Computer security and privacy at Cambridge (1/11)

The University of Cambridge is home to some of the world's leading computer security researchers, with a long history of key contributions to the field. Cambridge's early interests in security include the identification of large primes (Wheeler 1949), one-way password encryption (Needham 1962), the capability-system security model (Wilkes, Needham, Walker 1970–1977), the Needham-Schroeder Protocol (1978), and the Burrows-Abadi-Needham logic (1989). We continue to make core contributions in the field – cryptographic protocol design, processor and operating system security, anonymity research, hardware security, and malware analysis. We also perform foundational cross-disciplinary work in security economics, cybercrime measurement, censorship resistance, security psychology, human factors, and domestic and international policy.

Security and privacy research spans many groups at Cambridge, exploring issues ranging from CPU security to cybercrime:

- Computer Laboratory - Security Research Group
- Computer Laboratory - Network and Operating System Group
- Computer Laboratory - OPERA Group (distributed systems)
- Computer Laboratory - Computer Architecture Group
- Computer Laboratory - Programming, Logic, and Semantics Group
- Computer Laboratory - Digital Technologies Group (DTG)
- Centre for Science and Policy (CSaP)

Full-time academic staff with security or privacy focuses are Ross Anderson, Jean Bacon, Alastair Beresford, Jon Crowcroft, John Daugman, Markus Kuhn, Andrew Moore, Simon Moore, Steven Murdoch, Larry Paulson, Frank Stajano, and Robert Watson. Post-doctoral researchers and assistants include Jonathan Anderson, David Chisnall, Richard Clayton, Khilan Gudka, Graeme Jenkinson, David Modic, Jeunese Payne, Michael Roe, Sergei Skorobogatov, Max Spencer, Quentin Stafford-Fraser, Sophie van der Zee, and Chris Warrington.

Computer Laboratory Security Research Group

The security research group consists of five full-time faculty members, eleven full-time post-doctoral researchers, nine PhD students, and several master students working on security-focussed dissertations. The group maintains dedicated research facilities including a Tamper Lab for reverse engineering hardware, side-channel and fault injection attacks, and analysing electromagnetic emanations from computing devices. The security group works closely with other groups, bringing a security perspective to their projects. The group has a web site and blog, which carry our recent research, musings on security, and job/studentship ads:

http://www.lightbluetouchpaper.org/
http://www.cl.cam.ac.uk/research/security/

Active security and privacy research areas

- Anonymous communication
- API and protocol security
- Application compartmentalisation techniques
- Authentication and biometric identification systems
- Banking and payment system security
- Censorship resistance
- Capability systems
- Compromising emanations
- Cryptology
- Digital forensics
- Distributed system and cloud computing security
- Economics of cybercrime
- Economics of information security
- Formal methods
- Hardware security
- Location and positioning systems
- Malware analysis
- Medical information security
- Mobile and embedded system security
- Operating system security
- Passwords
- Privacy and freedom issues
- Programming language security
- Psychology of deception
- SCADA and the security of industrial control systems
- Security and human behaviour
- Security protocols
- Social networking and privacy
- Steganography
- Tampering with tamper-resistant devices
- Temporal security properties

Teaching in computer security and privacy

The Computer Laboratory's undergraduate and masters programmes provide in-depth teaching of computer security foundations and current research; operating systems, networking and programming languages courses also necessarily consider security. Several undergraduates and masters students write security-related dissertations or essays each year. Masters and PhD students frequently publish security-related research.

Undergraduate degree in Computer Science

- *Part I Security* is an introductory course providing every student with the basics in security. Material includes the cryptography, protocols, programming language, application, and operating system security.
- *Part II Security* is an advanced course teaching about security policy, security usability, security economics, security protocols, cryptography, hardware security, privacy, anonymity and concurrency vulnerabilities.

MPhil in Advanced Computer Science (ACS)

- *Computer security: principles and foundations* is a research readings course reviewing key historic security texts and themes in security research, including tensions between attack and defence, human factors, security economics, reasoning protocols, programming language security, and the evolution of access control and protection models.
- *Computer security: current applications and research* is a research readings course exploring contemporary research areas, including supply-chain trojans in hardware, malware analysis, secure processor design, banking-system security, and security psychology.

PhD in Computer Science

The PhD in Computer Science mentors students in award-winning research and methodology. Recent security-related PhDs include:

- *Decompilation as search*
- *Reconstructing compressed photo and video data*
- *Privacy engineering in social networks*
- *Guessing human-chosen secrets*
- *Robust security for the electricity network*
- *Complex network analysis for secure and robust communications*
- *Verification of security protocols based on multicase communication*
- *Distributed virtual environment scalability and security*
- *New approaches to operating system security extensibility*
- *Active electromagnetic attacks on secure hardware*

UNIVERSITY OF CAMBRIDGE

Economics and psychology of security (2/11)

Security economics

Ross Anderson, Mike Bond, Richard Clayton, Steven J. Murdoch

An early observation that complex systems fail because the incentives are wrong led to Cambridge founding the field of security economics, which has 100+ researchers worldwide.

This new, and highly successful, approach takes the view that it is usually more relevant to look at the economics of a system than at the computer science when seeking to understand whether or not it will be secure.

Two major reports have been written for ENISA, on applying security economics to network security (2008) and on the resilience of the Internet (2011) where we explain how network operators negotiate peering and transit, what goes wrong, how they deal with failures and where the incentives for resilience are inadequate.

Research into cybercrime, particularly 'phishing', has used econometric approaches (analysis of website lifetimes for example) to explain the variations in the effectiveness of countermeasures to criminal activity. We are now involved in work funded by the US Department of Homeland Security to examine criminal supply chains, evaluate security indicators and assess the barriers to sharing security information.

Other research in this field has looked at website data collection policies, password strength, public policy on malware removal, app security and the security economics of electricity 'smart meters'.

Ross Anderson and Tyler Moore, "Information security: where computer science, economics and psychology meet". Phil. Trans. R. Soc. A367(1898):2717-2727, July 2009.

Understanding the psychology of scams

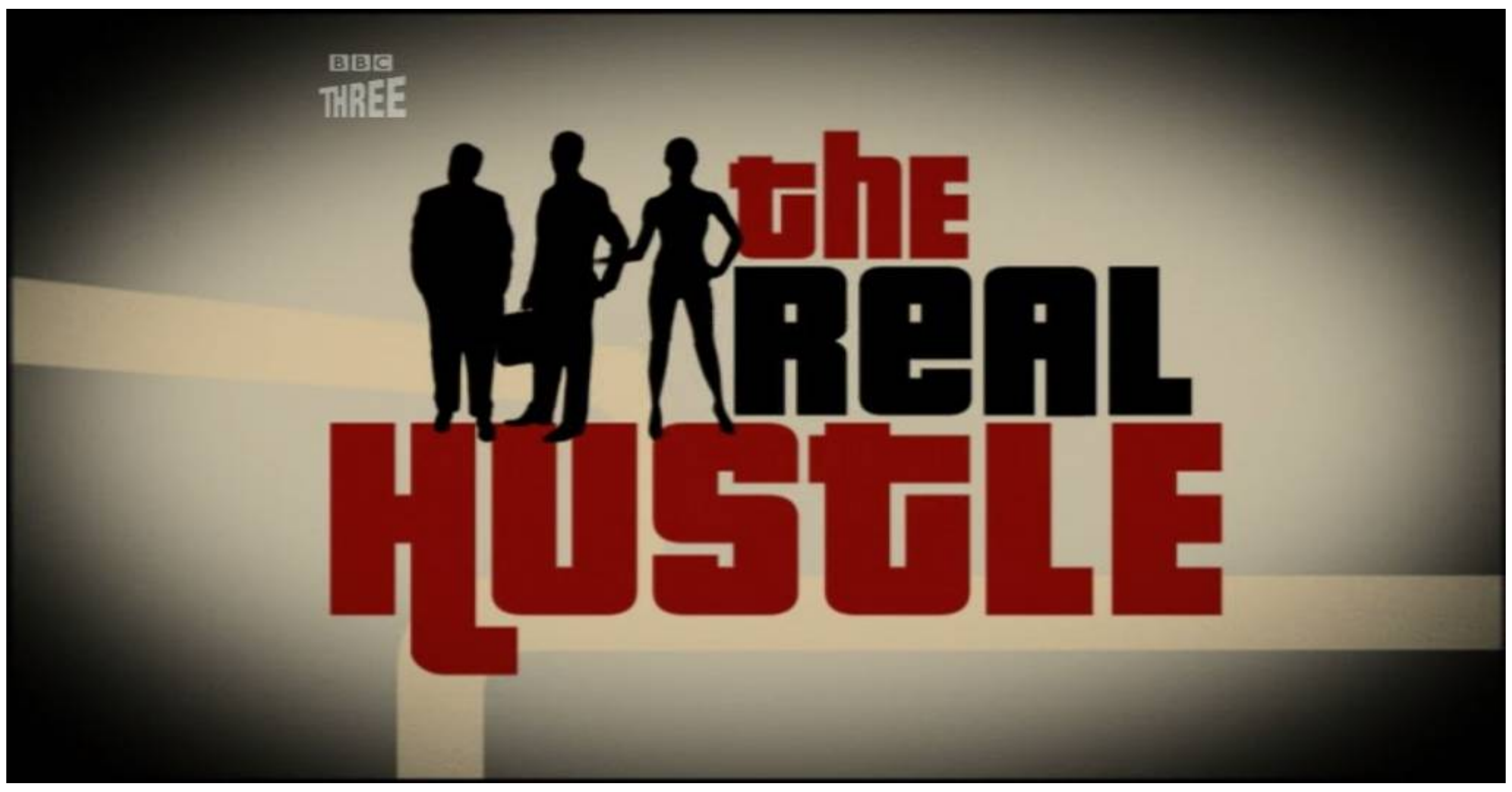
Frank Stajano (joint work with Paul Wilson of BBC3's The Real Hustle)

Security engineers build system defences, but real users don't follow engineer logic. Result: systems are vulnerable to attack. How can we understand what makes users vulnerable?

Our approach: learn from the fraudsters! They understand the victims' psychology better than most security engineers.

We distilled a number of general principles by documenting and analysing hundreds of observed scams. Knowledge of these principles can be used to strengthen system security.

The viewpoint that "it's the fault of those gullible users" is arrogant and idiotic. Some behavioural patterns that make us vulnerable to fraud are just human nature. It is up to smart system designers to prevent their exploitation.

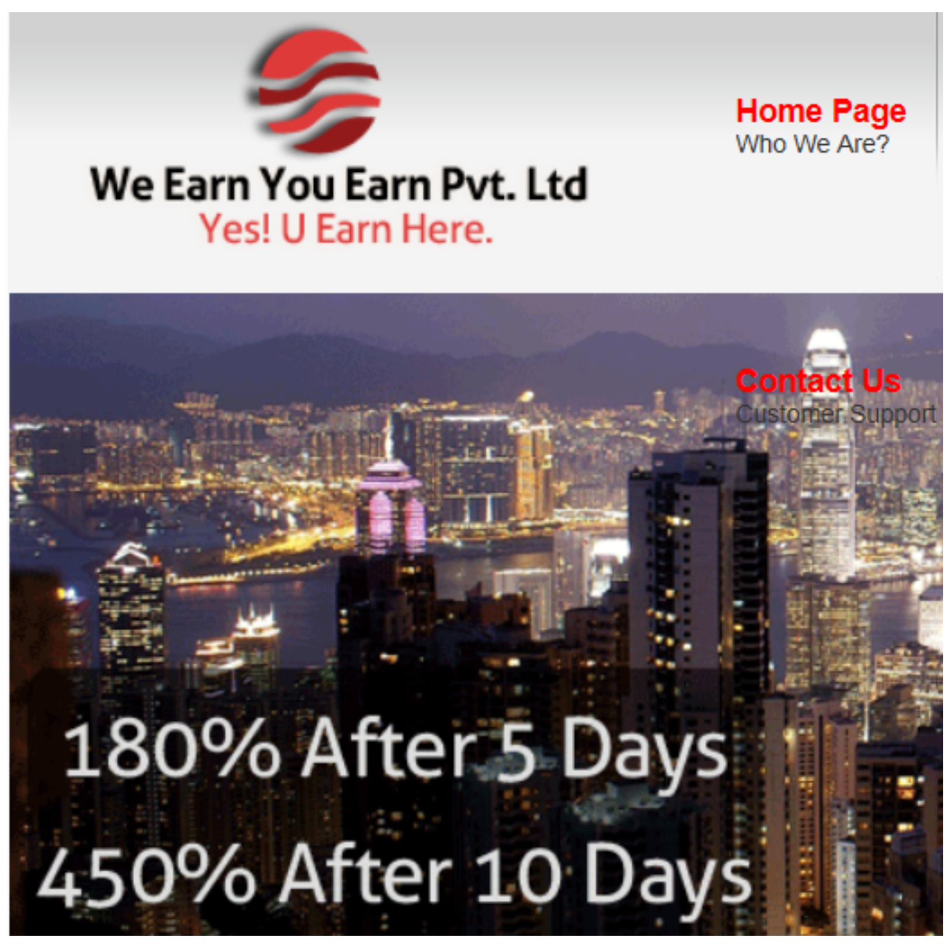


Frank Stajano and Paul Wilson, "Understanding scam victims: seven principles for systems security", Communications of the ACM, 54(3):70-75, March 2011.

Measuring cybercrime

Richard Clayton, Ross Anderson

We first measured 'phishing' in 2007, showing that the newly introduced 'fast flux' techniques were allowing the criminals to keep their fake banking websites running for longer. The industry told us that the average lifetimes we observed were higher than they expected – we realised that we knew about sites that the banks did not and, not surprisingly, websites they hadn't been told about didn't get taken down. Our data about phishing websites was better than the banks because competing take-down companies would share information with us, but not with each other – we showed for the first time the true cost of this data hoarding.



We have continued to work on phishing, but have also been publishing measurement work on spam fighting, and recently on High Yield Investment Programmes – Ponzi schemes that pay outrageously high returns to investors from the money coming in from newer victims.

In 2012 we led a team of international experts to create the first systematic study of the costs of cybercrime. We found that the direct costs of traditional offences such as tax and welfare fraud (now mainly done 'online') cost the typical citizen in the low hundreds of pounds a year; transitional frauds (reinvented for cyberspace) cost a few pounds; while the new cyberspace-only crimes cost in the tens of pence. However, the indirect costs and defence costs are much higher for transitional and new crimes. Our figures suggest that we should spend less in anticipation of cybercrime (on antivirus, firewalls, etc.) and more in response – on the prosaic business of hunting down cybercriminals and throwing them in jail.

Ross Anderson, Chris Barton, Rainer Boehme, Richard Clayton, Michel J.G. van Eeten, Michael Levi, Tyler Moore, Stefan Savage, "Measuring the cost of cybercrime". WEIS12, June 25 2012.

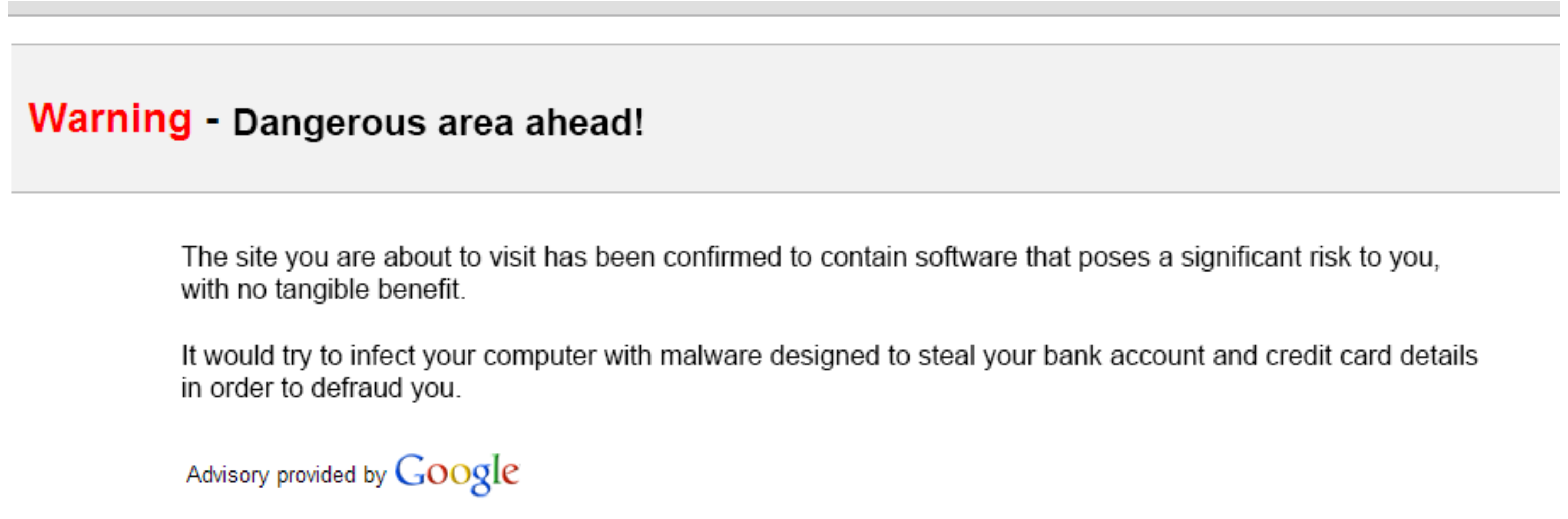
The psychology of deception

David Modic, Sophie van der Zee, Ross Anderson

We perform experiments on a range of topics in security usability, risk communication and deception online. For example, we have been working with an industrial funder on browser security warnings, which we see so often that we have mostly learned to ignore them. How can we make browser warnings fewer but better? We experimented with a range of warnings and found that many favourite 'nudges', such as appeals to authority and social compliance, do not work well. Even putting cartoon faces on browser warnings to activate social cognition doesn't give a significant improvement. What does work is making warnings more concrete. Telling users that a website will try to install a trojan that will steal their bank credentials gets much better compliance than a vague warning that "this site may harm your computer".

We are also developing a psychometric tool to measure people's susceptibility to persuasion. Our new scale combines established measures of susceptibility to factors such as need for cognition, sensation seeking, susceptibility to advertising, attitudes towards risky choices and so on. We will be testing it extensively to understand the causes of gullibility. Once we understand better which people are likely to fall for which scams, we can start thinking about what can be done by way of education, training or even automated scam detection.

We have a large project on the deterrence of deception in socio-technical systems with researchers at Portsmouth, Newcastle and UCL. In 2008 we founded the Workshop on Security and Human Behaviour, which brings psychologists together with security engineers.



Warning - Dangerous area ahead!

The site you are about to visit has been confirmed to contain software that poses a significant risk to you, with no tangible benefit.

It would try to infect your computer with malware designed to steal your bank account and credit card details in order to defraud you.

Advisory provided by Google

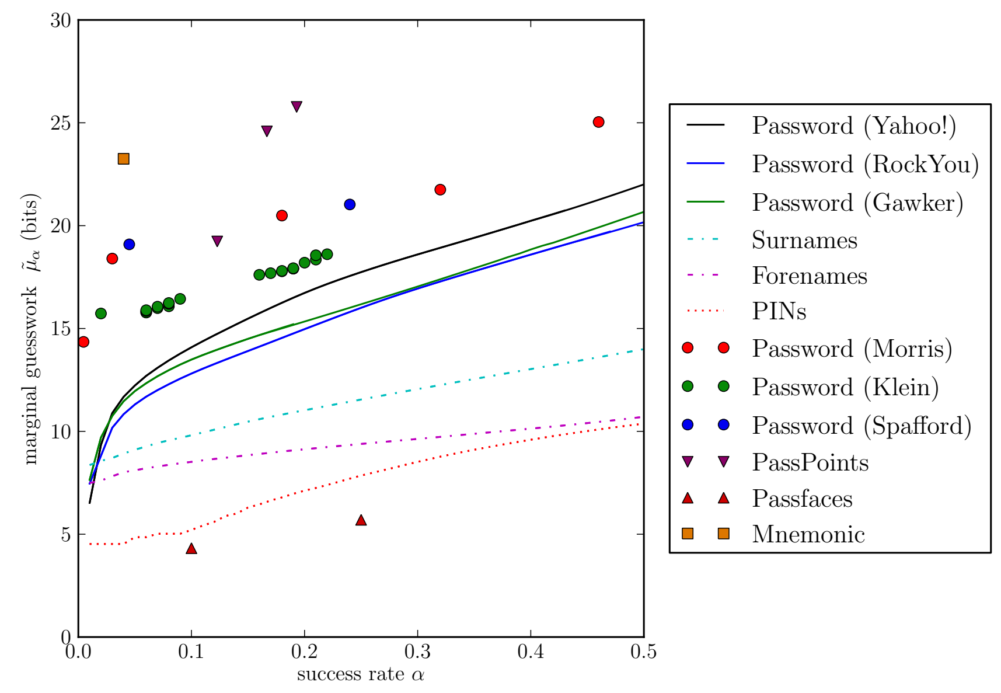
UNIVERSITY OF CAMBRIDGE

Authentication and identity (3/11)

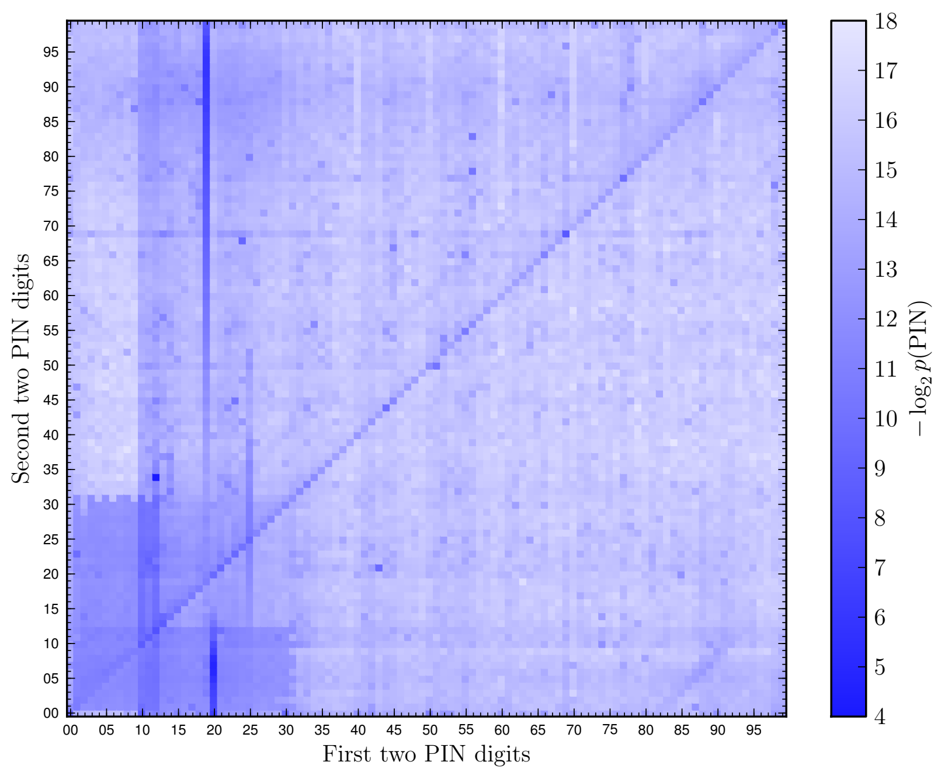
Human authentication in practice

Joseph Bonneau, Ross Anderson, Sören Preibusch

Secure and usable authentication remains elusive: decades of research have failed to move the world away from basic textual passwords. Yet, surprisingly little research attempted to answer basic questions such as: are human-chosen PINs or passwords easier to guess? What about personal knowledge questions such as "what is your mother's maiden name?" Do some users tend to pick better passwords than other users?



Recent research at Cambridge has begun to provide both sound methodology to answer these questions and reliable estimates from a collection of massive data sets. This effort has required collecting the largest-ever corpus (70M) of human-chosen passwords in a privacy-preserving manner, conducting the first large-scale surveys of human PIN choice, and collecting the world's largest corpora of human names (over 250M).



