MAY 2006

VIRUS BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

IN THIS ISSUE

LOCAL ISSUES

We hear about global threats attacking systems across the world every day. Meanwhile, however, we are also seeing an increase in targeted attacks that affect only specific geographic regions or languages. Kaoru Hayashi explains some of the issues surrounding regional threats.

page 4

A NEW METRIC

Oren Drori and Nicky Pappo present their model for calculating malware penetration probability during an outbreak.

page 6

A DIFFERENT ANGLE

Aleksander Czarnowski takes a look at one of the largely undocumented aspects of penetration testing: rootkit installation for profiling system security at local level.

page 10

vbSpam supplement

This month: anti-spam news and events; and Tomer Honen explains why being mugged at gunpoint could cost you less than being the target of the next multi-stage phishing attack.

ISSN 1749-7027

virus BULLETIN COMMENT

'The industry needs to look at what can be done to increase its defences, and provide innovative solutions.'

Joshua Corman
Internet Security Systems, USA

DON'T BRING A KNIFE TO A GUNFIGHT

During 2005 there was an evolutionary leap in the threat landscape as hackers moved toward sophisticated, well-funded attacks for profit. Unfortunately, security professionals and vendors seem to be falling further and further behind. Fighting this evolved threat with legacy anti-virus is like using a wooden shield against a rocket – or bringing a knife to a gunfight. Most organizations are woefully unprotected against this new, more sophisticated threat, and few of them understand how unprotected they really are.

The shift of hacker motives from glory to profit has driven substantial advances in their strategies and technologies – diminishing the effectiveness of legacy anti-virus security solutions dramatically. When glory was their motive, malcode writers wanted to infect hundreds of thousands of systems – but with financial or political motives, such visibility could jeopardize their objectives. Today's hackers want quality footholds, not quantity. They want stealth, not visibility.

When the driving motive is profit, 'total number of infected systems' is a poor indicator of malcode severity. Consider the Israeli espionage trojan that went undetected for nearly two years and in the process stole thousands of confidential documents. Insidious and damaging custom attacks are a rising trend. 'Designer trojans' are running rings around online banking providers. While it didn't make the Top 10 list on any AV website, bank-specific malcode is far more severe than yet another Bagle variant. What keeps us up at night is not Bagle.xyz. Enterprises fear making the same mistakes as *ChoicePoint*. Governments fear state-sponsored attacks. An AV signature can only be provided *after* successful fraud because a signature requires a 'patient zero'. With designer attacks, patient zero is the only victim – and a signature is like giving a vaccine to a corpse.

Today's malcode writers have studied legacy defences and have adapted. Two such adaptations were captured in a feature article in the September 2005 issue of *Virus Bulletin* – short span and serial variant distribution trends (see *VB*, September 2005, p.9). Short span showed how hackers complete 100 per cent of their infection drive before any AV signatures are released – relegating AV to a 'scan and remove' technology. Serial variant trends showed how releasing several samples extends vulnerability windows and serves to distract and overwhelm AV teams. Add a rootkit and AV may not be able to scan and remove either. Kernel rootkits like ShadowWalker, and VMBRs (virtual machine based rootkits) like SubVirt represent serious threats to any reactive defences. Once they take root, discovery and removal are costly and difficult (see http://www.eweek.com/article2/0,1895,1945808,00.asp). How do you fight an invisible opponent? The enemy has studied our defences, exploited our weaknesses, and changed the nature of the battle.

As Darwin taught us: that which fails to evolve, dies. To turn the tide, the security community needs to make an honest and grounded examination of this 'evolved threat'. The industry needs to look at what can be done to increase its defences, and provide innovative solutions.

The good news is that a handful of virus *prevention* systems routinely thwart designer attacks and prevent malcode from taking root. These virus prevention systems identify malicious behaviour in any code, new or unknown. The bad news is that the vast majority of us are unaware that effective behavioural technologies exist, and how badly we need them. The worst part is that the majority of the 'trusted security providers' have kept the public in the dark and failed to innovate or keep pace with this evolving threat. We face an escalating danger. Education is the first step – 'knowing is half the battle'.

The bottom line is: don't bring a knife to a gunfight. Your enemy has evolved and is using its latest and greatest arsenal. Are you?

Editor: Helen Martin

Technical Consultant: Matt Ham

Technical Editor: Morton Swimmer

Consulting Editors:
Nick FitzGerald, *Independent consultant, NZ*
Ian Whalley, *IBM Research, USA*
Richard Ford, *Florida Institute of Technology, USA*
Edward Wilding, *Data Genetics, UK*

NEWS

VB2006 CONFERENCE PROGRAMME REVEALED

VB has revealed the conference programme for VB2006, the 16th Virus Bulletin International Conference.

The three-day conference programme boasts an exceptional line-up of international anti-malware and anti-spam speakers, and as usual caters for both technical and corporate audiences. More than 40 presentations will cover subjects including: education, forensics, automated analysis, botnets, spam trends and filtering techniques, phishing techniques, Unix malware, Macintosh malware, fraud detection, corporate case studies, the CME initiative and much more. VB2006 takes place 11–13 October 2006 in Montréal, Canada. For the full programme see http://www.virusbtn.com/conference/vb2006/programme/.

SECURITY SURVEY & CHECKLIST

Businesses in the US have been urged to complete a survey issued jointly by the US Departments of Justice and Homeland Security. The aim of the survey is to gain a better understanding of the costs of computer security incidents.

The survey, which has been distributed to a wide range of industry sectors, covers a variety of security-related topics. For example, businesses are asked to describe the types of security incidents they have experienced, their current defence strategies and their concerns about information security. Encouraging businesses to reveal such sensitive information is notoriously difficult, but companies have been assured that the responses to this survey will be held strictly confidential, by law.

It is hoped that the results of the survey will provide enough information to establish some accurate data on the costs of computer security incidents and that they will help the federal government decide where to concentrate its resources in fighting cybercrime.

Meanwhile, the Department of Homeland Security's Cyber Consequences Unit has released the first draft of a checklist designed to help businesses focus on security best practices and on the consequences of security breaches. The Cybersecurity Checklist identifies potential avenues for attacks and recommends ways to protect against them. The list concentrates on six areas of vulnerability: hardware, software access, software supply, network, automation and human operators. According to the Unit's director Scott Borg, the list provides specific guidance for businesses while also recognizing economic realities – including items that are desirable, but which may be difficult and expensive to implement. No date has been given for the final approval of the draft.

Prevalence Table – March 2006

Virus	Type	Incidents	Reports
Win32/Netsky	File	79,897	39.71%
Win32/Mytob	File	65,687	32.65%
Win32/MyWife	File	22,865	11.36%
Win32/Mydoom	File	11,318	5.63%
Win32/Bagle	File	9,541	4.74%
Win32/Lovgate	File	2,951	1.47%
Win32/Bugbear	File	2,549	1.27%
Win32/Zafi	File	1,052	0.52%
Win32/Pate	File	929	0.46%
Win32/Funlove	File	787	0.39%
Win32/Sality	File	447	0.22%
Win32/Sdbot	File	444	0.22%
Win32/Valla	File	363	0.18%
Win32/Klez	File	308	0.15%
Win32/Feebs	File	283	0.14%
Win32/Mabutu	File	195	0.10%
Win32/Mimail	File	176	0.09%
Win32/Bagz	File	172	0.09%
Win32/Gibe	File	124	0.06%
Win32/Dumaru	File	102	0.05%
Win32/Chir	File	99	0.05%
Win32/Reatle	File	95	0.05%
Win32/Maslan	File	83	0.04%
Win32/Bobax	File	71	0.04%
Fortnight	Script	60	0.03%
Win32/Kedebe	File	44	0.02%
Win32/Mota	File	42	0.02%
Wonka	Script	36	0.02%
Win32/Agobot	File	35	0.02%
Win32/Elkern	File	35	0.02%
Win32/Rontokbro	File	34	0.02%
Win32/Swen	File	29	0.01%
Others[1]		351	0.17%
Total		201,204	100%

[1]The Prevalence Table includes a total of 351 reports across 61 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

FEATURE

REGIONAL THREATS

Kaoru Hayashi
Symantec Security Response, Japan

New families and new variants of threats, such as W32/Sober, W32/Blackmal and W32/Beagle, attack systems across the world every day. These threats attempt to infect any system they can reach and propagate worldwide. They can be referred to as global threats. Meanwhile, we are also seeing an increase in attacks that are targeted at specific geographic regions or languages.

TARGETED ATTACKS

An example of a targeted attack is W32/Fanbot.A@mm, which is an IRC bot worm based on the HellBot worm. It attempts to propagate by mass mailing and exploiting the MS05-039 vulnerability. The worm also attempts to propagate through file sharing and P2P networks. It has a list of words, such as 'kazaa', 'share' and 'download', and searches for folder names that contain any of the words in the list. If a match is found, the worm copies itself to the folder. The list includes several Chinese words, which suggests that the author of the worm is aiming to infiltrate Chinese language operating systems.

Recently, users of online games have been targeted by malware. The attacker attempts to steal account information by using trojans and can make money by selling items or virtual money through Real Money Trade (RMT) sites. Particular games tend to be popular in particular regions, hence malware targets these regions as well. Trojan/Okarag and PWSteal/Wayi are examples of trojans that attempt to steal information from certain games that are popular in Asia, and close the windows of several security products, including Chinese security products. Some adware and spyware programs also target specific regions or languages.

W32/ANTINNY

W32/Antinny is a worm that targets the *Winny* P2P file-sharing network. It is the most notorious worm in Japan. *Winny* was developed by an anonymous author called Mr 47, and is available only for Japanese versions of *Windows*. In much the same way as other P2P programs, a lot of people use *Winny* for file sharing.

When I first saw the W32/Antinny worm in 2003, I was interested in two points: the fact that it was the first worm for the Japanese P2P environment and the fact that it utilizes Japanese words for file names. Obviously this means that the worm cannot propagate globally and resides only on Japanese *Windows* with *Winny* – so, initially, I thought the worm wouldn't be a significant problem. However, I was wrong.

A few variants later, W32/Antinny had a payload that would cause significant social problems: information leaking. The worm searches many files, such as *Office* documents, text, email boxes, photos and movie files on the compromised computer. It archives those files in a zip file with a gripping name in Japanese and then copies the file to the upload folder, which allows other *Winny* users to search for and download it from the *Winny* P2P network. Therefore, *Winny* users were suddenly finding lots of private information in the download files instead of what they were expecting.

Date	Source	Information
December 9, 2005	Air Carrier	The password for restricted areas in an airport
December 9, 2005	University Teaching Hospital	Information about three patients, including patient name and history for both the patients and their families
December 8, 2005	Power Company	Nuclear plant meeting minutes and documents
November 6, 2005	Public Hospital	Information about a dozen patients
November 14, 2005	Prefectural Office	Personal details of 354 staff members, including names, addresses, and bank accounts
November 17, 2005	Police Department	Home addresses of 33 police officers
September 16, 2005	Power Company	Technical documents of a thermal power plant and customer information
August 30, 2005	Heavy Industry Company	Maps, photos and inspection report of a power plant
July 21, 2005	Public Agency	Inspection report of a nuclear power plant
June 27, 2005	Police Department	Documents of criminal investigations, including personal information about suspects and eyewitnesses
June 23, 2005	Electric company	Inspection reports and photos of 27 power plants, including seven nuclear power plants

Table 1: Examples of information leaked by W32/Antinny.

Table 1 contains some examples of the information leaked by W32/Antinny as reported in the media last year. In some cases, highly sensitive information was leaked by the worm.

According to *Virus Bulletin*'s malware prevalence table [1], instances of W32/Antinny have consistently been very low. For example, the number of instances of W32/Antinny in October 2005 was just 11, representing less than 0.01% of all virus reports in that month. However, in November 2005, *Microsoft* reported that in one month it had removed more than 200,000 W32/Antinny files from 110,000 computers using the Malware Removal Tool in Windows Update [2]. The company also stated that a few hundred thousand machines were still infected with the worm. This must be the first verified case of a regional threat having propagated on such a large scale.

TWO ISSUES WITH REGIONAL THREATS

Acquisition and analysis of samples can be problematic where regional threats are concerned. If an AV company does not have a local office or support, it may be difficult to acquire enough samples of the threat. Most AV and security companies provide customers with a variety of ways to submit samples, such as by email, or via the Internet. However, most of these methods are presented in English or another specific language. If the user is not familiar with the language, they may hesitate to submit the sample through those means.

Global threats, such as W32/Sober and W32/Beagle, are reported by many people in many different countries. But regional threats are reported by a limited number of people in limited places. Such samples may easily be overlooked or assigned a low priority.

The next issue is analysis. As mentioned earlier, many regional threats contain languages other than English. An unfamiliar language in a threat can cause delay in analysis or even insufficient analysis. For example, PWSteal/Bancos.AA checks the *Internet Explorer* window text and starts logging key strokes if it matches with certain strings. Most of the strings are in English, but there are a few strings in Russian as well. These could easily be missed or ignored.

Technology can also be an issue in cases where we find files that are developed with particular tools. Trojan/Kakkeys was originally written in Ruby script language and converted into a *Windows* PE executable file by Ruby-Exerb [3]. Hot Soup Processor (HSP) [4] is another example. It is a kind of basic language and is also able to create *Windows* PE executable files. We found some tiny trojans written in HSP. Both Ruby-Exerb and HSP were developed and used mostly in Japan. The PE file that is created with those tools contains huge runtime code so that the file seems clean at first glance. These files are likely to be false positives or false negatives.

Along with understanding foreign languages and technologies, analysts also need some knowledge of regional software usage. *Winny* is available only for Japanese versions of *Windows* and is the most popular P2P program in Japan. *QQMessenger* [5] is the most popular IM program in China.

