`http://www.cs.georgetown.edu/~denning/infosec/ids-model.rtf`

[46] R. Dingledine, N. Mathewson, P. Syverson: Tor: The Second-Generation Onion Router. In: Proceedings 13th USENIX Security Symposium, Aug 2004, pp. 303–320.

Tor is a distributed overlay network for anonymising TCP based applications such as web browsing. It provides a low-latency service and has been designed to be extremely easy to deploy and use. There is a working network which was about 30 nodes when this paper was written, but it has now (mid-2005) grown to about 200 nodes.

`http://freehaven.net/tor/tor-design.pdf`

[47] M. Dornseif: Government mandated blocking of foreign Web content. In J. von Knop, W. Haverkamp, E. Jessen (Ed.): Security, E-Learning, E-Services: Proceedings of the 17. DFN-Arbeitstagung über Kommunikationsnetze, Düsseldorf 2003, Lecture Notes in Informatics, pp. 617–648.

This paper looks at the blocking being performed by ISPs in the North-Rhine-Westphalia region of Germany. The material being blocked relates to the Nazi party – content which is illegal under German law. This paper discusses the wording of the legal order, which makes little technical sense, and observes that the major method used by the ISPs to block the sites is DNS poisoning. In practice, a number of errors have been made and hence web pages that should be blocked are not (44% error rate for one example) and pages that should not be blocked are blocked (56% error rate for one example).

`http://md.hudora.de/publications/200306-gi-blocking/200306-gi-blocking.pdf`

[48] R. Droms: Dynamic Host Configuration Protocol. RFC2131, IETF, Mar 1997.

This RFC describes the basic DHCP protocol used to pass configuration information to hosts, thereby avoiding the necessity to configure them all individually. A number of extensions to the original scheme have been developed down the years, so to fully understand the protocol it would be necessary to also read RFC2132, RFC2241, RFC2242, RFC2485, RFC2563, RFC2610, RFC2855, RFC2937, RFC2939, RFC3004, RFC3011, RFC3046, RFC3118, RFC3203, RFC3256, RFC3361, RFC3396, RFC3397, RFC3442, RFC3456, RFC3495, RFC3527, RFC3594, RFC3634, RFC3679, RFC3825, RFC3925, RFC3942, RFC3993, RFC4014, RFC4030 and RFC4039 – and doubtless more by now!

`http://www.ietf.org/rfc/rfc2131.txt`

[49] C. Dwork, A. Goldberg and M. Naor: On Memory-Bound Functions for Fighting Spam. In D. Boneh (Ed.): Advances in Cryptology – CRYPTO 2003, LNCS 2729, Springer Verlag 2003, pp. 426–444.

This paper looks at proof-of-work functions that are limited by memory speed rather than processor cycle time, as first proposed by Abadi et al. *[1]. The authors provide a formal model of the problem and an abstract function, then develop concrete instantiations and give experimental results of the different levels of performance achieved. They conclude that the fastest and slowest implementations differ only by a factor of four – far less than a CPU-bound system would manage.*

`http://research.microsoft.com/research/sv/PennyBlack/demo/lbdgn.pdf`

[50] C. Dwork, M. Naor: Pricing via Processing or Combatting Junk Mail. In E. F. Brickell (Ed.): Advances in Cryptology – CRYPTO '92, LNCS 740, Springer Verlag 1992, pp.139–147.

This paper was the first to propose a proof-of-work scheme for fighting email spam. Various possible moderately hard functions are considered for the proof-of-work rôle including calculating square roots and recycling various broken cryptographic signature schemes (since there is still significant computation in breaking them). The authors also propose that a centrally held "shortcut" should be created that which would permit authorised senders to despatch email in bulk without the necessity to perform all the computations. This latter feature has been ignored by later work, with whitelist schemes providing this functionality.

`http://www.wisdom.weizmann.ac.il/~naor/PAPERS/pvp_abs.html`

[51] S. T. Eckmann, G. Vigna, R. A. Kemmerer: STATL: An Attack Language for State-based Intrusion Detection. J. Computer Security, 10(1/2), 2002, pp. 71–104.

STATL is an extensible state/transition-based attack description language designed to support intrusion detection. The language allows one to describe computer penetrations as sequences of actions that an attacker performs to compromise a computer system.

`http://www.cs.ucsb.edu/~vigna/pub/2000_eckmann_vigna_kemmerer_statl.pdf`

[52] B. Edelman: Web Sites Sharing IP Addresses: Prevalence and Significance. Berkman Center for Internet and Society, Feb 2003.

Websites are seldom hosted on a machine that is dedicated solely to supporting a single site; for economic reasons, the vast majority share a machine with many other websites. With the advent of HTTP/1.0 these websites can also share a single IP address; rather than having the machine support multiple addresses. This report summarises the results from a study of the IP addresses associated with websites of the form `www.domain` *for all* `domain` *names within the top level domains of* `.org`, `.com` *and* `.net`*. Edelman shows that 87.3% of such websites share an IP address with one or more other websites, and 69.8% with 50 or more other sites. He is particularly interested in the effect of this IP address sharing on content suppression – and he concludes that there is a significant risk of "overblocking" with schemes that suppress undesirable content by methods based solely on IP address values.*

`http://cyber.law.harvard.edu/people/edelman/ip-sharing/`

[53] K. Egevang, P. Francis: The IP Network Address Translator (NAT). RFC1631, IETF, May 1994.

This RFC describes how NAT works, translating IP addresses and port numbers so as to permit multiple hosts to share IP addresses. It also describes the issues that arise with FTP and ICMP, and hints that other protocols will have problems with the breaking of the end-to-end nature of Internet communications.

`http://www.ietf.org/rfc/rfc1631.txt`

[54] H. Eidnes, G. de Groot, P. Vixie: Classless IN-ADDR.ARPA delegation. RFC2317, IETF, Mar 1994.

This RFC describes an extension to the classic delegation of `in\_addr.arpa` *described in [102] for networks that are smaller than a /24.*

`http://www.ietf.org/rfc/rfc2317.txt`

[55] Electronic Frontiers Australia: Privacy invasions: Blocked Calling Number Disclosure to ISPs. EFA, 5 Jul 2003.

The EFA (a non-profit national organization representing Internet users concerned with online freedoms and rights) explain how user settings on CLI disclosure are being ignored for calls to Australian ISPs. They argue that this is unlawful and that overriding individual privacy choices is "overkill". They submit that if this disclosure is to take place, then it should be a matter for public and parliamentary decision; rather than being decided by industry association or telecommunication providers.

`http://www.efa.org.au/Issues/Privacy/cndnomand.html`

[56] Electronic Privacy Information Center (EPIC): In the Matter of DoubleClick Inc. Complaint and Request for Injunction, Request for Investigation and for Other Relief, before the Federal Trade Commission, 10 Feb 2000.

This complaint to the FTC sets out how DoubleClick Inc. uses cookies to track users as they visit large numbers of websites. The issue raised by EPIC was that specific information about individuals obtained by one of these websites could then be available to all the other websites. In the event, the FTC criticised the way DoubleClick was operating, but took no action. A later Federal court case on the topic was thrown out, but the company was forced to settle an action brought by the Attorney Generals of ten states. However, the idea of personalising banner ads wasn't cost effective and DoubleClick quietly dropped the product from their portfolio in 2002.

`http://www.epic.org/privacy/internet/ftc/DCLK_complaint.pdf`

[57] C. Feather: The Hashed URI. IETF Internet Draft, draft-feather-hashed-uri-03.txt, 17 Sep 2002.

This document proposes a method of hashing a Universal Resource Identifier (URI) into another URI that conceals the original value. This would, for example, permit the publisher of filtering software to provide a set of URIs to be filtered without identifying the actual resources in question. NB: Although it is undesirable to cite Internet Drafts, this one did not make it to an RFC, so there's no other source to reference for this idea. Fortunately the document remains available online, albeit not preserved by the IETF.

`http://waterspprings.org/pub/id/draft-feather-hashed-uri-03.txt`

[58] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, IETF, Jun 1999.

This RFC documents the current version of the HTTP protocol used for access to hypertext, viz: the content of the web.