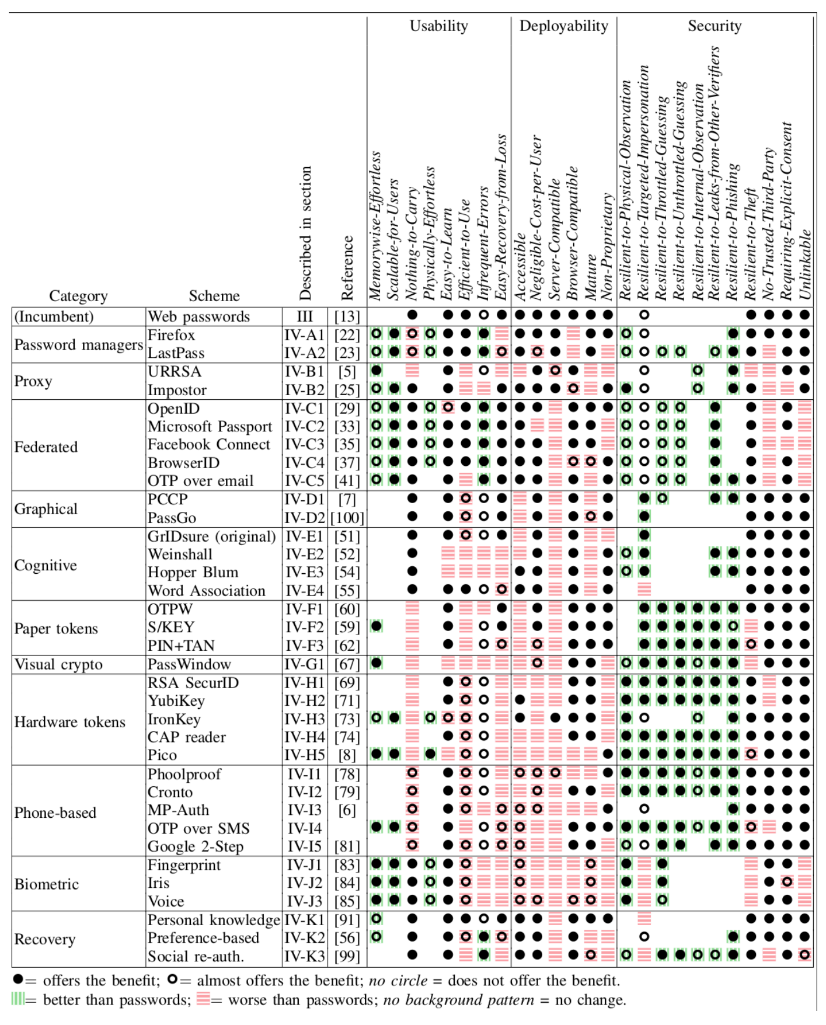
This project has provided some surprises, like older users picking stronger passwords than younger users or password strength policies making little impact on guessing difficulty. It has also provided some compelling numbers for security engineers to remember. Passwords are about equal to 10-bit random strings, or 3-digit decimal numbers, against a rate-limited attacker. If a thief steals a wallet with an ATM card and an ID listing date of birth, she has about a 10% chance of guessing the PIN.

Joseph Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords". 2012 IEEE Symposium on Security and Privacy.

A framework for comparative evaluation of password replacement schemes

Frank Stajano, Joseph Bonneau, Cormac Herley (Microsoft), Paul van Oorschot (Carleton)

Passwords have well-known security and usability problems. Over the past couple of decades, dozens of alternative schemes were proposed. Why, then, do we still use passwords so extensively? Don't the suggested replacements offer any improvements? This project offers a structured and well-researched answer.



				Usability								Deployability						Security										
Category	Scheme	Described in section	Reference	*Memorywise-Effortless*	*Scalable-for-Users*	*Nothing-to-Carry*	*Physically-Effortless*	*Easy-to-Learn*	*Efficient-to-Use*	*Infrequent-Errors*	*Easy-Recovery-from-Loss*	*Accessible*	*Negligible-Cost-per-User*	*Server-Compatible*	*Browser-Compatible*	*Mature*	*Non-Proprietary*	*Resilient-to-Physical-Observation*	*Resilient-to-Targeted-Impersonation*	*Resilient-to-Throttled-Guessing*	*Resilient-to-Unthrottled-Guessing*	*Resilient-to-Internal-Observation*	*Resilient-to-Leaks-from-Other-Verifiers*	*Resilient-to-Phishing*	*Resilient-to-Theft*	*No-Trusted-Third-Party*	*Requiring-Explicit-Consent*	*Unlinkable*
(Incumbent)	Web passwords	III	[13]			●		●	●	○	●	●	●	●	●	●	●		○						●	●	●	●
Password managers	Firefox	IV-A1	[22]	○	●	○	○	●	●	●		●	●	●		●	●	○	○					●	●	●	●	●
	LastPass	IV-A2	[23]	○	●	○	○	●	●	●	○	●	○	●		●		○	○	○	○		○	●	●		●	●
Proxy	URRSA	IV-B1	[5]	●				●		○			●	○	●				○			○		●			●	●
	Impostor	IV-B2	[25]	○	●	●		●			●	●	●	●	○		●	●	○			○		●	●			●
Federated	OpenID	IV-C1	[29]	○	●	●	○	○	●	●	●	●	●		●	●	●	○	○	○	○		●		●		●	
	Microsoft Passport	IV-C2	[33]	○	●	●	○	●	●	●	●	●			●	●		○	○	○	○		●		●		●	
	Facebook Connect	IV-C3	[35]	○	●	●	○	●	●	●	●	●	●		●	●		○	○	○	○		●		●			
	BrowserID	IV-C4	[37]	○	●	●	○	●	●	●	●	●	●		○	○	●	○	○	○	○		●		●		●	
	OTP over email	IV-C5	[41]	○	●	●		●		●	●	●	●		●		●	○	○	○	○		●	●	●		●	
Graphical	PCCP	IV-D1	[7]			●		●	○	○	●		●		●		●		●	○			●	●	●	●	●	●
	PassGo	IV-D2	[100]			●		●	○	○	●		●		●	○	●		●						●	●	●	●
Cognitive	GrIDsure (original)	IV-E1	[51]			●		●	○	○	●		●		●				●						●	●	●	●
	Weinshall	IV-E2	[52]			●							●		●		●	○	●				●	●	●	●	●	●
	Hopper Blum	IV-E3	[54]			●						●	●		●		●	○	●				●	●	●	●	●	●
	Word Association	IV-E4	[55]			●		●	●	○	○	●	●		●		●								●	●	●	●
Paper tokens	OTPW	IV-F1	[60]					●			●		●		●	●	●		●	●	●	●	●	●	●	●	●	●
	S/KEY	IV-F2	[59]	●				●		○	●		●		●	●	●		●	●	●	●	●	○		●	●	●
	PIN+TAN	IV-F3	[62]					●		○	○		○		●	●	●		●	●	●	●	●	●	○	●	●	●
Visual crypto	PassWindow	IV-G1	[67]	●									○		●	●		○	●	●	●	○	●	●		●	●	●
Hardware tokens	RSA SecurID	IV-H1	[69]					●	○	○					●	●		●	●	●	●	●	●	●	●		●	●
	YubiKey	IV-H2	[71]					●	○	○		●			●	●		●	●	●	●	●	●	●	●		●	●
	IronKey	IV-H3	[73]	○	●		○	○	○	○		●		●	●	●		●	○			○		●	●	●	●	●
	CAP reader	IV-H4	[74]					●	○	○					●	●		●	●	●	●	●	●	●	●	●	●	●
	Pico	IV-H5	[8]	●	●		●		○	○							●	●	●	●	●	●	●	●	○	●	●	●
Phone-based	Phoolproof	IV-I1	[78]			○		●	○	○		○	○	○			●	●	●	●	●	○	●	●	●	●	●	●
	Cronto	IV-I2	[79]			○		●	○	○			○		●	●		●	●	●	●	○	●	●	●	●	●	●
	MP-Auth	IV-I3	[6]			○		●	○		○	○	○				●		○					●	●	●	●	●
	OTP over SMS	IV-I4		●	●	○		●		○	○	○			●	●	●	●	●	●	●	○	●	●	○		●	●
	Google 2-Step	IV-I5	[81]			○		●	○	○	○	○			●	●		○	○	●	●		●	●	●	●	●	●
Biometric	Fingerprint	IV-J1	[83]	●	●	●	○	●	○			○				○		●		●						●	●	
	Iris	IV-J2	[84]	●	●	●	○	●	○			○				○		●		●						●	○	
	Voice	IV-J3	[85]	●	●	●	○	●	○			○	○		○	○		●		○						●	●	
Recovery	Personal knowledge	IV-K1	[91]	○		●		●	●	○	●	●	●		●	●	●								●	●	●	●
	Preference-based	IV-K2	[56]	○		●		●	○	●	○	●	●		●				○					●	●	●	●	●
	Social re-auth.	IV-K3	[99]			●		●		●		●	●		●	○		○		●	●	○	○	●	●		●	○

● = offers the benefit; ○ = almost offers the benefit; *no circle* = does not offer the benefit.
▥ = better than passwords; ▤ = worse than passwords; *no background pattern* = no change.

We arrange our data in a matrix. Across the columns we define a broad spectrum of 25 potential benefits in the areas of usability, security and deployability. Next, in the rows, we identify 35 representative schemes covering 11 broad categories. We then rate each scheme individually on whether it offers each benefit. The resulting matrix allows readers to compare features at a glance and to recognize general patterns.

Contrary to the optimistic claims of scheme authors, who often completely ignore some evaluation criteria, none of the examined schemes does better than passwords when rated on all 25 benefits of this objective benchmark.

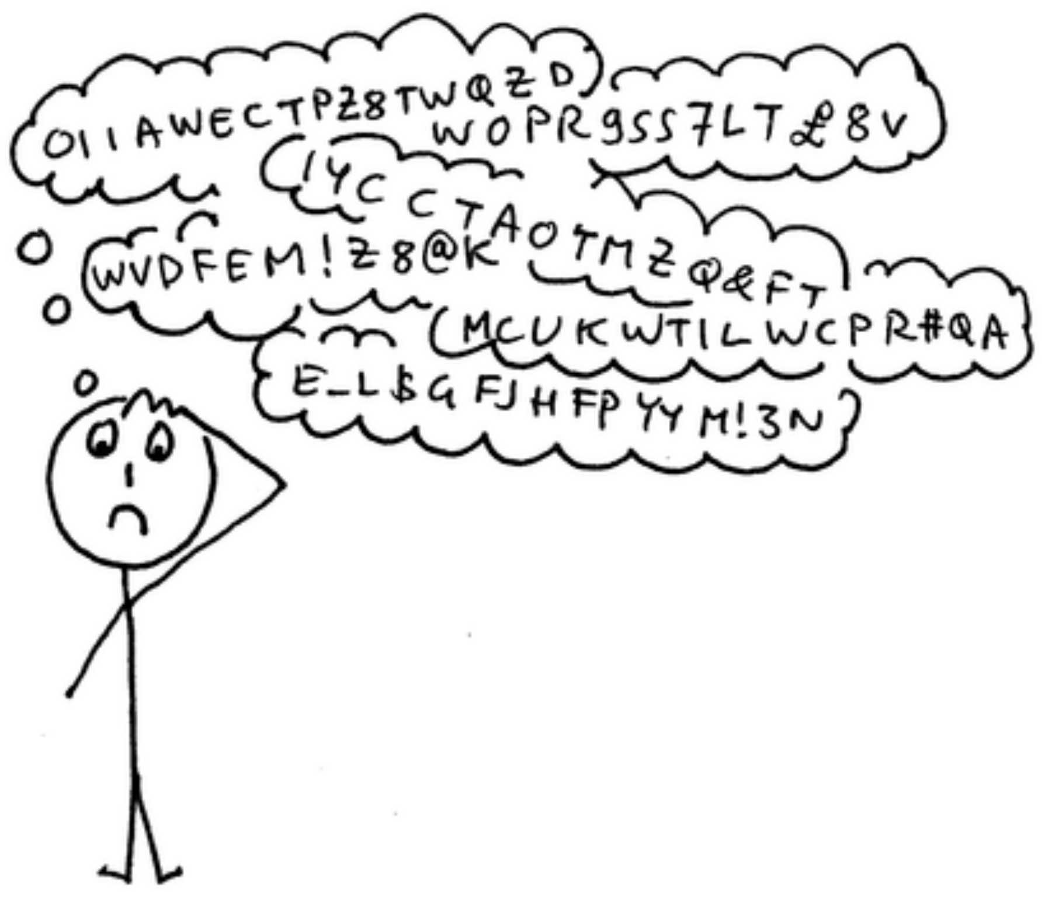
Many people repeat the mistakes of history because they didn't understand the history book. Here, we had to write one first! As was predicted during peer review, this widely cited work is now a foundational starting point for further research in the area and a useful sanity check for future password replacement proposals.

Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. "The quest to replace passwords: a framework for comparative evaluation of password replacement schemes". IEEE Security & Privacy 2012.

Pico: no more passwords!

Frank Stajano, Quentin Stafford-Fraser, Graeme Jenkinson, Max Spencer, Chris Warrington, Jeunese Payne

Users are told by security people that their passwords must be unguessable, must contain mixed-case letters, numbers and symbols, must not be written down, must all be different and must also be changed every couple of months. The intersection of all these constraints is the empty set. It's objectively impossible to follow all these directives at once – an unfair deal for users. As the number of online accounts per person keeps growing, passwords are not sustainable as a user authentication scheme.



Pico was designed so that you would not have to memorise secrets. It is a hardware token that can handle thousands of accounts. Besides usability, it also addresses many security problems: resisting brute-forcing, eavesdropping, phishing, keylogging etc. Pico unlocks in the presence of its owner by recognising miniature gadgets embedded in wearable items such as clothes and jewellery (Picosiblings). The user never has to type a PIN to unlock the Pico.

A competitive European Research Council grant worth over £1M funds the research team that is currently prototyping, validating and refining the design of Pico and will eventually produce a reference implementation, which will not be encumbered by patents or licensing fees.

Frank Stajano, "Pico: no more passwords!". Security Protocols Workshop 2011, LNCS 7114, Springer 2011.

The management of identity

David Evans (Derby), Jean Bacon, Alastair Beresford

Future applications will need to be able to identify places and objects, as well as people. We are interested in extending the traditional model of identity – one or more for each individual – to these scenarios. Here we bind context to person or a group, leading to a pseudonym that is linkable only by those knowing the context. This context might be an attribute of the physical location (such as bus number plates, pub names, or street names) or knowledge that is dependent on being in a particular location (e.g., the number of people standing on the pavement outside John Lewis).

A pseudonym for person *i* linkable by person *j* in context *n* is

$$P_{ij}^{(n)} = H_{K_{ij}}(\mathrm{ID}_i, C_n)$$

where K_{ij} is a key known only to users *i* and *j* and $H_K\langle\cdot\rangle$ is a message authentication code with key *K*. For a set of pseudonyms $G = \{P_1, P_2, \ldots\}$, $P_G = \langle P_1, P_2, \ldots\rangle_K$ where $\langle\cdot\rangle_K$ is a privacy-preserving set representation such as that by Hohenberger and Weis having key *K*.

Pseudonym Properties

- Given access to *P*, an attacker without the correct key cannot infer the ID or the context of the corresponding person.
- Given $P_{ij}^{(n)}$ and $P_{ij}^{(m)}$, user *j* can link the movement of user *i* from context C_n to C_m.
- Given access to $P_{ij}^{(n)}$, $P' = P_{ij}^{(m)}$ but not K_{ij} an attacker cannot determine that *P* and *P'* represent the same person.
- Given access to P_G and *K*, a user can check whether specific pseudonyms are in *G*.
- Given P_G and *K*, a user knowing the keys to some of the pseudonyms in *G* learns nothing about the status of pseudonyms whose keys are unknown.
- Unrestricted distribution of individual or group pseudonyms does not compromise real-world identity.

Hardware security and signal analsysis (4/11)

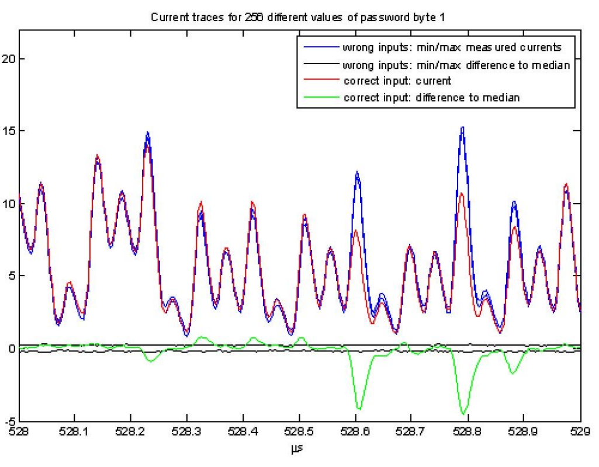
Hardware tamper resistance

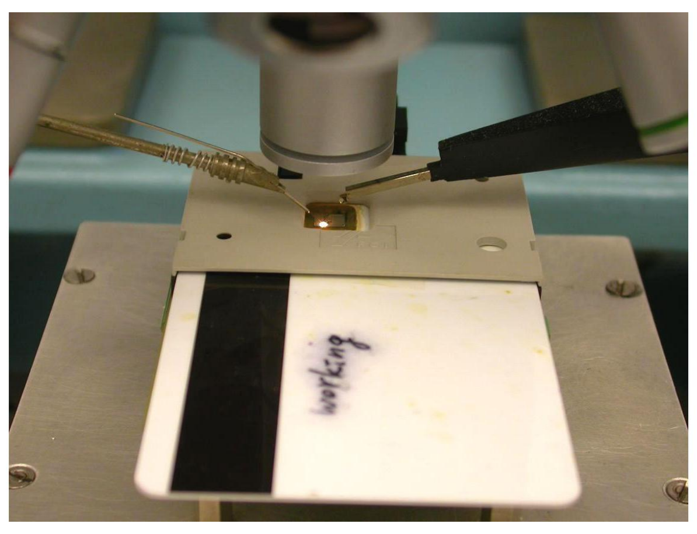
Sergei Skorobogatov, Markus G. Kuhn

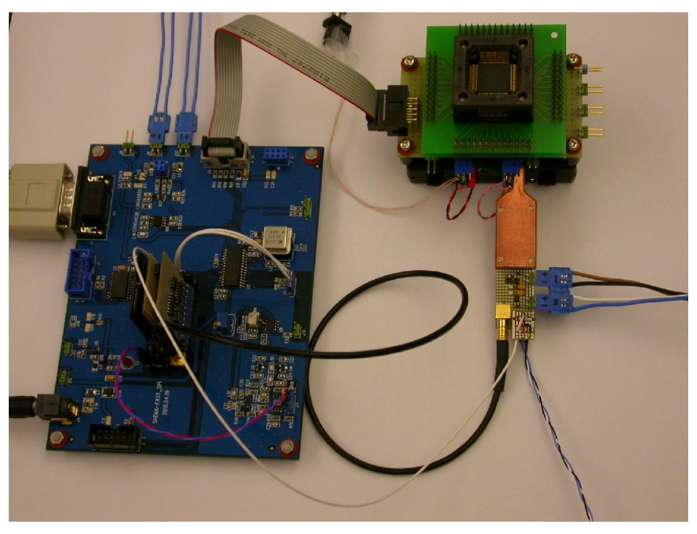
Protection against physical attacks has become an essential part of many modern system designs. These days we have a continuous battle between manufacturers who invent new security solutions learning from previous mistakes, and a hacker community that is constantly trying to break protection in various devices. The importance of security is dictated by the amount of valuable and sensitive information stored on the chip. This could be cryptographic keys, secret data, company secrets, intellectual property, electronic money or banking smartcards.

Our group have invented semi-invasive attacks (optical fault injection) which have forced the industry to rethink protection mechanisms in smartcards and amend Common Criteria requirements. As with invasive attacks (microprobing, chip modification), they require opening the chip in order to get access to its surface without destroying it or creating contacts to internal wires. Those attacks are as easy to implement as inexpensive non-invasive attacks (power analysis, glitching).

In collaboration with Quo Vadis Labs we developed a new side-channel analysis technique. This breakthrough approach means it is now possible to extract encryption keys from devices and systems up to a million times faster than state-of-the-art power analysis techniques, e.g. DPA. Our recent research is focused on Hardware Assurance – testing of silicon chips for backdoors and trojans. Because of the speed at which analysis can be performed, it becomes possible to identify hidden backdoors and trojans in silicon chips – a task that is not feasible with current DPA methods.





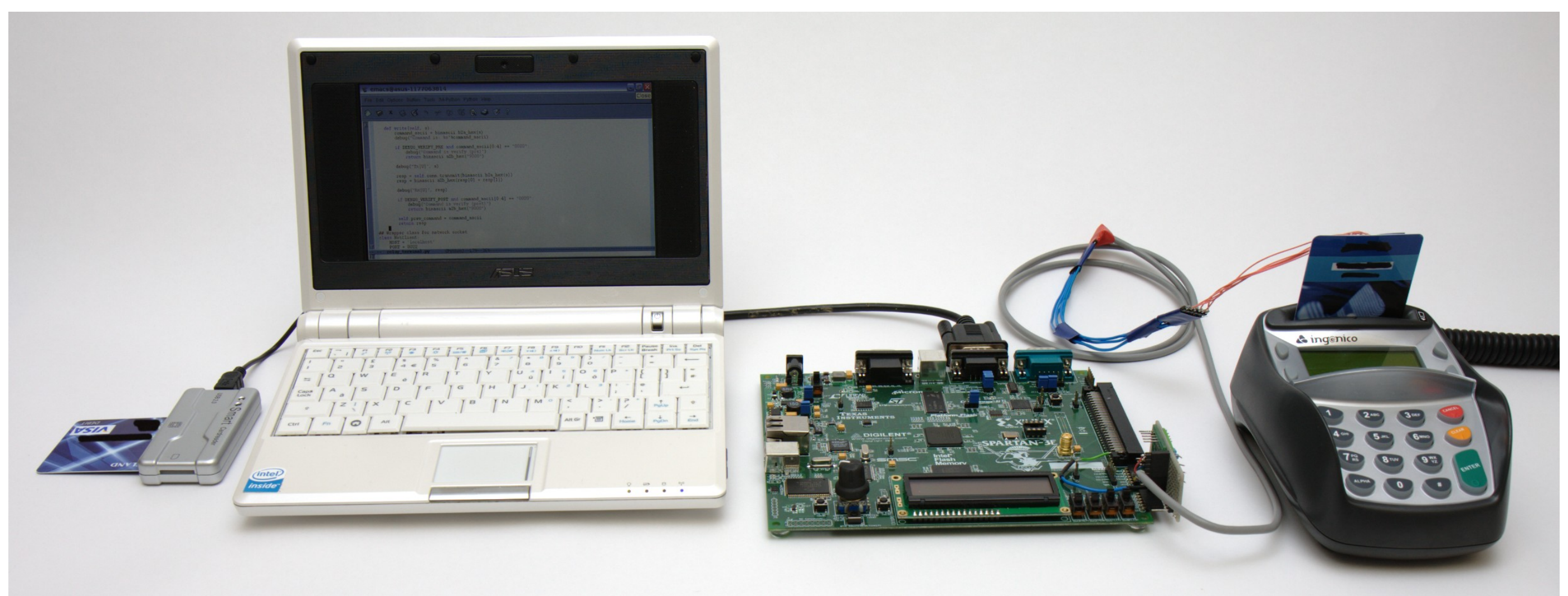


Banking and payment system security

Ross Anderson, Mike Bond, Omar Choudary, Steven J. Murdoch, Laurent Simon

Known in the UK as “Chip and PIN”, EMV (Europay, MasterCard, Visa) is the dominant standard for smart-card payments worldwide. Introduced to reduce card fraud, EMV is used throughout Europe; it is being introduced in the US, Canada and South America. EMVCo estimates that over a billion EMV payment cards are in circulation. EMV makes card transactions more secure by adding a chip to cards to make them harder to counterfeit and requiring customers to enter a PIN to authorize payment. While initially reducing fraud, criminals adapted to the change, resulting in increased losses.

Research at Cambridge has discovered numerous vulnerabilities in the deployed Chip and PIN system, including the ability for criminals to trick terminals into accepting an incorrect PIN for a stolen card, failures in the tamper resistance measures present in widely deployed Chip and PIN terminals, and ways for corrupt bank employees circumvent protections against insider attacks to discover customer PINs. We have developed methods to resolve these vulnerabilities and work with industry to have these improvements deployed. Smart Architects Ltd sell auditing equipment, developed at Cambridge, to detect these vulnerabilities. Methods for securing online banking against man-in-the-middle malware, developed at Cambridge, have been commercialised by spin-out company Cronto Ltd, since acquired by VASCO, and are in use at several banks.



