Technical Report

UCAM-CL-TR-666
ISSN 1476-2986

Number 666

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

A pact with the Devil

Mike Bond, George Danezis

June 2006

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

ISSN 1476-2986

A pact with the Devil

Mike Bond and George Danezis

University of Cambridge, Computer Laboratory
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
{Mike.Bond, George.Danezis}@cl.cam.ac.uk

6th June 2006

Abstract

We study malware propagation strategies which exploit not the incompetence or naivety of users, but instead their own greed, malice and short-sightedness. We demonstrate that *interactive propagation strategies*, for example bribery and blackmail of computer users, are effective mechanisms for malware to survive and entrench, and present an example employing these techniques. We argue that in terms of propagation, there exists a continuum between legitimate applications and pure malware, rather than a quantised scale.

1 Introduction

"*The stars move still, time runs, the clock will strike,*
The devil will come, and Faustus must be damned."

The Tragicall History of D. Faustus — Christopher Marlowe

Computer viruses and their payloads have developed through an interesting historical chain. Morris Jr.'s early example of a self-replicating program was caused significant damage to the early Internet through poor calibration of its propagation strategy [5]. In the age of floppy disks, memory persistence across warm reboots, and multiple disk drives were the vectors of the highly destructive viruses that evolved as weapons in a war of kudos between hackers. In the age of Internet worms, exploits are a dime a dozen, and the most successful viruses use multiple infection vectors in parallel. Meanwhile, the payloads (temporarily impotent during the exploratory "development" phase) now perform targeted acts of malice against one site on the net or another, or are simply about money making. Modern worms are more concerned with creating and controlling botnets and sending spam than they are with deliberately upsetting users.

The countermeasures used against the above schemes have also followed a similar evolution. They range from simple 'computer hygiene' (cold boot your computer), through user education, to sophisticated anti-virus software, which today include full virtual machines [6], thousands of virus detection templates, and constant patching to online systems. It is unlikely that in the near future any of these techniques will bring an end to malware propagation, and it is far more likely that the arms race between propagation and defence will continue *ad infinitum*. A key point is that, through all this evolution, it has been assumed that users are the enemies of the malware which (nearly by definition) acts against their interests.

But in the study of all the selfish schemes that use replicating code to achieve petty human end, have we in fact under-estimated the enemy? To better understand the spectrum of propagation techniques, we introduce the concept of the Satan Virus (named following the tradition from [7]). This virus propagates and survives not only using the conventional arsenal of exploits and deception, but also employs interaction with the user. We present a concrete example which employs *bribery* and *blackmail* to acquire and retain hosts.

Our key contribution, following recent work on the economics of information security [1], is to demonstrate that malware can provide enough incentives to users for them to willingly maintain it on their systems, and can again provide in the medium-term enough disincentives to them removing it. Users can therefore enter in "a pact with the devil" that confers on them some powers, that the virus shares with them, but as they soon realise, some heavy responsibilities too. Not surprisingly, it is the darker human traits that such malware seeks to foster and exploit – greed, curiosity, need for power, fear, shame, lust, to name but a few.

In section 2 we present and analyse an example design, and in section 3 we explore the full range of threats and rewards that viruses can use to influence user behaviour. In section 5 we consider countermeasures, and explore a disturbing truth: otherwise useful software containing malware or ad-ware, the free clients for peer-to-peer networks, and the numerous mainstream applications that bundle together desirable and undesirable features all come together to form a continuum from 'legitimate software' to the 'Satan Virus'.

2 The Satan Virus

The term 'Satan Virus' is somewhat symbolic – it is a conceptual super-virus that carries the malice of the devil, and will employ the most ruthless techniques to achieve its ends. While we hope that this ideal is beyond conception, many cruder images of it could be just around the corner. We can thus speak of a family of Satan Viruses, each credible and implementable. In this section, we explore one instantiation: a simple but concrete design for a virus that propagates and entrenches using bribery and blackmail; we then analyse its properties.