Even without understanding the specific language or the software that the threat targets, analysts can analyse the threat and recognize what it is or what it does. But it is likely they will provide insufficient analysis, and give a lower priority to the analysis of the threat. It is reported that more than 4 billion accounts of *QQMessenger* exist and 20 million users are online at any one time [6]. If a new worm that targets only *QQMessenger* is released, it will be a big problem – but only in specific regions.

CONCLUSION

Fortunately, only a few regional threats have become significant problems recently. However, anyone can obtain malicious code or ideas from the Internet and use them for profit-gain, and the number and variety of threats – including regional threats – will continue to rise. To gain profit, authors of malware don't need to create a threat that attacks computers worldwide. There is a lot of software that supports local languages only, and in certain regions (particularly China and Japan), such programs and services are more popular than ones that support only English.

AV and security companies need to be careful to acquire and analyse samples. If it's difficult to acquire samples from some regions, cooperation with other groups, companies or local organizations may be necessary in order to support customers in those regions. Even if a file looks clean, it's possible that the file spreads rapidly but only on specific language operating systems, hardware, software or services we have never heard of.

REFERENCES

[1] http://www.virusbtn.com/resources/malwareDirectory/prevalence/.

[2] http://www.microsoft.com/japan/presspass/detail.aspx?newsid=2504.

[3] http://exerb.sourceforge.jp/index.en.html.

[4] http://www.onionsoft.net/hsp/.

[5] http://www.tencent.com/.

[6] http://comm.ccidnet.com/art/1522/20051026/358525_1.html.

TECHNICAL FEATURE 1

MALWARE PENETRATION INDEX: A NEW VIRUS METRIC

Oren Drori and Nicky Pappo
Commtouch Software, Israel

This article presents a new model for calculating malware penetration probability during an outbreak. We suggest that the Malware Penetration Index (MPI) model should be adopted and developed further to become a standard calculation for the probability of malware penetration.

In this article we describe the motivation behind this initiative, elaborate on the theoretical model for the MPI and the methods for MPI estimation and calculation. Finally, we outline some possible real-life applications of the MPI model.

INTRODUCTION

Today, the majority of malware is weighted and described differently by different anti-virus vendors, IT forums and analysts; even the media gives different definitions or threat levels for the same attack. In most cases, the severity of an attack is determined by its incurred damage. As a result, the most prevalent/widespread malware gains public exposure (often resulting in a global panic), while less widespread malware may barely be acknowledged.

Unfortunately this gives a value to only one aspect of the attack. Given that an increasing number of high-risk threats are being propagated as collections of multiple low-risk malware, it is essential that a complementary metric is defined to help measure the potential risks faced by organizations.

The MPI was developed to empower the community of IT professionals worldwide. It aims to provide a reliable, unified and vendor-independent forecasting procedure that can determine how well an organization is protected against new malware.

MOTIVATION AND REQUIREMENTS

The MPI is a metric of a virus outbreak. Its aim is to indicate the chances of a virus-carrying email penetrating a user's mailbox.

The MPI paradigm has several desirable attributes:

Methodological and mathematically well-defined: its methodology makes it easy for users to understand what can and cannot be determined from this measure. The model's transparency invites open debate and scientific criticism.

Quantified: the MPI provides a comparison scale for virus outbreaks – for example, virus A has a larger or smaller MPI value than virus B.

Vendor-independent: in theory, any two people should be able to calculate the MPI for a given virus and reach the same answer (excluding errors in measurement). The MPI model is objective, and not based on the decisions of individual vendors or service providers.

DEFINING MPI

MPI is the probability that a random virus-carrying message received during an attack will penetrate an organization's anti-virus defence and arrive in an end user's mailbox.

DRIVING FACTORS

The MPI value is affected both by the distribution of the virus and by the level of defences that have been deployed against it (at least in the target population).

	Driving factor	Impact on MPI value
Attack parameters	Intensity	Increases
	Shape of distribution	Right tail distribution curve (most volume distributed early): increases
		Left tail distribution (most volume distributed late): decreases
	Attack speed	Increases
AV parameters	Responsiveness of AVs	Increases
	Number of effective AVs	Increases
	Identity of effective AVs	Large market share: increases
		Smaller vendors: decreases

Table 1: MPI driving factors.

THEORETICAL MODEL

The following semantics are used in the MPI model:

- ***I(t)*** Intensity level = the virus proportion (%) of the total traffic, at time *t*.
- ***M(t)*** Miss rate = the proportion of the population that is unprotected at time *t*.
- ***P(t)*** Penetration rate = at time *t*, the probability of a (random) received message being a successfully penetrating virus i.e. satisfies two conditions: the

message carries a virus, and the user is unprotected (at time *t*).

MPI (virus attack) Malware Penetration Index: $MPI := \frac{1}{T}\int_t P(t)$

T Attack length i.e. $t \in \{0,1,\dots T\}$.

N Number of available anti-virus engines i.e. $AV_n \in \{AV_1, AV_2,\dots AV_N\}$.

Miss rate at time *t*

The miss rate function, *M(t)*, is the percentage of users that are *unprotected*.

When *M(t)* is plotted against time, we typically see a descending step shape (see Figure 1). This is because every time a new anti-virus signature is released, an entire population of users change their status from unprotected to *protected*.

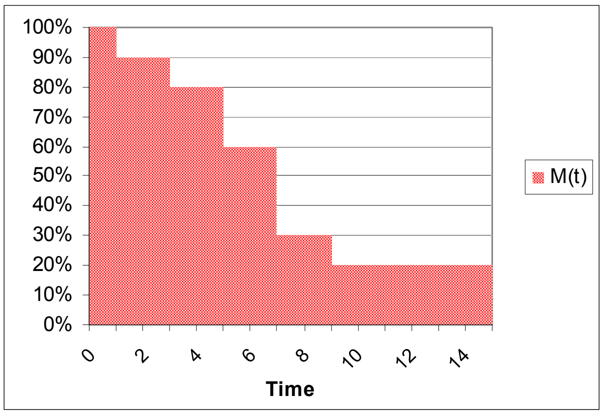


Figure 1: Miss rate, M(t), plotted against time.

The requirements for calculating *M(t)* include determining which anti-virus products provide protection against the virus at time *t*, and determining the relative market share(s) of the various anti-virus product(s).

Additional definitions:

$AV_n(t) \in \{1,0\}$ defines whether anti-virus engine *n* provides protection against the virus. The answer is time-dependent, and it can be assigned a value of zero (0) if the answer is *no*, or a value of one (1), if the answer is *yes*.

S_n is the market share of anti-virus engine *n*. This means the proportion of users across the world that use this specific anti-virus product.

Aggregating the miss rate across anti-virus engines yields the following calculation:

$$M(t) = 1 - \sum_{1}^{N} S_n AV_n(t)$$

Penetration rate and MPI

Penetration rate at time *t* is the product of the miss rate and the intensity (the prevalence of the virus, in percentage points):

$$P(t) = I(t)\cdot M(t) = I(t)\left(1 - \sum_{1}^{N} S_n AV_n(t)\right)$$

An example of a typical penetration rate graph is shown in Figure 2. It is calculated by multiplying the miss rate *M(t)* and the intensity *I(t)*.

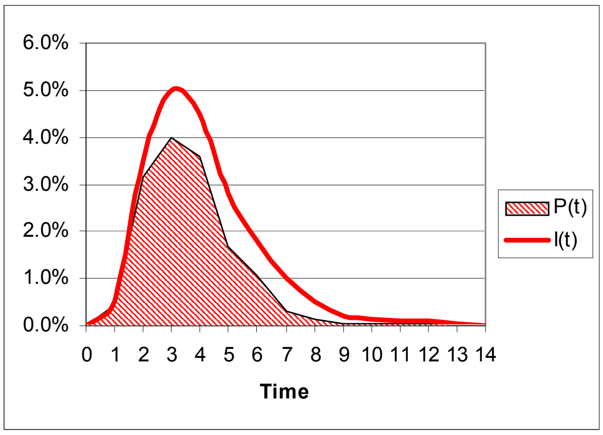


Figure 2: Intensity and penetration potential.

The MPI is the probability that a single (random) message that was sent at any time during an attack will become a *penetrating message*. In order to calculate this probability, it is necessary to calculate the proportion of the messages that were sent during that attack that answer the penetration criteria:

$$MPI = \frac{1}{T}\int_t P(t) = \int_0^T I(t)\cdot M(t)$$

Note: Division by *T* is required in order to normalize the index to (0,1) range (between 0 and 100%).

FROM THEORETICAL MODEL TO PRACTICAL ESTIMATION

In order to estimate the MPI for a message, we need to do the following:

- Approximate the $AV_n(t)$ – mapping which anti-virus engine was added and when.
- Approximate the S_n – market share of the various anti-virus engines.
- Calculate the approximated value for *M(t)* – miss rate at time *t*.

- Estimate $I(t)$ – distribution intensity through the attack.
- Estimate the penetration rate (at time t) and MPI.

Approximating $AV_n(t)$

In order to approximate $AV_n(t)$ we require a lab that has access to the various anti-virus products that are currently available, and which is updated throughout the attack. An independent lab such as *AV-test.org* could be used for this task. Such a lab might not cover every anti-virus engine on the market, but would provide a good approximation.

We also require a methodology for testing the different anti-virus engines at time t and repeatedly until the attack is over. *Commtouch* has built an automated mechanism that performs this action and then repeats it, once it is triggered.

More importantly, the above test must be performed from the beginning of the attack. A good zero-hour detection mechanism (signature- and update-independent) must be used for the trigger.

Approximating S_n

Approximating the market share of the anti-virus engines can be difficult for a number of reasons. First, the definition of market share is not trivial because analysts use different definitions of the market. In the MPI context the definition should be conclusive and include software solutions as well as hardware solutions, desktop as well as gateway products, consumer as well as business solutions.

Second, some users do not buy a pure anti-virus product, but rather a security package that includes virus protection. Anti-virus market researchers usually disregard these users.

Finally, market share is usually judged by revenue, while the MPI measurement requires market share to be determined by number of units.

In the example we have used, the approximate market share value (S_n) has been calculated by using a revenue breakdown as published by *IDC*.

Notes and reservations:

- This approximation will serve reasonably on a global level (MPI as a proportion of infected and penetrating emails out of world traffic), but should be modified if one wishes to calculate the MPI for a particular market segment, such as business users or consumers etc.
- Inclusion of integrated products in *IDC*'s report is very partial, which creates a certain bias.

Calculating the approximated value for $M(t)$

Using the approximated $AV_n(t)$ and S_n, you should have the required information to calculate the $M(t)$ variable. To do this, consider a four-hour attack, approximated market share data and protection indications across the attack timeline (provided by a lab e.g. *AV-test.org*). Table 2 illustrates the calculation of miss rate at various times – $M(t)$.

	S=share	AV(11:00)	SxAV(11:00)	12:00	SxAV(12:00)	13:00	SxAV(13:00)	14:00	SxAV(14:00)
Symantec	44.0%	0	0.0%	0	0.0%	0	0.0%	1	44.0%
McAfee	18.1%	0	0.0%	0	0.0%	1	18.1%	1	18.1%
Trend Micro	14.2%	0	0.0%	1	14.2%	1	14.2%	1	14.2%
Sophos	3.2%	0	0.0%	0	0.0%	0	0.0%	1	3.2%
Panda	2.8%	1	2.8%	1	2.8%	1	2.8%	1	2.8%
AV-6	2.5%	0	0.0%	0	0.0%	0	0.0%	1	2.5%
AV-7	1.5%	0	0.0%	1	1.5%	1	1.5%	1	1.5%
AV-8	1.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-9	1.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-10	1.0%	1	1.0%	1	1.0%	1	1.0%	1	1.0%
AV-11	0.8%	1	0.8%	1	0.8%	1	0.8%	1	0.8%
AV-12	0.8%	1	0.8%	1	0.8%	1	0.8%	1	0.8%
AV-13	0.7%	1	0.7%	1	0.7%	1	0.7%	1	0.7%
AV-14	0.7%	1	0.7%	1	0.7%	1	0.7%	1	0.7%
AV-15	0.7%	0	0.0%	1	0.7%	1	0.7%	1	0.7%
AV-16	0.5%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-17	0.5%	0	0.0%	0	0.0%	0	0.0%	1	0.5%
AV-18	0.5%	0	0.0%	1	0.5%	1	0.5%	1	0.5%
AV-19	0.5%	0	0.0%	0	0.0%	0	0.0%	1	0.5%
AV-20	0.5%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-21	0.5%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-22	0.2%	0	0.0%	0	0.0%	0	0.0%	1	0.2%
AV-23	0.2%	1	0.2%	1	0.2%	1	0.2%	1	0.2%
AV-24	0.2%	1	0.2%	1	0.2%	1	0.2%	1	0.2%
AV-25	0.2%	1	0.2%	1	0.2%	1	0.2%	1	0.2%
AV-26	0.2%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-27	0.1%	1	0.1%	1	0.1%	1	0.1%	1	0.1%
AV-28	0.1%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
AV-29	0.1%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
Others	2.7%	0.34	0.9%	0.48	1.3%	0.52	1.4%	0.72	2.0%
Aggregated Catch Rate			**8.4%**		**25.7%**		**43.9%**		**95.4%**
Aggregated Miss Rate			**91.6%**		**74.3%**		**56.1%**		**4.6%**

Table 2: Example approximation of market share, Sn, and miss rate, M(t).

Notes:

1. Market shares of the top five vendors are based on 'Worldwide Antivirus 2005–2009 Forecast and Analysis', IDC 2006.

2. Market shares of other vendors are approximated, using cross-match between a few sources – hence, the vendors are not specified by name.

