The future of virus detection

Richard Ford, PhD.
Research Professor, Florida Institute of Technology

Dept. of Computer Science, Florida Institute of Technology, 150 W. University Blvd, Melbourne, FL 32901, USA.

rford@se.fit.edu

Dr. Richard Ford graduated from the University of Oxford in 1992 with a D.Phil in Quantum Physics. Since that time, he has worked extensively in computer security and malicious mobile code. Previous projects include work on the Computer Virus Immune System at IBM Research and development of the largest web hosting system in the world, whilst Director of Engineering for Verio. He is currently Research Professor at Florida Institute of Technology's Center for Information Assurance, where he carries out research on a wide variety of security-related areas. He is currently Executive Editor of Reed-Elsevier's Computers & Security, and Virus Bulletin. He is also Scientific Advisor to the European Institute of Computer Antivirus Research.

Richard lives with his wife and two dogs in beautiful Florida and, when not trying to solve the world's malcode problems, plays both classical and jazz flute.

1. Introduction

When the average computer user can tell you the name of the latest major Internet worm, it becomes clear that we are not winning the war against Malicious Mobile Code. With the billions of dollars spent annually by the antivirus community and operating system manufactures on new and innovative countermeasures to stop viruses and worms, one is tempted to ask is the war even winnable? Is the issue simply one of careless users taking unnecessary risks, or is there something fundamentally broken in the way in which we attempt to defend against the onslaught of new viruses?

In this article, these issues will be examined by looking not at the current threat profile, but at the different ways in which we defend our systems and their weaknesses. By examining the problem from this direction, it is possible to build up a list of requirements for solutions of the future – essentially, what functionality tomorrow's anti-virus software will need to have to keep us one step ahead of the next generation of viruses and worms.

2. Overview of the solution space

While the antivirus industry has been steadily improving over the years, it may come as a surprise for some to realize that the fundamental technology used for detecting and removing viruses has changed very little over time. In this section, a brief discussion of current common virus solutions is outlined, so that a better understanding can be gained of what one can realistically expect from such countermeasures.

When viruses first became common, most samples were simple boot viruses – that is, small blocks of code, usually less than a couple of kilobytes in length that spread via the boot sectors of removable media. Due to the size limitations, and the naivety of the virus writers of the time, each copy of a virus "looked" the same, and the machine code was not obfuscated in any serious way. Furthermore, outbreaks generally took several weeks or even months to build, as the time between infection events was large.

It was in this environment that the scanner made its debut. At the time, most viruses were relatively static and slow moving. It seemed as though a signature-based virus scanning approach would thus be capable of arresting their spread, and thus this technology formed the basis of most virus scanners. Here, the scanner would look for a particular sequence of bytes in boot sectors and files. This approach was fairly accurate, as the virus code was simplistic. Furthermore, due to the slow spread, updates could be issued monthly, often via regular mail.

As the threat changed, so did the solution. While the idea of "signature" based scanning remained popular, scanner vendors improved performance by moving away from scanning every byte of an object to just scanning those around the entry point of the object. This optimization provided for significant speed improvements, which were critical given the memory and processor limitations of the day. Furthermore, by restricting the area of search the problem of false positives – where a clean file is inadvertently reported as infected – was reduced.

The next large change in detection strategy arrived with the advent of so-called polymorphic viruses. These viruses remain operationally unchanged with each replication, but employ variable decryption routines so that there is no constant string of

bytes that can be used for detection. This technique became more popular and forced most vendors to trace the path of execution into the virus code in order to detect an infected file. Perhaps the largest milestone – certainly the most well known – was the release of the Mutation Engine. This was a simple but effective polymorphic engine that was designed to be linked in to simple static file viruses; suddenly polymorphism was within the reach of any virus writer who knew how to leverage a link library.

The current situation is not fundamentally changed from this distant past. While anti-virus vendors no longer use strings of static bytes to detect viruses, the vast majority of viruses are detected using techniques that identify viruses by particular attributes that must be known before detection and removal can take place. Because of this, signature distribution has moved from a monthly event to an ongoing race, with vendors issuing updates via the Internet in real time.

