

Stopping the Slew of Mutating Malware

a report by

Dr Steven Hofmeyr

The Malware Problem

Today, a proliferation of malicious software (malware for short) is being seen, including Trojans, Rootkits, Spyware, worms and viruses. Malware is installed surreptitiously, without the explicit consent of the owner, and often turns the victim computer into a remote-controlled 'zombie'. These zombies are formed into vast BotNets that are used for a variety of nefarious purposes, such as stealing confidential information through keystroke logging or screen captures, functioning as servers for pornography or stolen data, launching denial-of-service attacks in extortion rackets, and acting as spam relays. Malware gets onto computers in a variety of ways, including through ill-advised downloads of particular software, through email attachments, through browser flaws when visiting malicious websites, through vulnerable network shares, and through flawed server applications, including standard system services. Sometimes the malware spreads automatically in the form of worms and can spread rapidly enough to cause serious network overload.

The malware problem is getting worse, driven by several trends. First, mobile and remote users are often outside the security of the corporate network, and their computers could easily be infected anywhere and then act as a transmission vector for malware into the enterprise. Furthermore, mobile users' computers are not always connected, making it hard to for them to get updates for signatures and patches in a timely manner. Second, there is an increasing disintegration of the secure corporate perimeter, driven by economic pressures, such as distributed applications (e.g. web services), partner networks, remote offices and wireless networks. Consequently, traditional, perimeter-based security that relies on network devices (such as firewalls) for protection is rapidly becoming ineffective. Finally, as more and more business is transacted online, attackers have a growing incentive for malicious behaviour. Attackers make money through identity theft, industrial espionage, and extortion rackets in which the victim is threatened with denial-of-service attacks.

The Information Technology Security Arms Race

Securing information technology (IT) systems is an arms race between attackers and defenders. Traditionally, the defenders have relied on signature-based anti-virus (AV) systems to secure the end-point, the host computer. Unfortunately, the AV approach is failing, for a variety of reasons. Signatures rely on identifying some constant part of static code, but malware is increasingly mutating so that there is no constant part from one instance to the next, and hence, nothing for the signature to match. Often, malware is customised for a particular attack, which means that no signature exists for it, and after the attack, it is never used again, which means there is no point in developing a signature. This illustrates one of the great failings of signature AV – there are no signatures for completely new attacks (zero-day attacks). This is particularly significant for fast-spreading malware, such as worms, which can compromise thousands of computers in minutes, far too fast for any signature to be developed in time.

To catch up in the arms race, the defenders have developed intrusion prevention systems (IPS). In general, IPS are proactive, stopping attacks immediately without relying on signatures. Hence, IPS protect against zero-day attacks and – because there are no dependencies on updates – provide continual protection to mobile and occasionally connected users. The power of the technology underlying IPS derives from a fundamental precept – IPS constrain the behaviour of running code, they do not scan static code, like AV systems do. After all, code can only do harm when it runs. Constraining behaviour escapes the limitations of static signature scanning because behaviour is much harder to spoof, obfuscate or mutate than static code; for example, it is possible to write code for keystroke logging thousands of different ways, but the behaviour is always the same. Constraining behaviour is also much more efficient, because only running code is monitored and there is no need to do slow, compute-intensive system-wide scans. Finally, behavioural constraints can prevent zero-



Dr Steven Hofmeyr is Chief Scientist of Sana Security, which he founded four years ago. He has served on many program committees including the Association for Computing Machinery's (ACM's) New Security Paradigms Workshop, the Artificial Immune Systems workshop at the IEEE World Congress on Computational Intelligence, and the Genetic and Evolutionary Computation Conference (GECCO). He has been an invited participant to several US Government workshops on future directions in technology and has served on National Science Foundation (NSF) panels deliberating on security research funding. In 2003, Massachusetts Institute of Technology's (MIT's) *Technology Review* named Dr Hofmeyr as one of the top 100 young innovators under 35, and in 2004 he was named one of the 12 innovators of the year by InfoWorld. He received a PhD in computer science from the University of New Mexico in 1999, where his research focused on immunological approaches to computer security. During his studies he was a visiting scholar at MIT's Artificial Intelligence Lab and associated with the Santa Fe Institute.



day attacks by blocking all behaviours that deviate from those normally seen or allowed, without requiring any specific knowledge of attacks.

Stopping Attacks

An attack can take two basic forms – pure malware, which is a program written expressly for malicious purposes, or co-opted malware, which is a legitimate program that is behaving maliciously after being compromised by an attacker exploiting a vulnerability in that program. It is possible to prevent legitimate programs from becoming malware by constraining their behaviour, whereas pure malware needs to be prevented from running and removed from the system.