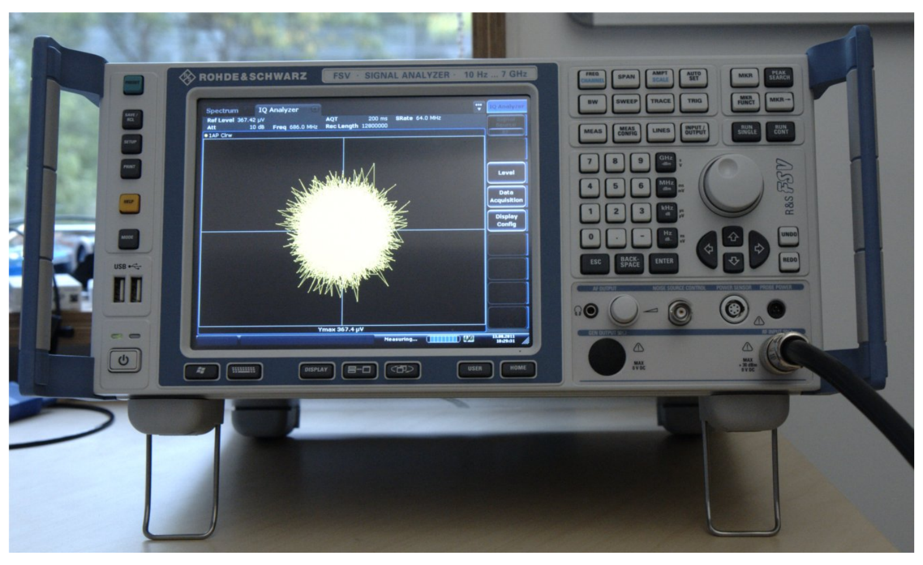
Compromising emanations and side channels

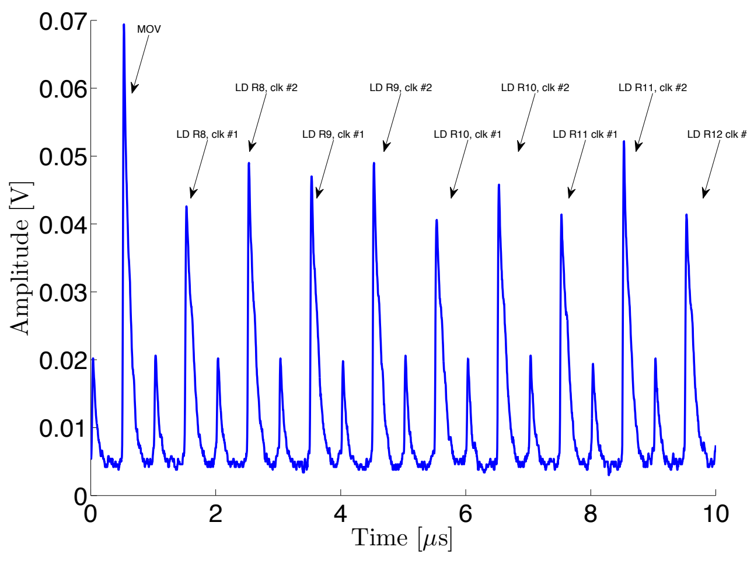
Markus G. Kuhn, Omar Choudary

Computers unintentionally emit information about the data they process on various parasitic analog side channels, such as high-frequency current fluctuations on power-supply lines, electromagnetic fields, as well as audible, ultrasonics and light emissions. These signals can be picked up nearby and decoded by well-equipped eavesdroppers, and used to circumvent cryptographic and access-control security mechanisms. We investigated the nature of compromising emanations of analog and digital computer displays, as well as television sets, and developed software techniques to manipulate these. We work on civilian emission-security standards, including one for voting machines by the Dutch government.

We optimized template attacks on security microcontrollers, a powerful statistical signal-processing technique to extract data from instruction-specific information leakage. The attacker first builds a high-dimensional multivariate model of both the information leakage and noise while observing the processing of known data, and then uses that model to perform maximum-likelihood estimation of unknown processed data in products involving the same type of processor.

We provide advice to manufacturers of TEMPEST equipment for government applications that require protection against electromagnetic eavesdropping. We preparing new digital signal-processing techniques to improve industrial emission-security measurement processes.



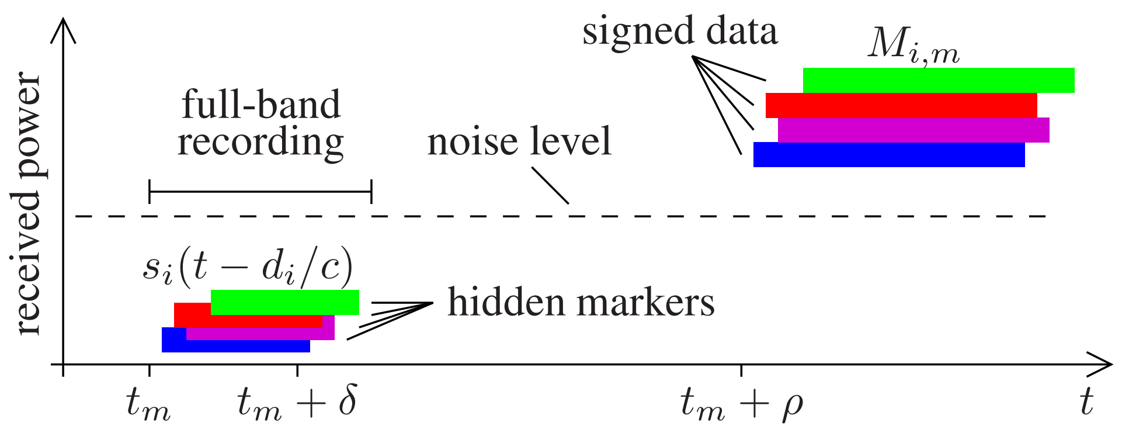


Omar Choudary, Markus G. Kuhn: Efficient Template Attacks. CARDIS 2013.
Markus G. Kuhn: Compromising emanations of LCD TV sets. IEEE Trans. on Electromagnetic Compatibility, Vol. 55, No. 3, June 2013.

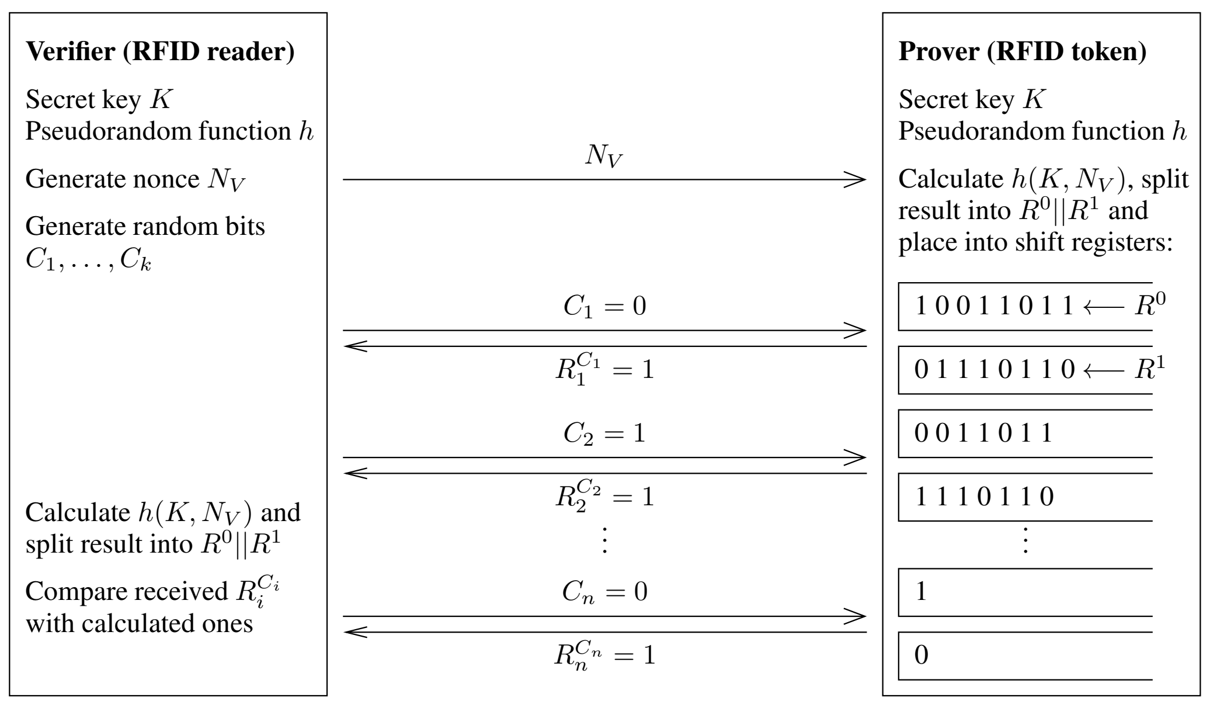
Location security: cryptography at the speed of light

Markus G. Kuhn

Global navigation satellite systems still lack signal-integrity protection for mass-market applications. A particular challenge is the need to protect not only the integrity of broadcast data, but also the nanosecond-accurate relative arrival times of signals from different satellites, from which receivers determine their position. We have proposed several new types of signal-authentication and spoofing detection techniques. A steganographic transmission scheme with delayed release of spreading keys provides the same asymmetric security that made digital signatures so useful: the ability to verify a signal does not lead to the ability to spoof it. This is of particular importance in civilian mass-market applications where the holder of the receiver may want to manipulate its reading: road toll, pay-as-you-drive car insurance, or offender-tagging. In addition, we have also proposed heuristics that help GPS receivers detected spoofed signals.



Distance-bounding protocols ascertain both the identity of a communications partner and also their location. They securely answer questions such as “is this smartcard really within 1 metre of the reader?”, by incorporating a rapid exchange of single challenge-response bits, where the speed of light is the main part of the round-trip time. We proposed the first distance-bounding protocol optimized for RFID applications with noisy wide-band channels.



Verifier (RFID reader)

Secret key K
Pseudorandom function h

Generate nonce N_V

Generate random bits $C_1, \ldots, C_k$

Calculate $h(K, N_V)$ and split result into $R^0||R^1$

Compare received $R_i^{C_i}$ with calculated ones

N_V

$C_1 = 0$

$R_1^{C_1} = 1$

$C_2 = 1$

$R_2^{C_2} = 1$

$\vdots$

$C_n = 0$

$R_n^{C_n} = 1$

Prover (RFID token)

Secret key K
Pseudorandom function h

Calculate $h(K, N_V)$, split result into $R^0||R^1$ and place into shift registers:

1 0 0 1 1 0 1 1 ⟵ R^0
0 1 1 1 0 1 1 0 ⟵ R^1
0 0 1 1 0 1 1
1 1 1 0 1 1 0
⋮
1
0

$\langle C_i \rangle = 01001100$ will return $\langle R_i^{C_i} \rangle = 11010111$

Digital forensics and malware analysis (5/11)

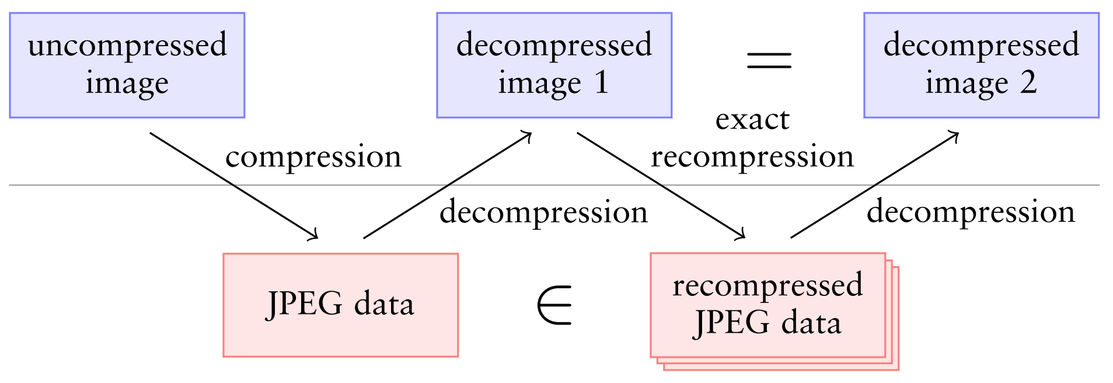
Forensic signal analysis

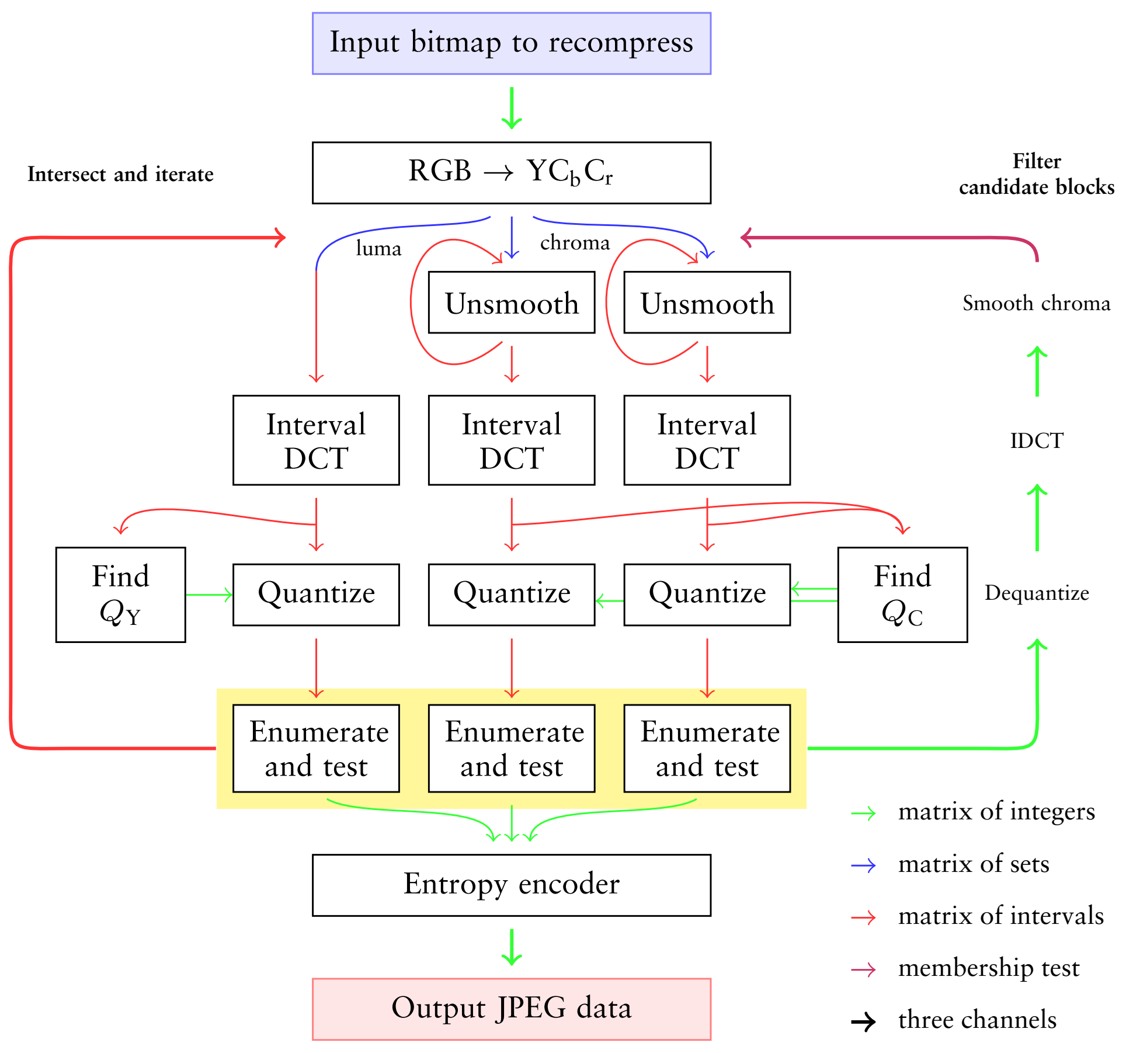
Markus G. Kuhn, Andrew B. Lewis

Widespread use of digital cameras, compression, high-capacity storage, and advanced tools (video editors, SDR, etc.) pose new challenges to forensic investigators, who need techniques to confirm claimed origins and processing histories of signal evidence, and recover data without the cooperation of the hardware owner or designer.

In collaboration with the Metropolitan Police, we designed a sophisticated tool for recovering compressed video data from fragmented storage where file-system metadata is unavailable. Manufacturers are unwilling to release documentation for proprietary file-systems, block allocation, and Flash wear-leveling algorithms, leaving investigators with a random puzzle of 4KB memory blocks to be assembled back into a video stream (H.264, etc.), a task that our new high-performance parser can handle for gigabytes of data.

Our *exact JPEG recompressor* uses iterative interval arithmetic to invert a JPEG decompressor, recreating the original compressed bitstream. It can help to identify/exclude particular decoders as a source, but also has applications in evaluating copy-protection schemes.





Andrew B. Lewis, Markus G. Kuhn, "Exact JPEG recompression". Proc. SPIE 7543, Visual Information Processing and Communication, 75430V, Januar 2010.
Andrew B. Lewis, "Reconstructing compressed photo and video data". Computer Laboratory Technical report UCAM-CL-TR-813, February 2012.

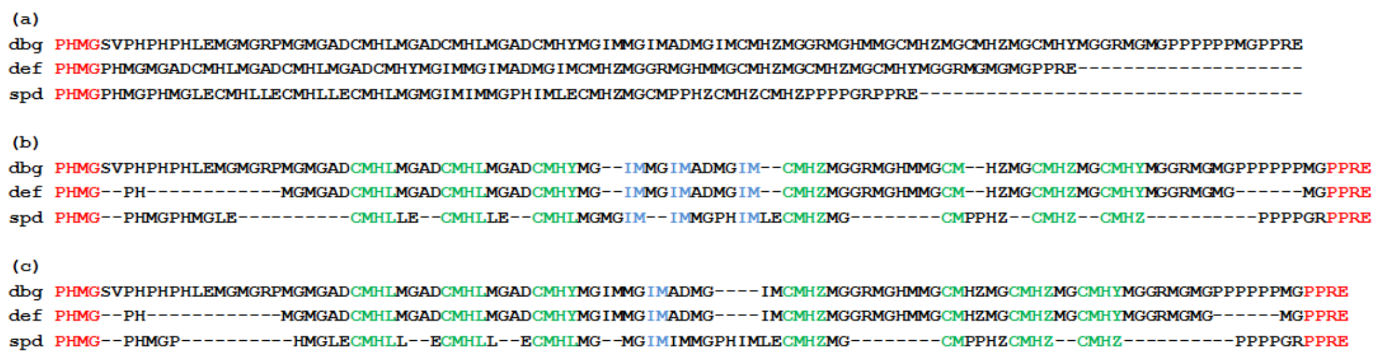
Phylogenetic-inspired techniques for malware analysis: attacks and defence

Wei Ming Khoo, Hyoungshick Kim, Pietro Liò

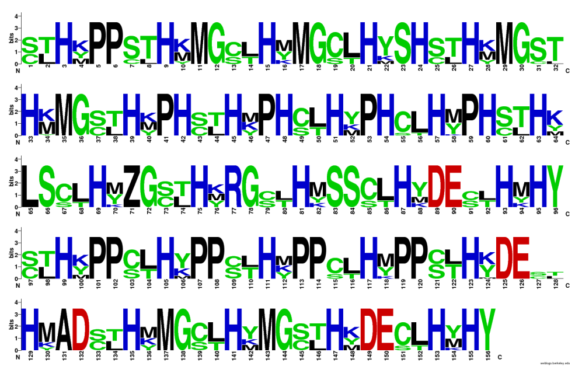
Today's malware is written to be persistent. Financial incentives are the dominant motivation for writing and spreading malware, and making sure that the malware remains as long as possible on the victims' machines. As a result of this, malware exist in families, often numbering in the thousands, in order to constantly evade antivirus products and operating system defences. However, malware is seldom written from scratch. Because new malware variants are usually inspired by previous ones, at some level they show a convergence of functionality.

We developed a framework for abstracting, aligning and analysing malware execution traces and are exploring the use of state of the art phylogenetic methods, whose strengths lie in pattern recognition and visualisation, to derive the statistical relationships within contemporary malware families. Some of methods used include phylogenetic trees and networks, motifs, logos, composition biases, and tree topology comparison methods.

Sequence alignment algorithms have recently found a use in detecting code clones, software plagiarism, code theft, and detecting polymorphic malware. This approach involves extracting software birthmarks, in this case sequences, from programs and comparing them using sequence alignment, a procedure which has been intensively studied in the field of bioinformatics. While sequence alignment may cope well with accidental DNA and protein mutations, we have shown that it can be vulnerable to specific insertion and deletion schemes and to concurrent programming.



```
(a)
dbg PHMGSVPHPHPHLEMGMGRPMGMGADCMHLMGADCMHLMGADCMHYMGIMMGIMADMGIMCMHZMGGRMGHMMGCMHZMGCMHZMGCMHYMGGRMGMGPPPPPPMGPPRE
def PHMGPHMGMGADCMHLMGADCMHLMGADCMHYMGIMMGIMADMGIMCMHZMGGRMGHMMGCMHZMGCMHZMGCMHYMGGRMGMGMGPPRE--------------------
spd PHMGPHMGPHMGLECMHLLECMHLLECMHLMGMGIMIMMGPHIMLECMHZMGCMPPHZCMHZCMHZPPPPGRPPRE----------------------------------

(b)
dbg PHMGSVPHPHPHLEMGMGRPMGMGADCMHLMGADCMHLMGADCMHYMG--IMMGIMADMGIM--CMHZMGGRMGHMMGCM--HZMGCMHZMGCMHYMGGRMGMGPPPPPPMGPPRE
def PHMG--PH------------MGMGADCMHLMGADCMHLMGADCMHYMG--IMMGIMADMGIM--CMHZMGGRMGHMMGCM--HZMGCMHZMGCMHYMGGRMGMG------MGPPRE
spd PHMG--PHMGPHMGLE---------CMHLLE--CMHLLE--CMHLMGMGIM--IMMGPHIMLECMHZMG--------CMPPHZ--CMHZ--CMHZ----------PPPPGRPPRE

(c)
dbg PHMGSVPHPHPHLEMGMGRPMGMGADCMHLMGADCMHLMGADCMHYMGIMMGIMADMG----IMCMHZMGGRMGHMMGCMHZMGCMHZMGCMHYMGGRMGMGPPPPPPMGPPRE
def PHMG--PH------------MGMGADCMHLMGADCMHLMGADCMHYMGIMMGIMADMG----IMCMHZMGGRMGHMMGCMHZMGCMHZMGCMHYMGGRMGMG------MGPPRE
spd PHMG--PHMGP----------HMGLECMHLL--ECMHLL--ECMHLMG--MGIMIMMGPHIMLECMHZMG--------CMPPHZCMHZ--CMHZ----------PPPPGRPPRE
```



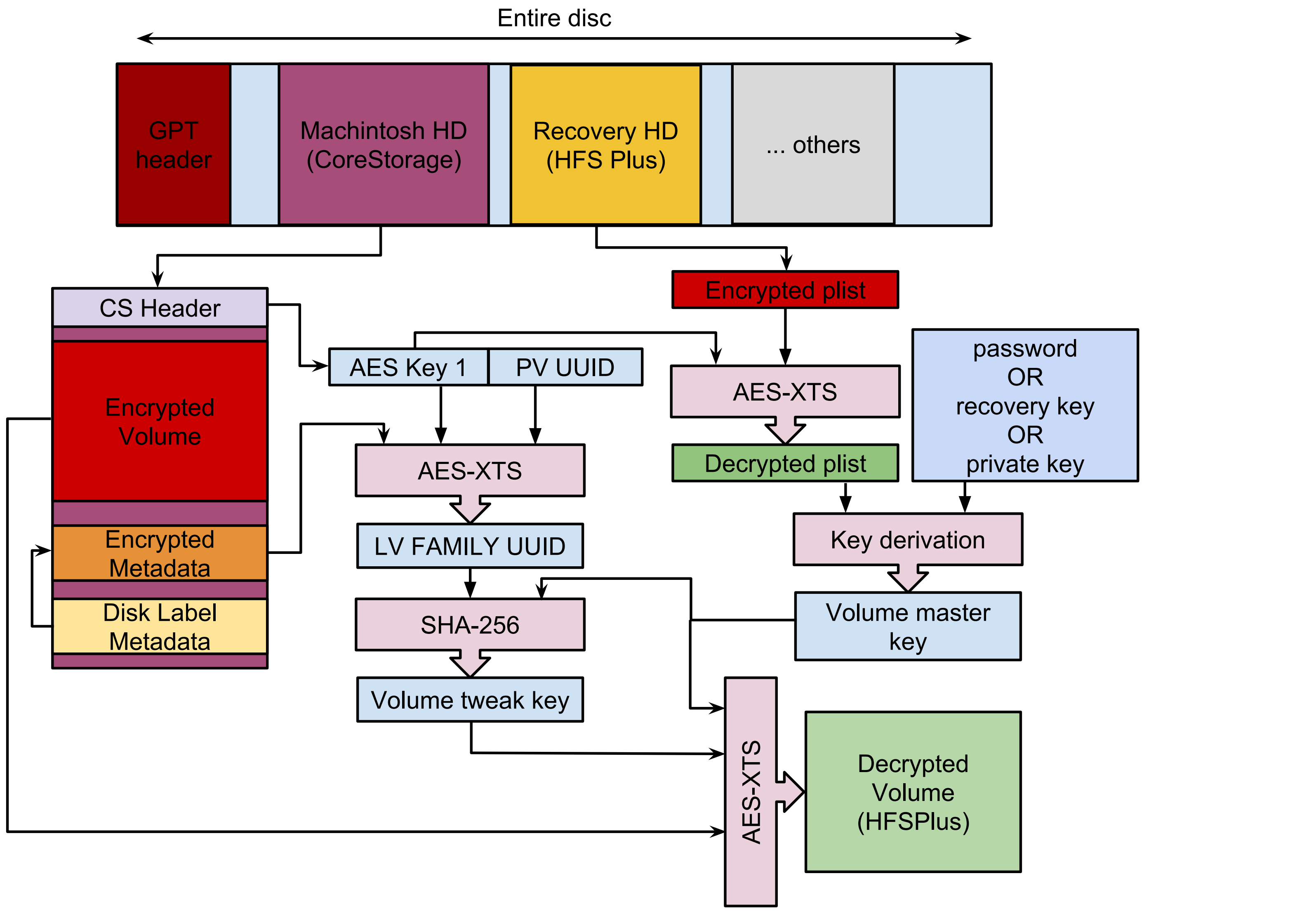
Wei Ming Khoo, Pietro Lio, "Unity in diversity: Phylogenetic-inspired techniques for reverse engineering and detection of malware families". 1st SysSec Workshop, 2011.

libfvde: analysis and forensic tools for FileVault 2

Omar Choudary, Felix Grobert (Google) and Joachim Metz (Google)

With Mac OS X 10.7 (Lion), Apple has introduced a volume encryption mechanism known as FileVault 2. We performed a detailed analysis of FileVault 2 which allows organisations and individuals to understand the mechanisms behind this system and be aware of the security level provided.