`http://www.ietf.org/rfc/rfc2616.txt`

[59] S. Fluhrer, I. Mantin, A. Shamir: Weaknesses in the Key Scheduling Algorithm of RC4. In S. Vaudenay, A. M. Youssef (Ed.): Selected Areas in Cryptography: 8th Annual International Workshop, SAC 2001, Toronto, Canada, 16–17 Aug 2001, LNCS 2259, Springer-Verlag GmbH, 2001, pp: 1–24

Several weaknesses in the key scheduling algorithm of RC4 are presented and a large number of weak keys are identified, in which knowledge of a small number of key bits suffices to determine many state and output bits with non-negligible probability. This led on to practical related key attacks on the 802.11 wireless encryption system WEP, that uses RC4 in what turns out to be an insecure manner. Having seen this paper, Stubblefield et al. *[132] took only a week to implement the attack.*

`http://www.math.psu.edu/mathnet/mdoc/md15/rc4_ksaproc-1.pdf`

[60] M. K. Franklin, D. Malkhi: Auditable Metering with Lightweight Security. Journal of Computer Security, 6(4), 1998, pp: 237–255.

The authors propose a "proof-of-work" scheme for visitors to a website; so that the website operator will find it expensive to boost their stats by fabricating fake visits.

`http://www.cs.huji.ac.il/~dalia/pubs/meter-ftp.ps`

[61] Gandalf the White: Spam FAQ or "Figuring out fake E-mail & posts" Rev 951212. Usenet Message-ID: `<gandalf-1212951731290001@199.0.12.35>`, 12 Dec 1995.

This is the earliest known "traceability" document. Equal weight is given to heuristics, such as examining message IDs, as to the step-by-step analysis we employ today.

`http://www.google.com/groups?selm=gandalf-1212951731290001@199.0.12.35`

[62] J. Goodman and R. Rounthwaite: Stopping Outgoing Spam. ACM Conference on Electronic Commerce, EC'04, 2004.

The authors analyse the economics of sending spam through an Email Service Provider (such as Hotmail). They conclude that their aim is to make it uneconomic over the long term, so there is no need to challenge every message. The paper is especially useful because it reports upon real-world pricing for the bulk sending of spam.

`http://research.microsoft.com/~joshuago/outgoingspam-final-submit.pdf`

[63] J. Goodman: IP Addresses in Email Clients. First Conference on Email and Anti-Spam (CEAS 2004), Mountain View CA, USA, 30–31 Jul 2004.

This paper considers how to pick out the email header field that records the sender of an email. This is trivial problem for email servers at gateways to an organization, but rather more complex for clients running on hosts within the organization.

`http://www.ceas.cc/papers-2004/162.pdf`

[64] Government of India, Ministry of Communications and IT: Licence Agreement for Provision of Unified Access Services After Migration. 11 May 2004, 79pp.

The Indian Government is moving to a unified licensing scheme for telecommunications. This is the proposed licence for the new regime. It is notable because it sets out very significant restrictions on the ability of end users to restrict the sending of caller line identification (CLI).

`http://www.dotindia.com/basic/`
`FINAL%20UASL%20MIGRATION%20metro(%2011.05.2004).doc`

[65] D. J. Greaves: ATM in the Home – and – the Home Area Network. IEE Colloquium on ATM in Professional and Consumer Applications, London, May 1997.

The ATM Warren project, at the University of Cambridge, investigated the use of ATM for home networking. One of the old PCBs from this project was used for the work described in Chapter 4.

`http://www.cl.cam.ac.uk/Research/SRG/netos/han/docs/IEESavoy.ps.gz`

[66] E. Hacker: Resynchronizing NIDS systems. InFocus, SecurityFocus, 22 Sep 2000.

The author argues that an IDS should be looking for both anomalous traffic and anomalous signatures. He suggests practical ways to detect abnormal traffic by arranging for normal traffic to have unusual characteristics. For example, the case insensitivity of web servers means that directories such as `cgi-bin` *can be given an unusual combination of upper and lower case letters, that scripted attacks would be unlikely to duplicate.*

`http://www.securityfocus.com/infocus/1226`

[67] M. Handley, V. Paxson, C. Kreibich: Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics. In: Proceedings 10th USENIX Security Symposium, Washington DC, USA, Aug 2001.

This paper introduces the concept of a "normaliser" that intercepts an IP packet stream and fixes up ambiguous packet contents – that may be interpreted differently by different systems – so as to ensure that an IDS system will be operating on the same data as a host that may come under attack. A detailed list of issues is considered at the IP and TCP levels, but the technique and methodology could be applied at higher protocol stack levels as well.

`http://www.usenix.org/events/sec01/full_papers/handley/handley.pdf`

[68] S. Hansell: E-Mail Message Blitz Creates What May Be Fastest Fad Ever. New York Times, 9 Jun 2003.

News story about using spam to promote the sale to the public of "Iraqi Most Wanted" decks of playing cards – which were issued to US soldiers in Iraq to help them remember which members of Saddam's regime they were looking for. In the first wave, one spammer sent 25 million messages and received 3 164 orders (0.013%). This was apparently about "four times" their usual response rate for products like printer ink.

`http://www.nytimes.com/2003/06/09/technology/09CARD.html`

[69] P. Hazel: The Exim SMTP Mail Server, Official Guide for Release 4, UIT Cambridge, 2003, 620pp, ISBN 0-954-45290-9.

Definitive book on the operation of the Exim email server, written by its author. The book itself is not online, the URL given here leads to reference material, such as "The Exim Specification" and "FAQ for Exim 4", that supplements the more expository material in the "Official Guide".

`http://www.exim.org/docs.html`

[70] Her Majesty's Stationery Office: Protection of Children Act 1978.

The PCA 1978 made it an offence to take, distribute, show or advertise indecent photographs of children under 16. It was amended by the Criminal Justice Act 1998 to cover possession. It was further amended by the Criminal Justice and Public Order Act 1994 to also cover "making" and "pseudo-photographs" and by the Sexual Offences Act 2003 to raise the age limit to 18 and provide a specialist defence for possession for the activity concerned with criminal investigations. Other amendments raised the tariffs for various offences. Acts of UK Parliament passed before 1988 are not currently online at official sites, so the URLs provided are to a privately run site that hosts copies of the initial 1978 Act and its current, amended, form.

`http://www.geocities.com/pca_1978/reference/pca_1978unam.html`

`http://www.geocities.com/pca_1978/reference/pca_1978amSOA.html`

[71] D. C. Hewes: I Can See You Behind Layer 2... Overcoming the difficulties of Packet Capturing on a Switched Network. SANS Institute, 21 May 2003.

This paper, produced as a practical assignment for SANS/GIAC certification, provides an overview of techniques for sniffing Ethernet traffic on a switched network.

`http://www.giac.org/certified_professionals/practicals/gsec/0112.php`

[72] The Honeynet Project and Research Alliance: Know Your Enemy: Tracking Botnets. 13 Mar 2005.

This "whitepaper" describes the way in which botnets are controlled through the use of commands issued on IRC channels.

`http://www.honeynet.org/papers/bots/`

[73] A. Householder, B. King, K. Silva: Securing an Internet Name Server. CERT Coordination Center, Aug 2002.

This document, based on a presentation by the DNS expert Cricket Liu, sets out, in quite considerable detail, the steps that need to be take to ensure the integrity of a DNS server such as BIND, Microsoft DNS or `djbdns` *and hence the authenticity of the data that it provides to requesters. Quite clearly, if a DNS server is providing incorrect information then there is significant potential to not only disrupt access to Internet content but also to impact the ability to provide reliable traceability.*

`http://www.cert.org/archive/pdf/dns.pdf`

[74] IEEE Computer Society: 802.3 IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements, Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications, The Institute of Electrical and Electronics Engineers, Inc., 8 Mar 2002, 1538pp.

This is the current version of the standard for Ethernet systems. As can be seen from the length of the document, Ethernets are extremely complex systems! although some of the length does result from describing systems that work at a number of different speeds, from 10 Mbit/s up to 1 Gbit/s. Since the standard is more than six months old it is available for free download.

http://standards.ieee.org/getieee802/802.3.html

[75] Intel Corporation: LXT901/901 Universal 10BASE-T and AUI Transceivers. Datasheet, Order Number: 249097-002, Intel Inc., Jun 2001.