3. Challenges in the solution space

From a description of the problem and solution space, it is fairly easy to see that there are a number of open issues for those interested in virus prevention. Note here the change in focus: anti-virus solutions are about return on investment and risk mitigation, not the complete removal of the virus problem. Thus, from a purely pragmatic worldview, the primary goal of anti-virus software is not ultimately virus detection, but reduction of the cost associated with viruses. While this almost always means detection, it does not have to be detection. This clarification is an important point, as we shall see later.

This clarification aside, there are a number of very unpleasant attacks one can carry out against current anti-virus products. In this section, we will examine two of the attacks we have already observed as opposed to future threats, respecting the long-standing tension within the anti-virus world between informing users of risks and actually arming virus writers with new ideas.

3.1. Speed of spread

Modern malware is fast. In almost the blink of an eye, an infected machine can send infected documents worldwide; worms like SQL.Slammer can become pandemic in just minutes [Moore2003]. Compare this to the virus problem before ubiquitous connectivity. Organizations like The WildList tracked outbreaks around the world, as infection spread from host to host via sneakernet. Those days are gone forever.

Of course, anti-virus software has taken advantage of connectivity too. Suspicious samples can be gathered from host machines quickly and solutions distributed worldwide in real time. There have even been some attempts – most notably the IBM Digital Immune System [White1999] – to automate parts of the solution generation process. However, with outbreak times now being measured in minutes not weeks, it is increasingly improbable that solutions that are reactive can hope to keep up with the virus problem. This was nicely illustrated in [Williamson2003], which showed that the lag between virus outbreak and solution dissemination is an important factor in determining outbreak size.

In addition to this work, we have created a network-aware simulator to study the challenges of solution dissemination in a degraded network environment. Detailed results of our work in this area are to be the subject of a later paper. Funded by a grant from the Cisco Critical Infrastructure

Assurance group, early results indicate that network degradation during an outbreak further weakens the abilities of reactive solutions. While speed of spread is the result of a number of different factors, the result of increased spread speed is very simple: signature-based solutions are becoming increasingly untenable.

3.2. Entry point obfuscation

The next big concern for scanner manufacturers is entry point obfuscation, or EPO. In order to understand why EPO is such a problem for the scanner developers, we must first gain a better understanding of how virus scanners work.

Once, when viruses were very simple, and each replication of a virus "looked" very much like every other, viruses could be detected by scanning files for simple hexadecimal strings. Such scanning, however, was slow and unwieldy – as any file that contained the magic hex signature was detected as infected. Given that most viruses modified the file in such a way that the virus code was located at a predictable position within the file, vendors moved from simplistic bulk scanners to scanners that only searched particular locations in files.

This improvement was rendered useless by the creation of "polymorphic" viruses. Such viruses contain a static code block that is trivially encrypted, and a "new" decryptor is built for each replication. Thus, as the static code is encrypted, the virus scanner is forced to search for the presence of the decryptor… which, as it is dynamically generated, is different for each replication of the virus. The result of this is that while each generation of a polymorphic virus is functionally identical, there may be no single code sequence found in every infected file.

In order to detect such files, anti-virus developers have shifted detection strategies and now heavily rely on tracing of the execution path within scanned objects. During this process, the scanner can determine if the code being executed is viral. This technique has proven very effective, and is significantly faster than bulk-scanning large files. However, its usefulness is predicated on the fact that binary file viruses are executed at the start of the execution process.

EPO viruses attack this assumption directly, embedding the jump to the virus code deep within a target executable. Thus, simply tracing the execution path of an EPO-infected file provides no guarantee that the virus code itself will ever be called. Even if the actual virus code is located at the end of the executable, the scanner is unable to detect its exact location in order to emulate it. Essentially, EPO raises the bar in terms of what is required for static detection, as it removes the ability for a scanner to trace into the virus code with surety.

4. New solutions and the future

Given that we have already demonstrated that reliable virus detection via static analysis is impossible and that current attacks are pushing the envelope in terms of manageability, new solutions to malcode are paramount if we are to create a robust Internet infrastructure. In the remainder of this paper, we will discuss the attributes of these new technologies, and from a requirements analysis position, attempt to work out what effective solutions might look like.