2.1 Design principles

Satan Viruses are based on two fundamental design principles:

1. **The carrot principle.** First, the virus convinces the user to execute it by conferring him or her with a certain advantage. This advantage is true and tangible, it is backed up by evidence that clearly demonstrates it can be provided, and should ultimately satisfy the user. There is no deception involved at this stage – and the user knowingly "sells his soul to the devil" to acquire this advantage. As long as he honours his side of the "pact" the advantage is provided. This first principle provides incentives for the user to execute and maintain the virus alive.

2. **The stick principle.** Second, the virus, in its co-existence with the user, gathers information about the user's activities, lifestyle and habits. It then tells the user

that if an attempt is made to remove the virus, the gathered information will be used to hurt the user. This provides further disincentives for the user to remove the virus.

In its purest form this Satan Virus does not deceive: it provides the advantages it claims, and does not gratuitously hurt the user – it fulfils its side of the contract. The main catch lies in the contract terms, which can be ever expanding: to maintain the virus alive on the "possessed" computer, to assist the virus in spreading, whatever lies within the imagination of the virus author.

Naturally there is no reason to expect such pure (and in some twisted way, honest) strategies to be employed in isolation, when mixing them with deception and other more automated ways of executing malware would provide the virus designer with better propagation characteristics. We again stress that the strategies presented here may appear a poor substitute for traditional mechanisms: it is because they are not intended to be a total replacement.

2.2 The instantiation

For the threat to be tangible, the virus must implement the carrot and stick principles in a robust way: the principles must respectively seduce the user compellingly, and resist trivial bypass. For this instantiation, we will use access to another user's files as the carrot, and revelation of this access to the party spied upon as the stick. Assume there are three parties: Alice, Bob and Charlie. Alice is already infected with the virus, and Bob and Charlie are related to her (employees, colleagues, friends or family). The virus propagates in the following manner:

1. **Temptation.** The virus sends an email from Alice to Bob, offering access to all of Alice's emails and documents. To make the offer more enticing, extracts from these documents containing Bob's name, or other interesting keywords can be included. Bob can chose to accept this offer, by downloading the virus (that can be hosted on Alice's computer or bundled in the email) and executing it. As a result he should have full access to Alice's documents, with a search interface to help locate files of interest.

2. **Monitoring.** As soon as the virus has installed itself, it starts recording everything that Bob does, and in particular the accesses to Alice's information. Crucially, this includes the search queries performed as well as logs of the documents retrieved. This information is sent back to Alice or another infected third party (that can be known through Alice) for safekeeping, but it is not revealed. The key intuition is that the virus avoids the hard problem of automatic detection of 'blackmail' material on Bob's computer, by collecting evidence on the unsavoury act of spying that it has tempted Bob to commit. The unauthorised access to Alice's computer, both in the files Bob views, and the search terms he uses (revealing his suspicions of Alice) should in most cases be incriminating material.

3. **Blackmail.** When a critical mass of incriminating evidence of unauthorised accesses from Bob to Alice's machine has been gathered, the virus emails Bob with a warning.

The warning specifies that if an attempt is made to remove the virus the information gathered will be revealed. A snippet of the information can also be provided to substantiate the threat. To safeguard the virus against retaliation, it sets up a life-line between Bob and Alice's machine (or a compromised third party holding the incriminating evidence), to monitor Bob's computer, and ensure that it remains infected. If Bob's computer does not appropriately respond, the evidence is released.

4. **Voluntary Propagation.** Bob is asked by the virus to provide a target to which it might spread. Bob selects Charlie. Bob is told that Charlie would have the ability to read Alice's documents (not Bob's) and that he would have the ability to read Charlie's documents. The 'invitation' will appear to be coming from the virus residing on Alice, in the form of an email tempting Charlie to read her documents. Thus the incentives are aligned for Bob to assist, and the virus propagates.

5. **Involuntary Propagation.** In case the virus has not propagated enough through the addresses provided by Bob, it considers that Bob has breached his side of the "pact", and sends itself to Bob's contacts, as harvested through emails, contact lists, documents, etc. The virus now encourages recipients to install it, using the incentive of access to Bob's files.