This is the datasheet for an Intel PHY chip. This was the particular PHY that was mounted on the "Warren" PCB that was used in the Ethernet collision experiment described in Chapter 4.

http://www.intel.com/design/network/products/LAN/datashts/24909702.pdf

[76] Inter-departmental Working Group on Computer Related Crime: Report. Security Bureau, Hong Kong, Sep 2000, 138pp.

This report deals with many issues related to cybercrime and to what extent Hong Kong's legal framework would be in conformance with the (at that time, still draft) Treaty on Cybercrime. There is a long section discussing the pros and cons of mandatory recording of CLI on dial-up access to the Internet, with the conclusion being that it should not be insisted upon.

http://www.hkispa.org.hk/pdf/ComputerRelatedCrime.pdf

[77] Internet Assigned Numbers Authority: Internet Protocol v4 Address Space. Constantly revised document, last updated 30 Jun 2005.

This document is the master list of IPv4 address space assignment. There is an equivalent document for IPv6.

http://www.iana.org/assignments/ipv4-address-space

[78] Executive Committee of ISPA – Internet Services Providers Association, LINX – London Internet Exchange, The Safety-Net Foundation: R3 Safety-Net: Rating, Reporting, Responsibility for Child Pornography & Illegal Material on the Internet. 23 Sep 1996.

This document presents a UK industry proposal for tackling illegal material on the Internet. The Rating part has never been seriously tackled, although several Government departments and some ISPs have ICRA http://www.icra.org/ *labels for their websites. The Reporting side was to set up the Internet Watch Foundation (IWF) to operate a hotline and relay reports to ISPs. The Responsibility section included the notion that "truly anonymous" Internet access accounts were "a danger" and that anonymous remailers operated by the ISPs (these did not exist then and still do not) should record* *details of the identity of users for access by the police. In practice, what the document said has always been far less important than the consensus it represents around the joint ownership between Industry, Government and Law Enforcement, of the problem of trying to police the Internet.*

`http://dtiinfo1.dti.gov.uk/safety-net/r3.htm`

[79] Internet Service Providers Association: ISPA statement on NHTCU request for ISP cooperation with US Atrocity investigations. 13 Sep 2001.

This is a brief statement from ISPA recording that the National Hi-Tech Crime Unit (NHTCU) had requested preservation of communications data (email logs etc.) in case it would help assist investigations in the events of the 11th September 2001. The initial request was for the logs to be preserved for a month, which was eventually extended to four months. The Information Commissioner had been consulted and advised, "the request from the NHTCU is lawful and proportionate in the circumstances".

`http://www.ispa.org.uk/html/media/statementdp1309.htm`

[80] Internet Systems Consortium: Internet Domain Survey, 2004.

Every six months, the ISC attempt to count how many hosts are connected to the Internet. It does this by seeing how many IP addresses are referred to within the global DNS system. In January 2004 this total was 233 101 481.

`http://www.isc.org/ops/ds/reports/2004-01/`

[81] Internet Watch Foundation: Annual Report 2003. 22 Mar 2004.

The IWF annual report provides statistics on the number of reports made to them of illegal material via the public hotline. Unfortunately, they do not seem to have an especially future-proof method of providing online access to the document.

`http://www.iwf.org.uk/documents/20050221_annual_report_2003.pdf`

[82] M. Jakobsson and A. Juels: Proofs of Work and Bread Pudding Protocols. In Proceedings of the IFIP TC6 and TC11 Joint Working Conference on Communications and Multimedia Security (CMS '99). Kluwer, 1999.

A "bread pudding protocol" is a proof-of-work scheme where the computational effort invested in the proof may be reused by the verifier to achieve a separate, useful, and verifiably correct computation. The paper develops an example using Rivest and Shamir's MicroMint scheme.

`http://www.rsasecurity.com/rsalabs/node.asp?id=2049`

[83] J. Jordaan: Pine setup on Windows. Usenet article ID: `<Pine.WNT.4.21.0005160950270.1028-100000@roamer>`, 16 May 2000.

In this article Ms Jordaan poses a question about Pine passwords, but, wishing to provide all possible information to those who might help her, she posted the contents of her `pine.pwd` file. Unfortunately, the contents are only lightly encrypted (obscured might be a better term) so that anyone who read the article would have been able to obtain her passwords. The error was pointed out about two and half hours later.

```
http://www.google.com/groups?
      selm=Pine.WNT.4.21.0005160950270.1028-100000@roamer
```

[84] A. Juels and J. Brainard: Client puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. In S. Kent (Ed.): Proceedings of NDSS '99 (Networks and Distributed Systems Security), 1999, pp. 151–165.

This paper proposes the use of a proof-of-work system as a way for a server to control the rate of creation of new TCP/IP connections by clients, so as to ameliorate denial-of-service attacks.

`http://www.rsasecurity.com/rsalabs/node.asp?id=2050`

[85] King Abdulaziz City for Science and Technology: Local Content Filtering Procedure. Internet Services Unit, KACST, 2004.

This web page describes the content filtering system for the Kingdom of Saudi Arabia as a "proxy farm system implementing a content filtering software". The system has a list of banned sites which is updated daily. The base list comes from the software provider but, "this list is not comprehensive due to the high proliferation and diversity of pornographic sites. Therefore, KACST maintains a web-based form that users can fill-out to report sites they feel should be blocked. Hundreds of requests are received daily. A team of full-time employees at KACST study these requests and implement them only if justified. As for non-pornographic sites, KACST receives orders to block them from related government bodies (refer to content of filtering policy)."

`http://www.isu.net.sa/saudi-internet/contenet-filtring/`
`filtring-mechanism.htm` [1]

[86] J. Klensin (Ed.): Simple Mail Transfer Protocol. RFC2821, IETF, Apr 2001.

This RFC updates the classic RFC821 to describe the current version of the SMTP protocol used for the transfer of email across the Internet.

`http://www.ietf.org/rfc/rfc2821.txt`

[87] T. Kohno, A. Broido, k. claffy: Remote physical device fingerprinting. IEEE Transactions on Dependable and Secure Computing, 2(2), 2005.

The authors measure clock skews (the rate of change of their clock values with respect to the correct time) on remote devices and use this as a fingerprinting method. One application, of many possible uses, is the counting of hosts behind a NAT device.

`http://www.cse.ucsd.edu/users/tkohno/papers/PDF/`

[88] B. Laurie, R. Clayton: Proof-of-Work Proves Not to Work. Third Annual Workshop on Economics and Information Security, WEIS04, Minneapolis MN, May 13–14 2004.

This is the paper that initially presented the work that is in Chapter 6 of this thesis. This version unfortunately contains an arithmetic error (using a figure of $4.35 rather than $43.50 for the profit margin that a spammer must achieve). This error has subsequently been corrected (and the underlying model somewhat improved), so that the correct figure is now believed to be $33.33.

`http://www.cl.cam.ac.uk/~rnc1/proofwork.pdf`

[1] yes, `contenet` is correct! as is `filtring`.

[89] R. Lemos: MSBlast epidemic far larger than believed. CNET News.com, 2 Apr 2004.

This news story reports that 16 million machines that had used Microsoft's Windows Update service had been found to be infected with the MSBlast virus. About 8 million systems were actually patched, suggesting the true infection rate lay between these bounds. Previously the anti-virus vendors had suggested the August 2003 worm might have infected about half a million machines.

`http://news.com.com/2100-7349_3-5184439.html`

[90] L. Lessig: Code and other laws of cyberspace. Basic Books, New York, 1999, ISBN 0-465-03913-8, 297pp.

This extremely interesting and carefully argued book considers the current nature of the Internet not as a natural state, inherently free and unregulated, but as reflecting its underlying "code" – the programs and protocols that implement the features we see. That code could, relatively easily, be altered so as to make the Internet a very highly regulated space, potentially far more oppressive to its users than the physical world has ever been. Lessig illustrates his thesis by carefully dissecting a number of cyberspace systems, showing how their nature is a result of specific decisions and is not pre-ordained. He remains at heart a US constitutional lawyer and often argues how things should *be by reference to the US constitution, a rather dissatisfying approach for a non-American.*

`http://www.code-is-law.org/`

[91] C. Linhart, A. Klein, R. Heled, S. Orrin: HTTP Request Smuggling. Whitepaper, Watchfire Corporation, Jun 2005, 23pp.