The first step towards preventing exploits is to constrain legitimate application behaviour with rules that define what an application can and cannot do; for instance, what files it can read and write, whether it is allowed to connect to the network, etc. Rules can be very effective at constraining the behaviour of compromised applications. However, determining the rules is not easy – you need to have expert understanding of both the application being protected and the system on which it is running. The more complex the application, the more complex the rule set, and complex rule sets are much more prone to error, such as rules that prevent legitimate behaviour (false positives). Frequently, rule sets become detuned or simplified to the point where they provide only a mediocre level of protection and therefore many attacks get through (false negatives). Furthermore, changes in the application (such as from software upgrades) or reconfiguration will often require changes in the rule sets. In general, rules should only be used when they are easy to write, which means they are best for protecting applications that are well-understood, reasonably simple and not heavily customised, such as common browsers, e-mail clients and standard system services.

For protecting more complex applications (such as Microsoft Exchange Server), non-standard applications, or heavily customised ones (most web-servers fall into this category), IPS cannot rely on human expertise alone. For these applications, autoconfiguration is essential – the IPS automatically learns a profile of normal behaviour for the application, and then prevents anomalies or deviations from the profile. This enables the system to prevent zero-day attacks and protect the application without requiring any specific knowledge of the application. The only downside of autoconfiguration is that the application must be fully exercised and not compromised

during the configuration period, which can sometimes take several days. On servers, this can be accomplished by doing the auto-configuration during the quality assurance (QA) process, or on fully patched applications out in a production environment.

Stopping Pure Malware

The behaviour-constraining approaches discussed thus far are good at protecting against co-opted malware (legitimate programs that have been compromised), but IPS also need to be able to protect against pure malware – the programs that are expressly written for malicious purposes. IPS stop some malware using global rules to catch egregious behaviours; for example, they might have a global rule to prevent disk formats, but global rules are far too general to offer comprehensive protection.

The problem is that the behaviour of malware and good applications overlaps, so an IPS cannot set global rules on most behaviour. Much behaviour will be exercised by both malware and legitimate applications, making it impossible to differentiate between good and bad on the basis of isolated behaviour. For example, installing a keystroke logger is something that many instant messaging (IM) clients do in order to determine whether the user is still active on the computer; in this case, the keystroke logger behaviour is acceptable. But Trojans also often install keystroke loggers in order to steal information; in these cases, the keystroke logger behaviour is definitely bad. However, although the IM client and the Trojan are similar in this behaviour, they will differ in many other ways. The key is to look at the program holistically, that is, to consider multiple behaviour.

The IPS will be able to monitor the composite behaviour of every running program on the system although often, for the sake of efficiency, the IPS will only monitor those running programs that communicate over the network. The composite behaviour consists of a set of carefully chosen behaviours that, when combined, can give an accurate prediction of whether or not the program is malware. The requirement for choosing a particular behaviour is that it is fundamental to the nature of malware, fundamental in the sense that any malware needs to carry out that behaviour in order to be useful to the malware author. For example, keystroke logging or screen capture is fundamental to Trojans, but so is the need to remain stealthy and hide from the users, and the need to survive system reboots. This behaviour combined can easily be used to differentiate between legitimate programs and malware; for

instance, IM clients do not need to remain hidden, although they may need to survive reboots.

An IPS will apply composite behavioural analysis to each running program, and those that are deemed acceptable will immediately be protected from compromise through rules and autoconfiguration, whereas those that are deemed to be malware will immediately be terminated. Termination of malware will often involve killing multiple processes, because most modern malware is not a single process, program or executable. The IPS will track the communications and interdependencies between the various malware components to ensure that it has killed all of them. Using composite behavioural monitoring, the IPS can effectively stop pure malware without requiring manual configuration or learning periods, and can do so regardless of how much the malware code on the disk is obfuscated or mutated.

Summary – New Threats Require New Defences

The world of the attacker is changing fast, with an ever-expanding variety of malware. Technological trends, such as the mobile work force, are making life easier for the attacker and harder for the defender. Consequently, what used to be the backbone of IT security on the end-point, the signature-based AV system, is becoming obsolete. IPS provide the next logical step in the arms race for the defenders. IPS rely on constraining the behaviour of running applications, and hence overcome the limitations of the old AV systems. IPS are seeing widespread adoption because they are the only way to secure the increasingly vulnerable end-point. And it is essential to secure those end-points, because without security, IT systems will be crippled and unable to deliver on the promise of the future. ■