As a side effect of the propagation method described above, the virus nodes can construct a peer-to-peer network, that can be used to propagate payloads or commands from their "master". The lifelines between them can also be used to manage the network, and make sure that it stays connected. While such peer-to-peer design problems must be tackled by the virus writer, we shall not discuss them further as they apply equally well to traditional non-interactive malware, which has already demonstrated some capability in this area. Some further implementation issues and open research problems are presented in section 4.

2.3 Analysis

We will spend some time considering the incentives of Alice, Bob and Charlie, in each case where action is solicited.

The *temptation* step of the propagation is by far the most uncertain and risky from the point of view of the virus. At this stage, a tempted Charlie can just say 'No' – without any repercussions. The challenge here is incentive design: the author of a virus exploiting interactive propagation must design lures which appeal to Charlie, and a framework for soliciting assistance from Bob. This might require Bob to aid in customising an email to tempt Charlie with Alice's files, or require him to surreptitiously visit Charlie's PC and click to open the attachment. Strategies commencing with a threat of harm on an identified 'significant other' of Charlie lie beyond the scope of this example.

Just as important as design of the lures, the virus author must calibrate the strategy to spread with the correct amplifying effect. Studying the influence of simple local tactics for involuntary propagation in existing virus code already creates a headache for analysts; the same challenge will meet the designer of interactive propagation routines. A partial solution could be to design lures which are good for propagation in particular directions through the social network; it will need lures to spread both up and down hierarchies in corporations, and from friend to friend across 'gossip bridges' between corporate networks.

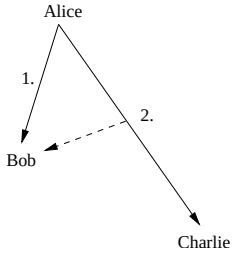

Figure 1: Bob is tempted by Alice's files (1) and then uses Alice's files to tempt Charlie (2). As a result Bob can access Alice and Charlie's files.

The generic lure of spying in this virus instantiation will likely have immediate appeal to Charlie if there is an asymmetric power relation between them (whether known to Bob, or data-mined [8]), so it will certainly travel effectively down hierarchies, and plausibly up as well. With the minor addition of a hint of what salacious material might be found (substantiated or unsubstantiated) people in peer groups can also be encouraged to spy. In essence, human curiosity is at the heart of this lure. We believe the algorithm proposed above is sufficiently simple to correctly calibrate for an amplifying effect.

During the *monitoring* phase Bob is probably aware that the virus might be giving other people access to his files (it is not yet apparent that the incriminating information to be actually used will be the browsing patterns on Alice's files). This might lead the user to moderate his activities, and delete incriminating information from his computer. If this is done after the virus has been installed it might be used to discern what material is interesting to use for blackmail purposes (deleted material is bound to be more interesting than random files). The material could have been deleted before the virus has been installed (as a precaution). This illustrates quite vividly the "virtuous sinner" paradox: if Bob does not perceive that he has something to hide, genuinely or because he has deleted the information, he will be more tempted to spy on Alice, since he is less worried about it happening to him.

When the user is notified of the *blackmail* material stored on third party computers he is pretty much stuck. Often the effectiveness of blackmail is severely limited by funnelling the profit to the blackmailer, but as we have blackmail from an automated system there is no exit from the user's dilemma[1]. Bob may well realize that all his activities are recorded and stop using the advantages provided by the virus. He might even chose to stop using the infected machine altogether. Yet in order not to face the repercussions he has no choice but to leave the machine connected to the network, and in contact with the other viruses. Survival is therefore achieved for the virus.

The *propagation* phase sees the virus exploit its edge on Bob to have him assist propagation, and of course is the mirror image of the temptation phase. Humans are much better at social engineering than any automated tool: the key aspect of our design is that Bob has a positive incentive to assist in infecting Charlie, through access to Charlie's files, as well as keeping his own virus satisfied, and placating his fears that it will take unilateral action. If Bob's support does not materialize, either because he does not want to assist,

[1]There are several examples of viruses which attempt to blackmail a user, for instance by encrypting their files, then demanding payment for the key; however virus authors plans for remuneration have not been coherent[14].

or because he does not understand, propagation happens automatically, and Bob is used as a bait to entice other users. Note that the failure of Bob to propagate the virus should not lead to the release of the blackmail material – this may threaten the survival of the virus on Bob's machine. The combination of a tangible reward for success and threats upon withdrawal is the same as that which sucks individuals into organised crime.