This whitepaper describes a scheme which involves sending two `Content-Length` *values within a single HTTP request. Proxy systems may use a different value than a remote website does, so that a blocking policy might be circumvented.*

`http://www.watchfire.com/resources/HTTP-Request-Smuggling.pdf`

[92] H. F. Lipson: Tracking and Tracing Cyber-Attacks: Technical Challenges and Global Policy Issues. Special Report CMU/SEI2-2002-SR-009, CERT/CC, Nov 2002, 85pp.

This report, which I discuss in Section 2.4.3, examines why tracing the source of attacks on the Internet is so hard, and then reviews current (for 2002) research on how tracking and tracing cyber-attacks might be improved. Lipson believes that a way forward would be to concentrate on "survivability" rather than security per se*. The policy issues mentioned in the title relate to the issues that would arise in negotiating international treaties for collaboration in dealing with attacks.*

`http://www.cert.org/archive/pdf/02sr009.pdf`

[93] K. Little: How to Obscure Any URL. Northwest Internet, 13 Jan 2002.

This web page documents numerous methods of obscuring URLs by expressing them in an unusual way, with a focus on the types of tricks that spammers use to obscure the destination of links within emails.

`http://www.pc-help.org/obscure.htm`

[94] D. L. Lough: A Taxonomy of Computer Attacks with Applications to Wireless Networks. PhD thesis, Virginia Polytechnic Institute, Apr 2001.

This thesis develops a taxonomy for computer attacks and applies it to 802.11 wireless networks. Lough observes that the same vulnerabilities and attacks recur down the years and suggests that by proper recording and classification of the community's experience we will be able to forestall old attacks working against new systems.

`http://scholar.lib.vt.edu/theses/available/etd-04252001-234145/unrestricted/lough.dissertation.pdf`

[95] G. Lowe: An Attack on the Needham-Schroeder Public-Key Authentication Protocol. Information Processing Letters, 56(3), 1995, pp. 131–133.

This paper describes an attack on the well-known Needham-Schroeder authentication protocol, that works by running two sessions in parallel and then using one of the principals as an oracle to decode a message and obtain the value of a nonce. This appears to be amongst the first papers to use the notion of "as an oracle" in this sense.

`http://web.comlab.ox.ac.uk/oucl/work/gavin.lowe/Security/Papers/NSPKP.ps`

[96] K. Lucke: Reading Email Headers. stopspam.org, 2004.

This web page explains how to read email headers in order to determine where an email has come from. Versions are available in English, French, Hungarian, Dutch and Slovenian – and there is an appeal for further translators to come forward.

`http://www.stopspam.org/email/headers.html`

[97] P. Lyman and H. R. Varian: How Much Information. School of Information Management and Systems, Univ. of California at Berkeley, Oct 2003.

The "How Much Information" study estimates how much new information is created each year. Newly created information is distributed on four storage media – print, film, magnetic, and optical – and seen or heard in four information flows – telephone, radio and TV, and the Internet. This study of information storage and flows analyses the year 2002 in order to estimate the annual size of the stock of new information on storage media, and heard or seen each year in information flows. The authors attempt to measure the amount of mailing list email. They do this by taking usage figures for LISTSERV, sampling to establish this is about one third of all lists and multiplying appropriately. This is undoubtedly an underestimate, since it ignores systems such as Yahoo! and ad hoc *distribution systems built out of Perl, PHP etc.*

`http://www.sims.berkeley.edu/how-much-info-2003`

[98] B. McWilliams: Cloaking Device Made for Spammers. Wired News, 9 Oct 2003.

This news story describes a network of end-user machines that are unknowingly running an extra executable. The network can be used to redirect HTTP requests in an untraceable manner. The Polish group who run it claim to control 450 000 machines and charge $1 500/month for their "invisible bulletproof hosting".

`http://www.wired.com/news/business/0,1367,60747,00.html`

[99] R. Mahajan, D. Wetherall, T. Anderson: Understanding BGP Misconfiguration. in: Proceedings of the ACM SIGCOMM 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, 19–23 Aug 2002, Pittsburgh, PA, USA, pp 3–16.

This is a quantitative study of BGP misconfiguration events, which turned out to be extremely common (between 200 and 1 200 prefixes were affected each day). ISPs were surveyed to establish the root cause; the authors believe better router user interface design would assist considerably.

`http://www.cs.washington.edu/homes/ratul/bgp/bgp-misconfigs.pdf`

[100] M. Mangalindan: For Bulk E-Mailer, Pestering Millions Offers Path to Profit. Wall Street Journal, 13 Nov 2002.

This news story covers Laura Betterley's spamming activities. It contains some useful figures on response rates: 275 out of 500 000 messages relating to a prize contest. It also provides data on profit margins for spammers: $35 on each sale of a 3D-glasses package; $50 for a mortgage lead; $85 for a cellphone sale.

`http://alanluber.com/storiesnolongeronweb/WSJ_com%20-%20For%20Bulk%20 E-Mailer,%20Pestering%20Millions%20Offers%20Path%20to%20Profit.htm`

[101] D. Mankins, R. Krishnan, C. Boyd, J. Zao and M. Frentz: Mitigating Distributed Denial of Service Attacks with Dynamic Resource Pricing. In Proceedings of 17th Annual Computer Security Applications Conference (ACSAC 2001), 2001.

This paper proposes to prevent DoS attacks by imposing dynamically changing prices for resources, to try and push back a cost onto requesters. Their scheme is intended to work both with micro-payment schemes and also with proof-of-work mechanisms.

`http://www.ir.bbn.com/~krash/pubs/mankins_acsac01.pdf`

[102] P. Mockapetris: Domain Names – implementation and specification. RFC1035, IETF, Nov 1987.

This RFC, along with RFC1034 which describes the concepts, remains the main standards document for the DNS system.

`http://www.ietf.org/rfc/rfc1035.txt`

[103] S. Moore, J. Srinivasan, D. Wilson: Hardware/Software Co-Design Teaching Package. University of Cambridge, Computer Laboratory, 19 Apr 2004.

Poster describing a port of Windows CE to the Altera EPXA1 development board.

`http://www.cl.cam.ac.uk/Teaching/current/ECADArch/extra/ WinCE-EPXA1-19april2004/EPXA1-poster.pdf`

[104] R. T. Morris: A Weakness in the 4.2BSD Unix TCP/IP Software. Bell Labs Computer Science Technical Report 117, 25 Feb 1985.

This paper shows how to "spoof" TCP connections by guessing the contents of the SYN/ACK packet sent in response to a SYN where the source address has been forged.

`http://www.pdos.lcs.mit.edu/~rtm/papers/117.pdf`

[105] Ofcom Network Interoperability Consultative Committee: Requirements on Communications Providers in Relation to Customer Line Identification Display Services and Other Related Services. ND1016:2004-09, PNO-ISC/SPEC/016, Ofcom, Sep 2004, 27pp.

This is a deeply technical description of how CLI interworking should be carried out, setting out the responsibilities of networks that originate, transfer or deliver calls. If the need is for a document written in end user terms, then it would be better to consult [106] instead.

http://www.nicc.org.uk/nicc-public/Public/interconnectstandards/spec/nd1016_2004_09.pdf

[106] Oftel: Guidelines for the provision of Calling Line Identification Facilities and other related services over Electronic Communications Networks Version 2. Office of Telecommunications, 11 Dec 2003.

"Oftel" was the UK telecommunications industry regulator until the start of 2004 when it was merged into the new communications industry regulator "Ofcom". This document sets out guidance for the handling of CLI by public telephone service providers. It proceeds from privacy rights for end users to set out the steps that the telcos must apply. An Annex sets out the rules for "presentation numbers".

http://www.ofcom.org.uk/advice-old/telecoms_ifc/telephony_con_guides/cli/

[107] OpenNet Initiative: Google Search & Cache Filtering Behind China's Great Firewall. Bulletin 006, OpenNet Initiative, 30 Aug 2004.

