Deconstructing malware: what it is and how to stop it

David Williamson
Director of Sales, United Kingdom and Ireland, Ubizen

David Williamson joined Ubizen as director of sales for United Kingdom and Ireland in August 2003. He is responsible for spearheading the company's growth as part of the company's strategic plan to capitalise on the growing demand for managed security solutions in the UK market.

David's experience runs back to 1983, when he held positions with Apple Computer and Digital Equipment Company before co-founding TCS Consultants Ltd. in 1992.

David possesses extensive expertise in the security arena and holds the prestigious CISSP qualification, amongst others. He is currently studying for his MBA with Henley Management College.

1. What is malware?

Malware – or, to give it its full title, malicious software – is a catch-all term that is used to describe software that intentionally causes damage to a computer, collection of computers or network infrastructure. Malware is created intentionally and is released by individuals with the intent of causing harm, either directly through the effects of the malware on the target systems, or as a by-product via propagation or impact on the resources of the target systems and thus causing a Denial of Service (DoS) Attack.

Malware can affect all three elements of the CIA trail – Confidentiality, Integrity and Availability – but it most often impacts the integrity and availability of the target systems, by exploiting vulnerabilities and causing system instability; malware also attempts to propagate and consume host and network resources, usually to the detriment of the other, legitimate processes on the system.

1.1. The various types of malware

The most common forms of malware are viruses and worms, but other types of malware exist including malicious mobile code (MMC), Trojan horses, spyware and root kits (sometimes known as backdoors).

Worms and computer viruses are either computer programs or scripts that spread from one computer to another without the consent, and often without the knowledge, of the legitimate user. The key difference between these two types of malware is that viruses spread when the user of the system inadvertently triggers them, whereas a worm is wholly independent and propagates autonomously.

The transport most often used by viruses is e-mail, triggered by the user opening an e-mail message. Due to the default configuration of many e-mail clients, malware can be embedded in an e-mail message as a script that executes automatically when the message is read. Examination of virus code shows that raison d'être of most viruses is simply to propagate as widely as possible; any harm to the host system is often incidental, a by-product of the mechanism and the use of resources. This does not, of course, excuse the behaviour of the authors of such malware, and writing and releasing a virus is rightly a criminal offence in many countries including the United Kingdom.

Unlike a virus, a worm propagates without user involvement. Typically, it exploits a previously unknown or recently discovered vulnerability on a widely deployed software package. A worm exploits the vulnerability, gaining temporary control of the host system, and then uses the host as a springboard for launching attacks on new systems.

The third category of propagating malware is that of MMC. This utilises active content as the attack vector, accessed via the web browser and leveraging weaknesses in ActiveX® or Java® applets. An example of such an attack is CoolWebSearch spyware, which installs using vulnerabilities in Microsoft's Java Virtual Machine (JVM).

Various approaches exist for mitigating the risk of these types of malware and these will be explored later in this article.

1.1.1. A brief history of worms

While the concept of worms dates back to the early 1970s, and the term itself is derived from a 1975 science fiction novel, the first worm released into the wild is

generally considered to be the Morris worm, launched in late 1988 to exploit a vulnerability in the Sendmail program on DEC VAX and Sun systems. Since then many worms have been released by various individuals, with varying degrees of impact. Notable examples include Nimda, Code Red, SQL Slammer, MS Blaster and most recently the Sasser worm.

One differentiating characteristic of these various worms is their virulence – the speed at which they propagate. Whereas older worms such as Nimda took days to build to peak infection rates, allowing system administrators to take countermeasures and protect their systems, newer worms – most notably SQL Slammer – spread much more quickly, reaching peak infection rate in just a few minutes. The concept of a 'hyper-virulent active worm', that would infect all vulnerable hosts in 15 minutes or under, is largely credited to Nicholas Weaver, a graduate student at the University of California at Berkeley, who coined the term 'Warhol worm' – a reference to Andy Warhol's comment that "In the future, everybody will be famous for 15 minutes."