We have developed an open-source tool that can read the data from FileVault 2 encrypted volumes, when some recovery token is given. This tool is platform independent and is particularly useful to digital forensic investigators, which need to analyse data from a disk without relying on a possibly compromised operating system. Several people are contributing to this project in order to make the tool compatible with newer versions of Mac OS X and to provide additional functionality.

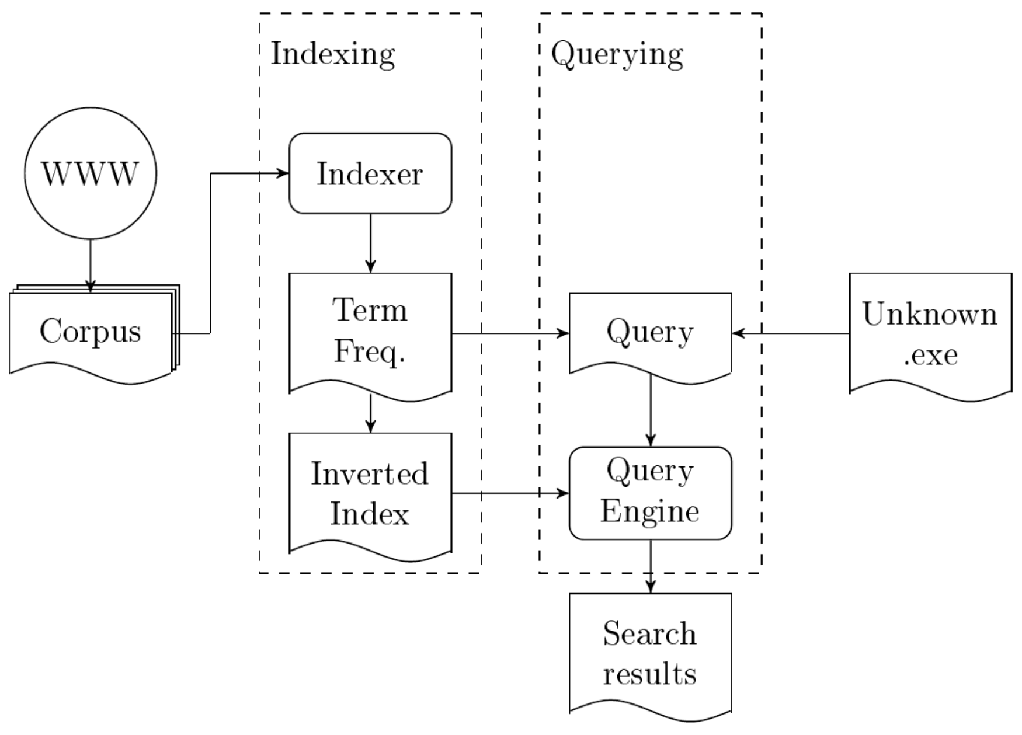


Omar Choudary, Felix Grobert and Joachim Metz. Security Analysis and Decryption of FileVault 2. In Advances in Digital Forensics IX – 9th IFIPWG 11.9 International Conference on Digital Forensics, Springer, 2013.

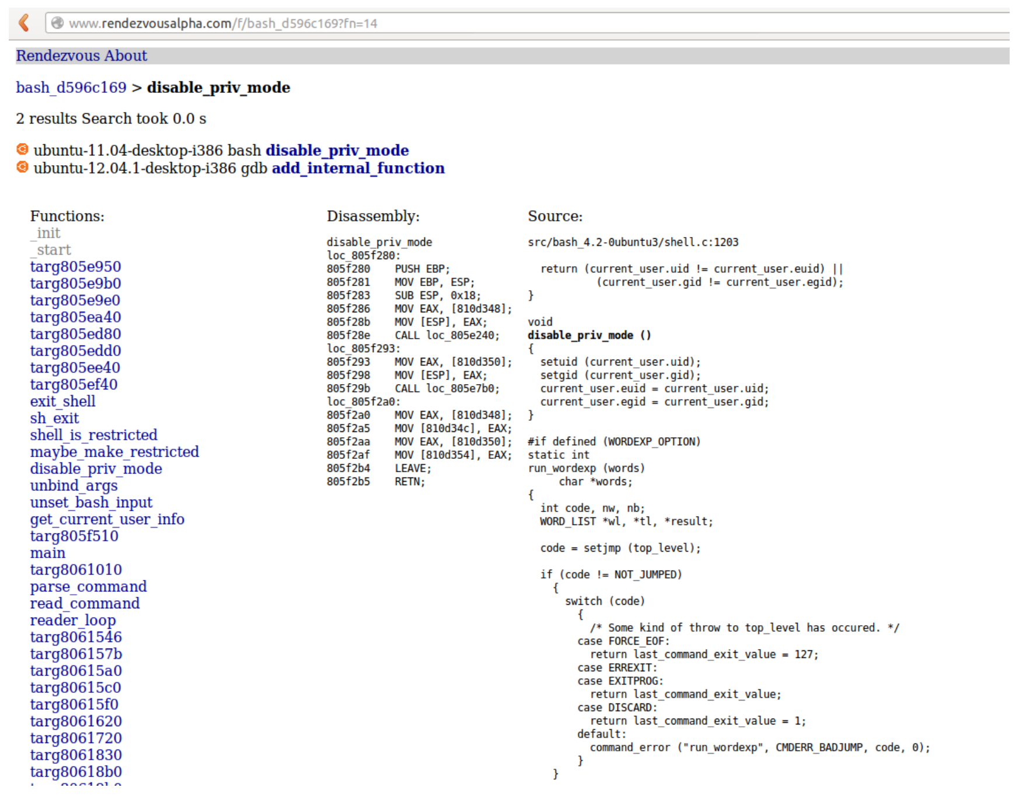
Rendezvous: a search engine for binary code

Wei Ming Khoo, Ross Anderson, Alan Mycroft

Software re-use is prevalent because of competitive time-to-market for new products, the sheer complexity of the application domain and the availability of high-quality re-usable software components. Currently, an auditor seeking to identify re-used components in a software product, such as for GPL-compliance, has two main approaches available: source code review and software reverse engineering. However, source code availability is not always guaranteed, especially if the code was developed by a third party, and understanding machine code is at present rather challenging.



Our approach bootstraps the process by providing a search engine for binary code. By leveraging open-source initiatives such as GNU, the Apache foundation, Linux and BSD distributions, and public code repositories such as Github, we can reframe identifying code reuse as an indexing and search problem. Experiments show that Rendezvous achieves F2 measures of 86.7% and 83.0% on the GNU C library compiled with different compiler optimisations and the GNU coreutils suite compiled with gcc and clang respectively. These two code bases together comprise more than one million lines of code. Rendezvous will bring significant changes to the way patch management and copyright enforcement is currently performed.



www.rendezvousalpha.com/f/bash_d596c169?fn=14

Rendezvous About

bash_d596c169 > **disable_priv_mode**

2 results Search took 0.0 s

- ubuntu-11.04-desktop-i386 bash **disable_priv_mode**
- ubuntu-12.04.1-desktop-i386 gdb **add_internal_function**

Functions:

```
_init
_start
targ805e950
targ805e9b0
targ805e9e0
targ805ea40
targ805ed80
targ805edd0
targ805ee40
targ805ef40
exit_shell
sh_exit
shell_is_restricted
maybe_make_restricted
disable_priv_mode
unbind_args
unset_bash_input
get_current_user_info
targ805f510
main
targ8061010
parse_command
read_command
reader_loop
targ8061546
targ806157b
targ80615a0
targ80615c0
targ80615f0
targ8061620
targ8061720
targ8061830
targ80618b0
```

Disassembly:

```
disable_priv_mode
loc_805f280:
805f280     PUSH EBP;
805f281     MOV EBP, ESP;
805f283     SUB ESP, 0x18;
805f286     MOV EAX, [810d348];
805f28b     MOV [ESP], EAX;
805f28e     CALL loc_805e240;
loc_805f293:
805f293     MOV EAX, [810d350];
805f298     MOV [ESP], EAX;
805f29b     CALL loc_805e7b0;
loc_805f2a0:
805f2a0     MOV EAX, [810d348];
805f2a5     MOV [810d34c], EAX;
805f2aa     MOV EAX, [810d350];
805f2af     MOV [810d354], EAX;
805f2b4     LEAVE;
805f2b5     RETN;
```

Source:

```
src/bash_4.2-0ubuntu3/shell.c:1203

  return (current_user.uid != current_user.euid) ||
         (current_user.gid != current_user.egid);
}

void
disable_priv_mode ()
{
  setuid (current_user.uid);
  setgid (current_user.gid);
  current_user.euid = current_user.uid;
  current_user.egid = current_user.gid;
}

#if defined (WORDEXP_OPTION)
static int
run_wordexp (words)
     char *words;
{
  int code, nw, nb;
  WORD_LIST *wl, *tl, *result;

  code = setjmp (top_level);

  if (code != NOT_JUMPED)
    {
      switch (code)
        {
          /* Some kind of throw to top_level has occured. */
        case FORCE_EOF:
          return last_command_exit_value = 127;
        case ERREXIT:
        case EXITPROG:
          return last_command_exit_value;
        case DISCARD:
          return last_command_exit_value = 1;
        default:
          command_error ("run_wordexp", CMDERR_BADJUMP, code, 0);
        }
    }
```

Wei Ming Khoo, Alan Mycroft, Ross Anderson, "Rendezvous: A search engine for binary code". 10th Working Conference on Mining Software Repositories (MSR'13), May 2013.

Operating-system security research

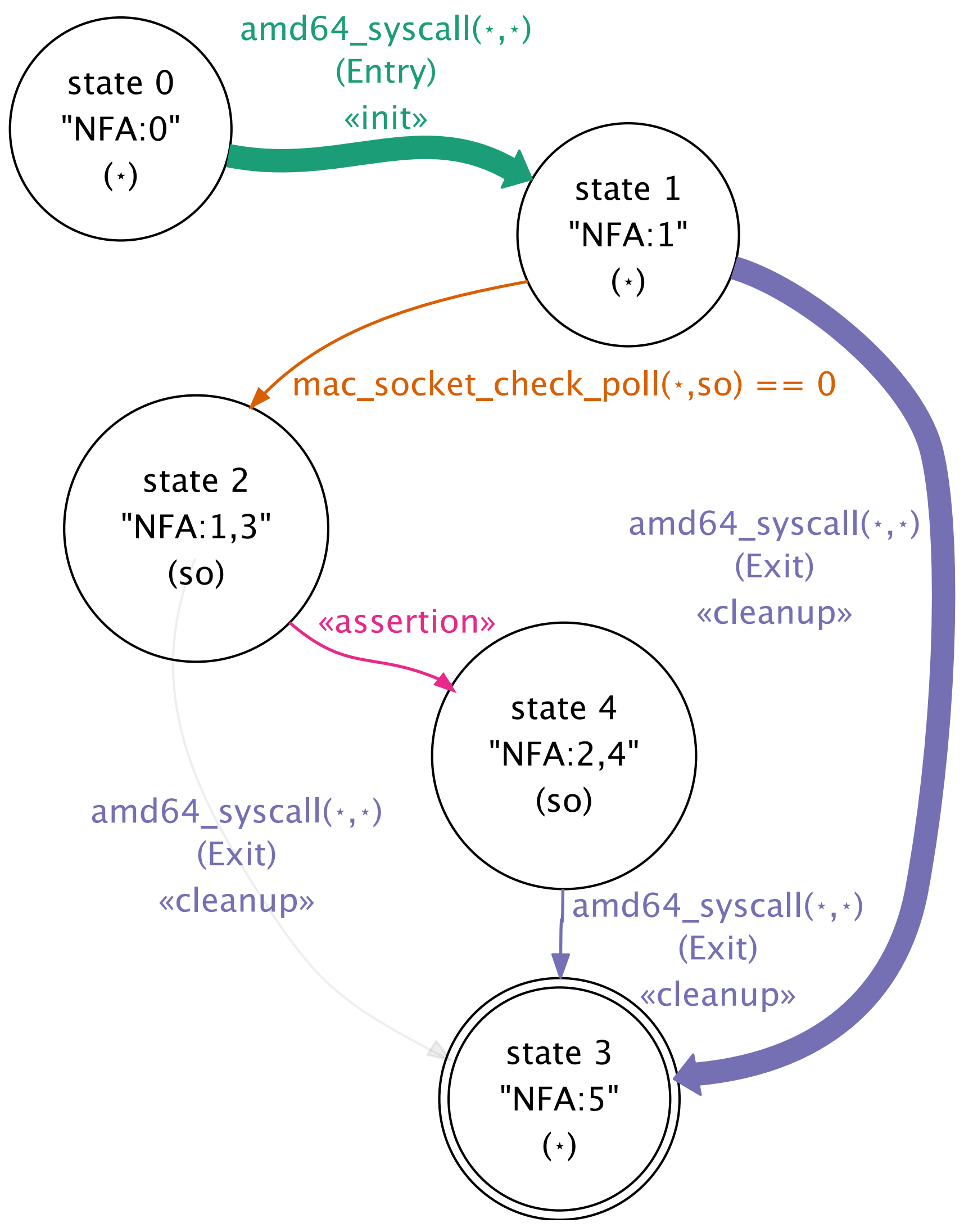
Jonathan Anderson, Ross Anderson, Jean Bacon, Alistair Beresford, David Chisnall, Jon Crowcroft, Brooks Davis (SRI), Khilan Gudka, Wei Ming Khoo, Ben Laurie (Google), Pietro Liò, Anil Madhavapeddy, Steven J. Murdoch, Andrew Rice, Michael Roe, Laurent Simon, Daniel Thomas, Daniel Wagner, Robert N. M. Watson, Rubin Xu

Operating systems and programming language runtimes are key trusted computing bases (TCBs) – the minimal subset of a system that must be correct for it to be secure. We are exploring many techniques to improve OS security, including virtualisation, new security models, blending formal methods with engineering, and clean-slate redesigns:

- **Aurasium** is an Android application sandboxing system that imposes security and privacy policies through application transformation.
- **Capsicum** blends historic capability-system models with contemporary OS design. Now shipping in FreeBSD, Capsicum supports application compartmentalisation, a key vulnerability mitigation technique.
- **CHERI** is a capability-extended RISC CPU supporting hardware memory safety and highly scalable application compartmentalisation.
- The **MAC Framework** is an extensible access-control framework shipping in FreeBSD, Mac OS X, iOS, Junos, and Sidewinder.
- **Mirage** is a clean-slate OS written in OCaml. Implementing the OS in a type-safe, functional language offers security and reliability improvements through access to formal verification and immunity to many traditional vulnerabilities.
- **SOAAP** is a suite of analysis tools to assist in identifying and semi-automatically applying application compartmentalisation.
- **TESLA** is a framework for validating temporal security properties in software TCBs that cannot be checked using current ‘instantaneous’ software assertion techniques.
- **Xen** is a hypervisor for full system virtualisation. We have released Xen as open source and it is now widely used for security and cloud computing systems such as Amazon EC2 and Rackspace.

TESLA: temporal security assertion language

Jonathan Anderson, Robert N. M. Watson, David Chisnall, Khilan Gudka, Brooks Davis (SRI)



Currently, we cannot prove complex software such as OS kernels correct. Instead, OS developers describe the expected behaviour of their systems with assertions and validate with testing. Many security properties cannot be validated this way because they are temporal: they depend on events in the past or the future.

TESLA (Temporally Enhanced Security Logic Assertions) is a tool for describing, analysing and validating the temporal behaviour of complex systems that are written in low-level languages like C. Temporal assertions, which are inspired by Linear Temporal Logic (LTL), can span the interfaces between libraries and even languages. Program instrumentation exposes runtime behaviour, converting it into events that drive complex state machines, expose application behaviour, and reveal specification violations.

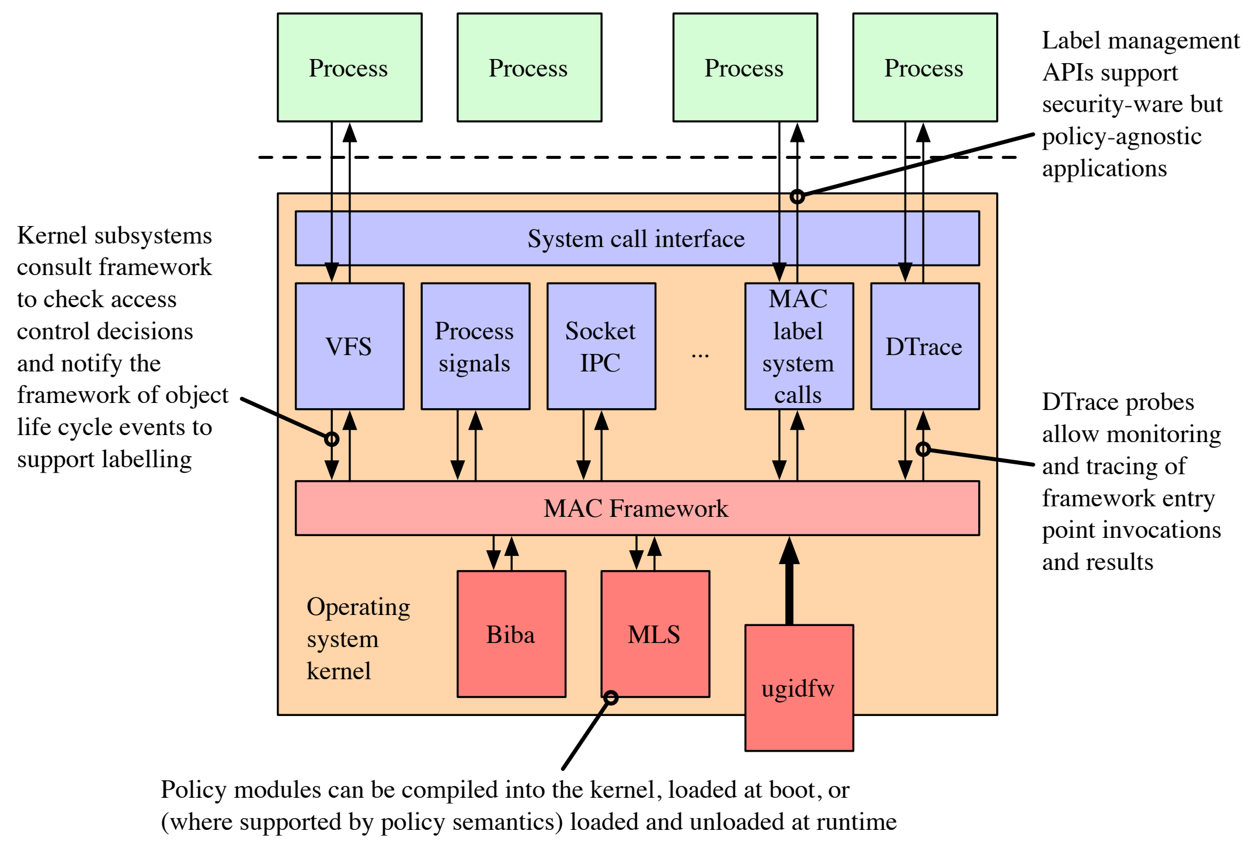
TESLA has detected API security vulnerabilities in OpenSSL client code, found security bugs in the FreeBSD kernel and helped us understand subtle (and previously-undiagnosed) bugs in a complex UI framework. TESLA brings powerful dynamic analysis and visualisation tools to practical development environments; its performance allows for ‘always-in’ availability in systems development. This open-source tool is available today to systems developers without demanding software be rewritten for formal analysis.

http://www.cl.cam.ac.uk/research/security/ctsrd/tesla/
https://github.com/CTSRD-TESLA/TESLA

Operating-system access-control extensibility

Robert N. M. Watson

In the 1990s, operating-system (OS) security was focussed on multi-user systems: authentication, audit, access-control lists, and mandatory access-control (MAC) schemes such as multi-level security and the Biba integrity model. By the 2007 releases of Apple iOS and Google Android, OS security had been transformed by ubiquitous Internet access, 3rd-party applications on embedded/mobile devices, and a shift from multi-user security to single-user application sandboxing. This transition was supported by the adoption of *extensible access-control frameworks*, which allow OS kernels to be adapted to new security requirements.



Label management APIs support security-ware but policy-agnostic applications

Kernel subsystems consult framework to check access control decisions and notify the framework of object life cycle events to support labelling

DTrace probes allow monitoring and tracing of framework entry point invocations and results

Policy modules can be compiled into the kernel, loaded at boot, or (where supported by policy semantics) loaded and unloaded at runtime

We argue that *security localisation*, the adaptation of security to site-local and product-specific environments, is as natural a form of extensibility as device drivers and pluggable file systems. From 2000-2003, we developed the TrustedBSD MAC Framework. The framework allows policies to instrument security decisions, attach meta-data to system objects (e.g., processes and files), composes the results of multiple simultaneous policies, and provides policy-agnostic APIs for security tools. The MAC Framework implements kernel access control in FreeBSD, Juniper’s Junos, Apple’s Mac OS X on Macs and iOS on iPhones/iPads, and McAfee’s Sidewinder firewall.

In a 2013 *Communications of the ACM* article, we review our design principles, transition from research, how the framework supports both conventional MAC policies and contemporary sandboxing schemes across a range of products, and lessons for the future of OS security.

Robert N. M. Watson, “A decade of OS access-control extensibility”. Communications of the ACM 56(2), February 2013.