One way or another propagation onwards from Bob will happen – and it is not the case, as it may appear, that the number of parties under observation, like Alice, remains small. Bob identifies targets such as Charlie, that are especially interested in observing Alice, in order to prevent the virus from using him as a bait to entice other users. As we have seen, Bob gains the ability to spy on Charlie in addition to Alice, and he can now entice further targets by offering access to Charlie's files. Meanwhile, the new infectee, Charlie, has a smaller pool of candidates to voluntarily install the virus, since people especially interested in Alice's files are bound to be limited – and there is one less candidate (Bob has nothing to gain from accepting an offer from Charlie). Therefore Charlie will have to start propagating the virus in different circles, or admit defeat and not propagate it at all. In these cases the virus on Charlie's machine will detect that it has not been propagated and will initiate the *involuntary propagation* phase. In this respect the virus works like a pyramid scheme, by providing early adopters the most benefits as they rise high up in the hierarchy, while leaving those at the lower levels exposed to a greater probability of failure.

Bootstrapping the spread of the virus is not difficult: manual hacking or botnet distribution techniques could be used to install the first few instances. More interestingly, the virus could be offered on the internet as a trojan that purports to simply allow remote monitoring of other users' computers. Here, Bob would surreptitiously install the trojan server on Alice's machine, and the client on his own. BackOrifice [9] is an example of such software that has made quite a few waves in the popular press [10]. At a later date, the trojan enters its monitoring, blackmail and propagation phases, and spreads from there.

3 Interactive propagation strategies

We now explore the spectrum of interactive propagation strategies – which is ever expanding as more of human activity moves into the networked world. The carrot principle and stick principle generalise into an array of possible threats and rewards, in the same way as the tactics of a negotiator or politician might be classified. Also, the fuller spectrum plays across the full lifecycle of a virus. Threats and rewards may be *execution phase* actions, which can be carried out in a short space of time as soon as a host is infected, or *survival phase* actions, which take some time to carry out successfully, and require the host to remain online. Finally, *lifecycle phase* actions are only meaningful in the context of the wider purpose of the virus, and success may not be guaranteed by persistence of infection on a particular host.

The table in figure 2 gives a few examples of the rewards the Satan Virus could provide to incentivize the user in order to keep it alive. Note that as the virus spreads, it can exploit the resources of others, so there are positive network externalities for its propagation. Consider a Satan Virus that shares MP3 music files amongst all infected users. As the number of users increases, the number of available files grows, and users get even more value by joining the network [11]. The cynical reader might already believe

Reward	Examples
Threat Enactment*	"I'll carry out threat X on Y, and you can watch!"
Privacy Invasion*	"You can browse X's hard drive"
	"You can read X's email archive"
	"You can watch X's webcam/mic"
Revelation*	"I'll tell you what X said to Y"
	"I'll tell you what I found on X's hard drive"
Fabrication*	"You can forge emails from X"
Mischief*	"You can seize control of X's PC"
Virtual goods	"You'll get tons of free porn"
	"You'll get free software"
Real-world goods*	"You'll get free goods to your door"
Innovation	"You can use this really cool feature"
Unsubstantiated	"You'll get seven years good luck"
	"Your true love will return to you"

Figure 2: Taxonomy of rewards (* marks cross-party rewards)

that this mechanism is the key to the success of peer-to-peer networks (we discuss the continuum between malware and other software in section 5). Figure 3 shows the broad categories of threats conceivable, giving one or two examples of each. Threats marked with a '*' are cross-party in that the enactment of the threat might be considered a reward to another, or that another party is explicitly involved in the process.

4 Implementation challenges and open questions

Whilst incentive design is indeed subtle and hard, the virus instantiation of section 2 is not much more difficult to *implement* than a conventional virus: it requires a state machine on each node, some basic communications protocols, and a dozen or so pre-prepared message boxes for communication with the user. It is not immediately obvious however that the more sophisticated propagation strategies of section 3 are credible; we briefly consider the challenges of architectural design.