1.2. Stacks, heaps and exploits

In its most basic form, malware consists of a way of entering or gaining control of a host system – by exploiting a vulnerability – and a payload and/or a means of replicating itself to another host system. The main type of vulnerability-exploited worms, are known as buffer overflow vulnerabilities. These occur when programmers don't use bounds and range checking when declaring input buffers, which allow malware developers to 'inject' their code into the target system's memory space where it is executed, typically with SYSTEM, or root, privileges. Two types exist – stack-based and heap-based.

The stack is an allocated area of memory inside each program that stores variables that are local to functions and acts as a virtual scratch pad. The heap is used to store dynamic data structures such as binary trees, where memory is allocated by the programmer (using the malloc command in the C programming language, for example).

Heap-based buffer overflows are broadly similar in concept; the operating system is tricked into running unauthorised, alien code and so compromises the system. Whereas it is comparatively easy to protect the stack, either through diligent programming or through additional software that monitors the stack and prevents software from executing, the heap is much harder to protect on Windows-based systems. On *nix systems, applications have but one heap apiece; the calls to manipulate the heap are relatively simple and on many variants there is a kernel-level switch that can be enabled to prevent code executing on the heap. Conversely Win32 applications can have multiple heaps; heaps can have different characteristics and the data and control blocks are allocated adjacently, which complicates attempts to monitor and protect heap operations.

While the majority of buffer overflow exploits utilise the stack as a way of injecting their code into the target system, heap-based buffer overflows have become increasingly common – for example, the Code Red worm of 2001 used a heap-based buffer overflow exploit as its primary vector. Win32 heap-based exploits are harder to write, as execution flow is not synchronous as in stack-based exploits, but similarly it is harder to develop software to prevent heap-based exploits succeeding, particularly so if system stability isn't to be compromised.

While stack and heap-based buffer overflow techniques have been known about for years, new approaches for compromising systems are emerging. These include so-called format string vulnerabilities, and return-into-libc overflows. Such new developments are set to provide whole new attack vectors for malware.

1.3. Zero-day attack

The concept of a 'zero-day attack' is often mentioned in the press. While rare, this refers to the concept of an exploit that exists – either in manual form, or automated as Malware – for a vulnerability that is unknown to the software vendor or to the information security industry at large.

Without warning – hence 'zero-day' – these attacks can be most damaging, with no patch to remedy the flawed software available from the vendor, and limited knowledge of the attack vectors used by the malware available to information security professionals. An example of such an attack was that used against the US Army in March 2003, one that exploited a previously unknown buffer overflow vulnerability in the Microsoft Windows 2000 WebDAV component used by Microsoft IIS.

1.4. Emerging trends

Blended threats, sometimes called Combination or Combo Malware, are on the increase. By combining the characteristics of different types of malware and using multiple attack vectors to infect a host system, blended threats can spread rapidly and typically cause widespread damage.

Some worrying trends identified by Ubizen®, through the research carried out by its Security Intelligence Lab and from information gathered from the thousands of security devices monitored on behalf of its enterprise clients, are a clear increase in reconnaissance – port scanning and fingerprinting to identify the type and configuration of systems accessible from the Internet. There has also been increasing discussion in the underground that malware is combining worms with either spyware or root kits. Analysis of these trends, and recent blended threats such as Sobig.F, indicates that threats seen in the wild so far are just the tip of the iceberg.

1.5. Sources of infection

In the dim and distant past – the 1980s – the primary means of sharing information between different computers was via floppy disk. Early viruses worked by infecting floppy disks, and the source of infection was almost certainly an un-scanned floppy disk. Organisations responded through education, a policy of mandatory scanning of floppy disks and installation of desktop anti-virus products.

As of the early 1990s networks became ubiquitous, connected to the Internet for browsing and e-mail, and the primary source of infection changed to be via e-mail, or the downloaded attachment (typically a .exe file). Organisations again responded, supplementing earlier controls with strong border defences – firewalls, intrusion detection systems, etc. – and scanning of in- and out-bound e-mail using products such as MIMEsweeper or managed services such as that from MessageLabs.

While these sources remain of concern today, they have been surpassed by different threats: the ignorant user and the unsafe partner.