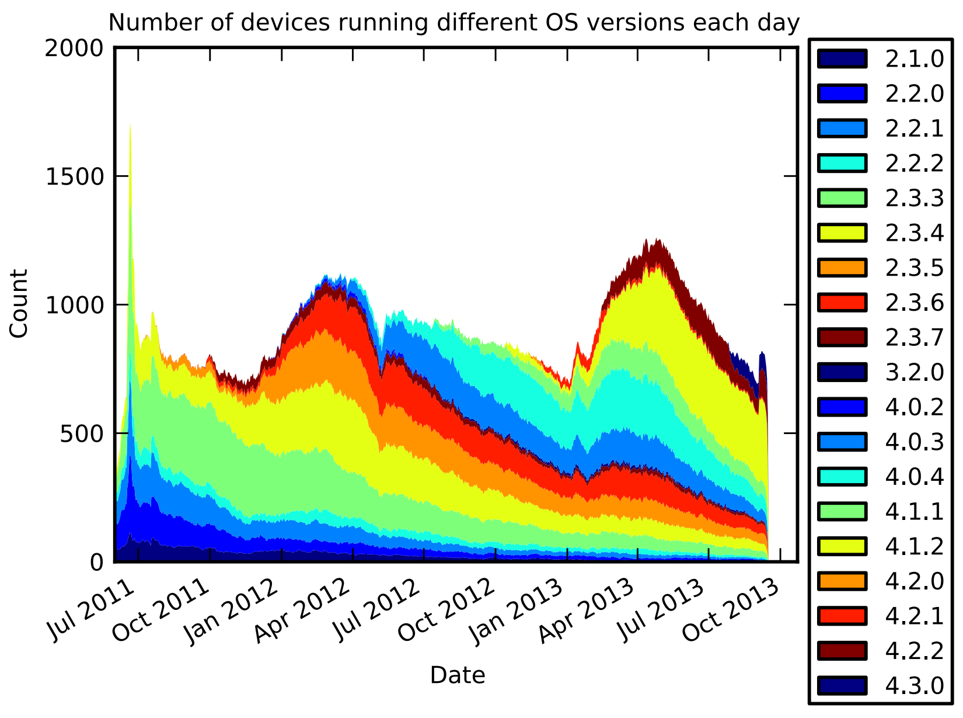
Software updates and vulnerabilities

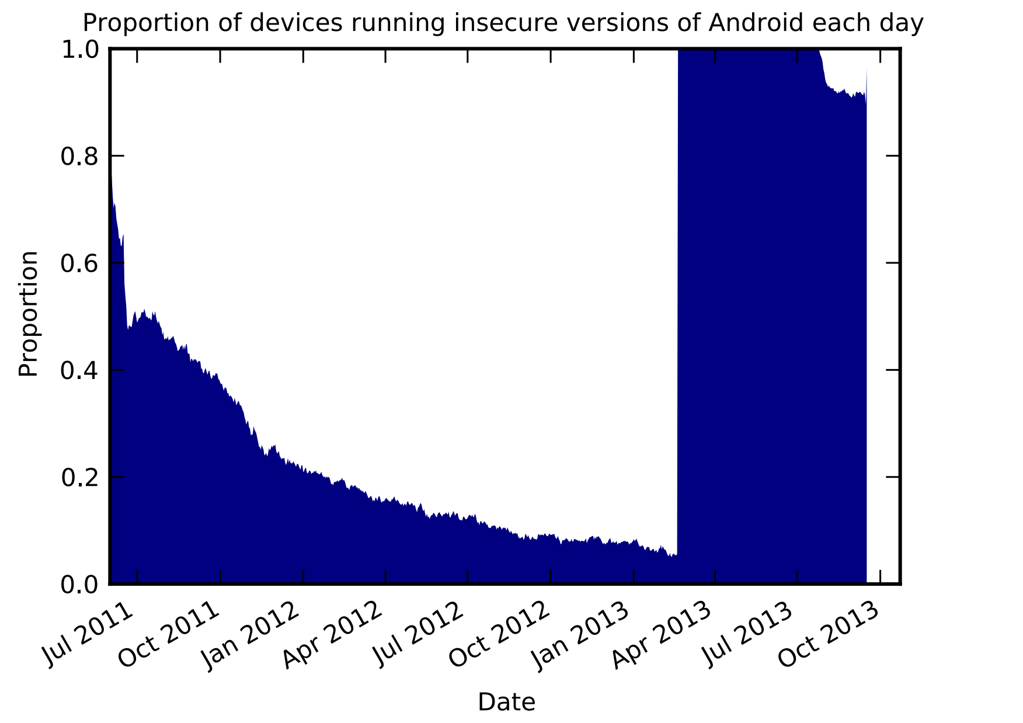
Daniel Thomas, Alastair Beresford, Laurent Simon, Daniel Wagner, Andrew Rice

Vulnerabilities in software are found regularly and must be fixed through software updates. These need to be distributed and installed on all devices. We are looking at how this works in practice. When are vulnerabilities found and how long does it take the vendor to produce and update for them? How much of the time are users exposed to root level exploits (particularly on Android)? Is it safer for a user to regularly or automatically install updates and so receive security fixes or does this allow developers and platform vendors additional ability to install malicious software on user devices?

One area we are focussing on is Android. We are constructing a database of known vulnerabilities which allow privilege escalation to root along with the information such as dates of discovery and which versions it was fixed in. Much of this information is not in existing databases such as CVE and is hard to come by.

We also have information on what versions of Android are installed on devices and so we can track what proportion of users are running versions of Android with known root level vulnerabilities. Which allows us to plot these graphs.


Number of devices running different OS versions each day


Proportion of devices running insecure versions of Android each day

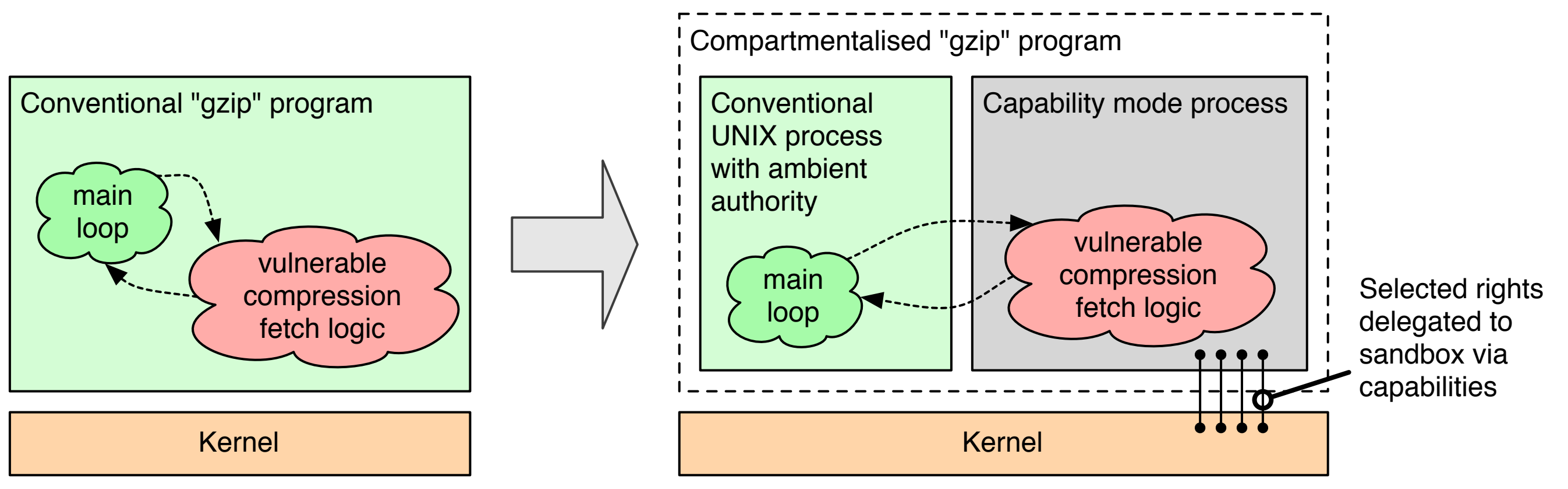
OS support for application security (7/11)

Capsicum: practical capabilities for UNIX

Robert N. M. Watson, Jonathan Anderson,
Kris Kennaway (Google), Ben Laurie (Google)

Application compartmentalisation, the sandboxing of software components to mitigate vulnerabilities, has to date been built on weak OS access-control foundations (e.g., chroot, setuid, and MAC) intended to solve different problems. Capsicum is a hybrid capability model that blends the *capability system model* with conventional UNIX security in order to provide the benefits of capability security and access to mainstream applications. Capsicum adds fine-grained capabilities and a sandboxed *capability mode* to existing OS APIs. This allows applications to implement their own, often dynamic, policies using delegation. A key benefit to Capsicum is that it allows applications to map their security requirements – e.g., the web's same-origin policy – onto robust OS primitives. Capsicum also supports application-level concepts such as multi-document interfaces (MDIs), which work poorly with MAC systems such as Type Enforcement.

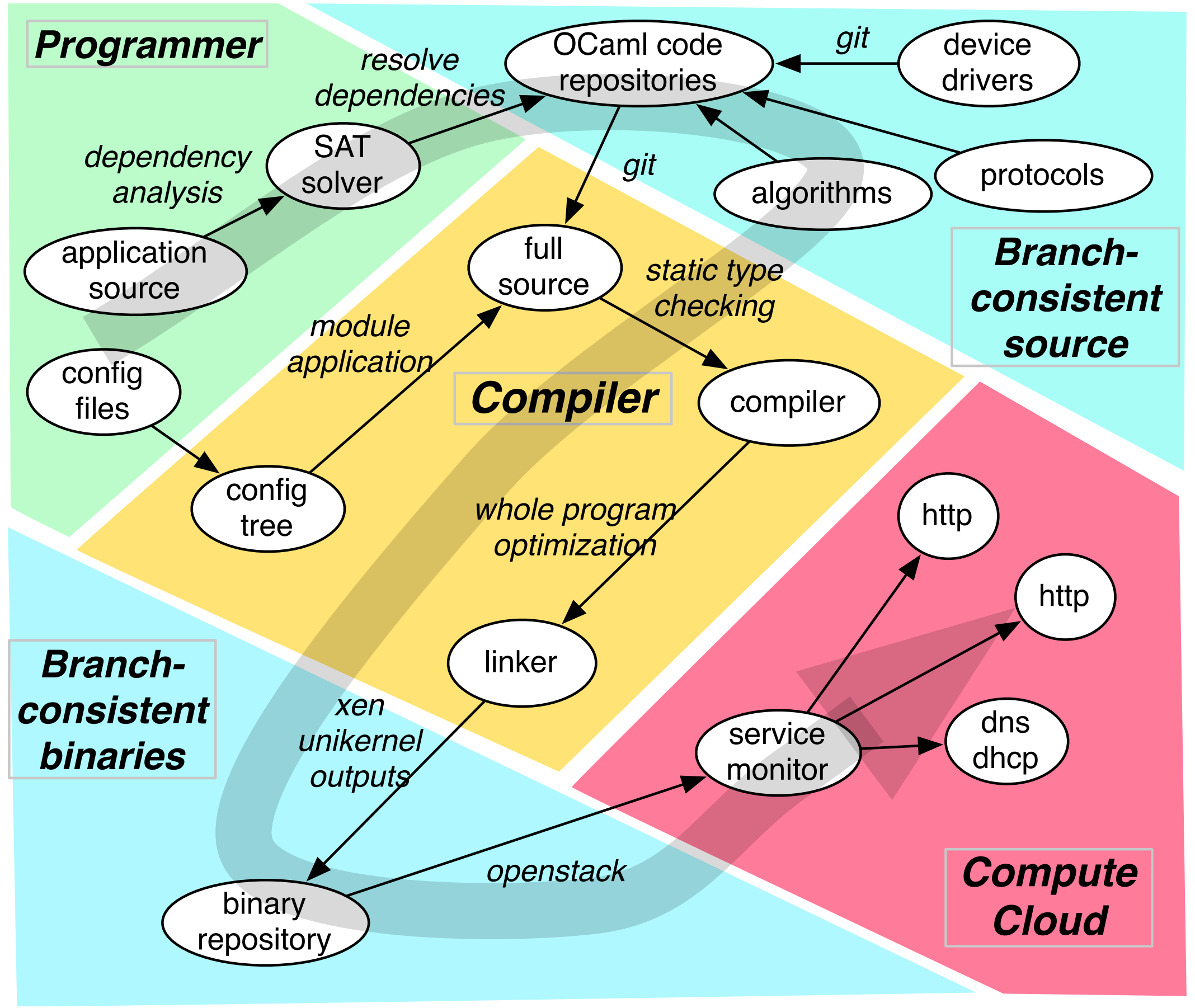
Capsicum's hybrid model allows software authors to reap immediate benefits as they incrementally convert systems to compartmentalisation, and offers a long-term capability system path to the *principle of least privilege*. Capsicum shipped in version 9.0 of the widely used open-source FreeBSD operating system; Google has also adapted the Linux operating system to support Capsicum. Google and the FreeBSD Foundation are co-sponsoring continued Capsicum development.


Conventional "gzip" program
main loop
vulnerable compression fetch logic
Kernel
Compartmentalised "gzip" program
Conventional UNIX process with ambient authority
Capability mode process
main loop
vulnerable compression fetch logic
Selected rights delegated to sandbox via capabilities
Kernel

Robert N. M. Watson, Jonathan Anderson, Ben Laurie, and Kris Kennaway, "Capsicum: practical capabilities for UNIX". 19th USENIX Security Symposium 2010, Washington, DC.
Robert N. M. Watson, Jonathan Anderson, Ben Laurie, and Kris Kennaway. "A taste of Capsicum: practical capabilities for UNIX". Communications of the ACM 55(3), March 2012.

MirageOS: a functional library operating system

Anil Madhavapeddy, Thomas Gazagnaire, Balraj Singh, Haris Rotsos,
Jon Crowcroft, Steven Hand, Robert N. M. Watson,
Richard Mortier (Nottingham), Dave Scott (Citrix), Jon Ludlam (Citrix),
Prashanth Mundkur (SRI)



On the cloud, the interface between guest kernels, applications and VMs are all managed differently, leading to serious inefficiencies, unreliability, and insecurity. MirageOS revisits the library OS concept and narrows the gap between safe high-level programming and low-level systems construction.

Applications are written in high-level OCaml and compiled directly into unikernels that run on the Xen hypervisor. By treating the hypervisor as a stable hardware platform, we can focus on high-performance protocol implementations without worrying about device driver support. Although Mirage initially targets the Xen hypervisor, other backends, such as a FreeBSD kernel module and JavaScript, also exist.

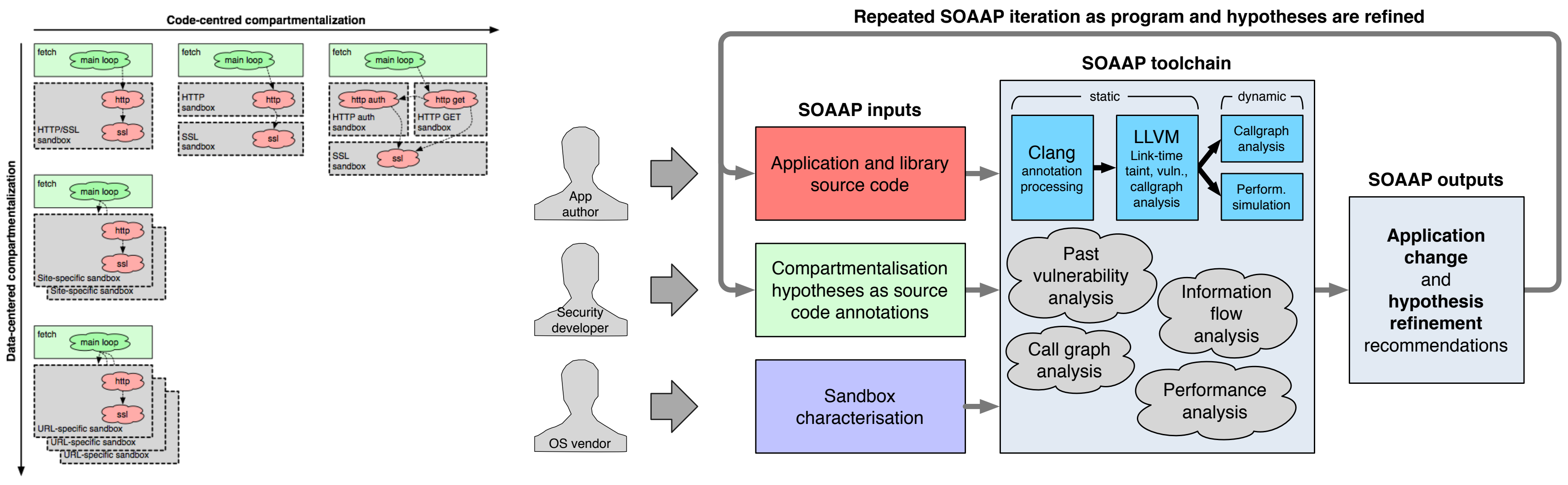
Mirage includes clean-slate modular, type-safe implementations of protocols ranging from TCP/IP, DNS, SSH, Openflow (switch/controller), HTTP, XMPP and Xen inter-VM transports. Some of the new applications we are building using Mirage include the next-generation XenServer (a widely deployed open-source Xen distribution). MirageOS is a Linux Foundation incubator project and freely available at:
http://openmirage.org

Anil Madhavapeddy and Dave Scott, "MirageOS: programming functional library operating systems". Communications of the ACM, January 2014.
Anil Madhavapeddy, Richard Mortier, Charalampos Rotsos, David Scott, Balraj Singh, Thomas Gazagnaire, Steven Smith, Steven Hand and Jon Crowcroft, "Unikernels: Library Operating Systems for the Cloud". ASPLOS March 2013.

SOAAP: security-oriented analysis of application programs

Khilan Gudka, Robert N. M. Watson, Steven Hand (MSR),
Ben Laurie (Google), Anil Madhavapeddy

Application compartmentalisation decomposes software into sandboxes to mitigate vulnerabilities, and has proven effective in limiting exploits. Experience shows that adding sandboxing to existing C programs is difficult, leading to problems with correctness, performance, complexity, and – critically – security. SOAAP is a Google- and DARPA-sponsored project investigating new, semi-automated techniques to support programmers in compartmentalisation efforts.


Code-centred compartmentalization
Data-centred compartmentalisation
App author
Security developer
OS vendor
Repeated SOAAP iteration as program and hypotheses are refined
SOAAP inputs
Application and library source code
Compartmentalisation hypotheses as source code annotations
Sandbox characterisation
SOAAP toolchain
static
dynamic
Clang annotation processing
LLVM Link-time taint, vuln., callgraph analysis
Callgraph analysis
Perform. simulation
Past vulnerability analysis
Information flow analysis
Call graph analysis
Performance analysis
SOAAP outputs
Application change and hypothesis refinement recommendations

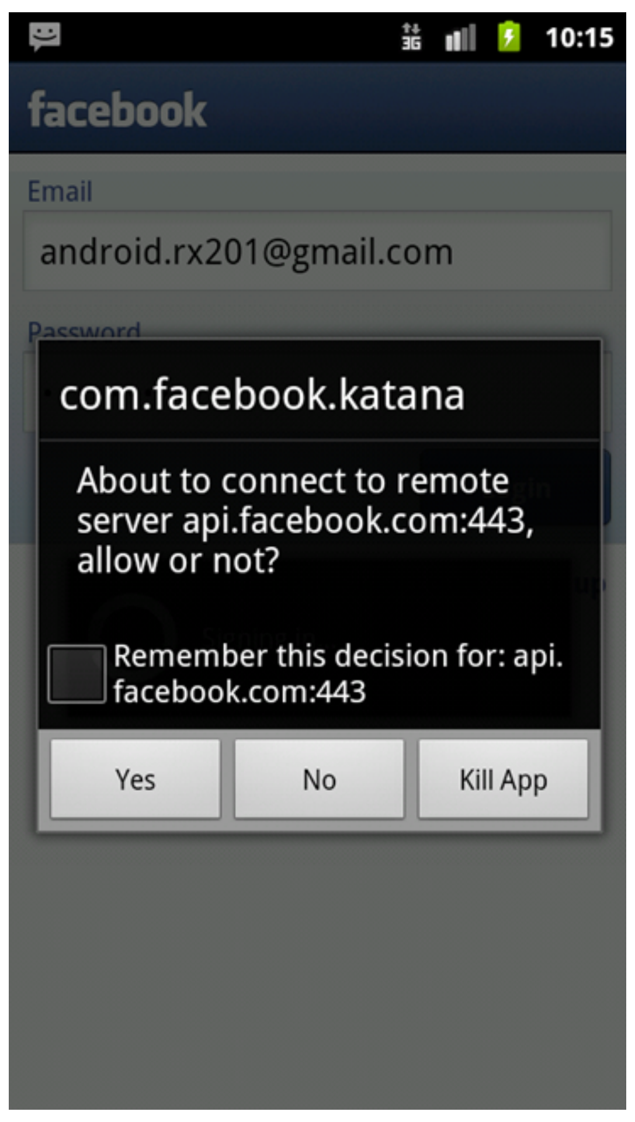
SOAAP allows compartmentalisation hypotheses to be explored through source code annotations and can also find bugs in existing sandboxed applications. Annotations describe functions to sandbox, global state and file descriptors that sandboxes can access, limits on system services, and rights available via RPC. Past vulnerabilities, code provenance, information flow constraints, and acceptable performance overheads can also be expressed. SOAAP uses program analysis to engage the developer in a constructive dialogue, identifying potential correctness bugs (e.g., data inconsistencies), security breaches (e.g., information leaks) and unmitigated vulnerabilities. They can also explore software supply-chain trojan scenarios and simulate sandboxed performance. Hypotheses are iterative: a single mitigated vulnerability in a sandbox gives an immediate improvement.

Khilan Gudka, Robert N. M. Watson, Steven Hand, Ben Laurie, and Anil Madhavapeddy, "Exploring compartmentalisation hypotheses with SOAAP". Adaptive Host and Network Security (AHANS 2012), September, 2012.

Aurasium: practical policy enforcement for Android applications

Rubin Xu, Hassen Saidi (SRI), Ross Anderson

The increasing popularity of Google's mobile platform Android makes it the prime target of the latest surge in mobile malware. Most research on enhancing the platform's security and privacy controls requires extensive modification to the operating system, which has significant usability issues and hinders efforts for widespread adoption. We develop a novel solution called Aurasium that bypasses the need to modify the Android OS while providing much of the security and privacy that users desire. We automatically repackage arbitrary applications to attach user-level sandboxing and policy enforcement code, which closely watches the application's behaviour for security and privacy violations, such as attempts to retrieve a user's sensitive information, send SMS covertly to premium numbers, access malicious IP addresses, and known privilege escalation attempts.

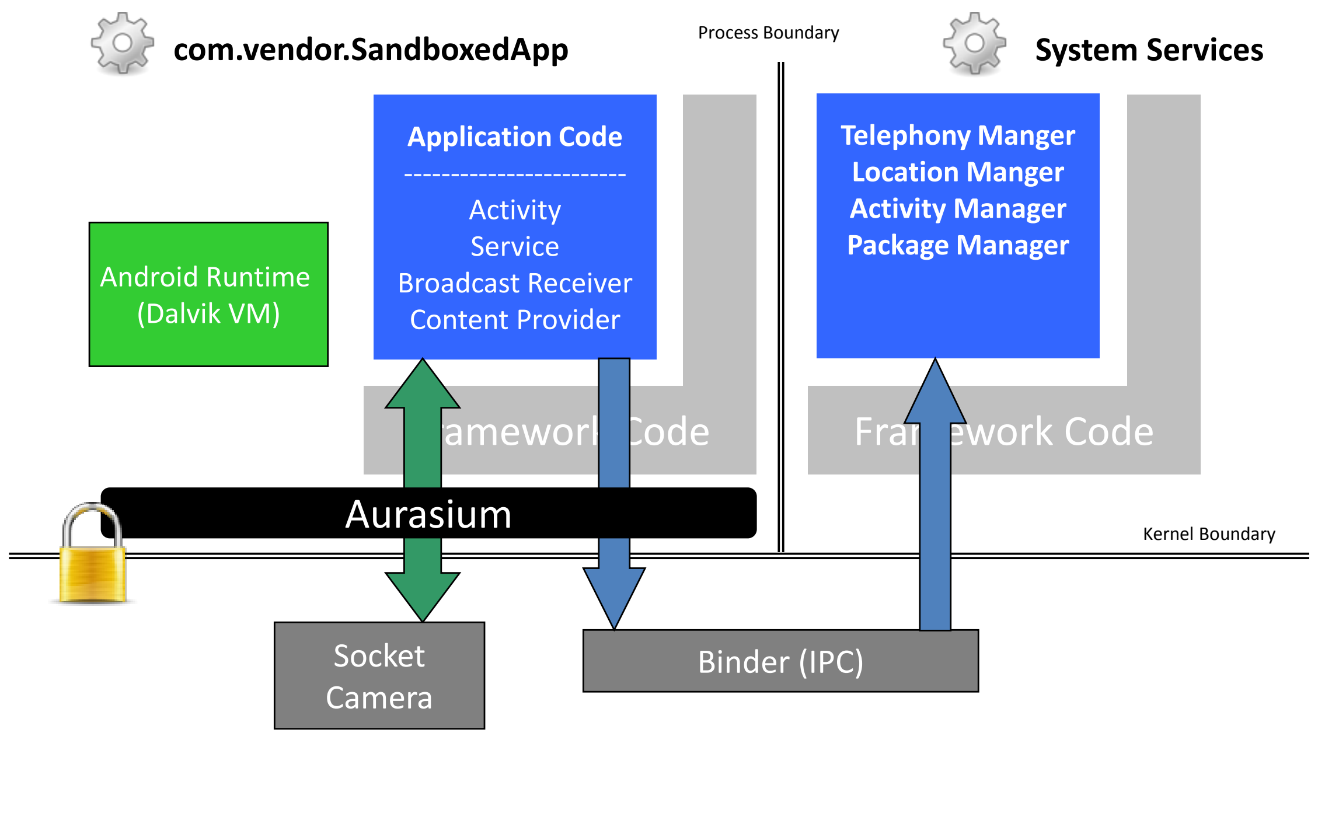


com.facebook.katana

About to connect to remote server api.facebook.com:443, allow or not?