3. 'Others' represent AV products for which we had no detection data: The total market share of these products is under 3%. Ignoring these products is equivalent to assigning them with 'zero' value. In order to avoid such bias, we are assuming their AV value is the average value in the industry at that time.

Estimating $I(t)$

Intensity must be estimated based on a sample group. If the sample group is large enough and unbiased, the estimated $I(t)$ will be an unbiased estimator.

The larger the sample group, the more accurate the estimation will be, and the more widespread the sample is (geographically, by email market segments etc.), the smaller the chances for bias will be.

$\hat{I}(t)$, the estimated intensity, is calculated as:

$$\hat{I}(t) = \frac{In(t)}{E(t)}$$

where $In(t)$ is the number of infected messages in the sample (between time t and t–1); $E(t)$ is the size of the email sample between time t and t–1 and E is the size of the total email sample, through the attack, that is:

$$E := \sum_{1}^{T} E(t)$$

Estimating penetration rate and MPI

$\hat{P}(t)$, the estimated penetration rate at time t, is calculated as follows:

$$\hat{P}(t) = \hat{I}(t) \cdot \hat{M}(t) = \frac{In(t)}{E(t)}\left(1 - \sum_{1}^{N} \hat{S}_n A \hat{V}_n(t)\right)$$

$$M\hat{P}I = \frac{1}{T}\sum_{t=1}^{T} \hat{P}(t) = \frac{1}{T}\sum_{t=1}^{T}\left(\hat{I}(t) \cdot \hat{M}(t)\right)$$

Alternatively, we can use the following simplified calculation, and more importantly, the process of data collection:

$$M\hat{P}I = \frac{Total\_Penetration\_Sample}{Total\_Email\_Sample} = \frac{1}{E}\sum_{t=1}^{T}\left(\hat{In}(t) \cdot \hat{M}(t)\right)$$

The two calculations are identical provided that email traffic is distributed evenly through the attack, i.e. $E(t) = E(t')$ for any t, t'. This assumption is more plausible when short attacks are considered, than for longer, slower attacks. Table 3 shows an example for estimation.

Time	E(t) - sample size	In(t) - infected sample	M(t) - Miss Rate	Penetration(t) = In(t)xM(t)
1	850,234	4,662	91.2%	4251.1
2	820,335	12,469	74.3%	9267.0
3	866,001	10,409	56.0%	5826.9
4	794,003	3,777	4.5%	171.1
E (total sample):	3,330,573		Total Penetration:	19,516
			(Total Penetration) / (Total Sample):	0.59%

Table 3: Example estimation of MPI.

$$\text{As: } M\hat{P}I = \frac{Total\_Penetration\_Sample}{Total\_Email\_Sample} = \frac{4{,}251.1 + 9{,}267 + 5{,}826.9 + 171.1}{3{,}330{,}573} = 0.59\%$$

APPLICATIONS

The above information is sufficient to draw the following conclusions:

- MPI = (by definition) the proportion of emails that were both infected and reached unprotected user. For example: '1 of every XX emails through this attack has infected a user'.
- MPI = the probability of a (randomly selected) individual message, sent sometime in the attack, being a successfully penetrating virus.
- For attacks A and B that lasted for similar periods of time: if virus attack A has a higher MPI value than virus attack B, the chances of getting hit by virus A are greater than virus B.
- Alternatively, it is possible to multiply the attack's MPI by the attack length (hours) and compare between any two attacks.

Probability of a user receiving a virus

The MPI can be used for estimating the probability of a random user receiving the virus, however several steps and assumptions are required:

- The probability must take into account the attack duration, and the number of emails received through that period of time.
- Per received message (through the attack timeframe) the probability of getting hit is the MPI and the probability of not getting hit is 1 – MPI. [*This is an approximation, ignoring the time of the specific message. If we consider the receiving time, t, of each message, the probabilities are P(t) and 1 – P(t).*]
- If we assume independence between the different emails received by the same user, and the approximation above (i.e. using *MPI* as hit probability for each message), the probability of not getting hit by any message = the probability of not getting hit by the first, *and* not getting hit by the second, etc. That is, $\Pr(safe) = (1 - MPI)^X$, where X is the number of messages received throughout the attack. The probability of getting hit through the attack, would then be: $\Pr(hit) = 1 - (1 - MPI)^X$.

For example, consider the following situation: a corporate user receives two emails per hour on a week day (about 50 a day). The attack takes place on a Tuesday and lasts 10 hours. The calculated MPI was 0.5%

Based on the above parameters, the total number of emails received was 20, and the probability of getting hit would be:

$$\Pr(hit) = 1 - (1 - 0.005)^{20} = 1 - 0.905 \approx 9.5\%$$

Sensitivity analysis

The hit probability responds dramatically to the first few emails received, while it is much less sensitive to whether 100 or 1000 messages have passed through:

MPI	0.5%			
Total emails	1	10	100	400
Hit probability	0.50%	4.89%	39.42%	86.53%

Table 4: Hit probability sensitivity to email volume.

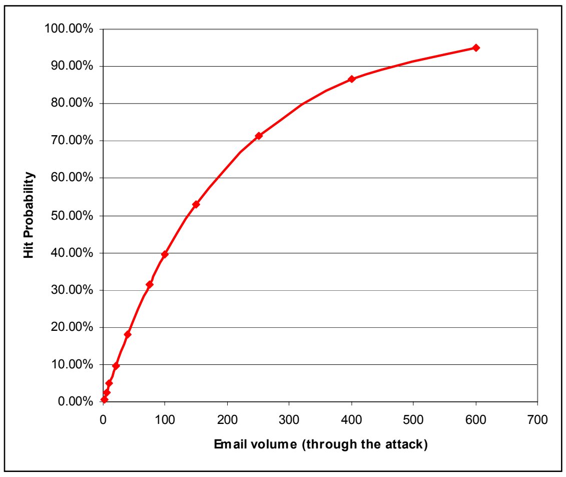


Figure 3: Sensitivity to email volume.

However, even small changes in the MPI have a dramatic effect on the hit probability. In the example shown in Table 5, a rise in the MPI from 0.1% to 0.5% increases the hit probability from less than one in ten users to almost 40%. If the MPI reaches the 3% level, the chances of getting hit are a near certainty:

Total emails	100			
MPI	0.1%	0.50%	1.00%	3.00%
Hit probability	9.52%	39.42%	63.40%	95.24%

Table 5: Hit probability sensitivity to MPI.

CONCLUSION

The MPI, a new virus metric, is a vendor-independent and quantified measure, focused on the penetration probability measure of a virus (or other malware) rather than on commonly examined aspects such as severity of damage. We have seen that, while there are challenges, it is possible to estimate the MPI in real life, providing one is willing to tolerate certain approximations.

The relevant application of the MPI for the IT community is the derivation of the hit probability, per user (based on the user's email usage pattern).

We welcome further discussion – please send us your feedback and related ideas to: orend@commtouch.com.

TECHNICAL FEATURE 2

A DIFFERENT LOOK AT THE ROOTKIT INSTALLATION PROCESS

Aleksander Czarnowski
AVET Information and Network Security, Poland

There is one method of threat modelling that is based on the actual vulnerability exploitation process. This approach has a crucial advantage over other methods: we don't have to estimate such values as discoverability, exploitability, impact or damage potential. Instead, we can use real data. The obvious drawback is that this method cannot be deployed rapidly and might require a lot of work. Furthermore, it is not feasible to use this approach in many cases due to time constraints and limited resources or knowledge. In this article, however, we will look at one of the largely undocumented aspects of this approach: rootkit installation.

THE PROCESS

What we were talking about is commonly called penetration testing (or pen testing). The aim of the process is to look at system security from an attacker's perspective and try to attack it. There are dozens of definitions of the pen test process. In our case we will divide the process into the following six stages:

1. System assets identification
2. Vulnerability identification
3. Vulnerability exploitation
4. Gaining further privileges within the system
5. Clean up
6. Report

In this article we will concentrate on stage 4 – looking specifically into using rootkits for profiling system security at local level.

We assume a situation where we have been able to gain access to the operating system and now our objective is to retain control over it. Kernel-level rootkits are the natural choice here. In fact, by subverting the system kernel we can profile and evaluate the system security level more effectively. This is important in today's world when operating system vendors and architects are trying to introduce anti-rootkit safeguards at the kernel level. One such example is the write protection of critical system structures in *Windows XP* (including x64) and *2003* [1]. In fact, while some concepts discussed here apply to other systems, the code examples and APIs used are for *Windows*.

KERNEL DRIVER INSTALLATION

To load a kernel driver we need to call the proper functions. In the case of *Windows*, we use the Service Control Manager (SCM) API. First we need to acquire a handle to the SC Manager with OpenSCManager – within any decent *Windows* assembler we can use the INVOKE macro to call the Win32 API function. In the case of FASM [7] it looks like this:

```
SC_MANAGER_ALL_ACCESS     equ    0x0F003F
invoke OpenSCManager, NULL, NULL, SC_MANAGER_ALL_ACCESS
```

If the function succeeds, EAX contains a non-zero value which holds the handle to SCM. With the handle to SCM we can load our driver with the CreateService() function:

```
hSCM                       dd ?
hRKService                 dd ?
[…]
mov [hSCM], eax            ;save valid handle to SCM
invoke CreateService, [hSCM], _szRootkitName,
_szRootkitName,\
                           SERVICE_ALL_ACCESS,\
                           SERVICE_KERNEL_DRIVER,\
                           SERVICE_DEMAND_START,\
                           SERVICE_ERROR_NORMAL,\
                           szPath,\
                           NULL,\
                           NULL,\
                           NULL,\
                           NULL,\
                           NULL
```

We can now start the service. The EAX register contains a handle to our newly created service (this value is non-zero):

```
.if eax <> 0
        mov [hRKService], eax
        invoke StartService, [hRKService], 0, NULL
.endif
```

To clean up we should close the SCM and our service handles:

```
invoke CloseServiceHandle, [hRKService]
invoke CloseServiceHandle, [hSCM]
```

There is a twist here that we should discuss. We have assumed that:

1. The driver (service) we are installing is not already installed on the system.
2. The driver (service) we are trying to start is not already started on the target system.

If the first assumption is incorrect, the CreateService function fails (EAX = 0) and GetLastError() returns ERROR_SERVICE_EXISTS. If the second assumption is incorrect it is the StartService function that fails and GetLastError() returns ERROR_SERVICE_ALREADY_RUNNING. In either event any further actions depend on your objectives. Generally during pen testing, a consultant would be using their own driver so the chances that the same service is installed or running on the system are small (if the name of driver is unique, i.e. not in conflict with one of the system services). Still, both cases should be supported and reported to provide a full audit trail of the pen test project. It is also a good idea to save the error number from GetLastError() in the event of any failure.

EXTRACTING FILE FROM EXE

Having everything in one file – the exploit to gain high privileges, rootkit (kernel driver) code and rootkit install/control application – has several advantages. As we have seen, you need a valid filename for the CreateService() function to load the driver into the kernel address space. One method [2] is based on using resources. Basically the idea is pretty simple – use the FindResource() and LoadResource() functions to get the handle to the resource containing your driver's code. While this method is quite simple and effective, I didn't like it for a number of reasons:

- Resources are easy to edit and extract.
- Resources are easy to spot within a file.
- When analysing a binary you see calls to the Win32 resources API functions either on the Import Address Table (IAT) entry level or during dynamic analysis when the binary is being run. You could, of course, use the LoadLibrary and GetProcAddress functions to call the resources API indirectly, which would result in a lack of those functions in IAT, but this is still easy to spot by looking at reference calls to GetProcAddress or at string tables/data sections.

One of the main objectives of this article is to show a different approach. In this case we will use the behaviour of the system loader when the CreateProcess() function is called to execute our binary in PE format. Some good discussions of the *Windows* system loader and PE format can be found in [3], [4] and [5]. In our approach we rely on several facts:

- A PE file can have many sections and we can control their content and alignment within the file.
- PE file sections are loaded into memory during process creation by the operating system.
- When building an executable file we can control the entry point address.
- The operating system creates additional structures within process memory in the userland area so we can access it without any problem from our code before loading any additional code into the kernel space.

This leads us to a simple conclusion: we can put all the required files into one EXE by inserting additional sections. To simplify our discussion we will assume that we put only one file into one section. This way we can use information within the PE file to get the section size and its location within memory and simply pass those as arguments to the WriteFile() function.

ADDING SECTIONS TO A PE FILE

First, we need to figure out how to add sections to our main file. This can be accomplished via several different methods. However, in our case this is very easy because we are building the final file from our sources, so we can leave that job to the compiler and linker. In this example I am using FASM [7], so adding a section is just a matter of a simple declaration in the source file:

```
section '_rootkit' code readable executable
```

An important fact about FASM is that no external linker is needed – the assembler will also do all the required linking when the 'format PE' declaration is used at the beginning of the source file. To add the content of another file we can use the 'file' command:

```
section '_rootkit' code readable executable
frootkit    file 'simpleice.sys'
```

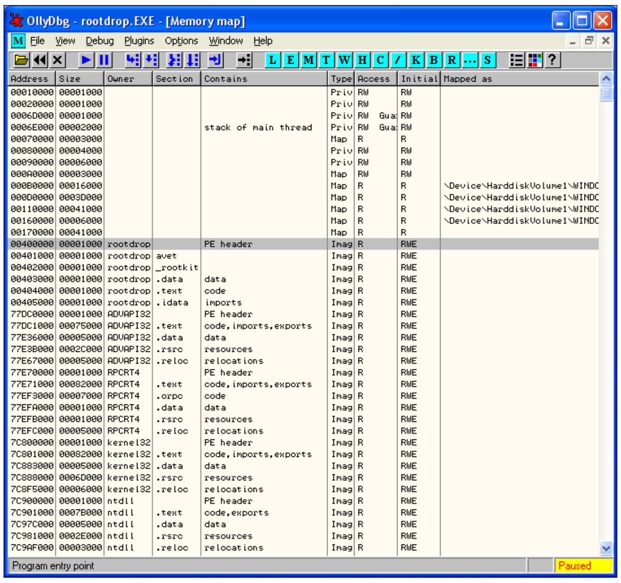
The above code will generate a PE file with the section '_rootkit' filled with the content of the simpleice.sys file. Our sys file is a compiled kernel service which we will load and start as discussed above. First, we will need to extract this file and recreate it on disk.