As increasing numbers of users have ditched their desktops, replacing them with laptops for home working and, as

organisations build their Web of interconnections with partners, suppliers and customers, the primary source of infection has shifted away from direct infection via the 'hard shell' of the network boundary to that brought into the 'soft inside' of the network: a user's laptop infected from their home DSL or dial-up connection; a partner's VPN connection for information exchange that bypasses the main firewalls; or USB memory sticks that are used to transfer information between otherwise unconnected systems.

A comparatively new threat is that of wireless networking, particularly that from 802.11n. As Wireless Access Points proliferate both within corporations and in public places such as coffee shops and airports, and laptops are fitted with WiFi networking as standard, users are opening little understood and often lightly monitored network connections and so exposing themselves to malware attacks that they were previously shielded from. Particularly in densely-populated areas – such as the City of London – is it often possible to access an external 802.11n network from one's desk inside the corporation's physical premises; the temptation to connect in order to access personal web sites or e-mail can be overpowering, but this can provide a way for malware to enter the corporation despite the best efforts of the information security team to secure the perimeter.

2. Mitigation and detection strategies

2.1. Have you been compromised?

Detection of infection by most types of malware is relatively easy. For a virus outbreak, the first notification is typically calls from recipients of your infected e-mails complaining that you're sending them a virus. Installing desktop anti-virus protection, with up-to-date signatures will reveal any infection. Worms can be harder to identify, but often they will impact the performance of your system or network or introduce instabilities to previously working and stable systems. If such circumstantial evidence is combined with the capabilities of network scanning and an Intrusion Detection System, worms can be quickly identified and mitigation and eradication steps taken accordingly.

Other types of malware can be much harder to identify, let alone eradicate. Of these the most sinister is the root kit. Root kits enable intruders to access the compromised system at will and take control over processes, access data and even use the system as a host as springboard to, for example, launch Distributed Denial of Service (DDoS) attacks. Different root kits work in different ways, but fundamentally they fall into two categories: user-mode and kernel-mode. User-mode root kits operate on the host system as visible processes, viewable under *nix using the ps command and on Win32 via the Task Manager. Kernel-level root kits are infinitely more devious, modifying the host operating system such that the presence of the root kit files – be they memory-resident or written to disk – are hidden from the operating system, and in turn from high-level applications and utilities that look for them.

The good news is that tools do exist to identify root kits, from freeware tools such as Rootkit Hunter through to commercial products such as PREVX Enterprise that includes compromise detection as part of its overall functionality.

2.2. Mitigation – protecting oneself

To the casual observer it would appear that most malware affects systems running

Microsoft operating systems and there is a myth that you can be safe from malware if you don't use Windows or other Microsoft products. While it's true that Microsoft products are the most heavily attacked, at least in terms of the number of vulnerabilities reported and the number, frequency and virulence of virus and worm attacks, this has probably got as much to do with the ubiquity of the Microsoft platform as it has with any inherent failings in the products or Microsoft's ability to create software that does not contain a plethora of exploitable vulnerabilities. That said, to cover all platforms in this section would require a much longer article, so we will focus for much of this article on Microsoft Win32 – and particularly Windows XP as a desktop operating system and Windows Server 2003 as a server operating system.

First, one should accept that it is not possible to achieve perfect security; whatever measures are taken, they will only mitigate the risk, not eradicate it. But an approach of layered, synergistic defences will significantly improve your security posture, combining appropriate defences at the network perimeter and at the host, as well as monitoring of the network for anomalous traffic.

When selecting baseline controls for host systems one should consider the context of the organisation, and the scope one has for hardening the standard build that is deployed in the enterprise. Clearly for home or SoHo users such restrictions are less of a problem, but it is always possible that critical applications have dependencies on components that one might wish to disable, so trial and error is often necessary.