4.1. Speed of response, revisited

Given the speed of spread of current malware, one of the obvious requirements is

that we need a solution that prevents pandemics from erupting. This implies either a solution that is capable of preventing fast attacks outright, or that slows attacks sufficiently that they become amenable to slower, but trusted, reactive techniques.

The "killer" solution to the virus problem has always been a method of stopping viruses that were unknown to the solution provider – that is, to provide generic detection of viruses rather than solutions that rely on prior knowledge of the virus.

Numerous attempts to provide such protection have been investigated by the anti-virus industry. Static heuristic analysis of files has been proposed, either by direct binary analysis, or by runtime simulation. In [Natvig2002] Norman Data Defense describes a system that emulates samples in a virtual environment, allowing viruses to "replicate" safely in a virtual machine. Such a technology is potentially extremely powerful, except for three significant weaknesses.

First, such an approach is incapable of detecting worms that rely on code injection – that is, where the worm code is contained in a data stream.

Second, this approach presupposes that the virus code actually gets to execute. With an EPO virus, this is not trivially the case, and the emulator is not practically able to execute all code paths within the target executable. Thus, EPO can potentially render such approaches useless.

Third, the approach is susceptible to viruses that are able to detect that they are under emulation, and thereby change their execution order to hide their intentions. Once free of the emulator and certified as clean, the virus code can do its job and infect the system. This last point is extremely important in virus prevention; so much so that the general case is explored below.

4.2. Try, try, try again attacks

One of the challenges with developing virus solutions is that it is somewhat like playing "Paper, Rock, Scissors" when your opponent has to move before you choose to. In such a scenario, the player who moves first always loses against best play. Similarly, with current generic solutions a virus writer is free to study the detection mechanism at his or her leisure and then play to win.

This drawback in detection leaves the door wide open to attacks. At the simplest level, an attacker may develop a virus that evades the proactive detection of a particular scanner. At a higher level, a nation state or similar entity could obtain copies of all proactive solutions and develop a single attack that avoids them all. In such a circumstance, proactive solutions become worthless and the network once again becomes susceptible to a crippling attack.

5. Generic virus detection?

Given the challenges faced by solutions that are based upon static analysis techniques, anti-virus researchers have been placing renewed efforts in provided for solutions that allow new viruses to be reliably detected. Indeed, a brief examination of most anti-virus vendors' web sites provides a raft of information that might lead one to think that the problem was well under control.

Unfortunately, a more in-depth analysis of current generic techniques shows that

this is far from the truth: the state of generic virus detection is very weak. In this section, we will show some of the techniques currently used for generic detection of viruses and illustrate their susceptibility to one or both of the attacks mentioned in the preceding section. In no particular order, we will therefore examine Checksummers, the Digital Immune System, Static Heuristic Analysis, Fuzzy Detection, the Norman Sandbox and Behavior Blockers.

5.1. Checksummers

Checksummers had been a vastly underused part of virus countermeasures, although they are somewhat tricky to deploy on a widespread basis. Essentially, a checksummer relies on the fact that a virus must change an object it infects. By creating a cryptographically strong hash of objects before infection, subsequent changes can be easily detected. Unfortunately, this strength of checksummers is also their biggest weakness: they are change detectors, not virus detectors. Furthermore, they rely on the system environment in which they running, making them vulnerable to viruses that implement a high degree of stealth. Finally, they are highly reactive, letting users know that something occurred, but doing little to prevent further infection.

5.2. Static heuristic analysis

Static heuristic analysis is the process whereby a file is searched for certain tell-tale indicators of viral infection. This technique can be useful as infected files are often fairly easily identifiable. Unfortunately, such heuristics fail to both "multiple attempt" attacks and EPO. In the former case, the virus writer can experimentally determine which factors trigger a heuristic response; in the latter, one simply needs to recognize that the entry point of the virus code must be located for meaningful static analysis to occur.

5.3. Fuzzy detection

Most computer viruses are not entirely new; rather, a virus writer often takes an existing virus and modifies it in some way. Given the similarity between many different virus variants there are some advantages to imprecisely detecting particular viruses – that is, detecting large groups of viruses and potential variants with one signature provides some level of proactive detection of new viruses based upon the original sample.