HOW NOT TO READ FROM DISK

One simple method of extracting content from any file would be to read it from disk and create another file. However, we have used PE sections so the code we want to extract is already in the memory of our process – there's no need to make additional reads from disk.

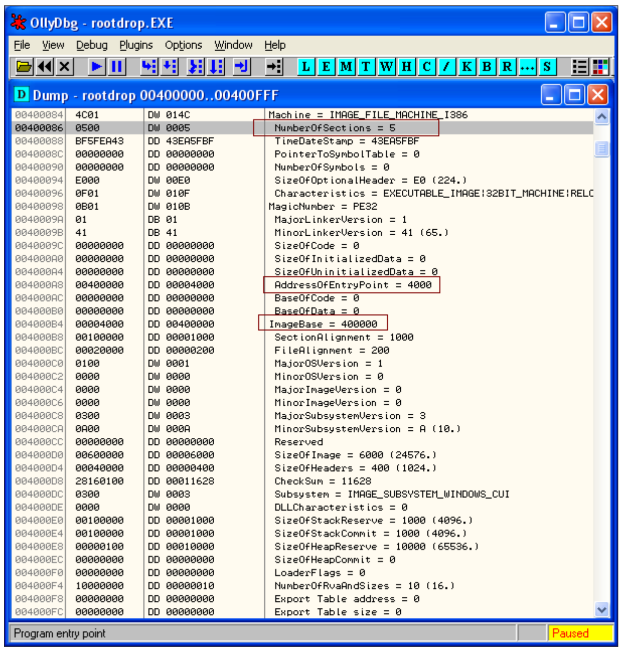
As shown in Figure 1 we can use *OllyDBG* to inspect process memory including all sections. One of the memory mapped sections is our process PE header. Again, we can use *OllyDBG* to inspect the content of this structure (see Figure 2).

As you see we have all the necessary information such as the number of sections inside the PE file, and ImageBase and Address of Entry Point values. Note that the default values for ImageBase and Address of Entry Point may be different from those in our example, however the ImageBase set to 0x400000 is the default for *Windows* operating systems. This is an important observation because we can use it in our code for finding the PE header:


OllyDbg - rootdrop.EXE - [Memory map]
File View Debug Plugins Options Window Help
L E M T W H C / K B R ... S
Program entry point
Paused

```
Address  Size     Owner    Section  Contains                Type Access  Initial Mapped as
00010000 00001000                                           Priv RW      RW
00020000 00001000                                           Priv RW      RW
0006D000 00001000                                           Priv RW  Gua RW
0006E000 00002000                   stack of main thread    Priv RW  Gua RW
00070000 00003000                                           Map  R       R
00080000 00004000                                           Priv RW      RW
00090000 00006000                                           Priv RW      RW
000A0000 00003000                                           Map  RW      RW
000B0000 00016000                                           Map  R       R       \Device\HarddiskVolume1\WINDC
000D0000 0003D000                                           Map  R       R       \Device\HarddiskVolume1\WINDC
00110000 00041000                                           Map  R       R       \Device\HarddiskVolume1\WINDC
00160000 00006000                                           Map  R       R       \Device\HarddiskVolume1\WINDC
00170000 00041000                                           Map  R       R
00400000 00001000 rootdrop          PE header               Imag R       RWE
00401000 00001000 rootdrop avet                             Imag R       RWE
00402000 00001000 rootdrop _rootkit                         Imag R       RWE
00403000 00001000 rootdrop .data    data                    Imag R       RWE
00404000 00001000 rootdrop .text    code                    Imag R       RWE
00405000 00001000 rootdrop .idata   imports                 Imag R       RWE
77DC0000 00001000 ADVAPI32          PE header               Imag R       RWE
77DC1000 00075000 ADVAPI32 .text    code,imports,exports    Imag R       RWE
77E36000 00005000 ADVAPI32 .data    data                    Imag R       RWE
77E3B000 0002C000 ADVAPI32 .rsrc    resources               Imag R       RWE
77E67000 00005000 ADVAPI32 .reloc   relocations             Imag R       RWE
77E70000 00001000 RPCRT4            PE header               Imag R       RWE
77E71000 00082000 RPCRT4   .text    code,imports,exports    Imag R       RWE
77EF3000 00007000 RPCRT4   .orpc    code                    Imag R       RWE
77EFA000 00001000 RPCRT4   .data    data                    Imag R       RWE
77EFB000 00001000 RPCRT4   .rsrc    resources               Imag R       RWE
77EFC000 00005000 RPCRT4   .reloc   relocations             Imag R       RWE
7C800000 00001000 kernel32          PE header               Imag R       RWE
7C801000 00082000 kernel32 .text    code,imports,exports    Imag R       RWE
7C883000 00005000 kernel32 .data    data                    Imag R       RWE
7C888000 0006D000 kernel32 .rsrc    resources               Imag R       RWE
7C8F5000 00006000 kernel32 .reloc   relocations             Imag R       RWE
7C900000 00001000 ntdll             PE header               Imag R       RWE
7C901000 0007B000 ntdll    .text    code,exports            Imag R       RWE
7C97C000 00005000 ntdll    .data    data                    Imag R       RWE
7C981000 0002E000 ntdll    .rsrc    resources               Imag R       RWE
7C9AF000 00003000 ntdll    .reloc   relocations             Imag R       RWE
```

Figure 1: OllyDBG: Location of PE header in process memory.


OllyDbg - rootdrop.EXE
File View Debug Plugins Options Window Help
L E M T W H C / K B R ... S
Dump - rootdrop 00400000..00400FFF
Program entry point
Paused

```
00400084 4C01     DW 014C        Machine = IMAGE_FILE_MACHINE_I386
00400086 0500     DW 0005         NumberOfSections = 5
00400088 BF5FEA43 DD 43EA5FBF     TimeDateStamp = 43EA5FBF
0040008C 00000000 DD 00000000     PointerToSymbolTable = 0
00400090 00000000 DD 00000000     NumberOfSymbols = 0
00400094 E000     DW 00E0         SizeOfOptionalHeader = E0 (224.)
00400096 0F01     DW 010F         Characteristics = EXECUTABLE_IMAGE|32BIT_MACHINE|RELC
00400098 0B01     DW 010B        MagicNumber = PE32
0040009A 01       DB 01           MajorLinkerVersion = 1
0040009B 41       DB 41           MinorLinkerVersion = 41 (65.)
0040009C 00000000 DD 00000000     SizeOfCode = 0
004000A0 00000000 DD 00000000     SizeOfInitializedData = 0
004000A4 00000000 DD 00000000     SizeOfUninitializedData = 0
004000A8 00400000 DD 00004000     AddressOfEntryPoint = 4000
004000AC 00000000 DD 00000000     BaseOfCode = 0
004000B0 00000000 DD 00000000     BaseOfData = 0
004000B4 00004000 DD 00400000    ImageBase = 400000
004000B8 00100000 DD 00001000     SectionAlignment = 1000
004000BC 00020000 DD 00000200     FileAlignment = 200
004000C0 0100     DW 0001         MajorOSVersion = 1
004000C2 0000     DW 0000         MinorOSVersion = 0
004000C4 0000     DW 0000         MajorImageVersion = 0
004000C6 0000     DW 0000         MinorImageVersion = 0
004000C8 0300     DW 0003         MajorSubsystemVersion = 3
004000CA 0A00     DW 000A         MinorSubsystemVersion = A (10.)
004000CC 00000000 DD 00000000     Reserved
004000D0 00600000 DD 00006000     SizeOfImage = 6000 (24576.)
004000D4 00040000 DD 00000400     SizeOfHeaders = 400 (1024.)
004000D8 28160100 DD 00011628     CheckSum = 11628
004000DC 0300     DW 0003         Subsystem = IMAGE_SUBSYSTEM_WINDOWS_CUI
004000DE 0000     DW 0000         DLLCharacteristics = 0
004000E0 00100000 DD 00001000     SizeOfStackReserve = 1000 (4096.)
004000E4 00100000 DD 00001000     SizeOfStackCommit = 1000 (4096.)
004000E8 00000100 DD 00010000     SizeOfHeapReserve = 10000 (65536.)
004000EC 00000000 DD 00000000     SizeOfHeapCommit = 0
004000F0 00000000 DD 00000000     LoaderFlags = 0
004000F4 10000000 DD 00000010     NumberOfRvaAndSizes = 10 (16.)
004000F8 00000000 DD 00000000     Export Table address = 0
004000FC 00000000 DD 00000000     Export Table size = 0
```

Figure 2: Snapshot of PE header from running process memory displayed by OllyDBG.

```
mov esi, 400000h
lodsw
cmp ax,'MZ'
jnz no_mz_header
add esi, 03ch - 2  ;esi = pointer to PE structure in
memory from MZ header
mov eax,[esi]
add eax,400000h
mov edi, eax
mov [e_lfanew], eax
```

```
cmp word [edi],'PE'
jnz no_pe_header
test word [edi + 2], 0
jnz no_pe_header
```

Keep in mind that the PE header contains an old MS-DOS MZ exe header too. The e_lfanew field contains a pointer to the real PE header. We now need the value of the NumberOfSection field to scan through all to identify the one we are looking for:

```
_szRootkitSectionName     db '_rootkit',0
xor ecx,ecx
mov word cx, [edi + 6]    ;edi+6 = pe->word = number
of section in PE
;dwSectionHeaderAddress = ImageDosHeader.e_lfanew +
sizeof(ImageNTHeader);
mov eax, [e_lfanew]
add eax, sizeof_ImageNTHeader
mov esi, eax
mov ebx, eax
cld
scan_section_table:
push ecx
mov edi, _szRootkitSectionName
mov ecx, 8        ;size of section name field in bytes
rep cmpsb
pop ecx
jz found_rk_section       ;we found our section
mov esi, ebx
add esi, sizeof_ImageSectionHeader
mov ebx, esi
loopd scan_section_table
```

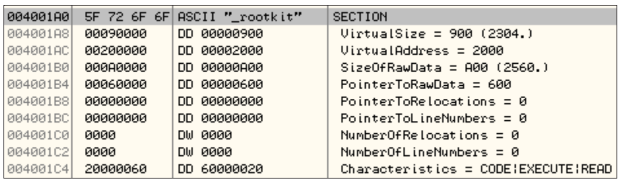
We loop through all the sections and we compare the current section name with the predefined _szRootkitSectionName. If the section name matches the _szRootkitSectionName we have found our section. The above code compares all eight bytes of the section name. The section name always occupies eight bytes of memory – if its name is shorter, the unused bytes are filled with zeros.

Every section header also contains a SizeOfRawData field (see Figure 3). We can use this field to calculate the size of the data. The VirtualAddress field tells us the address of the section within process memory. Keep in mind that this is the Virtual Address, so to get the location of the section in memory we need to add to this the value of ImageBase. Since we can get the value of ImageBase from the PE header (Figure 2), we have all the information we need at this point.

If we inserted only one file into the section, we have all the arguments we need to pass to WriteFile to create the .sys file that will be loaded with SCM.

PARSING PEB AND FS REGISTER

Two of the fundamental properties for a security tool are flexibility and stability. This is why we try to gather the information from the PE header in memory instead of hardcoding the base address and section address/size. This allows us to include any driver and any exploit in our tool. Such flexibility is important when we need to automate as much as possible in the pen test process. In the next section I will discuss further methods of making the code more flexible.



```
004001A0  5F 72 6F 6F  ASCII "_rootkit"  SECTION
004001A8  00090000     DD 00000900        VirtualSize = 900 (2304.)
004001AC  00200000     DD 00002000        VirtualAddress = 2000
004001B0  000A0000     DD 00000A00        SizeOfRawData = A00 (2560.)
004001B4  00060000     DD 00000600        PointerToRawData = 600
004001B8  00000000     DD 00000000        PointerToRelocations = 0
004001BC  00000000     DD 00000000        PointerToLineNumbers = 0
004001C0  0000         DW 0000            NumberOfRelocations = 0
004001C2  0000         DW 0000            NumberOfLineNumbers = 0
004001C4  20000060     DD 60000020        Characteristics = CODE|EXECUTE|READ
```

Figure 3: Section header in memory.

First – if we are not reading the PE file from disk – we need to get the base address. As stated previously, 0x400000 is the standard value for ImageBase. However, this could be changed to another value – possibly even accidentally (by the linker we are using, for example) – so it is wise not to assume any default values. One very old trick used in viruses and exploits is called a trampoline (not to be confused with what gcc generates on the stack to facilitate nested classes – which is also trampoline code):

```
start:
call trampoline
     real_start:
pop ebx                  ;get EIP value
[…]
     trampoline:
jmp real_start
;data section can be placed here for example
```

Running this code will result in having the base address (increased by the address of the entry point plus the address of the next instruction after the call opcode) in the EBX register. We need to remove less significant bytes to get a clean base address. Our section scanning loop will work perfectly well with this value (look at Figure 1 to see why).