So, what should you do? Start by disabling Microsoft Windows Scripting Host by finding and renaming wscript.exe; at your e-mail ingress point ensure that attachments ending with `.exe`, `.scr` and `.pif` are blocked; configure Internet Explorer's security level to High; if running Windows XP Professional, enable the Internet Connection Firewall (it is better than nothing, and Microsoft are promising improvements in Service Pack 2 for Windows XP); disable unnecessary processes running on the system; and, of course, keep your system up-to-date with patches using Microsoft's Windows Update site. This is by no means an exhaustive list of steps for hardening your Windows XP system; GIAC has a good guide for hardening your WinXP system that can be downloaded free of charge from their web site.

While the above are prudent steps, Microsoft themselves have a part to play. Much has been made of the fact that the new Windows Server 2003 product is "more secure" than its predecessor, Windows Server 2000. Indeed, Windows Server 2003 has built-in protection against stack-based buffer overflows using 'canaries' placed on the stack that enables a watchdog process to identify when an overflow has been attempted. Similar technology has been implemented in Microsoft Visual Studio.NET C++, and is expected in Service Pack 2 for Windows XP, due in mid-2004. While laudable in principle, and better than having no protection, Microsoft's implementation has been shown to be fallible and ways to circumvent it have already posted on the Internet by David Litchfield of NGS Software.

Aside from hardening the system and relying on improvements from the vendors, conventional wisdom is that the best practice for guarding against infection is to update the signature file for desktop anti-virus software daily. While this would appear to be obvious, the reality is that the vast majority of viruses and worms are not

actually seen in the wild: the December 2003 list as maintained by F-Secure, in conjunction with The WildList Organisation International, lists just 631 viruses and worms that have been seen 'in the wild' by one or more of the 75 participants in the scheme, which is approximately 1% of the 65,000+ that are listed in Symantec's Norton Anti-Virus product. Why the apparent discrepancy? Because only a fraction of the viruses known to researchers actually make it into the wild; many fail because of poor coding or their inability to propagate through common defences, or they are simply never released into the wild having been discovered through underground research and infiltration. Given that new malware typically peaks in infection during the first 24 hours – which is the time delay permitted to anti-virus software vendors by the ICSA Labs to produce a new signature – the reality is that by the time the new signature is made available, the greatest danger of infection has already passed. While it is worth updating signatures to provide protection against any legacy infection, this can be performed on a weekly basis – there is little demonstrable benefit to updating desktop anti-virus signatures on a daily basis, provided that the e-mail gateway is updated as soon as signatures are available.

Another contentious area is that of patching. Over the past few years the number of patches released has been increasing steadily, with a corresponding increase in the burden on system administrators to patch the systems under their control. Today the workload stands at a level that is all but untenable for even the most zealous or automated organisation. Of course application of patches is a vital component of any strategy to reduce risk from malware and an appropriate patch regime should be considered mandatory for all organisations. But certainly the use of other controls – and software products – can significantly ease the burden of patching, and the rush to apply a new patch as soon as it's released by the vendor should be resisted until such time that the impact of the patch on critical systems can be evaluated, and a back-out plan developed and tested in case the application has unforeseen consequences.

2.3. Third party products

In addition to the above baseline controls to reduce the likelihood of infection from malware, various software products can be used to further improve one's security posture. Aside from established defences such as Firewalls and Network Intrusion Detection Systems, various products can be deployed on the desktop or laptop to reduce the threat from malware.

Firstly, of course, is the desktop anti-virus product. There are many vendors of anti-virus (AV) software including Symantec, Network Associates (McAfee), Kaspersky Labs and F-Secure. Most AV software is ICSA Labs approved, meaning it has been tested to stop 100% of viruses ever found in the wild, so the choice of software will largely come down to features such as usability and cost.

Next is the personal firewall. Key players include Check Point, with their recently acquired Zone Alarm, and Symantec with their Personal Firewall. Products of this type provide useful protection against zero-day attacks, assuming that the policy and configuration of the firewall is such that it prevents un-trusted inbound connections.