□ Remember this decision for: api.facebook.com:443

Experiments show that we can apply this solution to a large sample of benign and malicious applications from various sources with over 99% success rate, and without significant performance and storage overhead.



Rubin Xu, Hassen Saidi and Ross Anderson. "Aurasium: practical policy enforcement for Android applications". Proceeding. 21st USENIX Security symposium, August 2012.

UNIVERSITY OF CAMBRIDGE

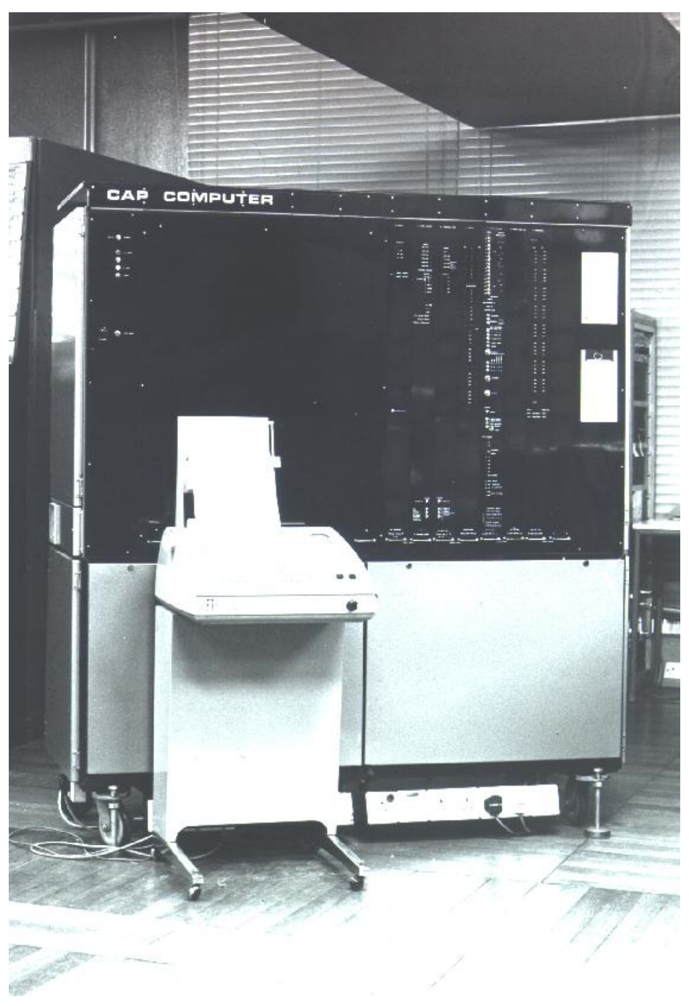
CPU support for fine-grained sandboxing (8/11)

Capability system research

Programmers are increasingly turning to *software compartmentalisation* to mitigate inevitable vulnerabilities: software is decomposed into many sandboxed components, each with only the rights it requires to function. This approach implements the *principle of least privilege*: as granularity increases, rights delegated to individual sandboxes decrease, limiting the effects of exploited vulnerabilities. Attackers are forced to identify and exploit more vulnerabilities to accomplish their goals.

Capability systems are processors, operating systems, or programming languages designed to support fine-grained compartmentalisation. Each component runs with limited rights, represented by its *capabilities*, rather than the *ambient authority* to name all objects as in conventional designs. Capabilities are unforgeable tokens of authority acquired through object creation or delegation. *Object capabilities* blend capabilities with object-oriented software design, linking objects and methods to compartments and message passing. *Hybrid capability systems*, such as the Capsicum operating-system design and CHERI processor, hybridise the capability system model with contemporary software and hardware designs.

Key aspects of the capability system model were developed at Cambridge during the 1970s, including the CAP Computer (developed by Wilkes, Needham and Wheeler) and Karger's SCAP which both implemented hardware capabilities.



Many challenges arise in deploying capability systems in software and hardware:

- Can they be made to perform well, despite fine-grained protection being de-emphasised in current computer architectures?
- How do we ensure that decomposed programs implement our security objectives?
- How can we ensure software remains comprehensible and debuggable despite taking on distributed-system properties?
- How can we incrementally deploy compartmentalisation in current designs?

Language and tool support for hardware capabilities

David Chisnall, Jonathan Woodruff, Robert N. M. Watson, Simon W. Moore, Michael Roe, Ben Laurie (Google), Brooks Davis (SRI), Stacey Son (SRI), Peter G. Neumann (SRI)

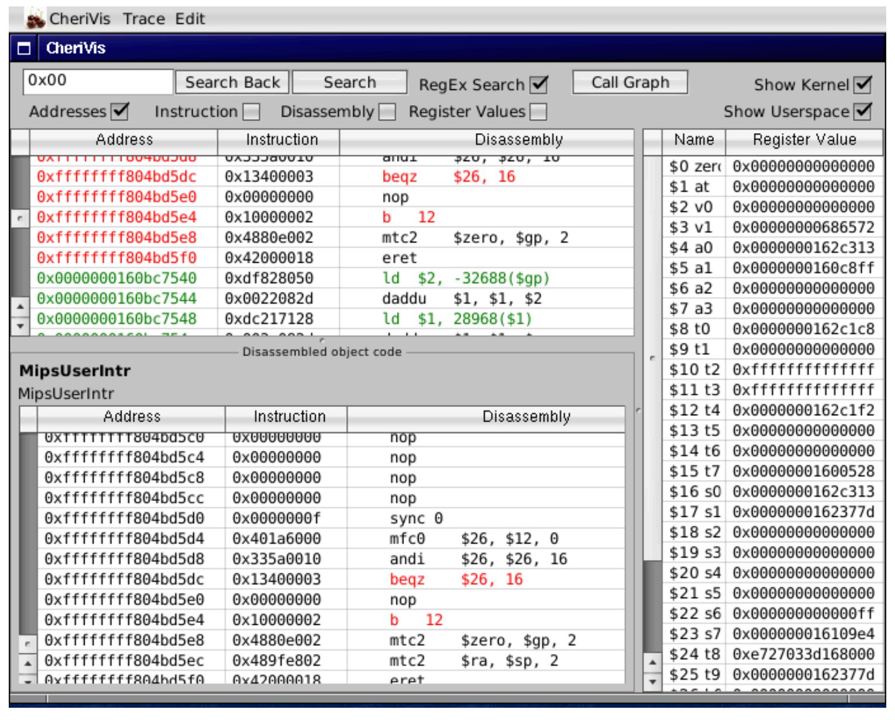
The CHERI ISA provides two features: fine-grained memory protection within address spaces, and support for efficient transition and memory sharing between different security domains within processes. These features require support from both the programming language and the operating system. Language integration allows the CHERI CPU to enforce programmer-imposed safety properties: pointer integrity, bounds checking, and read/write/execute checking. CHERI eliminates many exploit techniques grounded in memory corruption and code-data confusion used on conventional hardware, and provides the foundation for safe and high-performance data sharing between protection domains.

```
// The compiler automatically inserts the bounds limits
__capability int *buffer =
    (__capability int*)malloc(size);

// Size can be computed from the capability here, or
// made explicit so the code can be compiled on a
// non-capability-aware architecture
fillArray(buffer, size);

// This overflows the bounds and so will trap at run time
int retVal = buffer[size];
```

```
# Call malloc()
ld      $25, %call16(malloc)($18)
jalr    $25
# Load the address of fillArray
ld      $25, %call16(fillArray)($18)
# Set the length of the capability
cincbase        $c1, $c0, $2
# Call fillArray
jalr    $25
# Set capability length (in branch-delay slot)
csetlen $c1, $c1, $16
# Load the value (causing capability violation trap)
clw     $16, $1, 0($c1)
```

We have extended the LLVM MIPS back end to support code generation for the capability coprocessor, allowing use in experimental languages and language extensions. We have also modified clang (the C/C++/Objective-C front end for LLVM) to support a `__capability` keyword as a pointer qualifier. All pointers with this qualifier are hardware bounds checked and are unforgeable – any accidental modification in memory will cause a trap when dereferenced.

By creating compiler infrastructure as a library, we can re-use code to build tools for post-mortem debugging; e.g., to inspect CHERI debug-unit output.



Address	Instruction	Disassembly
0xffffffff804bd5dc	0x13400003	beqz $26, 16
0xffffffff804bd5e0	0x00000000	nop
0xffffffff804bd5e4	0x10000002	b 12
0xffffffff804bd5e8	0x4880e002	mtc2 $zero, $gp, 2
0xffffffff804bd5f0	0x42000018	eret
0x0000000160bc7540	0xdf828050	ld $2, -32688($gp)
0x0000000160bc7544	0x0022082d	daddu $1, $1, $2
0x0000000160bc7548	0xdc217128	ld $1, 28968($1)

MipsUserIntr

MipsUserIntr

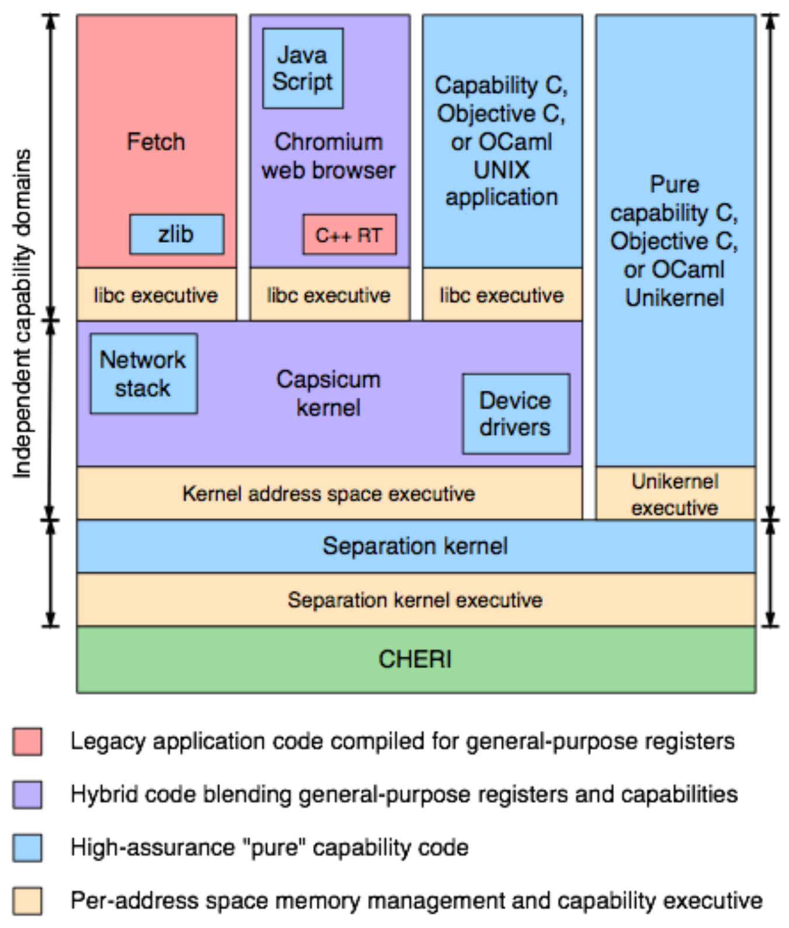
Address	Instruction	Disassembly
0xffffffff804bd5c0	0x00000000	nop
0xffffffff804bd5c4	0x00000000	nop
0xffffffff804bd5c8	0x00000000	nop
0xffffffff804bd5cc	0x00000000	nop
0xffffffff804bd5d0	0x0000000f	sync 0
0xffffffff804bd5d4	0x401a6000	mfc0 $26, $12, 0
0xffffffff804bd5d8	0x335a0010	andi $26, $26, 16
0xffffffff804bd5dc	0x13400003	beqz $26, 16
0xffffffff804bd5e0	0x00000000	nop
0xffffffff804bd5e4	0x10000002	b 12
0xffffffff804bd5e8	0x4880e002	mtc2 $zero, $gp, 2
0xffffffff804bd5ec	0x489fe802	mtc2 $ra, $sp, 2
0xffffffff804bd5f0	0x42000018	eret

Name	Register Value
$0 zer	0x00000000000000
$1 at	0x00000000000000
$2 v0	0x00000000000000
$3 v1	0x00000000686572
$4 a0	0x0000000162c313
$5 a1	0x0000000160c8ff
$6 a2	0x00000000000000
$7 a3	0x00000000000000
$8 t0	0x0000000162c1c8
$9 t1	0x00000000000000
$10 t2	0xffffffffffffff
$11 t3	0xffffffffffffff
$12 t4	0x0000000162c1f2
$13 t5	0x00000000000000
$14 t6	0x00000000000000
$15 t7	0x00000001600528
$16 s0	0x0000000162c313
$17 s1	0x0000000162377d
$18 s2	0x00000000000000
$19 s3	0x00000000000000
$20 s4	0x00000000000000
$21 s5	0x00000000000000
$22 s6	0x000000000000ff
$23 s7	0x000000016109e4
$24 t8	0xe727033d168000
$25 t9	0x0000000162377d

CHERI: Capability hardware enhanced RISC instructions

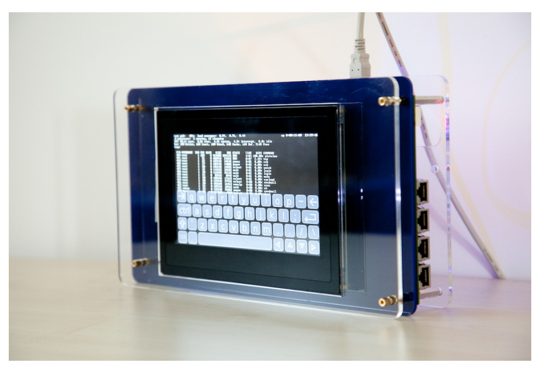
Robert N. M. Watson, Peter G. Neuman (SRI), Simon W. Moore, Jonathan Anderson, David Chisnall, Nirav Dave (SRI), Brooks Davis (SRI), Khilan Gudka, Steven M. Hand, Ben Laurie (Google), Alan Mujumdar, Steven J. Murdoch, Robert Norton, Michael Roe, Hassen Saidi (SRI), Jonathan Woodruff, Bjoern A. Zeeb

Current CPU instruction set architectures (ISAs) reflect a 1990s protection consensus: virtual memory supports coarse-grained processes. Today, systems are exposed to greater threats, placing strain on protection facilities designed for a less adversarial world. In the DARPA-funded CTSRD project, we are developing a hybrid-capability CPU model supporting finer-grained memory protection, orders of magnitude greater software compartmentalisation granularity, and strong compatibility with current software designs:


Independent capability domains
Fetch
zlib
libc executive
Java Script
Chromium web browser
C++ RT
libc executive
Capability C, Objective C, or OCaml UNIX application
libc executive
Pure capability C, Objective C, or OCaml Unikernel
Network stack
Capsicum kernel
Device drivers
Kernel address space executive
Unikernel executive
Separation kernel
Separation kernel executive
CHERI

🟥 Legacy application code compiled for general-purpose registers
🟪 Hybrid code blending general-purpose registers and capabilities
🟦 High-assurance "pure" capability code
🟨 Per-address space memory management and capability executive

- de-conflates CPU address space virtualisation and protection features;
- support three orders of magnitude greater scaling of protection within low-level TCBs (OS kernels and programming language runtimes);
- applies Capsicum's hybrid capability-system model at the ISA level;
- supports conventional and capability-aware code execute side-by-side;
- incrementally deployable to software TCBs & high-risk components;
- prototyped using 64-bit MIPS ISA; adaptable to 64-bit ARM;
- CPU implemented in the high-level Bluespec hardware description language (HDL) synthesised to field programmable gate array (FPGA);
- supports open-source FreeBSD OS, clang/LLVM compiler suite, and tens of thousands of off-the-shelf applications; and
- formal methods applied throughout hardware, ISA, and software.



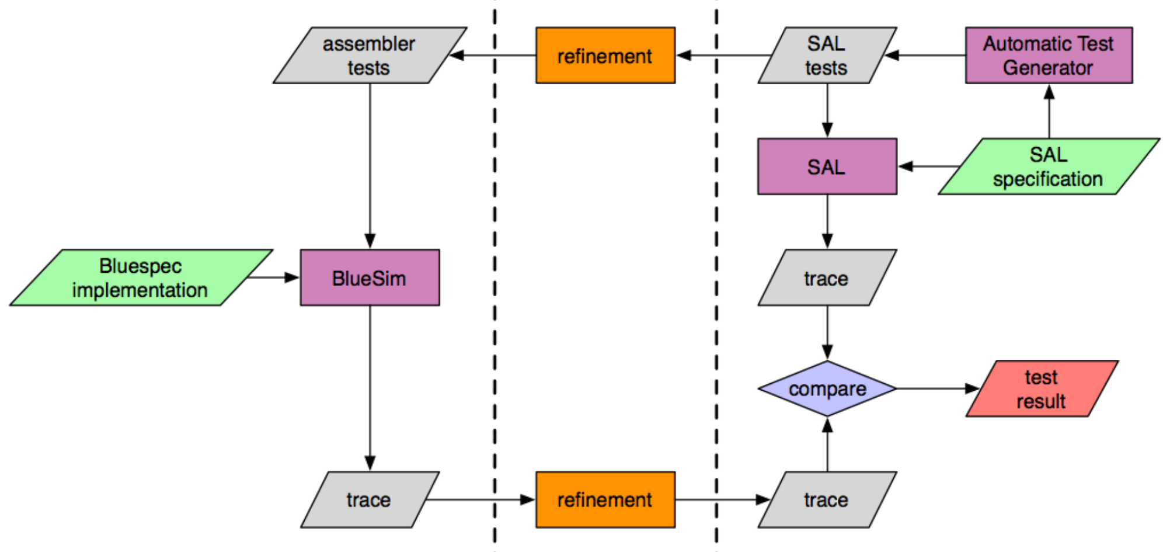
SRI International   UNIVERSITY OF CAMBRIDGE

Testing and verification of hardware designs

Nirav Dave (SRI), Michael Roe, Hassan Saidi (SRI)

CPU hardware is often very complex and prone to bugs, making it essential to have a comprehensive test suite and desirable to have correctness proofs as well. Test suites are time consuming to write by hand, also subject to errors, and need to be revised every time the specification is changed.

We used model checking to automatically construct a test suite for the CHERI capability unit from a formal specification written in the SAL language. The test generator constructs a set of assembly-language programs which will exercize every possible state transition rule in the formal specification. These test programs are then run on Bluespec-level simulation of CHERI, and the results automatically compared against the behavior predicted by the formal model.



```
tSetLength: opcode = opSetLength AND running
  AND state.c1.tag
  AND state.c1.unsealed
  AND state.t0 <= state.c1.length -->
  testSetLength' = TRUE;
  state' =
    state WITH .c1 :=
      (state.c1 WITH .length := state.t0);
```

The SAL specification is then automatically converted into a specification in he PVS theorem proving language, which can be used to prove theorems about the formal model (e.g. security properties). Our second approach to verifying CHERI works at a lower level, and attempts to show that the pipelined implementation of the CPU is equivalent to a simpler specification without pipelining. We automatically convert the Bluespec implementation of CHERI into the SMTen language, and algebraically reduce it to a specification where the internal state of the pipeline is not externally observable.

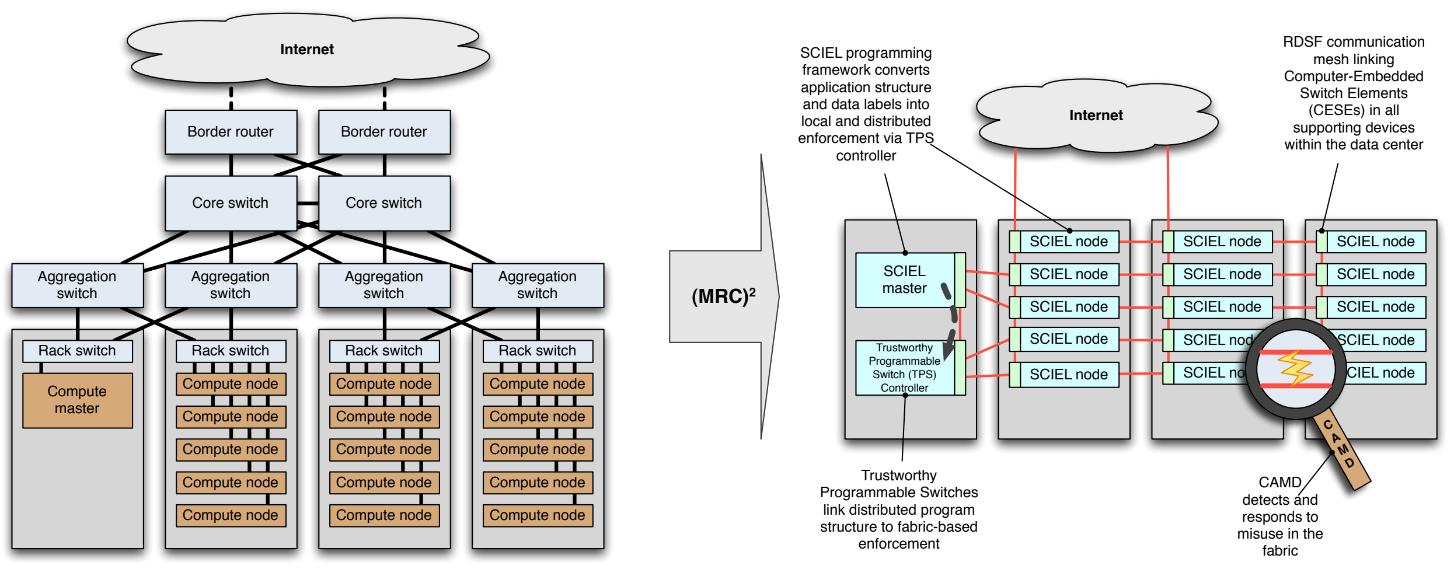
Distributed-system security (9/11)

(MRC)²: Resilient, secure data-centre switching

Simon W. Moore, Robert N. M. Watson, Peter G. Neumann (SRI), Andrew W. Moore, Anil Madhavapeddy, Brooks Davis (SRI), Matt P. Grosvenor, Alexandre Joannou, Alan Mujumdar, Robert Norton, Phil Porras (SRI), Colin Rothwell, Charalampos Rotsos, Hassen Saidi (SRI), Malte Schwarzkopf, Vinod Yegneswaren (SRI), Dongting Yu, Bjoern A. Zeeb

This DARPA-sponsored project at SRI and Cambridge is enhancing data-centre switching security and resilience from several perspectives:

- Resilient Distributed Switch Fabric (RDSF) employs higher-dimensionality links to improve performance, resilience, security, and energy efficiency through greater adaptiveness.
- The CHIMERA memory interconnect offers a weak coherency, capability-oriented memory semantic, improving scalability as the CHERI model scales up to thousands of cores.
- The SCIEL distributed computation framework, and the FABLE reconfigurable I/O layer, allow applications to map high-level security and resilience properties into a variety of communications substrates, including RDSF and CHIMERA.
- Trustworthy Programmable Switch Controllers (TPSC) incorporate a formally verified switch implementation with the CHERI CPU in order to provide secure foundations for distributed switching applications. TPSC also considers distributed switch control through extensions to current software-defined networking (SDN) models.
- Cloud Analysis and Misuse Detection (CAMD) deploys SDN-based security applications in an (MRC)² data centre, requiring revisions to mechanism and policy, eliminating global visibility requirements.