There is another method that is used in *Windows* shellcodes, which is based on parsing of the PEB block. In the case of real-life Win32 shellcode we are doing real parsing of PEB, but in our case we just need to execute a few mov instructions:

```
push dword [fs:30h]
pop eax
test eax, eax
js its_not_nt
nt:
mov ecx,[eax + 0ch]      ;_PEB_LDR_DATA
mov eax,[ecx + 0ch]      ;address of first module
```

The above code works because [FS:30h] is always a pointer to the PEB structure. We can use PEB to gain knowledge about every loaded module including our PE file. (Note: the push dword [fs:30h] trick could cause false positives on some *XP Home* edition systems.)

To fully understand how this code works you can use *WinDbg* from the *Microsoft Windows Debugging Tools* package [8], which is free to download. Keep in mind that *WinDbg* is probably the most unfriendly debugger on earth – at least for *Windows* systems (unless you are trying to use GDB without the source code of the debugged target).

There are two things you need to remember when working with *WinDbg*: always use the newest version available, and always load symbols – which will make *WinDbg* worth all the work you'll need to learn the thing.

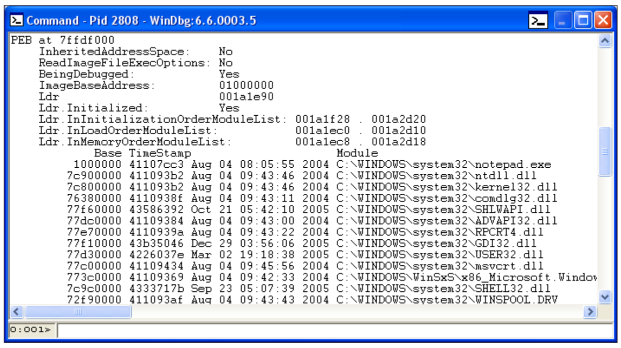
The easiest way to load symbols when you start *WinDbg* for the first time (if you are connected to the Internet) is to issue the following commands at the debugger command prompt:

```
.sympath srv*DownstreamStore*http://
msdl.microsoft.com/download/symbols

.reload
```

Keep in mind that you can load symbols in the kernel and user mode debugging session. To look at all the structures you might be interested in you need to select the 'Kernel debugging' option (Ctrl+K). With the most recent version of *WinDbg* you can perform kernel debugging using one system – something *SoftICE* [9] was capable of light years ago.

To get started, after loading the symbols, simply run *Notepad* or any other simple application and use the 'Attach to process' option (F6). This will change the debugger context to point at the right structures in memory. The first structure we are interested in is PEB – we can display its content by using the !peb command (Figure 4).

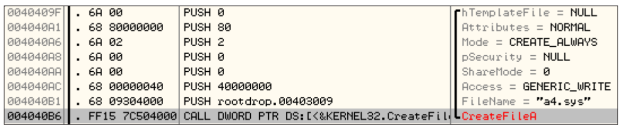


```
PEB at 7ffdf000
    InheritedAddressSpace:    No
    ReadImageFileExecOptions: No
    BeingDebugged:            Yes
    ImageBaseAddress:         01000000
    Ldr                       001a1e90
    Ldr.Initialized:          Yes
    Ldr.InInitializationOrderModuleList: 001a1f28 . 001a2d20
    Ldr.InLoadOrderModuleList:           001a1ec0 . 001a2d10
    Ldr.InMemoryOrderModuleList:         001a1ec8 . 001a2d18
            Base TimeStamp                     Module
         1000000 41107cc3 Aug 04 08:05:55 2004 C:\WINDOWS\system32\notepad.exe
        7c900000 411093b2 Aug 04 09:43:46 2004 C:\WINDOWS\system32\ntdll.dll
        7c800000 411093b2 Aug 04 09:43:46 2004 C:\WINDOWS\system32\kernel32.dll
        76380000 4110938f Aug 04 09:43:11 2004 C:\WINDOWS\system32\comdlg32.dll
        77f60000 43586392 Oct 21 05:42:10 2005 C:\WINDOWS\system32\SHLWAPI.dll
        77dc0000 41109384 Aug 04 09:43:00 2004 C:\WINDOWS\system32\ADVAPI32.dll
        77e70000 4110939a Aug 04 09:43:22 2004 C:\WINDOWS\system32\RPCRT4.dll
        77f10000 43b35046 Dec 29 03:56:06 2005 C:\WINDOWS\system32\GDI32.dll
        77d30000 4226037e Mar 02 19:18:38 2005 C:\WINDOWS\system32\USER32.dll
        77c00000 41109434 Aug 04 09:45:56 2004 C:\WINDOWS\system32\msvcrt.dll
        773c0000 41109369 Aug 04 09:42:33 2004 C:\WINDOWS\WinSxS\x86_Microsoft.Window
        7c9c0000 4333717b Sep 23 05:07:39 2005 C:\WINDOWS\system32\SHELL32.dll
        72f90000 411093af Aug 04 09:43:43 2004 C:\WINDOWS\system32\WINSPOOL.DRV
0:001>
```

Figure 4: Using WinDBG to view the PEB structure of the running process.

WinDBG has one important function for inspecting system structures, accessible both from kernel space and userland. This function is called 'display type' and it is accessible by using the 'dt' command. To display the PEB structure as it is seen by the operating system type 'dt nt!_PEB'.

Another important structure from the process perspective is EPROCESS. Again, the dt command will help – type 'dt nt!_EPROCESS' in the command prompt. (Note: the *WinDbg* !processfields command is not available in *Windows XP* and later versions. Instead, use dt command.)



```
0040409F  . 6A 00          PUSH 0                                  rhTemplateFile = NULL
004040A1  . 68 80000000    PUSH 80                                 |Attributes = NORMAL
004040A6  . 6A 02          PUSH 2                                  |Mode = CREATE_ALWAYS
004040A8  . 6A 00          PUSH 0                                  |pSecurity = NULL
004040AA  . 6A 00          PUSH 0                                  |ShareMode = 0
004040AC  . 68 00000040    PUSH 40000000                           |Access = GENERIC_WRITE
004040B1  . 68 09304000    PUSH rootdrop.00403009                  |FileName = "a4.sys"
004040B6  . FF15 7C504000  CALL DWORD PTR DS:[<&KERNEL32.CreateFil LCreateFileA
```

Figure 5: Inner workings of INVOKE macro.

A close inspection of this structure shows that the pointer to the PEB structure can also be found here. As you have already seen PEB is visible from userland processes, however kernel level structures also keep track of it. In fact, the EPROCESS structure is being used by rootkits to hide processes loaded into memory.

Coming back to our discussion, at least one field in the PEB structure is important for us: the ImageBase address. Other fields might be interesting as well – depending on what we want to accomplish. It is worth noting that under *Windows 2000* and *XP*, PEB is always mapped to the 0x7ffdf000 address in memory.

FINAL NOTES

FlatAssembler was chosen as the development environment for implementing the above ideas. This decision was based on the following functionality of FASM in comparison with other assemblers:

- It is an open source project.
- It provides better control over some aspects of code generation for PE.
- It supports the x64 (also called AMD64) architecture.
- It supports cross-compiling – this gives us the ability to create Win32/64 PE files on *Linux* machines for example.

One of the 'magic' assembler macros used in this article is INVOKE (or invoke to be more strict with FASM syntax). To understand how it works on a CPU level take a look at Figure 5. This is the disassembly of code generated by the assembler due to the use of the following Win32 API call from the assembly source code:

```
invoke CreateFile, _szRootkitName,\
                              GENERIC_WRITE, 0, NULL,\
                              CREATE_ALWAYS,\
                              FILE_ATTRIBUTE_NORMAL,\
                              NULL
```

In the pen test process it is wise to use drivers that provide an unload option. In terms of driver code this comes to a few additional lines (unless you are doing 'strange things' to the kernel code):

```
#include "ntddk.h"
VOID OnUnload(IN PDRIVER_OBJECT acpDriverObject)
{
      DbgPrint("driver unload");
}

NTSTATUS DriverEntry(IN PDRIVER_OBJECT
acpDriverObject, IN PUNICODE_STRING acpRegPath)
{
      acpDriverObject->DriverUnload = OnUnload;
//register unload function
      return STATUS_SUCCESS;
}
```

As you see, both pen test and threat modelling processes are becoming increasingly complicated – even seemingly simple or basic tasks require good planning. A thorough understanding of under-the-hood system structures can be very helpful here. In fact, it would not be possible to accomplish some projects without proper automation and tools support.

Some parts of rootkit technology can be used to understand system security better and to strengthen systems against future attack – not only on a configuration level, but also on kernel and compiler levels.

BIBLIOGRAPHY

[1] Kernel Patch Protection: Frequently Asked Questions http://www.microsoft.com/whdc/driver/kernel/64bitpatch_FAQ.mspx.

[2] *Rootkits: Subverting the Windows Kernel*, Greg Hoglund, Jamie Butler, Addison Wesley 2006, ISBN: 0321294319.

[3] *Inside Microsoft® Windows® 2000*, Third Edition, David A. Solomon and Mark E. Russinovich , Microsoft Press, 2000, ISBN 0-7356-1021-5.

[4] 'An In-Depth Look into the Win32 Portable Executable File Format', http://msdn.microsoft.com/msdnmag/issues/02/02/PE/default.aspx.

[5] 'What Goes On Inside Windows 2000: Solving the Mysteries of the Loader', http://msdn.microsoft.com/msdnmag/issues/02/03/Loader/.

[6] Microsoft Portable Executable and Common Object File Format Specification, http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx.

[7] FlatAssembler, http://www.flatassembler.net.

[8] Microsoft Debugging Tools, http://www.microsoft.com/ddk.

[9] SoftICE Drivers Suite, http://www.compuware.com/products/driverstudio/softice.htm.

PRODUCT REVIEW

eTRUST INTERNET SECURITY SUITE

Matt Ham

CA (formerly known as *Computer Associates*) has been one of the lesser-known giants of the security software industry for a number of years – its products in this field being overshadowed both by other enterprise offerings and by boardroom antics.

In the past, *Virus Bulletin* has tended to concentrate on the enterprise-level versions of *eTrust*, which makes it something of a change, on this occasion, to review what can be considered a consumer-friendly package. The *eTrust Internet Security Suite* (*ISS*) combines several products within one package, the applications within it each being available as standalone purchases. As such, the product manifests itself in three main ways: through a central console, the individual application GUIs and, as has come to be expected, the Windows Security Center.

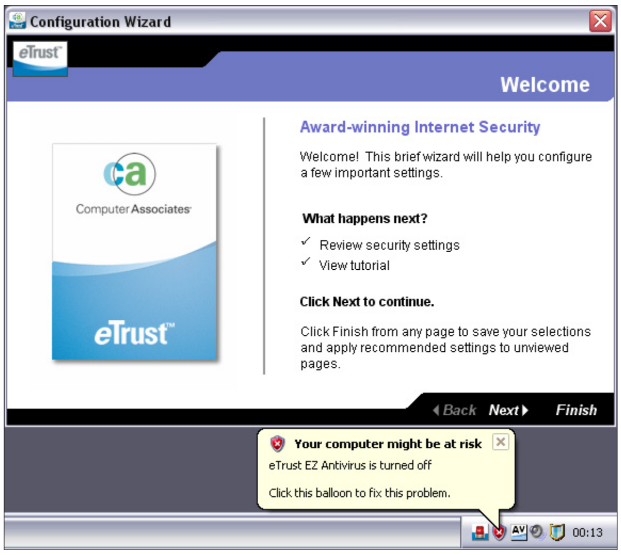
The applications contained within the package are currently *eTrust EZ Antivirus*, *eTrust Anti-Spam*, *eTrust PestPatrol Anti-Spyware* and *eTrust Personal Firewall*. This covers most of the security functionality that an average user would realise existed and, one hopes, desire to put into place.

INSTALLATION AND CONFIGURATION

The product was installed in a variety of ways, some more effective than others. In all cases, *Windows XP Professional* was used as the operating system, with *Service Pack 2* applied. Most testing was performed on a machine attached to the Internet via an ADSL connection, though several installations were performed on an isolated network.

The isolated machines were used first, with the first test using a trial version of *ISS*. The installation process was accompanied by a few warnings due to the lack of connectivity. Eventually the suite seemed to have been installed successfully, with the exception of the anti-virus functions which were disabled. Installing a full version of the anti-virus application on top failed to remedy the situation, with the software declaring simply that the anti-virus engine did not exist.

This lack of anti-virus operation is not unheard of in *CA*'s products – the *Vet* product, for one, refuses to operate if it has not been updated sufficiently recently. Since the anti-virus component within *ISS* is provided by the *Vet* engine, it is not unreasonable to suspect that the same feature is at work here. The presumed logic behind this is that users will attempt to update more frequently if their software is rendered useless otherwise. However, my


Configuration Wizard

Welcome

Award-winning Internet Security

Welcome! This brief wizard will help you configure a few important settings.

What happens next?

✓ Review security settings
✓ View tutorial

Click Next to continue.

Click Finish from any page to save your selections and apply recommended settings to unviewed pages.

Your computer might be at risk

eTrust EZ Antivirus is turned off

Click this balloon to fix this problem.

suspicion is that most people are not likely to care so much about one more red blob on their task bar. With the situation far from useful for testing purposes, all further tests were performed on the Internet-connected machine.

The relative sizes of the software packages within *ISS* were intriguing, although at least the total package size was pretty close to the sum of its parts. *ISS* is a combination of *eTrust EZ Antivirus* (5.5 MB), *eTrust Anti-Spam* (2.5 MB), *eTrust PestPatrol Anti-Spyware* (13.5 MB) and *eTrust Personal Firewall* (8.0 MB). The combined *ISS* package amounts to 27.0 MB.

Although each of the single products was installed during testing, the process of installing the overall *ISS* package is covered in detail here. *CA* has ever been a stickler for licensing, and this applies equally to *ISS*, where a licence key request is the first interaction in the installation procedure.