Assume the virus is adequately seeded, bootstrapped from an existing botnet or through manual infection. It may then organise itself into a communicating network in a peer-to-peer fashion. Once it has an adequate grasp on the host computers of a user base, it begins harvesting information about the users. It performs traditional activities such as logging keystrokes, but also analyses email archives and instant chat client configurations. It performs social network analysis to automatically determine significant links from person to person.

Once it has an awareness of the social structure of the human network above its hosts [13], it can start to approach humans for assistance in furthering its spread, and entrenching itself deeper into existing hosts. To do this, the virus spawns "demons" – communicating conversational clients which interact with a specific user via email or instant messaging, with the goal of obtaining their assistance, and additionally enacting whatever overall payload the virus may carry.

The demons manipulate the users through a system of rewards and threats, harvested

Threat	**Examples**
Data Destruction	"I'll delete all your files"
Privacy Invasion*	"I'll forward emails from your archive daily to X"
Revelation*	"I'll tell X that you said Y to Z"
	"I'll tell X about the porn I found in your web cache"
	"I'll put all your data on the net, searchable from Google"
Fabrication	"I'll make up an email telling X you slept with Y"
Desecration	"I'll distort all your digital camera photos and reduce them to 1/4 size"
Framing	"I'll plant illegal images on your computer"
Hardware Damage	"I'll blow-up your monitor"
	"I'll re-flash your BIOS and kill your PC"
Security Exposure	"I'll sabotage your security" (Kleptographic attacks [12])
	"I'll harvest all your keystrokes and passwords"
	"I'll get your credit card number and abuse it"
Real-world*	"I'll order something using your credit card for my friend"
Unsubstantiated	"I'll get your sister beaten up"
	"I'll give you seven years of bad luck"
Nuisance	"I'll phone your mobile at all times of day and night"
Unwanted goods*	"I'll send unwanted goods to your door"
Reporting	"I'll report your illegal downloads to the RIAA"
Access Denial	"I'll prevent you from using Word or Excel"
Composite	"I'll bug you with a threat or reward every 15 mins"
	"Now we're going to play truth or dare"

Figure 3: Taxonomy of threats (* marks cross-party threats)

from the compromised nodes it controls. If a user does not react to rewards, maybe she will react to threats. When a computer is compromised, the virus scans its content and produces "threat modules" and "reward modules" based on the content and resources found. These broadcast out their existence via the peer-to-peer system, and the demons barter for them. The specific task of each node are as follows:

- Nodes host *demons* that engage in conversation with others. Each demon binds to one person, is instantiated once the system determines a good chance of success in manipulating that person, and is then responsible for controlling that person's fortunes.

- Nodes accept *incoming reward and threat modules*, which may be time locked, or locked with some cryptographic key. When a reward package is unlocked it may confer a simple benefit, for instance releasing an MP3 or pornographic image to the owner of the host, or it might release some more sophisticated reward, such as access to the email archives of another user close by in the social network.

- Nodes scan themselves continuously, looking for new potential *outgoing reward and threat modules* to construct. Nodes then broadcasts the availability of these over the peer-to-peer network to demons who might wish to make use of them in manipulating users.

- Nodes measure parameters of their host, such as resources available (e.g. bandwidth) and also the number and vigour of attempts to remove the infection. These are called *triggers* and are broadcast back to demons in the same way as threats and rewards. Demons can use triggers as an extra channel of observation of the user, in addition to direct communication. Demons can thus detect attempts to damage the peer-to-peer system, and threaten the user appropriately, in order to make her desist.

4.1 Human interaction

Human perception of computer viruses is subjective. Infectees consider a virus as an entity that resides on their machine; anti-virus people consider strains of viruses or consider the author of the virus directly. However, once viruses engage in non-trivial communication with humans, perception of the virus becomes more important. In the majority of cases (but not all) the virus will have compromised the user's PC, yet the demon corresponding to a user will likely not entirely reside on it, lest it be vulnerable to deletion attempts. The virus may chose to be honest about internal social structure, for instance if it resides on a single different machine (for instance claiming "I am your Boss's virus"), or it could claim to reside ephemerally on the network. This captures the fact that demons have some shared state, but in other ways are quite separate. Most importantly, the virus should engineer the user's perception to gain an advantage, and have maximum psychological impact.