This bulletin describes a series of experiments that establish how China's "Great Firewall" is partially blocking access to the Google web search engine (and its associated cache) by looking for particular keywords in the HTTP protocol elements.

http://www.opennetinitiative.net/bulletins/006/

[108] M. Ossmann: WEP: Dead Again. SecurityFocus, Part 1: 14 Dec 2004, Part 2: 8 Mar 2005.

In this pair of articles Ossmann explains the current lamentable state of wireless security as it relates to WEP. The initial flurry of attack tools in 2001 were mainly blunted by vendor changes that ensured that there were limited numbers of "interesting" packets to attack. However, since summer 2004 much improved statistical methods have become available which, allied with new active attacks to generate traffic, mean that keys may be vulnerable within a few minutes.

Part 1: http://www.securityfocus.com/infocus/1814
Part 2: http://www.securityfocus.com/infocus/1824

[109] D. C. Plummer: An Ethernet Address Resolution Protocol – or – Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. IETF, RFC826, Nov 1982.

This RFC describes the ARP protocol used for mapping IP addresses to station addresses on an Ethernet. The associated RARP protocol is described in RFC903.

http://www.ietf.org/rfc/rfc826.txt

[110] J. Postel (Ed.): Internet Protocol. RFC791, IETF, Sep 1981.

This RFC is the base specification for version 4 of the IP protocol.

http://www.ietf.org/rfc/rfc791.txt

[111] J. Postel (Ed.): Transmission Control Protocol. RFC793, IETF, Sep 1981.

This RFC is the base specification for TCP.

http://www.ietf.org/rfc/rfc793.txt

[112] K. Poulsen: Cracking down on Cyberspace Land Grabs. Security Focus, 11 Jun 2003.

News story giving considerable detail about the hijacking of IP address space from Trafalgar House and Los Angeles County.

http://www.securityfocus.com/news/5654

[113] T. H. Ptacek, T. M. Newsham: Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection. Technical Report, Secure Networks Inc., Jan 1998.

This paper examines a number of attacks on the effectiveness of Intrusion Detection Systems by considering how invalid streams of packets can persuade the IDS to see one thing whilst the destination of the traffic sees another. Particular attention is paid to the difficulties of stream re-assembly at the IP and TCP layers.

http://downloads.securityfocus.com/library/ids.ps

[114] Radicati Group Inc.: Market Numbers Quarterly Update, Q1 2004.

The Radicati Group is a technology market research firm. It publishes quarterly updates to its surveys of anti-spam systems, giving forecasts and estimated market share. The actual reports are only available for a substantial fee, but accompanying press releases usually give valuable "headline" figures on spam volumes and trends.

http://www.radicati.com/

[115] Rain Forest Puppy: A look at whisker's anti-IDS tactics. 30 Dec 1999.

Whisker is a tool for accessing web resources via HTTP that attempts to hide its traffic from an Intrusion Detection System (IDS) that is monitoring the network link. It does this through various forms of obfuscation, both at the HTTP level and at the level of individual URLs.

http://www.wiretrip.net/rfp/pages/whitepapers/whiskerids.html

[116] T. Richardson: Net industry urged to co-operate following London bombings. The Register, 11 Jul 2005.

News article reporting the NHTCU memo requesting telcos to preserve traffic data and "the content of email servers" in the wake of the 7th July 2005 bombings in London. A further request, not widely reported in the media, was made after the further incidents which occurred on the 21st July.

http://www.theregister.co.uk/2005/07/11/ispa_preservation/

[117] C. Rigney, S. Willens, A. Rubens, W. Simpson: Remote Authentication Dial In User Service (RADIUS). RFC2865, IETF, Jun 2000.

This RFC describes the RADIUS protocol used by a NAS client to query a RADIUS server and determine if the credentials (typically username/password) provided by a dial-up user are acceptable and if so how the NAS should be configured for further Internet access by the user. The original RFC description of RADIUS was in RFC2058 (January 1997) reflecting work started in the IETF in April 1994, but the protocol originated at Livingston in 1991-2.

`http://www.ietf.org/rfc/rfc2865.txt`

[118] D. Roelker: HTTP IDS Evasions Revisited. Sourcefire Inc. Sep 2004.

This whitepaper discusses two generic IDS evasion techniques, sending invalid protocol elements and using coding tricks to obscure the request. These are explained with reference to HTTP, showing how the initial systems deployed in whisker [115] are still relevant – and still complex to deal with – today.

`http://docs.idsresearch.org/http_ids_evasions.pdf`

[119] SafeWeb Inc.: TriangleBoy Whitepaper. Safeweb Inc., 2001.

This rather short web-only document appears to be the only remaining authoritative description of Triangle Boy, a firewall evasion technology (apart from what may be gleaned by reading the source of the implementation). A requestor connects to one of a myriad of volunteer machines. They send the request to a content server which delivers the content direct to the requestor, but with the source address spoofed to be that of the volunteer. ACKs etc. are also passed to the server by the volunteer. A firewall near the requestor would have to block all volunteers in order to prevent access to the server. Note that Safeweb has now been taken over by Symantec and hence the only archive of the whitepaper is on the `archive.org` *system.*

`http://web.archive.org/web/20030417171335/`
`http://www.safeweb.com/tboy_whitepaper.html`

[120] SecuriTeam: Thompson (Alcatel) SpeedTouch Home ADSL Modem Predictable TCP ISN Generation. Beyond Security Ltd, 9 Aug 2004.

This article describes a security problem (CAN-2004-0641) with an ADSL modem, such that the initial sequence number it generates for a TCP connection is highly predictable. It is notable not because it is likely to lead to penetration of any site (the modem's identity (expressed merely as an IP address) is unlikely to be trusted by anyone) but because the underlying security issue has been known since 1985 [104] and was the subject of a CERT Advisory in 1995 [28].

`http://www.securiteam.com/securitynews/5LP032KDPS.html`

[121] A. Serjantov, S. Lewis: Puzzles in P2P Systems. 8th CaberNet Radicals Workshop, Corsica, Oct 2003.

This paper suggests a proof-of-work system could provide incentives for peer-to-peer users to behave in a uniform way, such as sharing content or performing "community" tasks.

`http://www.newcastle.research.ec.org/cabernet/workshops/`
`radicals/2003/papers/puzzles3.pdf`

[122] SETI@home: The Search for Extraterrestrial Intelligence.

This website is used to co-ordinate the use of Internet-connected computers to do background processing of radio telescope data in the hope of detecting patterns that would lead to the discovery of intelligent life elsewhere in the universe.

`http://setiathome.ssl.berkeley.edu/`

[123] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Ken, W. T. Strayer: Hash-Based IP Traceback. ACM SIGCOMM'01, San Diego, CA, USA, Aug 2001.

This paper proposes that multiple hash digests should be created for each packet that passes through a router, with a single bit being set in a local Bloom filter for each of these digest values. Interrogators can then determine whether a particular packet they have received may have passed through a particular router. This is useful for traceback of DoS attacks but is sensitive enough, if necessary and the filters are tuned correctly, to handle the tracing of a single packet.

`http://www.acm.org/sigs/sigcomm/sigcomm2001/p1-snoeren.pdf`

[124] J. Snyder: Test: Spam in the wild. Network World Fusion, 15 Sep 2003.

This article presents the results of a test of 16 anti-spam products on a live production network in June 2003. The best filtering product, when tuned for a false positive rate of 1%, stopped 94% of incoming spam.

`http://www.nwfusion.com/reviews/2003/0915spam.html`

[125] Society of Cable Telecommunications Engineers: Data-Over-Cable Systems Radio Frequency Interface Specification 1.1. ANSI/SCTE 23–1 2002, 2002.

This is the standard for the use of the cable medium by a DOCSIS cable modem. It was originally developed by MCNS Holdings (later CableLabs).

`http://www.scte.org/documents/pdf/ANSISCTE2312002DSS0209.pdf`

[126] P. Sommer: Intrusion Detection Systems as Evidence. First International Workshop on the Recent Advances in Intrusion Detection (RAID'98), 14–16 Sep 1998 Louvain-la-Neuve, Belgium.