It is noteworthy that a licence key is required not only for full products but also when installing demonstration versions. In addition, the demonstration versions are activated on the principle of a user supplying credit card details and then opting out later if they do not want to purchase the software. This is one of the more irritating ways of supplying demonstration software. On principle I would usually steer well clear of such demonstration versions.

Assuming a correct licence key is supplied, the next step is to specify an installation location. Following this comes the decision as to which of the four main components to install. For the purposes of testing all four were selected, although the test machine did not have any configured email clients present.

An installation progress box appears after this selection has been made, which lies dormant for a disturbingly long period before springing into action. At this point the user is bombarded by a rapid succession of additional progress bars erupting on the screen, as the components download updates. The *PestPatrol* updates seemed to take longest at this stage – perhaps not surprising given the relative size of the applications.

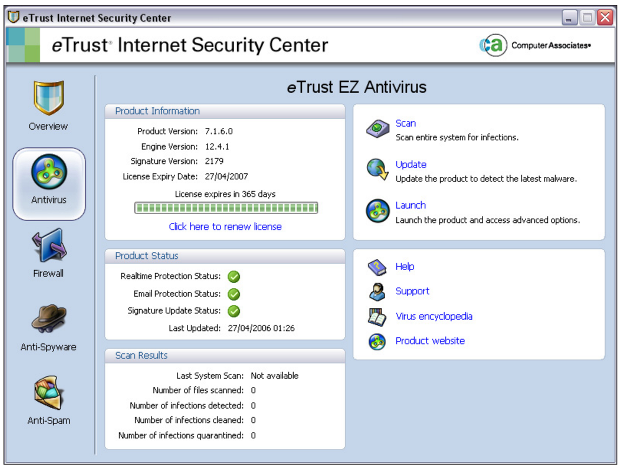
A further page of registration details also appears, which is dominated by a list of other products for the user potentially to be lured into buying. Since a reboot dialog eventually appears on top of this page, and the page is obscured by progress bars to a certain degree, this form of marketing is perhaps more subliminal than its designers would have hoped.

Any user completing the registration will, by default, be subjected to four different streams of email information – a service which I would have preferred to actively opt *into*, rather than out of. Rather than subject this registration page to even more abuse, however, rebooting to complete installation seemed a wise option.

The level of interactivity during the installation procedure so far is not exactly great. This seeming oversight is dealt with by the Configuration Wizard which now appears. During this configuration process the machine is isolated from any network connections, thus protecting the user during a potentially vulnerable period. As noted in the splash screen, the configuration process may be aborted at any time, resulting in the default security settings being applied.

First to be configured is the Program AlertAdvisor, which is, in fact, the list of applications allowed to operate through the firewall as trusted entities.

An online whitelist is used here, which can be applied automatically, used as a recommendation for user decisions


eTrust Internet Security Center

eTrust EZ Antivirus

Product Information

Product Version: 7.1.6.0
Engine Version: 12.4.1
Signature Version: 2179
License Expiry Date: 27/04/2007
License expires in 365 days
Click here to renew license

Product Status

Realtime Protection Status: ✓
Email Protection Status: ✓
Signature Update Status: ✓
Last Updated: 27/04/2006 01:26

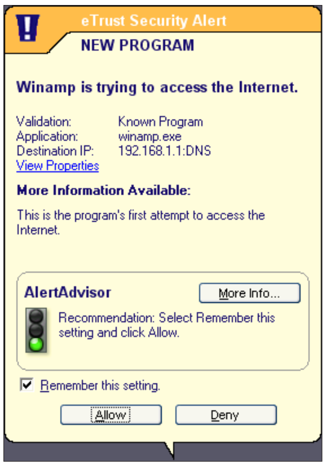
Scan Results

Last System Scan: Not available
Number of files scanned: 0
Number of infections detected: 0
Number of infections cleaned: 0
Number of infections quarantined: 0

Scan
Scan entire system for infections.

Update
Update the product to detect the latest malware.

Launch
Launch the product and access advanced options.



Winamp is trying to access the Internet.

Validation: Known Program
Application: winamp.exe
Destination IP: 192.168.1.1:DNS
View Properties

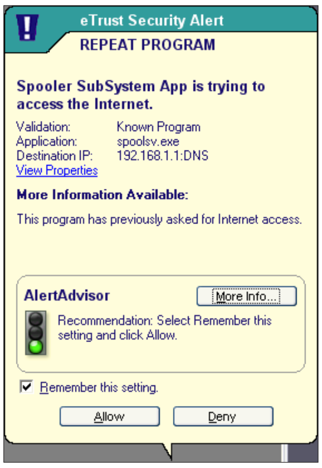
More Information Available:

This is the program's first attempt to access the Internet.

AlertAdvisor

Recommendation: Select Remember this setting and click Allow.

☑ Remember this setting.



Spooler SubSystem App is trying to access the Internet.

Validation: Known Program
Application: spoolsv.exe
Destination IP: 192.168.1.1:DNS
View Properties

More Information Available:

This program has previously asked for Internet access.

AlertAdvisor

Recommendation: Select Remember this setting and click Allow.

☑ Remember this setting.

or simply not consulted, depending on the user's preferences. Although the automatic setting is the default here, I chose the user consultation setting in order to see what information this whitelist provided on various day-to-day applications. The setting proved to be rather irritating, in that permission was requested for every application I accessed – even the ones that I had whitelisted. I very quickly reverted to the less sanity-testing default setting.

The next two configuration settings are slightly less obvious, involving whether pop-ups, banners and cookies will be blocked and whether the Cache Cleaner function will be activated. I ignored both these settings, which default to being turned off in any case. With *SP2* installed and *FireFox* as the primary web browser, popups have not been a problem on the test machine for some time. Likewise, disk space is not really a consideration and I have a natural distrust of applications that delete anything automatically – however much effort they may claim to save me.

This ends the firewall configuration procedure, after which a 'Tutorial' is launched through a series of ten dialog boxes. This is not entirely convincing, seeming just slightly too technical to be comfortable for a truly naïve user, yet not offering much more advice than to follow the firewall's blocking recommendations when in doubt.

Next, the firewall dialog returns to configuration – offering all detected networks and asking whether these should be in the Internet or Trusted zone. I have my suspicions that the bulk of home users with a router will be confused as to whether this is a trusted or untrusted connection – thinking that the router must surely be trusted, as it sits just next to the computer in their living room. While the firewall has this process of post-install configuration, the other components are all set to defaults for later tweaking, so at this stage installation can be considered complete.

As a general rule, it seemed that installing *ISS* over other security products was not perceived to be a problem. Thus it was quite possible to run, for example, *Windows OneCare*, *Microsoft AntiSpyware* and *eTrust Antivirus* simultaneously with no warning messages appearing during the installation of the *eTrust* product.

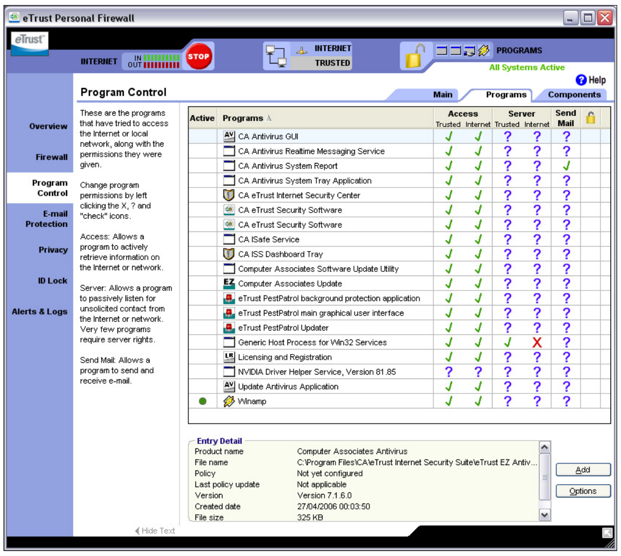
On the other hand, installing different versions of *CA* products does prompt warnings. This is especially noticeable since the downloaded versions have older definitions than a version installed on an Internet-connected machine that receives automatic updates. The result is that the user is bombarded with numerous, pretty much unavoidable, warnings that the application to be installed is older than the one currently running. It is reasonable to expect that some users will buy *eTrust Antivirus* and upgrade later to *ISS*, triggering these warnings in the process.

The warnings advise uninstallation before the addition of new versions, so this feature was also tested. Uninstallation was not a particularly arduous task, though it did require a reboot. To its credit, the uninstall facility removed all immediately obvious extraneous files, doing a better job than most such applications.

On the other hand, reinstallation of *ISS* after it had been removed did not progress smoothly. Updating of the product did not seem to occur at all during the initial download period. Updating was therefore performed after the installation reboot.

DOCUMENTATION AND WEB PRESENCE

ISS is designed primarily to be downloaded from the *CA* sales website and used without any hard copy being



Program Control

These are the programs that have tried to access the Internet or local network, along with the permissions they were given.

Change program permissions by left clicking the X, ? and "check" icons.

Access: Allows a program to actively retrieve information on the Internet or network.

Server: Allows a program to passively listen for unsolicited contact from the Internet or network. Very few programs require server rights.

Send Mail: Allows a program to send and receive e-mail.

Active	Programs	Access		Server		Send Mail
		Trusted	Internet	Trusted	Internet	
	CA Antivirus GUI	✓	✓	?	?	?
	CA Antivirus Realtime Messaging Service	✓	✓	?	?	?
	CA Antivirus System Report	✓	✓	?	?	✓
	CA Antivirus System Tray Application	✓	✓	?	?	?
	CA eTrust Internet Security Center	✓	✓	?	?	?
	CA eTrust Security Software	✓	✓	?	?	?
	CA eTrust Security Software	✓	✓	?	?	?
	CA ISafe Service	✓	✓	?	?	?
	CA ISS Dashboard Tray	✓	✓	?	?	?
	Computer Associates Software Update Utility	✓	✓	?	?	?
	Computer Associates Update	✓	✓	?	?	?
	eTrust PestPatrol background protection application	✓	✓	?	?	?
	eTrust PestPatrol main graphical user interface	✓	✓	?	?	?
	eTrust PestPatrol Updater	✓	✓	?	?	?
	Generic Host Process for Win32 Services	✓	✓	✓	X	?
	Licensing and Registration	✓	✓	?	?	?
	NVIDIA Driver Helper Service, Version 81.85	?	?	?	?	?
	Update Antivirus Application	✓	✓	?	?	?
●	Winamp	✓	✓	?	?	?

Entry Detail

Product name	Computer Associates Antivirus
File name	C:\Program Files\CA\eTrust Internet Security Suite\eTrust EZ Antiv...
Policy	Not yet configured
Last policy update	Not applicable
Version	Version 7.1.6.0
Created date	27/04/2006 00:03:50
File size	325 KB

involved. A CD version is available, though this comes at an extra cost. As such, most users can be expected to be using product documentation in its online and soft-copy formats.

The degree to which documentation is supplied varies greatly between the products. For the firewall, for example, the tutorial represents the only easy-to-find documentation. Both the anti-virus and anti-spyware components have help files available from the Start programs tree, with an HTML readme file also being available for the anti-virus component.

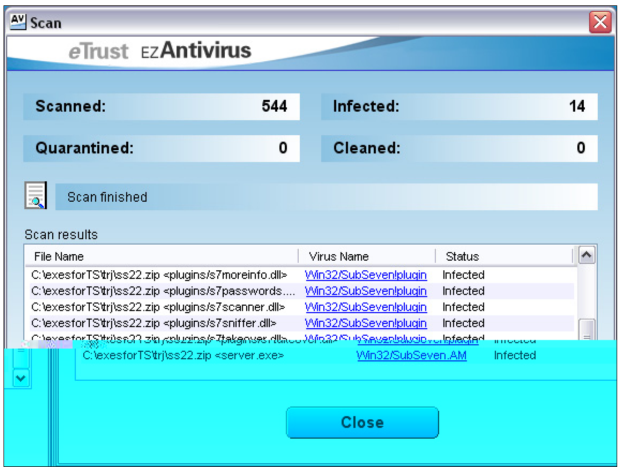
The readme file demonstrates one particular weakness in many of the help functions I examined, in that it contains numerous hyperlinks to the main *CA* site. This is understandable, in that it means that the most recent versions of FAQs, howtos and other documents are always ready for use. Having to depending on a network connection for this information is, however, less than ideal. In case of a firewall misconfiguration or error it is entirely likely that the machine will have no active connection – there is even a specific 'panic button' which will cut the connection in case of suspected infections. Since contact details are also supplied via a web link, this could leave an already worried user feeling very isolated.

The readme does note that documentation is available in PDF format – despite this not being downloaded automatically. In this case, however, only a file name is given, rather than a download link. Such minor frustrations would put off many people instantly – myself included, were I not wearing my reviewer's hat. Sadly, there was no sign of the PDF in any easily discoverable part of the site and my curiosity was vanquished after a fruitless half-hour search. The lack of accessible documentation can hardly be said to be a high point in the presumed intention to make the product easy to use.

The help files were next to be examined, these being the most conveniently available source of documentation. It was quite a relief to note that the implementation here is very good. Full use is made of both internal and external links, though the internal links contain all the necessary product information that I required in my tests.

There is also a troubleshooting menu, which offers some good advice on how to deal with some common queries and problems. The only disadvantage here is that, as is usual with help files, it is very hard to print out a full manual for reference purposes. This should not really be necessary given the availability of the files for browsing, but may annoy some.