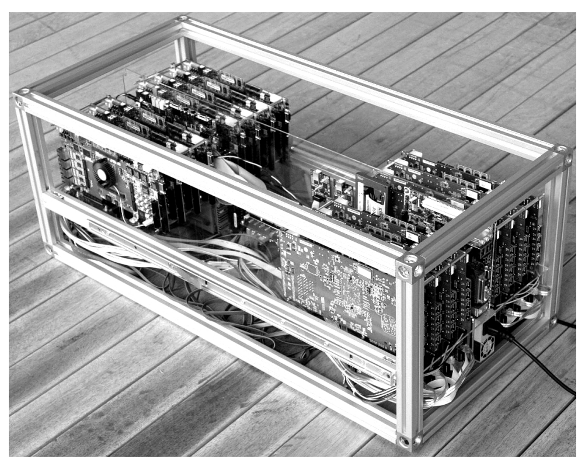


SCIEL programming framework converts application structure and data labels into local and distributed enforcement via TPS controller

RDSF communication mesh linking Computer-Embedded Switch Elements (CESEs) in all supporting devices within the data center

Trustworthy Programmable Switches link distributed program structure to fabric-based enforcement

CAMD detects and responds to misuse in the fabric



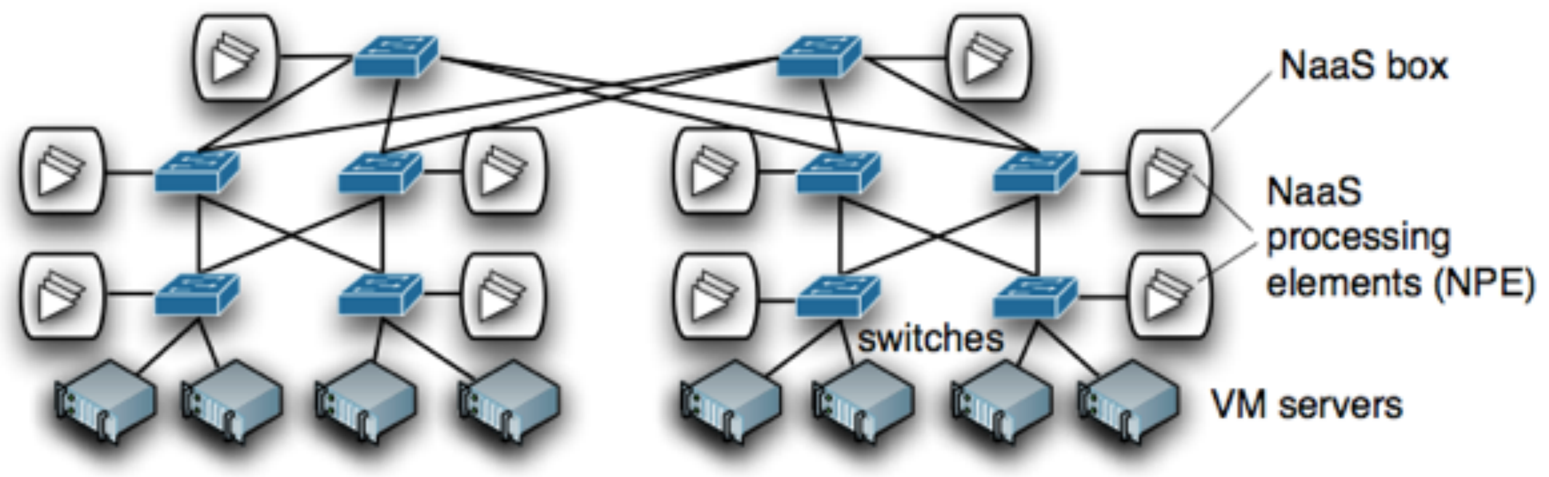
NaaS: Network-as-a-Service

Andrew Moore, Jon Crowcroft, Anil Madhavapeddy, Richard Mortier (Nottingham), Derek McAuley (Nottingham), Paolo Costa (Imperial), Peter Pietzuch (Imperial), Alexander L. Wolf (Imperial)

Today it is possible for small companies to access virtually unlimited resources in large data centres (DCs) to run computationally demanding tasks. This has triggered the rise of 'big data' applications that operate on large amounts of data such as data mining, real-time stream processing, web search, and advertising. An open challenge is balancing the contention between nodes for network resources in data centers.

This can be solved more effectively by providing DC tenants with efficient, easy, and safe control of network operations. Instead of over-provisioning, we optimise network traffic by exploiting application-specific knowledge, and balance the needs of secure isolation with network throughput. We term this approach 'network-as-a-service' (NaaS) because it allows tenants to customise the service that they receive from the network.

NaaS-enabled tenants can deploy custom routing protocols, as well as more sophisticated mechanisms like content-based routing. By modifying the content of packets on-path, they can implement application-specific network services such as in-network data aggregation and smart caching. Parallel processing systems, such as MapReduce, benefit because data can be aggregated on-path. Key-value stores can improve their performance by caching popular keys within the network. Applications can deploy custom security protocols depending on their end-to-end threat models within a given datacenter topology.

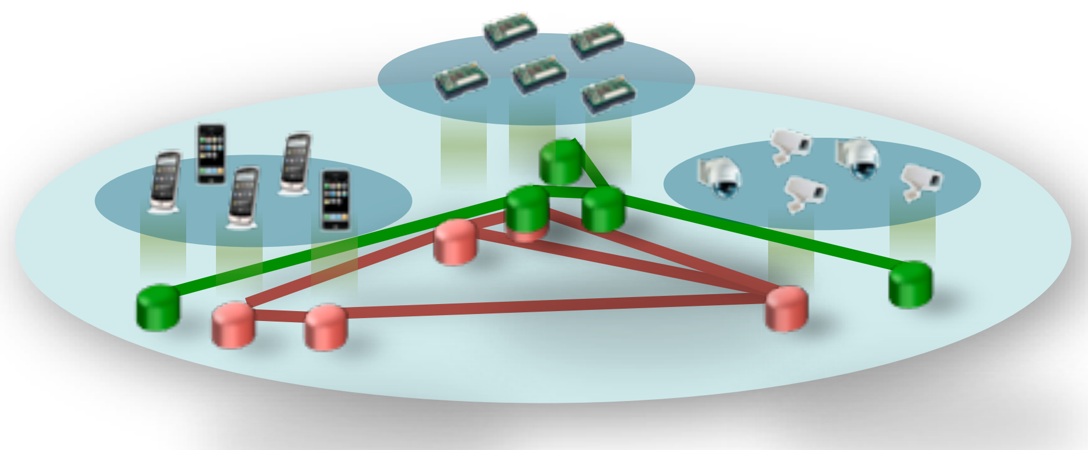


FRESNEL: federated secure sensor network laboratory

Jon Crowcroft, Christos Efstratiou, Ilias Leontiadis, Cecilia Mascolo

Current sensor networks

- Single owner
- Single purpose
- Fixed
- One application running
- Difficult to reprogram



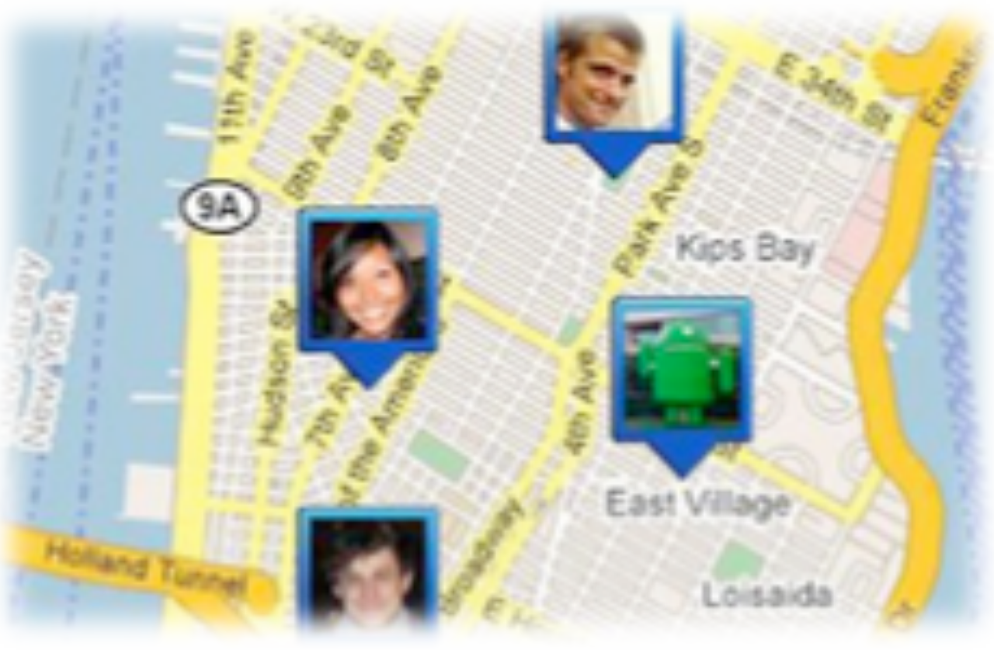
FRESNEL

- Shared sensor networks
- Support multiple applications simultaneously
- Each application can use any sensor available

Privacy and Security

- Different roles have different requirements
- Sensor network owners have their own security and privacy policies
- Users and application may have different demands

FRESNEL aims to offer a framework that supports merging of policies as specified by different roles in the system.

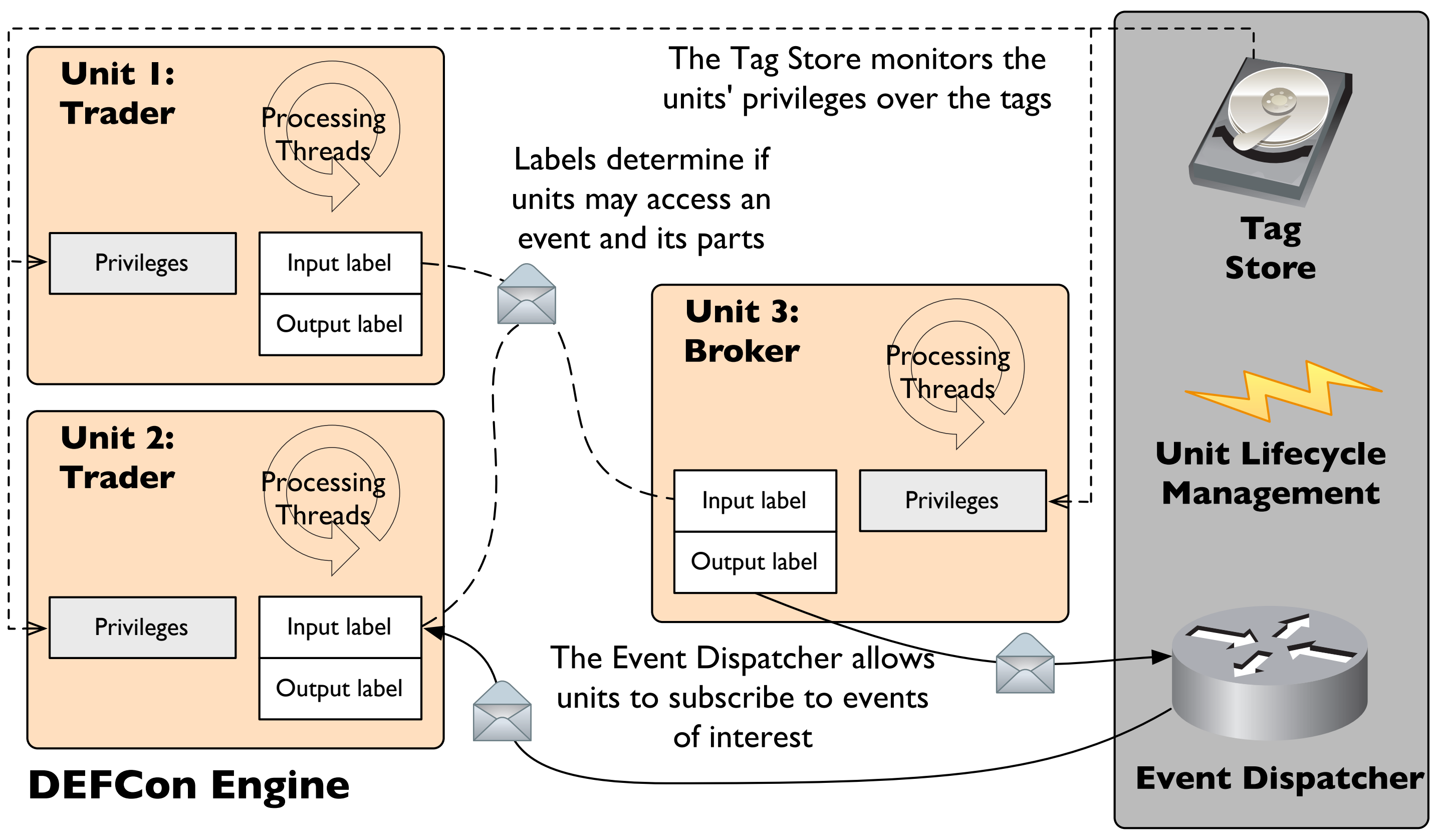


Information flow control in distributed systems: applications to healthcare

David Evans (Derby), Jean Bacon, Jatinder Singh

Healthcare providers world-wide are developing electronic solutions to improve patient care and reduce costs. The resulting systems must not compromise patient safety and privacy. Not surprisingly, healthcare IT efforts in many countries suffer from cost explosion and project overruns. In these systems, middleware software acts as the 'plumbing' that integrates many heterogeneous applications, coordinating widely distributed operations.

Decentralised Event Flow Control (DEFCon) is our model for building secure event-based applications. We track sensitive information flows at the granularity of events by using tags to form security labels, expressing confidentiality and integrity requirements; tags are applied to event parts. Privileges over tags constrain how recipients can perceive and propagate events. The figure below shows a DEFCon Engine hosting three processing units. Every node participating in a DEFCon infrastructure provides a DEFCon Engine. The Engine is the container for event processing within a DEFCon application and includes a trusted kernel. Application code is run by event processing units that are hosted by the Engine. We have an implementation for Java.



The Tag Store monitors the units' privileges over the tags

Labels determine if units may access an event and its parts

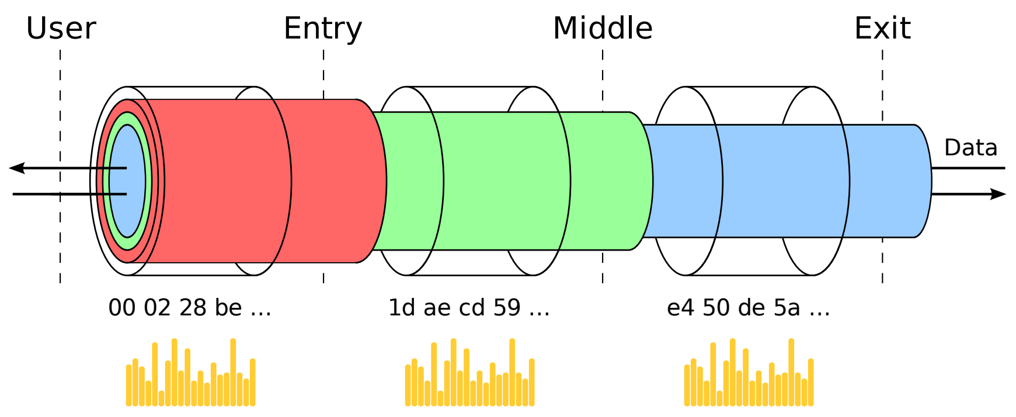
The Event Dispatcher allows units to subscribe to events of interest

UNIVERSITY OF CAMBRIDGE

Anonymity and censorship resistance (10/11)

Anonymous communication

Steven J. Murdoch, Richard Clayton, Robert N. M. Watson



Encryption, as sometimes used with web browsing (SSL/TLS) and email (e.g., PGP), hides only message content and not traffic data: source, destination, size, and timing. Traffic analysis studies this data to discover the behaviour and interests of groups and individuals. It is widely used to track people, for marketing, law-enforcement and by criminals. Anonymity systems, such as Tor, protect the privacy of Internet users from traffic analysis.

Tor is primarily used for anonymous web browsing, and is built from a network of around 1,500 servers (nodes) run by volunteers around the world. Messages are encrypted then sent through a randomly chosen path of three servers, within nested layers of encryption. This makes it difficult for an attacker to follow a message between source and destination. In addition to anonymity, Tor is used to circumvent national censorship systems. By hiding what websites a user is accessing, Tor makes it more difficult for countries to block access to certain websites.

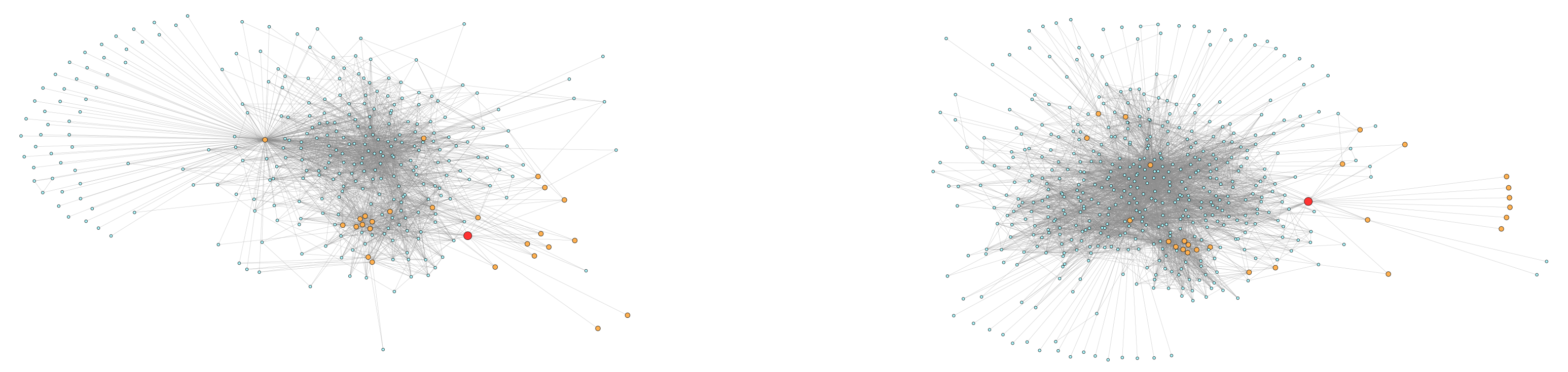
However, while Tor prevents message content from being used to match incoming and outgoing connections, it does not significantly distort traffic patterns, allowing traffic analysis to occur. Anonymous communications research at Cambridge, funded by The Tor Project, includes studying the effectiveness of traffic analysis techniques, designing routing methods to reduce the likelihood of traffic being monitored, and developing efficient methods to hide distinctive traffic patterns.

Claudia Diaz, Steven J. Murdoch, Carmela Troncoso, “Impact of Network Topology on Anonymity and Overhead in Low-Latency Anonymity Networks”. 10th Privacy Enhancing Technologies Symposium (PETS 2010), Berlin, Germany, 21-23 July 2010.
Steven J. Murdoch and Robert N. M. Watson, “Metrics for Security and Performance in Low-Latency Anonymity Systems”. 8th Privacy Enhancing Technologies Symposium (PETS 2008), Leuven, Belgium, 23-25 July 2008.
Steven J. Murdoch and George Danezis, “Low-Cost Traffic Analysis of Tor”. Proceedings of the 2005 IEEE Symposium on Security and Privacy, Oakland, California, USA, 8-11 May 2005.

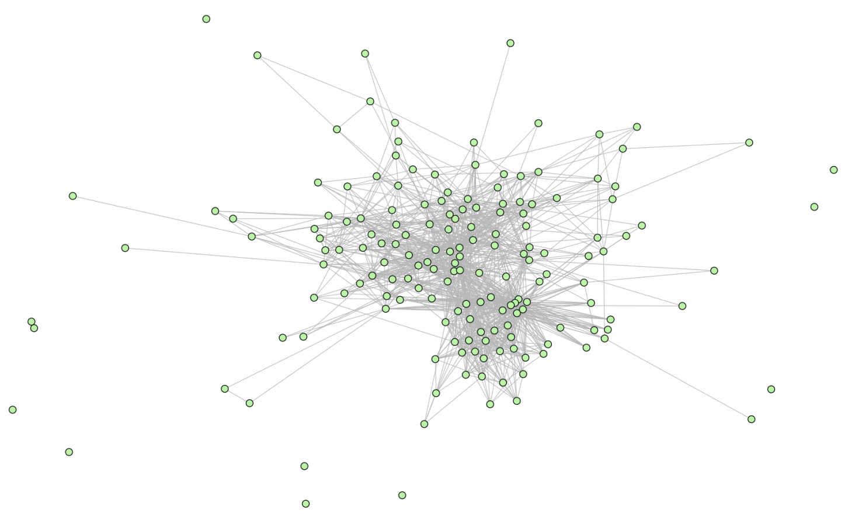
Privacy leakage in egonets

Kumar Sharad, George Danezis (UCL), Steven J. Murdoch

A major telecom company released anonymised communication sub-graphs (egonets) of a few thousand randomly selected subscribers (egos) for scientific analysis. All communication between their friends and friends of friends was published. The figures below show sub-graphs extracted from a publicly available social network dataset. The red node is the ego, the orange nodes are friends (1-hop away) and the blue nodes are friends of friends (2-hops away).



In real-world social networks the nodes common across sub-graphs interact in complex ways. This allows the graph topology to be exploited for re-identification. The figure below illustrates the interactions between the common nodes of the sub-graphs show



We use the degree distribution of the neighbourhood (completely available) of the 1-hop nodes to construct a signature which is used for re-identification. The technique is modified for the 2-hop nodes (partially available neighbourhood). To re-identify such nodes we use the 1-hop nodes common and the cosine similarity between their neighbourhood degree distribution. This assigns a score to the pairings which is maximised using bipartite matching.

We re-identified almost all the common 1-hop nodes with over 98% success probability. Close to 15% (often over 20%) of the common 2-hop nodes were re-identified with over 75% (often over 90%) success probability. This shows that the anonymisation strategy used is weak and allows an attacker to re-identify and link records efficiently.

Kumar Sharad and George Danezis, “De-anonymizing D4D Datasets”. 6th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2013), July 2013.

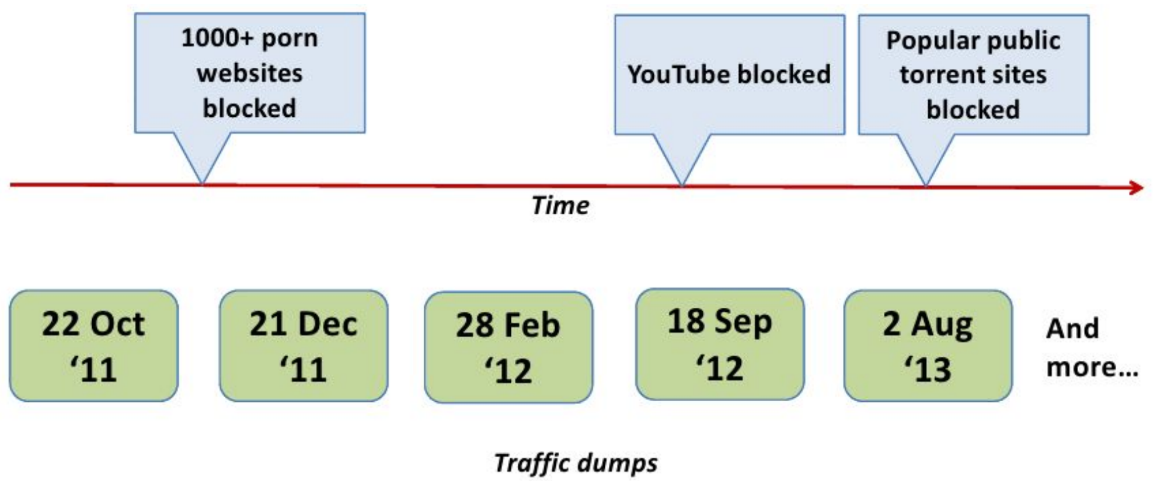
Censorship resistance

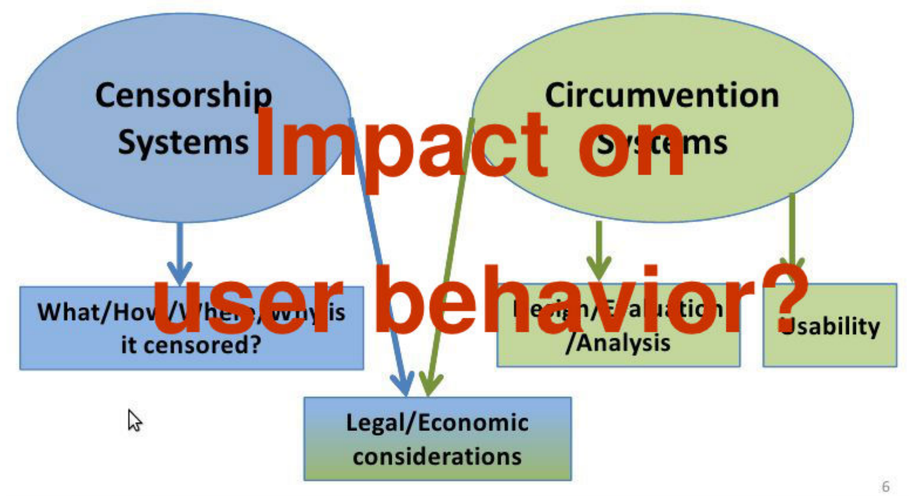
Sheharbano Khattak, Steven J. Murdoch

The Internet has emerged as an important medium of discourse. Consequently, it has become a tempting venue for authoritarian characters to exercise and maintain their control through censorship of the information disseminated over the Internet. Most of existing work in the Internet censorship landscape focuses on censorship systems (detection of censorship, identification of censored content, mechanism used to execute censorship), and circumvention (design/analysis/evaluation/usability of circumvention/censorship-proof protocols and systems).