IDS systems are designed to detect intrusions rather than to provide forensic quality evidence for a prosecution. This paper discusses the issues that arise with evidence from the network with specific examples of issues that arose in the "DataStream Cowboy" case, the intrusion into the US Air Force Rome Laboratories.

`http://www.raid-symposium.org/raid98/Prog_RAID98/`
`Full_Papers/Sommer_text.pdf`

[127] P. Sommer: Evidence in Internet Paedophilia Cases. In A. MacVean, P. Spindler (Ed.): Policing Paedophiles on the Internet, New Police Bookshop, 2003, ISBN 1-903639-12-2.

Sommer's chapter is mainly about computer evidence, disk forensics, network forensics and traceability. However, it also has a detailed account of UK law relating to indecent images of children, covering the statute's various revisions and the evolution of case law.

`http://isig.lse.ac.uk/news/evi.html`

[128] SORBS: Spam and Open Relay Blocking System.

SORBS publishes a list of IP addresses known to be capable of sending spam, whether because they are owned by spammers, or whether they are insecure and may be hijacked or proxied via in some manner. They also list machines with other spam related security issues, such as insecure formmail scripts. The number of addresses listed now comfortably exceeds a million.

`http://www.sorbs.net/`

[129] SpamCop.net

SpamCop is a spam reporting website. It was originally independent, but in November 2003 it was purchased by IronPort (see also [9]) who sell email handling hardware. The SpamCop website provides a range of tools for determining the origin of unwanted email and the company also provides real-time access to an extensive database of IP addresses that are known to be sources of spam.

`http://www.spamcop.net/`

[130] S. J. Stolfo, S. Hershkop, K. Wang, O. Nimeskern and C. Hu: A Behavior-based Approach to Securing Email Systems, Proceedings Mathematical Methods, Models and Architectures for Computer Networks Security (MMM-ACNS-2003), LNCS 2776, Springer, 2003, pp. 57–81.

The Email Mining Toolkit (EMT) works with MET [19] by processing email archives. It builds models of legitimate activity by users, and groups of users, paying specific attention to the presence of attachments, and then aims to detect anomalous or errant email behaviour.

`http://www1.cs.columbia.edu/ids/publications/EMT-ACNS03.pdf`

[131] M. St. Johns: Identification Protocol. RFC1413, IETF, Feb 1993.

This RFC describes the IDENT protocol that can be used to interrogate a server to obtain the identity of a user on that server who is at the other end of a TCP connection to the interrogator. The protocol returns a string that describes the user who is using a particular port. The string can only be fully understood by the server and is only as reliable as the server permits it to be. It is therefore of limited assistance to the remote machine in understanding who is using it, but may sometimes be of some help to the server owner in tracking down who is responsible for an action when a complaint arrives from the remote machine.

`http://www.ietf.org/rfc/rfc1413.txt`

[132] A. Stubblefield, J. Ioannidis, A. D. Rubin: Using the Fluhrer, Mantin and Shamir Attack to Break WEP. Network and Distributed System Security Symposium (NDSS02), San Diego, CA, USA, 6–8 Feb 2002.

This paper describes a practical implementation of the Fluhrer-Mantin-Shamir attack on RC4. A 128-bit WEP key was recovered from a deployed 802.11 wireless network. An earlier version appeared as an August 2001 AT&T Labs Technical Report (TD-4ZCPZZ) which contains fewer equations and a far more chatty description of the practical issues involved in the attack.

`http://www.isoc.org/isoc/conferences/ndss/02/proceedings/papers/stubbl.pdf`

[133] Supreme Court of the United States: Ashcroft, Attorney General, *et al.* v. Free Speech Coalition *et al.* 535 US 234, decided 16 Apr 2002.

The Supreme Court decided (on a 6–3 majority – and for some aspects a 7–2 majority) to strike down parts of the Child Pornography Prevention Act 1996 in so far as it was unconstitutional to restrict free speech involving visual depictions of minors engaging in sexual acts where there were no actual children involved, e.g. if the images were computer generated. In the UK such images are illegal, whether actual children are involved or not.

`http://laws.findlaw.com/us/000/00-795.html`

[134] D. Tarrant, T. Riddell: How To Increase Your E-Mail Response Rates… A best practices approach! Harte-Hanks Inc., 26 Oct 2004.

A "webinar" for an email marketing company. Gives typical click-through response rates for email campaigns: general marketing 0.7%; market research 2.6%; sales promotion 0.7%; subscription 0.9%; webinar 0.9%. Considerable variation (0% to 32%) occurs.

`http://www.hartehanksmi.com/content/pdf/Email_Webinar10-26-04_final.pdf`

[135] Telecom New Zealand: Calling Station ID/Calling Line ID. Informer WS 2003-02-10, Telecom New Zealand, 10 Feb 2003, 2pp.

This document is aimed at Telecom NZ staff who are dealing with ISPs and explains that if consumers withhold CLI then an ISP will generally not be provided with the calling number (in some cases the leading 4 digits of 8 will be provided). It is made clear that Telecom NZ will have records with the full number, which they will be prepared to disclose to the police on production of a search warrant.

`http://www.telecom.co.nz/binarys/ws_2003-02-10.pdf`

[136] Telecommunication Standardization Section of the International Telecommunications Union: ITU-T Recommendation Q.763. Signalling System No. 7 – ISDN user part formats and codes. International Telecommunication Union, Dec 1999, 122pp.

This document is one of the series Q.761–Q.764 which define the SS7 protocol used by telco switches to set up and tear down telephone calls. This particular part documents the protocol payloads and in particular the details of the Caller Line Identification (CLI) information sent from switch to switch.

`Not available online.`[2]

[137] Telecommunication Standardization Section of the International Telecommunications Union: ITU-T Recommendation Q.931. Digital Subscriber Signalling System No. 1 – Network Layer, ISDN user-network interface layer 3 specification for basic call control. International Telecommunication Union, May 1998, 329pp.

This document is one of the series Q.930–Q.939 that define the network layer of DSS1 (the protocol for ISDN equipment that talks to telco switches).

`Not available online.`[2]

[138] Telecommunication Standardization Section of the International Telecommunications Union: ITU-T Recommendation Q.951. Digital Subscriber Signalling System No. 1 – Stage 3 Description for Supplementary Services Using DSS 1. Clauses 3–6. International Telecommunication Union, Mar 1999, 34pp.

This document describes the handling of CLI information within DSS 1 (the protocol for ISDN equipment that talks to telco switches).

`Not available online.`[2]

[139] Telenor Norge: Telenor and KRIPOS introduce Internet child pornography filter. Telenor Press Release, 21 Sep 2004.

This press release announces that Telenor, a major Norwegian ISP, is to block access to child pornography. Telenor is to be responsible for the technical solutions and KRIPOS, the Norwegian National Criminal Investigation Service, is to provide lists of websites distributing such material. There are few details of the technical solution, but it is at least partly based on a web cache. Upon access to a blocked site a message "will automatically pop up, containing information about the filter, as well as a link to KRIPOS".

`http://presse.telenor.no/PR/200409/961319_5.html`

[140] F. Thernelius: SIP, NAT, and Firewalls. Master's Thesis, Dept. of Teleinformatics, Royal Institute of Technology in Stockholm (KTH), May 2000, 69pp.

This thesis discusses the issues that arise with the SIP protocol, used for VoIP signalling, when it has to work with NAT and firewalls.

`http://www.cs.columbia.edu/sip/drafts/Ther0005_SIP.pdf`

[141] P. F. Tsuchiya, T. Eng: Extending the IP Internet Through Address Reuse. ACM SIGCOMM Computer Communication Review, 23(1), 1993, pp: 16–33

This article introduces the idea of Network Address Translation – portraying it mainly as a method of preserving IP address space.

`http://portal.acm.org/citation.cfm?id=173944`

[142] US Department of Justice: Background on Operation Web Snare: Examples of Prosecutions. Press Release, 27 Aug 2004.

This press release describes a number of US cyberspace cases including "Operation Cyberslam" that indicted Jay R. Echouafni and others who were alleged to have perpetrated DDoS attacks against business competitors. Echouafni has absconded and is still, summer 2005, being sought by the FBI.