The usual *CA* corporate site at www.ca.com does not refer directly to *ISS* except though a succession of links. Links from within *ISS*, on the other hand, point to a sales site operated by a third party. This isolates the user from any useful information other than how to purchase further *CA*



Scanned: 544
Infected: 14
Quarantined: 0
Cleaned: 0

Scan finished

Scan results

File Name	Virus Name	Status
C:\exesforTS\trj\ss22.zip <plugins/s7moreinfo.dll>	Win32/SubSeven!plugin	Infected
C:\exesforTS\trj\ss22.zip <plugins/s7passwords....	Win32/SubSeven!plugin	Infected
C:\exesforTS\trj\ss22.zip <plugins/s7scanner.dll>	Win32/SubSeven!plugin	Infected
C:\exesforTS\trj\ss22.zip <plugins/s7sniffer.dll>	Win32/SubSeven!plugin	Infected
C:\exesforTS\trj\ss22.zip <plugins/s7takeover.dll>	Win32/SubSeven!plugin	Infected
C:\exesforTS\trj\ss22.zip <server.exe>	Win32/SubSeven.AM	Infected

products. This is irritating at best, and downright frustrating if technical information is being sought.

FEATURES

The features available within *ISS* may be accessed easily in three ways: through the startling number of tray icons, by launching the applications individually, or through the use of the *ISS* central console.

Generally, the console offers a simpler view of the applications, with control over the functionality being limited. Status is well reported but, for example, the anti-virus and anti-spyware portions allow only a full system scan or update to be performed without recourse to launching the independent applications. While this simplicity does have some advantages, it would be nice to have at least the ability to scan a more finely tuned area of the machine, rather than the current all-or-nothing option.

The firewall portion of the console is even more limited, offering status information, the option to launch the main firewall application and nothing else. The anti-spam application is unique in existing only through its tab in the console – though being listed as an individual application. However, anti-spam functionality was not tested at this time.

Once launched, the *EZ Antivirus* application has a rather more respectable number of options on offer. Four tabs control operations, divided amongst Virus Scanning, Update, Tools and Help. Help is dominated by version information, though it does also offer links to the *CA* website.

One of these links points to the Virus Encyclopedia, which is far more useful than the other links mentioned earlier. Here can be found the common fare of information on new threats, alerts and a variety of tools. Online scanning, sample submission, utilities and updates are available here. It would be more useful to have this as the primary product site.

The control offered on the Virus Scanning tab allows scheduling, scanning of browsed folders and the adjustment of settings. In reality, the settings that can be adjusted are limited to whether disinfection or quarantining will occur, and to the setting of exclusions. Updating is similarly streamlined, with schedules and proxies being the configurable areas.

Tools proved more interesting, offering views of quarantined items, logs and an overall system report for use if issues occur that require recourse to *CA*'s technical support. The log viewer is interesting, in that its output looks far more easily parsed than that from *CA*'s corporate products, which seems unusual. Examination of the raw log files proved that logging here is indeed in plain text, and thus eminently preferable to that of *CA*'s corporate offerings.

The *PestPatrol* application, when launched individually, offers much the same level of control, though the gathering of prevalence data is also supported here. This collates, on a voluntary basis, the results of user scans in an attempt to provide wider statistical data. There is a large quantity of data available and Internet links are provided to this.

One unfortunate issue was that *PestPatrol* crashed twice while testing. The instability was limited to the GUI, however, with real-time scanning unaffected.

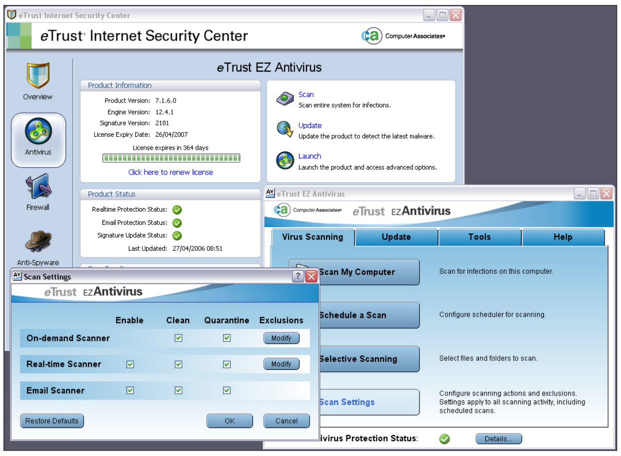
CONCLUSION

As a security suite designed for home use, *ISS* certainly contains all the standard features, plus a few which are less common, such as the privacy protection functionality.

Such a wide range of utility does, however, present problems to the developers when designing an interface. Since the *ISS* contains applications which can be installed and operated independently of one another, a single monolithic application is not really feasible. The situation is aggravated by the fact that the components have been developed by widely separated teams (originally by different companies).

From a more practical point of view, it would also be rather overwhelming for a potential user to be faced with all of the options available for each of the suite components. The use of a central console thus makes logical sense but I would suggest that it has not yet been taken far enough in *ISS*. The large number of tray icons is one example of the integration being not quite as deep as might be hoped.

On a less obvious note, updates are performed individually for each suite component rather than as conglomerated batches. This is most notable for the anti-virus and anti-spyware components which require frequent updates – often updates which cover the same malware. If the


eTrust Internet Security Center

eTrust EZ Antivirus

Product Information
Product Version: 7.1.6.0
Engine Version: 12.4.1
Signature Version: 2181
License Expiry Date: 26/04/2007
License expires in 364 days
Click here to renew license

Scan — Scan entire system for infections.
Update — Update the product to detect the latest malware.
Launch — Launch the product and access advanced options.

Product Status
Realtime Protection Status: ✓
Email Protection Status: ✓
Signature Update Status: ✓
Last Updated: 27/04/2006 08:51

Scan Settings

	Enable	Clean	Quarantine	Exclusions
On-demand Scanner		✓	✓	Modify
Real-time Scanner	✓	✓	✓	Modify
Email Scanner	✓	✓	✓	

Scan My Computer — Scan for infections on this computer.
Schedule a Scan — Configure scheduler for scanning.
Selective Scanning — Select files and folders to scan.
Scan Settings — Configure scanning actions and exclusions. Settings apply to all scanning activity, including scheduled scans.
ivirus Protection Status: ✓

products involved had originally been developed in-house it might be feasible to use unified updates here but as it stands this would no doubt be an unpleasant technical challenge.

From a wider perspective, the gulf between the two types of detection is becoming increasingly blurred, such that it would perhaps make more sense to combine the two applications fully rather than have them perform the same operations, looking for the same files but independently. This problem is by no means unique to *ISS*, since anti-virus companies have long tended to expand their security offerings by buying appropriate smaller companies.

This matter is particularly noticeable in *ISS*, due to the number of options having been cut in the components, especially the anti-virus configurations. This gives an impression of a vast, sprawling set of applications, with each only having a very small number of options within it. One hopes that some stream-lining will be possible in future.

These comments aside, there is no real faulting the contents of *ISS*, barring the instability issues noted with *PestPatrol*. Printable documentation could be more logically supplied but that is not a dire sin, since operation is, by and large, fairly easy to understand to all but total novices. A total novice, however, can simply accept defaults in all cases and have little need to interact with *ISS* – very much as would be hoped for by general users.

Technical Details

Test environment: 1.6 GHz Intel Pentium machine with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drive running *Windows XP Professional SP2*. AMD64 3800+ machine with 1 GB RAM, 80 GB hard disk, DVD/CD-ROM and 2 MBit ADSL Internet connection running *Windows XP Professional SP2*.

Product: *eTrust Internet Security Suite 7.1*

Developer: CA Inc., One CA Plaza, Islandia, NY 11749 USA. Tel: +1 631 342 6000; email sales@CA.com, web http://www.ca.com/.

END NOTES & NEWS

The Seventh National Information Security Conference (NISC 7) will take place from 17–19 May 2006 at St. Andrews Bay Golf Resort & Spa, Scotland. See http://www.nisc.org.uk/.

The 2006 IEEE Symposium on Security and Privacy will be held 21–24 May 2006 in Oakland, CA, USA. For details see http://www.ieee-security.org/TC/SP2006/oakland06.html.

AusCERT 2006 takes place 21–25 May 2006 in Gold Coast, Australia. For details see http://conference.auscert.org.au/.

The Fourth International Workshop on Security in Information Systems, WOSIS-2006, will be held 23–24 May 2006 in Paphos, Cyprus. For details see http://www.iceis.org/.

CSI NetSec '06 takes place 12–14 June 2006 in Scottsdale, AZ, USA. Topics to be covered at the event include: wireless, remote access, attacks and countermeasures, intrusion prevention, forensics and current trends. For more details see http://www.gocsi.com/.

The SecureLondon Seminar will be held on 20 June 2006 in London, UK. The SecureParis event has been postponed until the autumn. For details see https://www.isc2.org/cgi-bin/isc2event_information.cgi.

The First Conference on Advances in Computer Security and Forensics (ACSF) will be held in Liverpool, UK, 13–14 July, 2006. The conference aims to draw a wide range of participants from the national and international research community as well as current practitioners within the fields of computer security and computer forensics. For details, see http://www.cms.livjm.ac.uk/acsf1/.

Secure Malaysia 2006 will be held 24–26 July 2006 in Kuala Lumpur, Malaysia. Secure Malaysia is co-hosted by National ICT Security & Emergency Response Centre (NISER).The show will be held alongside CardEx Asia and Smart Labels 2006. See http://www.protemp.com.my/.

Black Hat USA 2006 will be held 29 July to 3 August 2006 in Las Vegas, NV, USA. See http://www.blackhat.com/.

The 15th USENIX Security Symposium takes place 31 July – 4 August 2006 in Vancouver, B.C., Canada. A training programme will be followed by a technical programme, which will include refereed papers, invited talks, work-in-progress reports, panel discussions and birds-of-a-feather sessions. A workshop, entitled Hot Topics in Security (HotSec '06), will also be held in conjunction with the main conference. For more details see http://www.usenix.org/.

ECCE2006 will be held 12–14 September 2006 in Nottingham, UK. This will be the second E-Crime and Computer Evidence Conference to be held in Europe. For full details, including a call for papers, see http://www.ecce-conference.com/.

The Gartner IT Security Summit 2006 takes place 18–19 September 2006 in London, UK. For full details see http://europe.gartner.com/security/.

HITBSecConf2006 will take place 18–21 September 2006 in Kuala Lumpur. Further details and a call for papers will be announced in due course at http://www.hackinthebox.org/.

The SecureLondon Workshop will be held on 3 October 2006 in London, UK. For details see https://www.isc2.org/cgi-bin/isc2event_information.cgi.

Black Hat Japan 2006 takes place 5–6 October 2006 in Tokyo, Japan. Unlike other Black Hat events, Black Hat Japan features Briefings only. For more information see http://www.blackhat.com/.

The 16th Virus Bulletin International Conference, VB2006, will take place 11–13 October 2006 in Montréal, Canada. Email vb2006@virusbtn.com for details of sponsorship opportunities. The full programme is now available at http://www.virusbtn.com/.

RSA Conference Europe 2006 takes place 23–25 October 2006 in Nice, France. See http://2006.rsaconference.com/europe/.

AVAR 2006 will be held 4–5 December 2006 in Auckland, New Zealand. See http://www.aavar.org/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover < $100 million): $1,000
- Corporate (turnover > $100 million): $2,000
- *Bona fide* charities and educational institutions: $175
- Public libraries and government organizations: $500

Corporate rates include a licence for intranet publication.

See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

Editorial enquiries, subscription enquiries, orders and payments:
Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England

Tel: +44 (0)1235 555139 Fax: +44 (0)1235 531889
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

NEWS & EVENTS

OECD CALLS FOR COORDINATION AND COOPERATION

The Organization for Economic Cooperation and Development (OECD) has called on governments and industry across the world to step up their coordination to combat the global problem of spam.

A new set of recommendations issued by the OECD last month calls for the establishment of clear national anti-spam policies and for anti-spam law enforcement bodies to be given greater power and resources globally. The 'Recommendation on Cross-Border Cooperation in the Enforcement of Laws against Spam' urges governments to ensure that their laws enable authorities to share information with other countries. It also recommends that each country establish a single national contact point to facilitate rapid and effective international cooperation.

The set of recommendations form part of the OECD's Anti-Spam Toolkit, which is designed to provide policy makers from across the world with a comprehensive package of regulatory approaches, technical solutions, and industry initiatives to fight spam. The Anti-Spam Toolkit is available online at http://www.oecd-antispam.org/.

VOIP PHISHING SCAM

A new variety of phishing scam was spotted last month: VoIP phishing. Instead of coercing victims into entering their confidential details on a fake website as 'traditional' phishing scams do, the new type of attack cons victims into providing their information on a fake customer support number. To date, however, only a small number of the VoIP-style phishing attacks have been reported.

SPAM PAPERS AVAILABLE

Papers and slides from the 2006 Spam Conference held at the end of March are now available online. The organizers recommend that the papers are downloaded, burned to a CD-ROM and even suggest that you deposit a copy in your local academic library. Webcasts of the conference are also available at http://spamconference.org/.

EVENTS

INBOX 2006 will be held 31 May to 1 June 2006 in San Jose, CA, USA. The event will cover all aspects of email including topics such as 'has CAN-SPAM failed us?', 'what can ISPs do to fix spam?', 'how not to be a spammer' and 'new directions in identifying spam'. For more information see http://www.inboxevent.com/2006/.

The EU Spam Symposium will be held 15 June 2006 at the University of Maastricht, the Netherlands. In addition to discussing technical issues, the symposium will discuss the effect of spam on business and what policymakers can do to contain the spam problem. An ex-spammer will also be present to reveal the psychology of spamming from the spammers' point of view. Full details can be found at http://www.spamsymposium.org/.

The Anti Phishing Working Group (APWG)'s European Summer General Meeting and SuperSummit will be held 27–28 June 2006 in Brussels, Belgium. The event will run alongside the Messaging Anti-Abuse Working Group's (MAAWG) 7th General Meeting. There will be two full days of presentations, discussions, roundtables and industrial working-sessions. Some of the sessions will be open only to APWG Members, while others will be open to non-members – all conference registrants will be vetted by the APWG organizers. Full details can be found at http://www.antiphishing.org/.