Meanwhile, particular demons must perceive humans, and the manner of perception hinges on both psychological and technical issues. One possibility is to define a person as an active email address that is used for at least some personal mail. However such a perception ignores the benefits of interaction through instant messaging clients, and direct interference with the runtime of a users computer. Various Japanese dating simulators exist [4] which operate through email communication, and this indicates that if people are willing to converse with email bots for enjoyment (and indeed for social training), that it is a sufficiently rich form of communication to deal with emotional issues, but also not too rich for the bot to expose itself.

Finally, a base communication link can be augmented with more intrusive channels, for instance using SMS gateways to interact with mobile phones, or using Voice-over-IP gateways to send pre-recorded instructions by telephone, or simply to make nuisance calls.

5 Defence and conclusions

Are we already adequately equipped to deal with an outbreak of the Satan Virus? What existing countermeasures would be effective, and what new countermeasures complement the new propagation strategies?

Clearly such viruses can be fought to some extent using conventional anti-virus software. The virus may try to use its edge on the user to have her disable anti-virus measures, but in a corporate environment the user may not have total control over her PC, thus the virus is threatening the wrong person. If the signature of the virus code becomes known, anti-virus software will warn the user when they give in to temptation. The virus might thus switch to granting spying capabilities up front, for instance through the user's

web browser, then monitor and perform blackmail in order that the user disables countermeasures *before* virus installation. Either way, conventional anti-virus software remains a significant threat, but a known threat, and one largely beyond the scope of this discussion.

Communication will be the key new battleground – this distinguishes Satan Viruses from existing strains. Gaining access to, and then the attention of the real user is the new challenge. Anti-virus or OS software may try to mask the attacks of the virus from the user by filtering out tempting messages. The name of the game is then spam detection, but can the virus stay ahead for long if it is cut off from a human creator, given the rapidity of spam filter evolution?

Often in tandem with a primitive virus outbreak comes a 'good virus' propagating through human incentive alignment, warning of the bad virus on the way, for example "If you see email with subject X, don't read it, delete it straight away". We call these *viral prophets*, which (generally considered a nuisance fighting current viruses) may be effective at fighting Satan Viruses, by spreading a naive but effective 'morality'.

The peer-to-peer architecture creates a new set of weak spots. Damaging the virus' perception of the outside world could cause it to push its energy into infecting imaginary new hosts. Limiting communications to other nodes (through temporary or permanently blocking), might imbalance the system with limited collateral damage through threat enactment. A final category is infiltration attacks – fighting the virus on its own communications infrastructure using sabotaged nodes and links. In our example architecture, offering of dummy rewards and threats could make a whole pool of demons impotent. However, if viruses embrace instant messaging before spam prevention techniques in these area are fully developed, they may be able to do more damage than they would via email.

Maybe the only long term solution is to give users the proper tools to compartmentalise their PCs, and maintain their privacy. Initiatives such as *trusted computing* may both help and hinder – they could provide hard boundaries to keep a virus from illicit access to address books and privately marked files, however they could also protect the virus, rendering it invulnerable to reverse engineering, and entrenching it more deeply.

Finally it is worth noting that a practical way to develop defensive measures is through experiencing weakened attack (indeed this is how much human virus immunisation is performed). The technical sophistication required to craft the *ideal* Satan Virus is quite a hurdle, thus potential deployers of the ideas in this paper will likely teach the community more about defence than scar it with the damage they would initially do. Aside from traditional skills, such as systems and network programming, a virus writer is required to conceive a data mining engine capable of constructing effective reward and threat packages. This would require implementing cutting edge tools from the fields of information retrieval, natural language processing, social network analysis, epidemiology, etc: a set of skills that are not widely available in the computer underground.