`http://www.usdoj.gov/criminal/fraud/websnare.pdf`

[143] US District Court for the Eastern District of Pennsylvania: CDT, ACLU, Plantagenet Inc. v Pappert, 337 F.Supp.2d 606, 10 Sep 2004.

This is the judgment that struck down as unconstitutional a 2002 Pennsylvania statute requiring ISPs to block access to child pornography websites.

`http://www.cdt.org/speech/pennwebblock/20040910memorandum.pdf`

[2]Note that although the ITU publications are not openly available online, it is currently possible to download a small number of documents from their electronic bookshop without charge.

[144] D. Verton: The Hacker Diaries: Confessions of Teenage Hackers. Osborne McGraw-Hill, 2002, ISBN 00-722-2364-2, 272pp.

This is a racy account of several hacking cases, with a detailed account of the February 2002 'MafiaBoy' case where a 14-year-old boy in Montreal ran DoS attacks against several eCommerce sites. NB: the URL is for a page containing lengthy extracts.

`http://www.witiger.com/ecommerce/mfiaby.htm`

[145] G. Vigna, W. Robertson, D. Balzarotti: Testing Network-based Intrusion Detection Signatures Using Mutant Exploits. In B. Pfitzmann, P. Liu (Ed.): Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS 2004), Washington DC, USA, Oct 2004, pp.21–30.

Many IDS systems work by constructing a model of an exploit and then creating a signature that is sufficient to detect it occurring. The signature, and indeed the model, may be over-specific and hence similar exploits will fail to be detected. This paper quantitatively evaluates IDS signatures by creating mutant versions of an exploit.

`http://www.cs.ucsb.edu/~vigna/pub/2004_vigna_robertson_balzarotti_CCS04.pdf`

[146] P. Walker: Re: BT CLI availability ? Usenet message ID: `<789a0I$8sl$1@plug.news.pipex.net>`, 22 Jan 1999.

When he posted this Usenet article, Walker was Director Technology at Oftel. He states that it is Oftel's policy to encourage subscriptionless ISPs to be passed CLI even if the user has withheld it, justifying this as "the only way to trace unsuitable material".

`http://www.google.com/groups?selm=789a0I$8sl$1@plug.news.pipex.net`

[147] R. Walker: Cable Modem Troubleshooting Tips – Swapping computers on the cable modem. 2002.

This web page, part of large collection of practical advice on cable modem issues, reports on the ability of NTL (a UK ISP providing services over cable) to automatically detect when new computers (with different MAC addresses) are connected to a cable modem; yet restrict the number of changes permitted. No details are given as to how this is achieved.

`http://homepage.ntlworld.com/robin.d.h.walker/cmtips/swap.html`

[148] L. Wall, T. Christiansen and J. Orwant: Programming Perl, 3rd Edition, O'Reilly, Jul 2000, 1092pp, ISBN 0-596-00027-8.

The 'camel' book, describing the Perl scripting language.

`http://safari.oreilly.com/?XmlId=0-596-00027-8`

[149] L. Wilcox: mail.internet.com. Usenet message ID: `<a802up$bug$1@newsg4.svr.pol.co.uk>`, 28 Mar 2002.

Mrs Wilcox was seeking help with a dialog that popped up within Outlook Express and referred to `mail.internet.com`*. Unfortunately, wishing to provide as much technical detail as possible, she included her password in the article. Her error was pointed out to her within 70 minutes of posting.*

`http://www.google.com/groups?selm=a802up$bug$1@newsg4.svr.pol.co.uk`

[150] Y. Zhang, V. Paxson: Detecting Stepping Stones. In: Proceedings 9th USENIX Security Symposium, Denver CO, USA, Aug 2000.

Attackers often use intermediate machines ("stepping stones") so that tracing back will fail to locate them. This paper considers how to detect stepping stones by correlating the timing of streams of traffic (when they are active or idle). Since the discriminator is timing patterns, the technique works even when some streams are encrypted (using SSH or similar systems).

`http://www.cs.utexas.edu/~yzhang/papers/stepping-sec00.pdf`

[151] J. Zittrain, B. Edelman: Documentation of Internet Filtering Worldwide. Harvard Law School. 24 Oct 2003.

This webpage was created as part of the OpenNet initiative and is intended to provide pointers to further information on filtering systems around the world; however it concentrates specifically on Saudi Arabia and China. It therefore misses the systems now deployed in Australia, Iran, Singapore, Burma, Norway and elsewhere.

`http://cyber.law.harvard.edu/filtering/`

[152] A. Zugenmaier, M. Kreutzer, G. Müller: The Freiburg Privacy Diamond: An Attacker Model for a Mobile Computing Environment. In K. Irmscher, K-P. Fährich (Ed.): KiVS 2003, Kommunikation in Verteilten Systemen '03, Leipzig 25-28 Feb 2003, VDE Verlag 2003, pp.131–142.

This paper introduces the 'Freiburg privacy diamond' as an attacker model that considers the links between users, devices, locations and actions. The ability of an attacker to compromise these links and hence link users to actions is considered not only in absolute terms but also in a probabilistic manner.

`http://www.vs.inf.ethz.ch/publ/se/FPDinKiVS03.pdf`

[153] A. Zugenmaier, A. Hohl: Anonymity for Users of Ubiquitous Computing. 2nd Workshop on Security in Ubiquitous Computing, Ubicomp 2003, Seattle 12 Oct 2003.

This "extended abstract on work in progress" revisits the 'Freiburg privacy diamond' introduced in [152] and lists the different combinations of relationships that must be secured in order for a user and action to be unlinkable. It then extends the model to consider multiple diamonds all existing at the same time, as would occur for a user with multiple devices at their disposal.

`http://www.vs.inf.ethz.ch/publ/se/secubi03_p06.pdf`

Glossary

You think that I don't even mean
A single word I say...
It's only words
And words are all I have
To take your heart away

— Barry, Robin and Maurice Gibb, 1967

Although I have been careful to define acronyms and other technical terms when they are first encountered in the text, a reader who is unfamiliar with the jargon may prefer to use this glossary when they encounter the terms later in the text. The definitions are not intended to be all-encompassing, but to give a flavour of how the terms are used within this thesis.

/11	A network with a fixed prefix of 11 bits, containing 2^{21} IP addresses.
/16	A network with a fixed prefix of 16 bits, containing 2^{16} IP addresses.
/24	A network with a fixed prefix of 24 bits, containing 2^{8} IP addresses.
/26	A network with a fixed prefix of 26 bits, containing 64 IP addresses.
/32	A network with a fixed prefix of 32 bits – exactly one IP address.
802.3	The IEEE 802.3 working group develops standards for CSMA/CD (Ethernet) local area networks. Hence 802.3 is used as shorthand for the standards themselves.
802.11	The IEEE 802.11 working group develops standards for wireless local area networks in the 2.4 GHz (and latterly the 5 GHz) band. Hence 802.11 is used as shorthand for the standards themselves.
ActiveX	ActiveX is Microsoft's software technology whereby self-contained programs can be fetched from the web and run on a local machine.
ADSL	Asymmetric Digital Subscriber Line; a widely-deployed technology for high-bandwidth (512 kbit/s up to 8 Mbit/s) transmission of digital data over telephone lines.

ARM	Name of a RISC (Reduced Instruction Set Computer) microprocessor architecture and the company (ARM plc) that produced it. ARM originally stood for Advanced RISC Machines.
ARP	Address Resolution Protocol; used to map IP addresses onto MAC addresses on an Ethernet.
AS	Autonomous System; routing entity used by BGP corresponding, more or less, to a single ISP's network.
ATM	Asynchronous Transfer Mode; a digital switching technology using dedicated connections. It was originally aimed at voice traffic, but is now used in some ISP core networks.
BCP	Best Current Practice; document explaining how it is currently believed that things should be done.
BGP	Border Gateway Protocol; protocol used to negotiate the inter-provider routing of IP traffic.
BIND	Berkeley Internet Name Domain; widely used DNS server program.
Browser	Program used for viewing files on the web.
BT	Large UK telecoms company; once called British Telecom.
CERT/CC	From Computer Emergency Response Team Coordination Center; the name of the premier incident response team for the Internet, based in Pittsburgh, PA, USA.
CleanFeed	Internal project name for a system deployed by BT to block access to illegal images of children.
CLI	Caller Line Identification; provision of the telephone number of the calling party to other parts of the telecoms system.
CLIR	CLI Presentation Restriction; the setting that blocks CLI.
CPE	Customer Premises Equipment; a cable company customer's computer, television, etc.
CPU	Central Processor Unit; the arithmetic/logical core of a computer that interprets and executes instructions.
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance; a media access strategy that can be used for wireless networks.
CSMA/CD	Carrier Sense Multiple Access with Collision Detection; the media access strategy used for Ethernet networks.