However while imprecise detection sounds attractive, it can pose some difficulties with respect to disinfection. Furthermore, such an approach is pragmatic, but not effective against a targeted attack: as the detection algorithm can be obtained by the virus writer by simple trial and error, creating a variant that goes undetected is not difficult. Thus, while fuzzy detection is a useful aid in prevention, we should not consider it to be in any way a cure all.

5.4. The Norman Sandbox

One highly effective solution to proactive virus detection is to run a file in a virtual environment, and see what changes it makes on the system. Such approaches are becoming increasingly attractive as processors become faster, as it is now feasible to carry out such simulation in real time.

Furthermore, if one takes the changes and simulates the result of those changes, false positives can be dramatically reduced, as only files that have some properties of self-replication can be tagged. One excellent – and highly effective – example of such a

system is found in [Natvig2002], where a fully functional virtual environment is demonstrated.

As we have previously discussed above, however, this approach is susceptible to attacks based upon EPO and against targeted attacks. By detecting that one is inside the sandbox, it is possible to avoid malicious behavior until the file has been determined to be clean. Similarly, if the malicious behavior occurs deep within the program of study, the Sandbox may never create the conditions that cause it to be executed. Thus while the Sandbox is an excellent technique, it is relatively easy to target.

5.5. The Digital Immune System

One approach that hopes to combine the best of generic detection with the robustness of virus-specific techniques is the Digital Immune System. This technology, first developed by IBM [White1999] and later acquired by Symantec [Symantec2001] seeks to provide innate immunity from new viruses by automatically generating signatures from infected objects. These signatures are then distributed to uninfected machines. The concept is that by combining generic detection with innate immunity, outbreaks can be handled automatically in real time.

While this approach sounds highly attractive, there are several issues that limit its effectiveness. Consider a typical detection by the immune system. Initially, generic techniques are used to identify objects that are suspicious. These objects are sent to a central analysis center automatically. Here, they are analyzed and replicated; a signature is generated and distributed first to the infected machine and then to all machines, providing innate immunity.

Unfortunately, this cycle limits the immune system in a number of ways. First, the speed of signature distribution required is probably untenable – especially in the face of a network-based virus attack that causes significant network congestion. Second, the immune system is limited by its front-end generic techniques; thus, it is not a method of generic virus detection per se, but a method for turning generic detection into virus-specific detection.

5.6. Behavior blockers and analyzers

One approach which is receiving more attention is the concept of runtime behavioral analysis. In this approach, each process or process tree on the machine is monitored for evidence that it is malicious in nature. Should a process be detected as malicious, it can be stopped.

Historically, such approaches have fallen prey to a number of different weaknesses. Most importantly, the challenge of false-positives has proven difficult to overcome. It is difficult, without hardcoding of rules, to determine that a “deltree” command is allowable but that hierarchical file deletion by a program may not be. Additionally, by the time a process is determined to be malicious, it is likely that some changes have already been made to the system, leaving the machine in an unknown and untrusted state. Thus, behavior based systems have historically been more virus detection than virus prevention, as manual work may be required for system recovery.

Fortunately, advances in processing power have made this approach more tenable, as improvements are available for the detection and removal algorithm. Such a system has been developed at Florida Institute of Technology and we use it below to introduce some important concepts in the future of virus detection.

6. Gatekeeper

Consider a behavior-driven virus prevention system. For such a system, the "faster" a virus is – that is, the more focused its propagation mechanism – the easier it is to detect it behaviorally. Thus, in the age of rapid malware, behaviorally systems are, at face value, attractive. However, historically such systems have not been very successful. Part of the challenge is that there is a tension between detection and prevention. In order to score well in detection, the longer the program of study runs the more reliable detection is, and the more behaviors can be monitored. Thus, the later detection occurs, the more reliable it is likely to be. Conversely, there is a need to prevent damage, so early detection is highly preferable.

One way to resolve this tension is to use a behavior monitor that tracks and is capable of undoing actions on the host machine. We have implemented such a solution – known as Gatekeeper [Ford2003] – at Florida Institute of Technology. Results so far have been highly encouraging – detection rates of >90% of new In the Wild viruses have been achieved with very low false positives. Behavioral systems are not fooled by EPO and most methods of avoiding behavioral systems rely on slowing the rate of spread dramatically. This is desirable, as if spread rate can be reduced, effective reactive countermeasures can be put in place.