5.1 The propagation continuum

Our explorations through this paper have given us a glimpse of the full attack surface for propagating software, and our example Satan Virus of section 2 shows these avenues are not flights of fancy but could quickly become quite real. But more important than acknowledging the variety of techniques, in studying how virus propagation interacts with user incentives, we discover *the propagation continuum*. Viruses, botnets, peer-to-

peer systems and conventional applications are not dichotomous, they are data points on a scale of malice. They all attempt to survive on end user systems, by providing both positive functionalities, penalties for being removed and often some negative side effects. Do the benefits of a peer-to-peer client such as the ad-ware ridden KaZaA [3] outweigh the problems? Will a user coexist in relative harmony with a search-hijacking, click-counting tool because of the risk of damage to their OS during a fudged uninstallation? When someone sends you a word document that you cannot read with your viewer, is Microsoft Word propagating itself in the same way?

Accepting this continuum, may lead us to do away with the already unwieldy raft of terminology for propagating software. The term *virus*, originally a medical one, is now irrevocably stigmatised with bad and undesirable effects, meanwhile modern computer viruses are better described as worms. However the worm definition falls short when considering the symbiosis of propagating software with its users. Meanwhile classification based on harm to the user will become more difficult (is a particular piece of *malware* truly bad?); as the aggression ever mounts in the media player wars, and the bundling ramps up, how will bundle-ware side-effects be classified?

So we could now argue that there are devils everywhere, spreading software that we might not approve of: they are bribing with great new features, legal or otherwise, and blackmailing with threats of revelation, or just with high switching costs [2]. In this paper, we have explored the farthest and most malicious end of the propagation continuum, and we stress that the ideal Satan Virus concept we find is no cause for panic – real-world implementations will inevitably fall far short of the mark. Once research into phishing, anti-spam and anti-spyware technology integrates more fully with virus defence, we may enter a new realm of content approval and adjudication. Therein lie fascinating new avenues for research that could help users properly gain control of their PCs, and understand what they are letting themselves in for when they click 'yes' – be it on temptation from the Satan Virus, or on a twenty page licence agreement. One thing remains certain: until this day comes, running other peoples software will remain an activity to be undertaken with caution.

References

[1] Camp, L. Jean; Lewis, Stephen (Eds.). "Economics of Information Security", Series: Advances in Information Security, Vol. 12, 2004.

[2] Carl Shapiro, Hal R. Varian. "Information Rules", Harvard Business School Press (November 1, 1998).

[3] NS Good, A Krekelberg. "Usability and privacy: a study of Kazaa P2P file-sharing" In Proc. CHI 2002, CHI Letters, 2003.

[4] "Japanese Men Date Hot, Sexy Bots", Wired, available at `http://www.wired.com/news/culture/0,1284,40369,00.html`

[5] Eugene H. Spafford. "The Internet Worm Program: An Analysis", Purdue Technical Report CSD-TR-823, 1989.

[6] Peter Szor. "The Art of Computer Virus Research and Defence", Addison-Wesley, 2005.

[7] RJ Anderson, RM Needham. "Programming Satan's Computer", in Computer Science Today, 1995.

[8] David Lyon. "Surveillance Society: Monitoring Everyday Life", Open University Press, 2001.

[9] "Back Orifice 2000", available from `http://www.bo2k.com/`

[10] James Glave. "Back Orifice a Pain in the ...?", Wired, July 1998.

[11] Andrew Odlyzko and Benjamin Tilly. "A refutation of Metcalfe's Law and a better estimate for the value of networks and network interconnections", Manuscript, March 2, 2005.

[12] A. Young, M. Yung, "Kleptography: Using Cryptography Against Cryptography", Eurocrypt '97, page 62-74, Springer-Verlag, LNCS #1233, ISBN 3-540-62975-0

[13] N. Eagle (2005), "Machine Perception and Learning of Complex Social Systems", Ph.D. Thesis, Program in Media Arts and Sciences, Massachusetts Institute of Technology.

[14] "Virus Hold Computer Files 'Hostage' for $200", Fox News, 24th May 2005, available at `http://www.foxnews.com/story/0,2933,157469,00.html`