Cyberspace	The notional environment within which electronic communications occur. The word was coined by William Gibson in 1982.
DDoS	Distributed Denial-of-Service; a DoS attack carried out from many sites simultaneously.
DES	Data Encryption Standard; a well-known cryptographic cipher standard from the 1970s.
DHCP	Dynamic Host Configuration Protocol; a protocol for providing systems with IP addresses and other configuration data from central servers.
DNS	Domain Name Service; a distributed database mainly used for mapping hostnames to IP addresses.
DOCSIS	Data-Over-Cable Service Interface Specification; suite of standards for the provision of data services on cable TV networks.
DoS	Denial-of-Service; the overloading of a cyberspace service through deliberate over-use.
DSLAM	Digital Subscriber Line Access Multiplexer; equipment at the telephone exchange that handles the conversion between ADSL traffic on the subscriber local loop and packets on an ATM network.
EHLO	Modern keyword for the initial message from a client in the email transfer protocol. Hence, by extension, this whole initial message.
Ethernet	Generic name for a family of LAN (Local Area Network) data transport protocols.
FPGA	Field Programmable Gate Array; a generic integrated circuit that can be programmed to provide a variety of different logic functions.
FTP	File Transfer Protocol; protocol for the bulk transfer of data.
Hash	A short message digest that is numerically derived from a message. A strong cryptographic hash will have numerous useful properties such as it being impracticable to derive the message from knowledge of the digest value.
HELO	Keyword for an initial message from a client in the SMTP email transfer protocol. Hence, by extension, this whole initial message.
HTML	HyperText Markup Language; the authoring language for web documents.

HTTP	HyperText Transfer Protocol; the transfer protocol for web documents.
IANA	Internet Assigned Numbers Authority; the organization responsible for assigning IP addresses and other identifiers used by Internet protocols.
ICMP	Internet Control Message Protocol; the Internet protocol for error reporting and diagnostics.
IDS	Intrusion Detection System; a security device for detecting unusual activity indicative of unauthorised access.
IEEE	Institute of Electrical and Electronic Engineers; professional organization for electrical engineers.
IETF	Internet Engineering Task Force; volunteer organization that develops Internet protocol standards.
IOCTL	Input/Output Control; interface for passing control commands, via the operating system, to a device driver.
IP	Internet Protocol; the underlying protocol for all Internet traffic.
IP address	End-point identifier for the Internet Protocol. Often, rather loosely, equated with a particular computer or interface.
IPv4	"Version 4" of the Internet Protocol, dating from 1981.
IPv6	"Version 6" of the Internet Protocol, dating from 1994.
IRC	Internet Relay Chat; text-based method for holding distributed real-time conversations.
ISDN	Integrated Services Digital Network; digital communications standard for sending voice and data over normal telephone wires.
ISP	Internet Service Provider; organization that provides connectivity to the Internet (and often many other services as well).
ITU	International Telecommunications Union; international organization within the United Nations system where governments and the private sector co-ordinate global telecom networks and services.
IWF	Internet Watch Foundation; operator of the UK hotline for the reporting of illegal content on the Internet.
JANET	JANET is the private, government funded, network for education and research in the UK; originally the Joint Academic Network.

Java	An object oriented high-level programming language designed for use in a distributed environment.
JavaScript	An interpreted scripting language, originally designed by Netscape, which provides programming functionality within HTML pages.
LAN	Local Area Network; network covering a relatively small area, such as a home or an office.
LINX	London Internet Exchange; a co-operative that owns the premier UK Internet peering point, sited in London Docklands.
Log	A file that records information about events that have occurred whilst a service is being provided.
MAC	Media Access Control; component of, for example, an Ethernet interface responsible for mediating access to the transport medium.
MIB	Management Information Base; stylised representation of the statistics available about a system or sub-system.
MIT	Massachusetts Institute of Technology; famous University based in another Cambridge.
MX	From "mail exchange"; a type of DNS record giving details of the hosts that accept email for a particular domain.
NAS	Network Access System; the equipment at an ISP, often thought of as "the ISP's modems", which handles dial-up access to the Internet.
NAT	Network Address Translation; the translation of IP addresses when packets flow between one network and another.
NHTCU	National Hi-Tech Crime Unit; specialist UK police squad dealing with cybercrime.
NIR	National Internet Registry; organization that distributes IP addresses within a single country.
OSI	Open Systems Interconnection; networking suite worked upon by the International Organization for Standardization (ISO).
PABX	Private Automatic Branch Exchange; a telephone switch on customer premises.
PC	Personal Computer; computer designed for use by one person.

PCB	Printed Circuit Board; insulator with pre-printed conductive paths onto which electronic components are placed.
PHY	From “physical”; subsystem of, for example, an Ethernet device that is concerned with the coding and decoding of data to be transmitted across the medium.
PTR	From “pointer”; type of DNS record that points to another part of domain name space.
PVC	Permanent Virtual Circuit; a software defined path across an ATM network.
RADIUS	Remote Authentication Dial In User Service; a protocol for handling authentication, authorisation, and configuration information.
RAM	Random Access Memory; high-speed computer data storage.
RFC	Request For Comments; self-deprecating name for a series of documents that describe Internet protocols and systems.
RIR	Regional Internet Registry; one of the five organizations that handle the allocation of IP addresses on a continent-wide basis.
Routing	Sending of packets to appropriate destinations; the preferred UK spelling is *routeing*, but the US form is used to avoid the unnecessary jarring of the sensibilities of trans-Atlantic readers.
RR	Resource Record; generic term for a record within the DNS.
SIP	Session Initiation Protocol; a signalling protocol used for VoIP.
SMTP	Simple Mail Transfer Protocol; protocol used for email transfer.
SOCKS	A networking proxy protocol. It is often used by proxy servers to accept local requests for web access, prior to forwarding them to the open Internet.
Spam	In this thesis the term means Unsolicited Bulk Email; it derives from the chanting of some Vikings in a ‘Monty Python’ sketch.
SS7	Common Channel Signalling System Number 7; a global standard for telecoms signalling.
Sysadmin	System administrator; all powerful being reduced to configuring computers for a living.
TCP	Transmission Control Protocol; reliable stream oriented communications protocol layered over IP.

TFTP	Trivial File Transfer Protocol; a lightweight file transmission protocol layered over UDP.
Trojan	A program that has an unexpected, unwanted, secondary functionality (often breaching security) in addition to a more obvious primary function. Often extended to any program functionality which is not immediately apparent. Shortened from *trojan horse*, which in turn derives from the legend of the fall of Troy.
TTL	Time To Live; remaining number of hops which an IP packet may be forwarded over.
TV	Television; radio with moving pictures.
UDP	User Datagram Protocol; simple (unreliable) data transmission protocol layered over IP.
UK	The United Kingdom of Great Britain and Northern Ireland.
URL	Uniform Resource Locator; the global address of an item on the worldwide web.
USA	The United States of America.
Usenet	A worldwide system that provides many tens of thousands of specialised forums for sharing articles on topics of mutual interest.
VoIP	Voice over IP; a voice telephony service carried over the open Internet using TCP/IP protocols.
Web proxy	An intermediate server for web content that relays requests to the actual server, unless it can satisfy the request itself.
Windows CE	Operating system that is broadly compatible with the Windows desktop operating system, but aimed at low-end devices such as Personal Digital Assistants (PDAs) and mobile phones.
xDSL	Generic term for (A)symmetric Digital Subscriber Line (ADSL, SDSL) along with other "broadband" technologies such as High Data-Rate DSL (HDSL) and Very High Speed DSL (VDSL).