Furthermore, there is a fairly convincing argument that such a system does not need to be perfect in order to be effective. For example, consider a system that adds increasing amounts of latency to processes that are deemed suspicious – such a system is described in [Somayaji2002]. Despite the fact that a virus within such a system may successfully infect another machine, if suppression is above a certain key threshold, pandemic spread can be avoided, providing time for reactive solutions to be put in place. Essentially, a behavioral system need not be perfect in order to be effective.

7. The future of virus detection

While behavioral systems may well be the future, so far they only address one part of the problem space. Even if reliable detection of rapid malware can be accomplished, how can such a system defend itself from highly-motivated attackers? The critical weakness seems to be that an attacker can try as many times as he likes with zero risk.

One way of defending this attack is to distribute the solution more widely – that is, that the behavior of the entire system of protection cannot be easily inferred from a study of just one endpoint. Such a requirement is key if we are to defend our systems from a potentially catastrophic worm outbreak. Furthermore, focus needs to move from simply protecting systems to protecting the infrastructure that those systems run on. As we read in [Griffin2003], the SQL.Slammer worm had a profound impact on BGP – this at some level measures the stability of the Internet. If such a benign worm can cause such widespread instability, one can only speculate as to the impact of a worm that was designed to cause as much network disruption as possible.

Based upon the rapidly changing threat profile, it is difficult to be specific about the future of virus detection, but the high-level requirements are clear. Generic suppression of viruses – be that via detection or simply reduction of spread rate – must be applied at a network level in order to protect the routes by which innate immunity can be disseminated.

Furthermore, solutions will become increasingly distributed, providing protection from well-motivated attackers who, through reverse engineering of detection algorithms, desire to cause catastrophic loss of connectivity and communications on a global scale.

8. Conclusion

In this article, the state of the art in virus detection has been examined and shortcomings of our current defense scheme highlighted. Based upon the requirements for a robust solution, some possibilities for the future of generic virus detection have been discussed, and a challenge issued to researchers: provide for reliable generic virus detection without allowing an attacker who knows the algorithm and deployment details to evade detection.

Based upon these requirements, it seems likely that the next generation of anti-virus products will be distributed in nature. Furthermore, it is possible that focus will move away from perfect endpoint protection – something that is demonstrable impossible – to a more holistic view. By taking such an approach, it should be possible to prevent catastrophic network failures, and provide a robust and reliable foundation for the electronic interactions of tomorrow. One thing is for sure: current solutions are not up to the task. We are well advised to reach out for new countermeasures before the current inadequacies become not a matter of discussion but history.

Bibliography

Ford 2004, Ford R.A, Thompson H, The Future of Proactive virus Prevention. From the proceedings of the 2004 EICAR Conference 2004.

Griffin 2002, Griffin T, and Zhuoqing M, Internet Routing Streams, Workshop on the Management and Processing of Data Streams 2003.

Moore 2003, Moore D, Paxson V, Savage S, Shannon C, Staniford S, and Weaver N, The Spread of the Sapphire/Slammer Worm. CAIDA. Available online at http://www.caida.org/outreach/papers/2003/sapphire/sapphire.html (2003).

Natvig 2002, Natvig K, Sandbox II: Internet, From the Proceedings of the Virus Bulletin Conference, 2002.

Somayaji 2002, Somayaji, A. Operating System Stability and Security Through Process Homeostasis, Ph.D. Dissertation, University of New Mexico, July 2002.

Symantec 2001, Symantec, The Digital Immune System, Symantec Technical Brief. Available online at http://securityresponse.symantec.com/avcenter/reference/dis.tech.brief.pdf. July 2001.

White 1999, White S.R, Swimmer M, Pring E, Arnold W, Chess D, and Morar J, Anatomy of a Commercial-Grade Immune System. International Virus Bulletin Conference, Vancouver, Canada (Sept. 28-30, 1999).

Williamson 2003, Williamson M, Leveille J, An Epidemiological Model of Virus Spread and Cleanup. From the Proceedings of the Virus Bulletin Conference, 2003.