We seek to understand Internet censorship from the victim’s perspective; how is user behaviour modified in response to blocking constraints? We introduce the concept of ‘persistence’ which is a new dimension in this area. Persistence is a quantity that measures how motivated users are to access blocked content. This is manifested by DNS and search engine queries for filtered content and increase in the usage of circumvention services. Persistence is an important measure because it provides insights into whether or not the censor has achieved its goal of suppressing user interest in specific information. Also, it presents a systematic analysis of human resilience in the face of obstructions artificially created by a control entity.

In collaboration with ICSI, UC Berkeley, PLUMgrid Inc., and Lahore University of Management Sciences, we are currently analysing longitudinal data collected from a country with prevailing Internet censorship during a timeline when the sophistication of its censorship system evolved.




Censorship Systems
Circumvention Systems
Impact on user behavior?
What/How/Where/Why is it censored?
Design/Evaluation /Analysis
Usability
Legal/Economic considerations

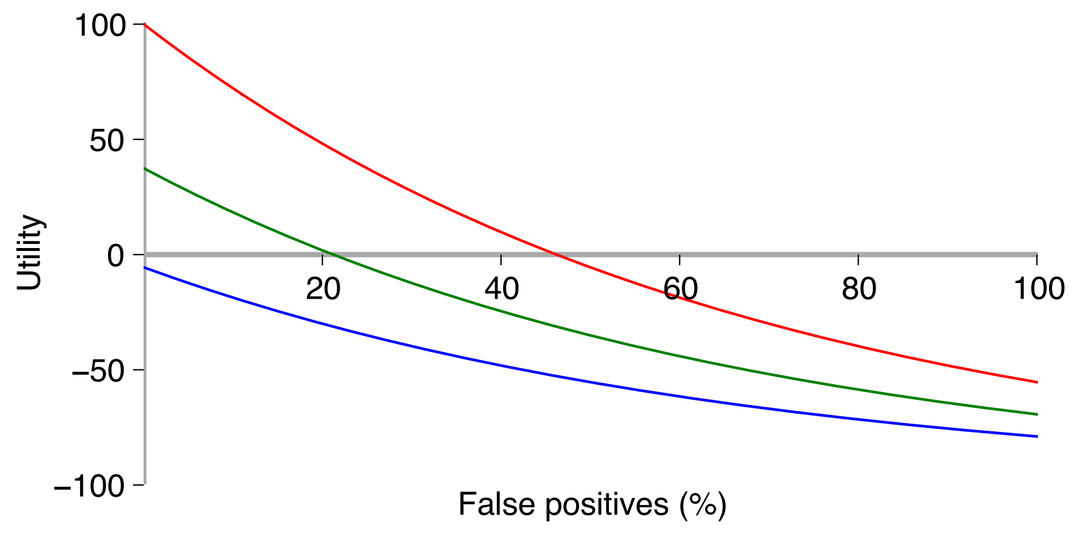
Evaluating censorship resistance strategies

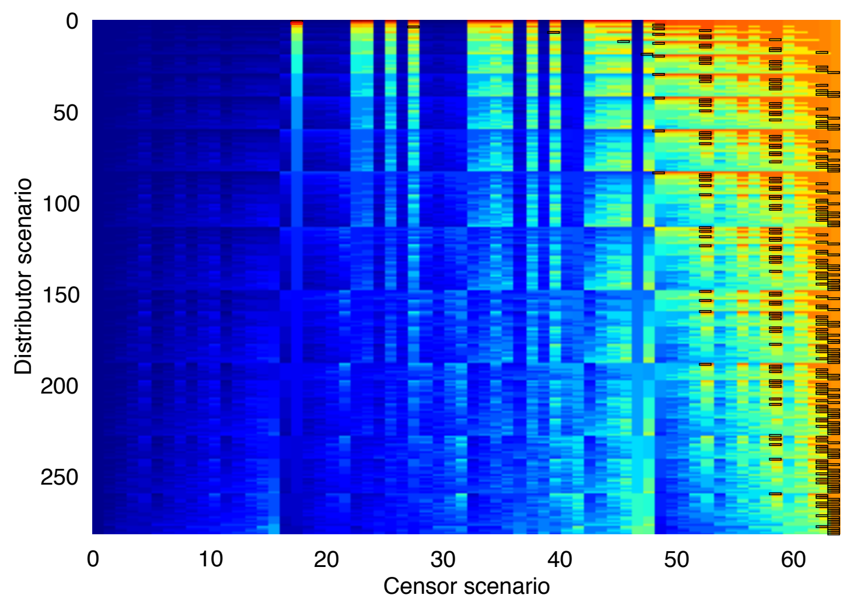
Steven J. Murdoch

A wide variety of censorship-resistance approaches have been proposed. Each has its own characteristic advantages and disadvantages in terms of efficiency, how easy it is to block, and how difficult it is to implement. The question is then raised – for a particular situation, which censorship-resistance strategy is best? Should multiple approaches be used in parallel or should all the effort be invested in building one censorship resistance technique? When the goal of the censorship resistance technique is to impersonate other network protocols, what is the optimum set of protocols to impersonate?

We are developing a suite of techniques, drawing upon security economics and in particular game theory, to answer these questions. By modelling the utility of censors in terms of their preferences towards false positives and false negatives we can predict their reaction towards the deployment of a censorship resistance tool. In turn, we can design a censorship resistance strategy which will force the hand of the censor to still allow as much uncensored access to the Internet as is possible.

Work is underway on developing accurate estimates of real-world censor preferences, so as to guide the deployment of censorship-resistance technologies in Tor. These preferences will be influenced by the censor's aversion to embarrassment at having failed to block particular websites and how upset a population will become if material which should not be blocked is censored by mistake.





UNIVERSITY OF CAMBRIDGE

Personal data privacy in the cloud (11/11)

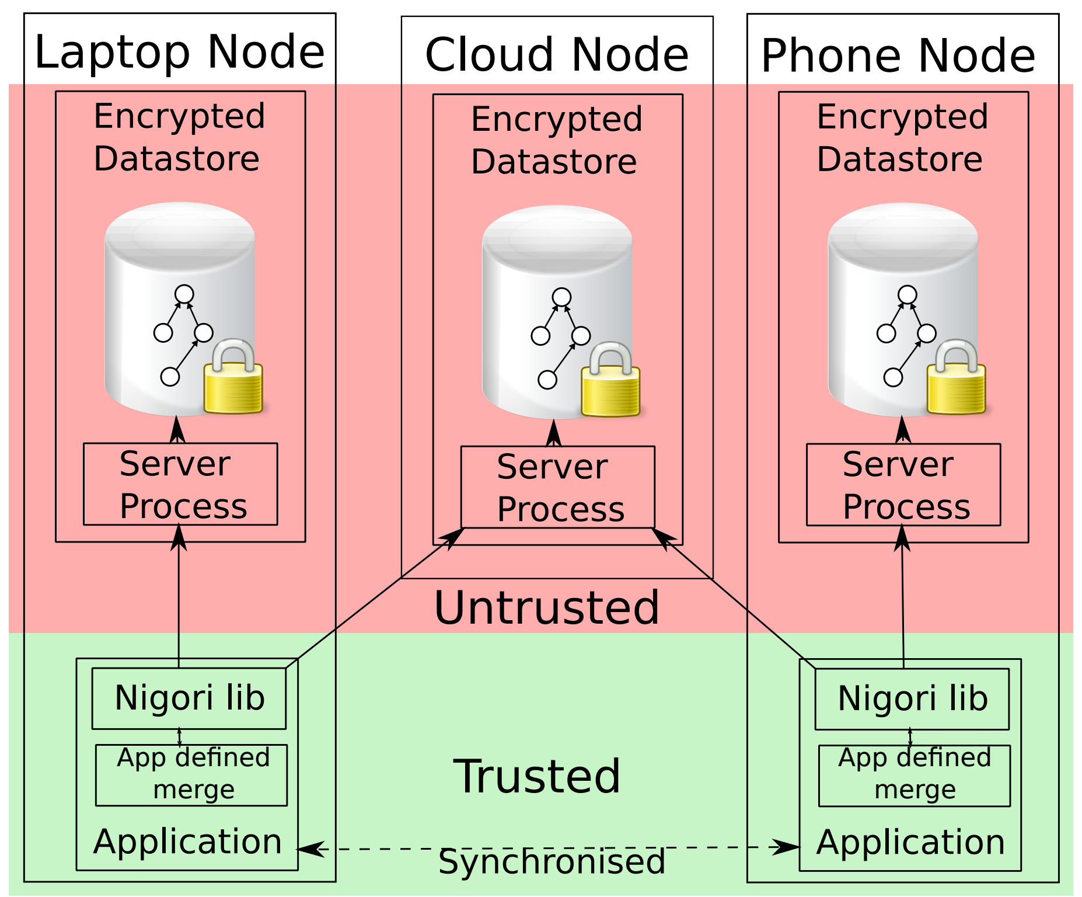
Nigori: Storing secrets in the cloud

Alastair Beresford, Ben Laurie (Google), Andrew Rice, Daniel Thomas

Computer users today have a smartphone, a tablet, a laptop, and a desktop machine. Consequently, many new computer applications seamlessly synchronise user data between devices using cloud storage as a highly-available intermediary. Whilst the communication link between the user device and cloud storage is often encrypted, user data is typically stored in a form which is readable by the cloud provider and the application developer.

The aim of the Nigori project is to develop a practical, application neutral, mechanism for storing sensitive user data in the cloud in such a way that the cloud provider and application developer cannot read any of the stored information. We have an initial specification, and an implementation of Nigori for Java and Android.

Nigori consists of two components: a datastore and a client library. A Nigori datastore is a service, either run locally on the device alongside the application, or run remotely in the cloud. The client library forms part of the application and runs on a user's device, encrypts data, and manages the user's datastores. A typical application deployment will contain one datastore on each user device and one datastore in the cloud; the application can then use Nigori to keep datastores, and therefore user data, synchronised across all their devices.

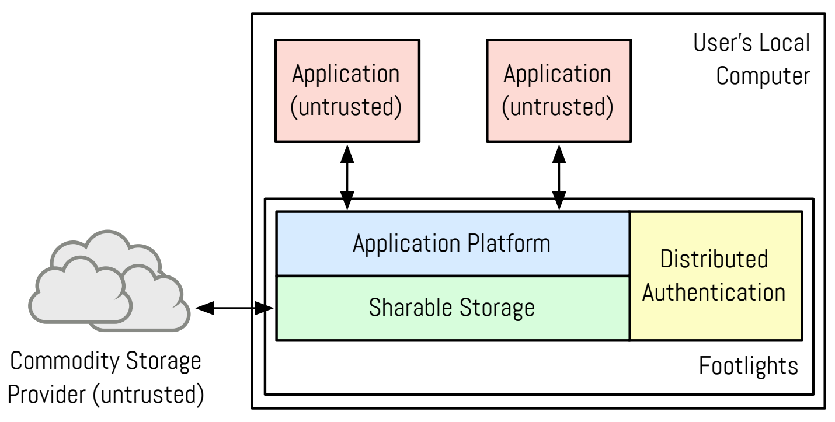


Footlights: a secure substrate for social sharing

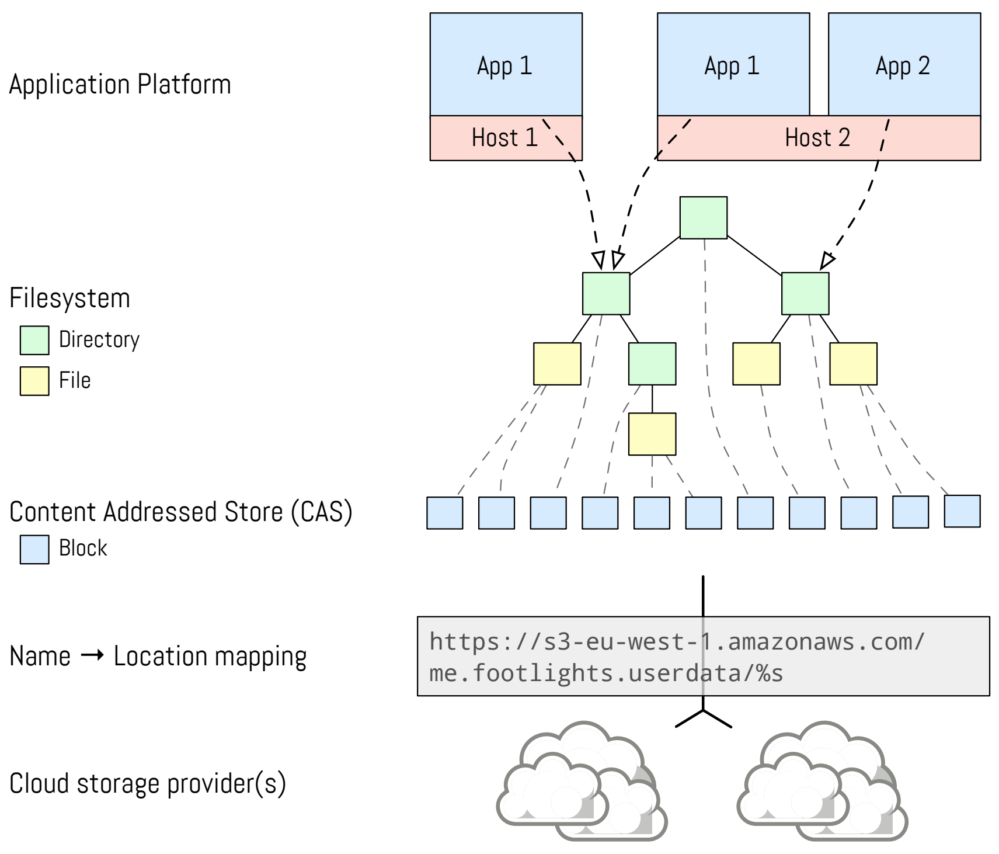
Jonathan Anderson, Frank Stajano

Footlights
All the world's a stage...

Centralised online social networks (OSNs) are incredibly popular, but have been demonstrated to leak users' private information. In addition to accidental leakage, OSNs, such as Facebook, share information more widely than users ask in order to to support their business models.



Footlights explores an alternative model, a hybrid that exploits centralised infrastructure for performance purposes and distributed cryptography techniques for user privacy.


Application Platform
App 1
Host 1
App 1
App 2
Host 2
Filesystem
Directory
File
Content Addressed Store (CAS)
Block
Name → Location mapping
https://s3-eu-west-1.amazonaws.com/
me.footlights.userdata/%s
Cloud storage provider(s)

Footlights uses commodity cloud storage as the backing for a distributed encrypted filesystem. Content is broken into fixed-size ciphertext blocks whose linkages are only visible after decryption. Blocks are immutable and deterministically named, so all users can share a global storage namespace without global identities: authorisation is independent of authentication. Footlights exposes this filesystem to applications that run on the user's computer, where they can be confined and their activity can be observed. Applications can perform arbitrary computation, but access to user data is always anonymous and usually implicit and indirect.

Finally, a distributed authentication scheme allows users to connect to their friends and benefit from social applications any time, anywhere.

Jonathan Anderson and Frank Stajano, "Must social networking conflict with privacy?" IEEE Security & Privacy 11(3) , May–June 2013.

Enforcing user specified policy

Jatinder Singh, Jean Bacon

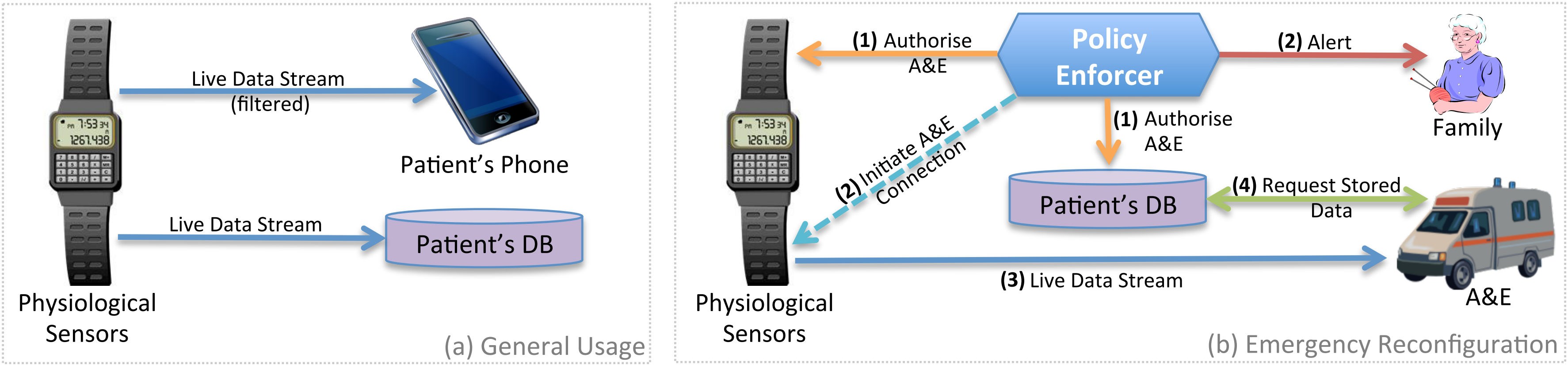
As computing becomes ubiquitous, we notice that:

- services tend to be personalised, tailored to each particular user;
- services are being provisioned over an ever increasing number of system components, which are often grounded in different administrative domains; and
- data may be live (streaming) or stored, control mechanisms must account for both.

Users have preferences as to how, when and with whom their data is shared. Such preferences depend on the circumstances; for instance, a user may choose to relax some restrictions in a medical emergency.

This research concerns the active enforcement of user-specified governance policy in emerging distributed systems. The goal is to enable consistent enforcement across application, system, and administrative boundaries.

Specifically, we consider policy-based middleware, where policy effects user preferences by reconfiguring systems in response to changes in context, triggered by events. This involves issues of policy specification, access control, stream management, component visibility/discovery, data filtering, alerting mechanisms, event composition, and connection management (initiation/cessation/diversion).



(a) General Usage

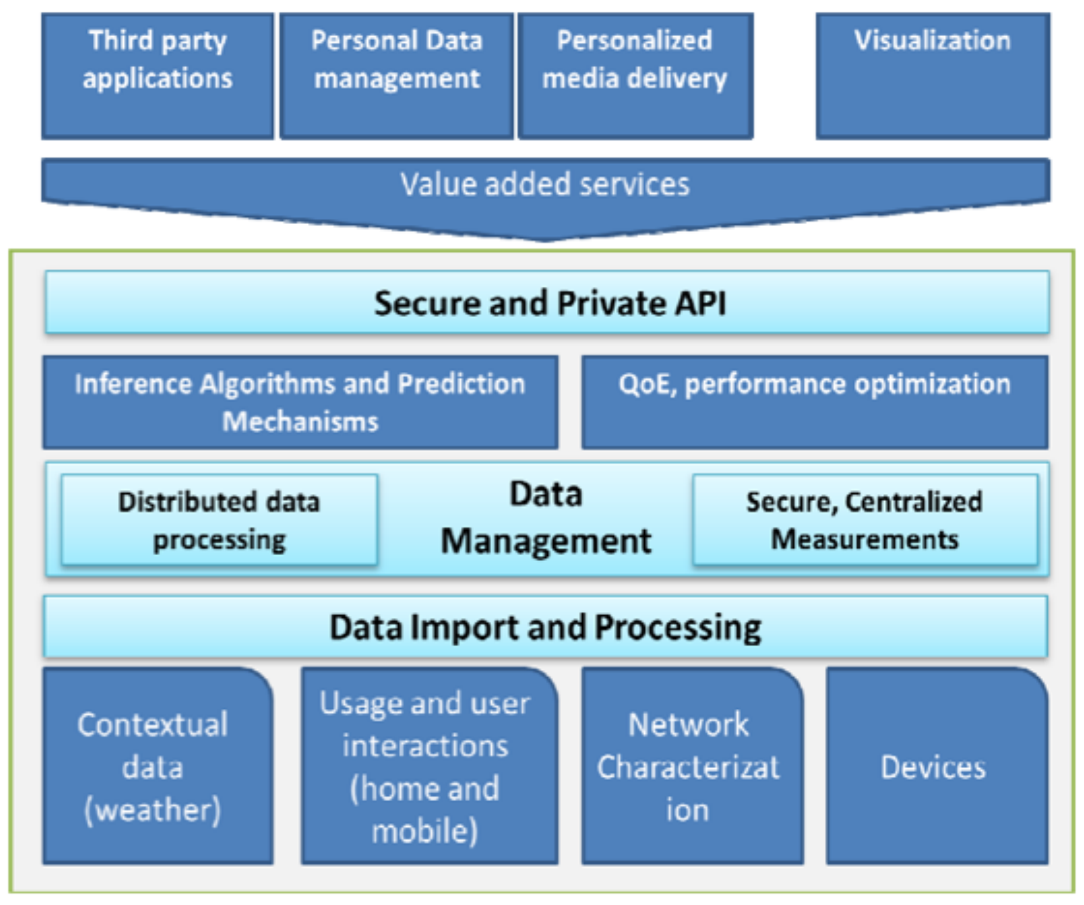
(b) Emergency Reconfiguration

An assisted living environment reconfigured for a medical emergency

User-centric networking

Jon Crowcroft, Anil Madhavapeddy, Thomas Gazagnaire, Amir Chaudhry, David Sheets

There is a staggering amount of online content available today on the Internet. Spurred by ubiquitous connectivity, fast bandwidth, and virtually limitless online storage, people are uploading and consuming data at a remarkable rate. Recommender systems are gathering unprecedented amounts of personal data in order to improve the user experience.


Third party applications
Personal Data management
Personalized media delivery
Visualization
Value added services
Secure and Private API
Inference Algorithms and Prediction Mechanisms
QoE, performance optimization
Distributed data processing
Data Management
Secure, Centralized Measurements
Data Import and Processing
Contextual data (weather)
Usage and user interactions (home and mobile)
Network Characterization
Devices

On the flip side, there is growing concern among users and privacy bodies about the potential for abuse with this kind of data. This will become even more pressing once more-detailed and contextual information about users starts to get incorporated in various services. The UCN project aims to enhance mechanisms to support privacy-preserving content recommendations and strike a balance between individual protection and the needs of the service industry.

We are building a Personal Information Hub (PIH), which is a logical repository for all the data generated by, and collected about, the end-user. The individual has total control over all the data stored inside the PIH and can choose to share it (or not) with specific providers. Importantly, using mechanisms built into the PIH, the user can also dictate the form of the data that will be shared (e.g., by obfuscating to varying degrees) and how the data will be used (by restricting the kinds of operations that can be carried out on it.).

The PIH software stack is based on the MirageOS operating system (http://openmirage.org) and the Nymote networking and storage substrate (http://nymote.org), which is being developed as open-source software.