Turning to the emerging category of Host Intrusion Prevention System, key products include Cisco Security Agent

(formerly OKENA, which Cisco acquired in January 2003), PREVX Enterprise,[1] NAI's Entercept and Sana Security's Primary Response. Of these, Cisco Security Agent is the most widely-deployed and includes an optional firewall component, which enables detection of port scans and integrates with the Cisco VPN client software to ensure that only users that have CSA installed, running and with a valid policy can connect over a VPN to the corporate network.

PREVX Enterprise offers extensive protection for the host system and also includes capabilities for detecting kernel-level root kits on Linux and Solaris systems. It works by assuming that the operating system cannot be trusted to truthfully report what processes are running, what files are on the disk and what network ports have been opened, and has code that directly interrogates the underlying components to reveal the actual configuration – not that reported by the potentially compromised operating system.

As described earlier, the key attack vector for many types of malware – particularly worms – is that of vulnerabilities. Regular scanning of exposed systems using Vulnerability Assessment tools such as the open-source Nessus or commercial products such as Qualys' QualysGuard or Tenable's NeWT Pro (the commercial version of Nessus) can provide valuable insight into the actual status of your systems.

Lastly, security intelligence products such as Symantec's DeepSight, TruSecure's IntelliShield or Sintelli's ALERT! enable the security professional to keep abreast of the latest vulnerabilities and gain insight and early-warning of emerging threats.

[1] Full disclosure: the author was a co-founder of PREVX, designing the architecture and features of the original product, and still retains an equity position in the company.

3. Recovering from an outbreak

To misquote Burns, despite the best-laid plans of mice and men (and women, for that matter) things oft go awry. In that case – an infection by virus or worm, or suspicion of other malware such as a root kit compromise – having an Incident Response (IR) programme consisting of an identified team and a pre-approved plan ready to go is critical to successfully managing the situation.

An organisation's IR team is typically assembled from key personnel within the organisation who have been trained in IR doctrine and are familiar with the IR plan. It is extremely rare for an organisation to have a dedicated IR team, even for large financial services institutions that have fraud and compliance departments.

The purpose of the IR plan is to provide a pre-approved and clear set of guidelines of what to do in the event of a security breach, which can be followed in a time of high stress. Organisations should recognise that there may be two conflicting goals – whether to mitigate the impact of the incident and recover the organisation to an acceptable state as quickly as possible, or whether to gather evidence in a way that facilitates identification of a perpetrator and their prosecution. When dealing with a virus or worm outbreak, time to return to normal operation is usually the priority, but when a root-kit compromise has been identified then the latter approach may be more appropriate, building an evidentiary chain from an in-depth forensic investigation.

When an incident has been detected and the decision is taken to invoke the IR

programme, the first action is to assemble the IR team and then follow the IR plan. Ubizen recommends that organisations develop plans that follow a five-step process:

1. Identification – what is the impact of the incident on the business, be that internal, on partners, and/or on customers? What is the source of the incident? What is the nature of the incident?
2. Containment – how can we contain the incident such that it ceases to spread to previously unaffected systems, or impact systems or processes elsewhere?
3. Eradication – having contained the incident, how can we eradicate its effects from the affected systems and restore those systems to a trustworthy state – both operationally, and in terms of integrity of data?
4. Learning – what can we learn from this incident?
5. Improvement – how can we apply this learning to improve our security posture such that future attacks, be they of this type or other, are less likely to lead to an incident?

As a provider of incident response services, including as an approved Incident Response Assessor for Visa and Mastercard, Ubizen has undertaken dozens of investigations for organisations worldwide. One of the common themes in situations where the breach is from hackers (as opposed to malcode) and seen in around 75% of investigations, is that the breach could have been thwarted by a more diligent and comprehensive security regime applied at the network perimeter. Conversely, in the majority of malcode events, the source of the outbreak was traced back to a source inside the organisation, often introduced inadvertently as a result of working from home or at a remote site.

References

[1] http://www.rootkit.nl

[2] http://www.giac.org/practical/GSEC/Zach_Groves_GSEC.pdf

[3] http://usa.visa.com/media/business/cisp/Qualified_CISP_Incident_Response_Assessor_List.pdf

[4] http://www.f-secure.com/virus-info/wild.html

[5] http://www.wildlist.org/