The third Conference on Email and Anti-Spam, CEAS 2006, will be held 27–28 July 2006 in Mountain View, CA, USA. The conference encompasses a broad range of issues relating to email and Internet communication. Full details can be found at http://www.ceas.cc/.

The Text Retrieval Conference (TREC) 2006 will be held 14–17 November 2006 at NIST in Gaithersburg, MD, USA. More details of the TREC 2006 spam track including information on how to participate can be found at: http://plg.uwaterloo.ca/~gvcormac/spam/.

FEATURE

TURN OFF YOUR PC

Tomer Honen
Aladdin Knowledge Systems, Israel

He knows you by name and probably knows some of your friends' names as well. He knows where you work, what your company's products are and your annual income. He knows your website by heart and by Googling for an hour he can also find out where you live and at which bank your company's main account is stored.

Without getting off his chair he can devise a most insidious attack against your company. Or he can clear out your bank account in the time it takes you to eat your breakfast. Not too long ago he was a script kid, but he grew up and realized how easy it is to bankrupt you and your company.

Multi-stage targeted phishing attacks are too attractive for malicious coders to ignore. They are an unholy joining of Trojan, spyware and phishing techniques, utilizing the best (or worst) of each to launch an attack. Mindless computer worms infecting PCs at random may be a challenge, but they won't buy you a brand new 4x4. In a sense, these phishing attacks are the perfect crime, promising substantial profit at minimum risk – the dark side of rags to riches, if you will.

If your security systems are just a tad short of perfection you may be the target of the next multi-stage phishing attack. So turn off your PC, leave the office and take a stroll downtown. Today, getting mugged at gunpoint will probably cost you less.

NUISANCE

Computer virus analysts usually take great care when writing articles – it's very easy to cause public hysteria and frighten anyone who is not 'in the know' with stories about some super-powerful new virus. Most of the headline-making viruses (especially those that make it to the nine o'clock news) seem very complex and quite sinister. In reality, even the most notorious malware outbreaks might be caused by the simplest of worms. The truth is that most malware is nothing more than a nuisance.

Think about it: all they usually do is get the user to install them and spread to other systems. Cleaning them (at least to the point where they can't ever be executed again) can often be done manually by those savvy enough to know what drive C looks like and how to press Alt, Ctrl and Delete simultaneously, open the Task Manager, find the suspicious application, close it and then delete that same file from wherever it might have landed on your PC. End of story. Registry entries might still attempt to run this file on every startup until the end of time – but they will always fail; the file has ceased to be, expired and joined the choir invisible.

Some might say that this is an over-simplification of the problem – they would be wrong. Of this year's worms and last year's outbreaks over 90% were simple self-propagating code that can do little else. As for the rest of them, only a fraction pose a real threat to home users and the corporate world alike. While virus analysts may tread carefully where other threats are concerned, phishing – especially the latest incarnation – is a subject that cannot be sugar-coated. While few people are likely to be exposed to such extreme threats, those that will be targeted are risking everything they own if they fail to take adequate measures to protect themselves.

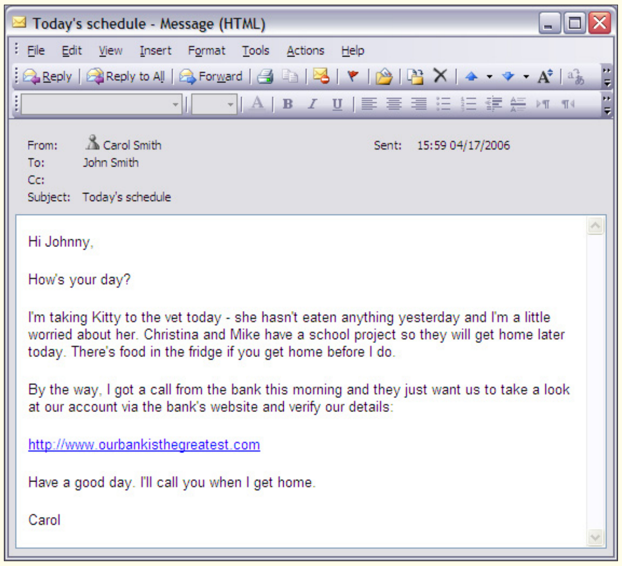
LET'S GO PHISHING

The term 'phishing' was coined in the mid-1990s by hackers, describing an attempt to 'fish' for usernames and passwords from *AOL* dial-up customers (in hacker-slang 'ph' often replaces the letter F). The stolen accounts would then be used for spam, hacking and a myriad of other illegal activities.

The modus operandi for obtaining the account information was for the hacker to pose as an *AOL* staff member and contact users via *AOL Instant Messenger*. They would then ask users to 'verify their account' by submitting their account information and even the credit card details used to pay for the account. The effectiveness of this method forced *AOL* to take several measures to protect itself and its customers.

For a few years following these incidents the phishing phenomenon took second place to other types of malware. Then, just a few years ago, phishing attempts started popping up in users' mailboxes spammed as messages from popular services such as *PayPal* and *eBay*. But the real devastating potential of phishing has only recently been realized with the adoption of a relatively new technique: targeted attacks.

A few widely publicized targeted attacks came into the media's spotlight last year. Hackers somehow managed to coerce otherwise careful users in an organization to execute a seemingly harmless file. The file then opened a backdoor on the infected system, allowing the hackers to log keystrokes, steal documents and sensitive data and do almost anything an authorized user sitting in the infected station can do. What separates these attacks is that they were not launched at random, against an anonymous user. They were carefully thought out and planned to affect a specific target, even a specific user within an organization. It is safe to assume the publicized attacks were only the tip of the iceberg, as most of these incidents were discovered by chance.


Today's schedule - Message (HTML)
File Edit View Insert Format Tools Actions Help
Reply | Reply to All | Forward

From: Carol Smith Sent: 15:59 04/17/2006
To: John Smith
Cc:
Subject: Today's schedule

Hi Johnny,

How's your day?

I'm taking Kitty to the vet today - she hasn't eaten anything yesterday and I'm a little worried about her. Christina and Mike have a school project so they will get home later today. There's food in the fridge if you get home before I do.

By the way, I got a call from the bank this morning and they just want us to take a look at our account via the bank's website and verify our details:

http://www.ourbankisthegreatest.com

Have a good day. I'll call you when I get home.

Carol

Figure 1: The email entices the recipient into clicking on a link.

Targeted phishing attacks work in the same way as spammed phishing; both try to persuade the user to visit a specific website and enter the requested details (username and password, credit card information etc.). Targeted attacks are far more disturbing as they are usually sent only to a single user whom the attacker knows to some extent. If the target is a public figure, knowing a thing or two about his personal life is quite easy. If the target is a private person the attack may originate closer to home.

SETTING UP THE BAIT

For example, let's say John is the president of a large company. Having been interviewed by many magazines and TV stations, he is probably a well-known public figure. A would-be attacker knows John's name, his company's profile and the names of a few of his employees. The attacker also knows who John's wife is, where she works, what his kids are called and the name of all his pets.

With these seemingly unrelated pieces of information, the attacker could compose a simple email to John, to his wife or to one of his employees. The email will entice the recipient into clicking on a link, entering his or her financial details and hitting the 'Submit' button – all without arousing suspicion. An email from : 'John's wife' to him might state that she intends to take the cat to the vet, that there's food in the fridge, that the kids will be home late from school today, and asking if he would mind taking a look at their online bank account details and verifying the information there – link included.

Clicking on the link would lead John to a specially crafted website that looks identical to his bank's website, where he would be greeted by the usual login screen. Punching in his details and hitting the enter key would simultaneously log him into his bank account (this time, the real thing) and make one hacker very happy indeed; John just handed his bank account over to a complete stranger. In a few hours John may discover the ruse, but by then it will be too late.

Multi-stage targeted phishing attacks work in the same way as described above, but in this case the crafted website will download spyware invisibly onto the target's PC. There's little or no need for any user interaction – the spyware is programmed to 'phish' for data (financial details, proprietary information etc.) automatically. For the attacked party, it can take months to figure out from where the information leak originates and how to seal it.

In the above example, let us focus our attention for a moment on Irene, Chief Financial Officer at John's company. Although she is not a public figure, her picture, name and occupation appears on the company's website – this information is sufficient for the hacker to launch the attack. While John is abroad on a business trip or on vacation, Irene receives an email apparently addressed to other employees as well (but actually sent only to her). In the message, 'John' will state that he will not be available in the next few hours and that the link below shows a local website that holds some interest to the company. A short note following the link states that since the address is local he's not sure if it will work for all recipients.

Being the obedient employee, Irene will click on the link, have spyware installed on her system without her knowledge, and be redirected to a false error message stating that the website is only available to local IP addresses – which is exactly what 'John' stated in his email. Since this is a minor issue and not really related to her work, Irene will quickly forget about it and never remember to bring this up in the future when speaking to John.

The most disturbing aspect of these attacks is that they are so incredibly easy to perform. All a hacker needs to do is read a few articles and create a credible story. With some coding experience, some patience, attention to detail and just a bit of luck, anyone can become rich (albeit in a criminal way) with just a few days' work.

If only the solution to these attacks was as simple.

NEW, BETTER, SMARTER PHISH

It's never a bad idea to treat every unsigned electronic communication with a certain level of suspicion. Email, after all, is a very simple tool that can be manipulated easily. Browsing the Internet is not in any way safer. The solution must first come from the users. The most sophisticated anti-malware application is useless when a

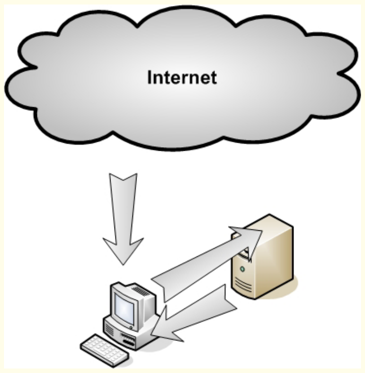


Figure 2: Content first gets into the system and only then inspected.

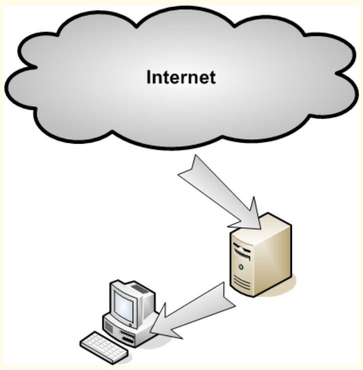


Figure 3: Content is inspected before it gets into the system.

user disables it, forgets to update it or just doesn't use it at all because it hogs system resources.

Like in many other fields and professions, understanding the problem is half the battle to find the cure. But users cannot eliminate such threats on their own. Without an adequate anti-malware solution even the most security-aware person cannot be perfectly safe.

Unfortunately, targeted attacks use unique malicious code that is entirely different from any other virus or worm ever released; so desktop-based anti-virus products that rely on signatures can be tossed right out of the window.

Since that specific malware is not in the wild (in active circulation around the world) and never will be, a signature-based solution will not detect it. A better solution would be proactive – a product that is able to detect threats based not on their unique signature but rather on what they were actually designed to do. A product like this will be able to block such threats as it will recognize suspicious activities like key logging or the theft of certain documents.

Another problem is that anti-virus solutions (both desktop and network-based) can easily be manipulated by certain malware. Many worms are capable of disabling security-related processes on the infected system. Adding this feature to a targeted phishing attack is both easy and highly effective, leaving the user completely exposed – not only to this attack but to any future threats as well. Most users will not be able to resolve this issue without taking some radical action such as to reformat the infected machine.

Even network-based anti-virus solutions are not enough to protect an organization completely against such threats (see Figure 2). While a system administrator may eventually find out about a system exposed to hostile actions it may simply be too late. For corporate users, the best solution to such problems is to employ a gateway-based proactive solution.

NO PHISHING ZONE

In an environment where all content is inspected before it arrives at the user's system (see Figure 3), the targeted system is much safer. This is simply because recognized malware will be blocked before it has a chance to be executed.

A gateway product that offers a proactive response to both known and obscure threats is the best solution for phishing. All content is scanned and the product can identify suspicious trends in a file as it is being downloaded. If John's company used a proactive gateway solution, things would work out differently. He would receive the email from his 'wife' but will not be able to get anywhere by clicking on the link – because the system would have identified suspicious code embedded within the website John wanted to visit. Instead, he will be presented with an error message explaining why the website was blocked.

Since gateway filtering systems are designed to support a large number of computers, owning one at home is a little over-the-top – not to mention expensive. Most home users are connected to the Internet using an Internet Service Provider. Since all data already passes through the ISP's servers before being sent onward, it only makes sense to perform content filtering on the data before it reaches customers. Users may have to pay a little extra for this service, but the alternative is much more expensive.

Surviving these incidents is not impossible, but it requires users and administrators alike to stay vigilant and even expect phishing attacks to come. If your company is using proactive content filtering at the gateway, you can probably breathe easy. You may find that getting mugged at gunpoint somehow sounds less and less appealing.

Phishing facts

Phishing is the fastest growing threat in the world today. The damage already caused by current phishing attacks is estimated in the billions of US dollars. The following are a few statistics gathered by the *Anti Phishing Working Group* (*APWG*) (http://antiphishing.org).

January 2006 phishing statistics:

- 17,877 unique reports (12,845)
- 9,715 unique phishing websites (2,560)
- 184 unique password-stealing apps (77)
- 31 days – longest time online for phishing website

Note: The numbers in parenthesis represent the statistics from the same period last year, i.e. January 2005.