Global Information Assurance Certification Paper

Poly (morphic) want a server…or Runaway worm.

by

Michael Desrosiers

November 15, 2001

Abstract

I could have easily named this paper future exploits or something lame along those lines, but I find this subject to be not only fascinating but frightening at the same time from a security professionals point of view. With the explosion of high bandwidth Internet connections for consumers, and the implementation of tens of thousands of SOHO (small office, home office) networks, the possibilities for more far reaching attacks and its consequences are not a question of if but of how damaging they will be. I will be examining the concept of worm propagation, and what I see the future worm to look like, out in the wild. I also want to address what steps can be taken to limit its effectiveness. As I make my way through the SANS curriculum, I will be trying to gain the specialized knowledge, in understanding what to look for in preventing the attacks of the future. I also will be re-engineering worms and or malware to better understand not only how they are progressing from being a mail distributed mechanism, to becoming a more pseudorandom generated, multi-headed monster that will be unleashed without warning onto the Internet. The notion here is that we have to go back to our roots as an entity. I am currently dusting off my assembly language textbooks from college. My belief is that we will see more damaging exploits written in assembly language simply because it can live on the first layer of the OSI stack or the physical portion of it, and be delivered and spread much more efficiently. This exploit using binary numbers, which is the state of a digital circuit being either in a *1* or *0* state, could then manifest itself into the computers resident memory or physical infrastructure. Also as the security products that are being written, focus more on the higher-level application and operating system languages, microcode can be the preferred method to transport the worms, and allow a greater infestation footprint. This is why Runaway is a good choice to describe what the future exploit holds. So strap yourself in, because fascination has won out!

Preamble

There was a time not so long ago, where life was not only simpler, but had a more straight forward list of both checks and balances. If you wanted to purchase an item or correspond with someone, you had to solicit that person or company. In today's world there is so much that is unsolicited, it is sometimes almost comical. I have a friend, who is happily married by the way, who last year was the unwitting participant in spreading the Melissa Worm throughout his company. It wasn't that he was reckless, but a guinea pig in a kind of weird techno-experiment. Now, you might ask what does this have to do with worms, and how they will be delivered? The answer is very simple. Humans by

nature are a very curious bunch, so if the delivery mechanism is innocuous, the more wide spread the exploit. Because the mechanism of B to C (business to consumer) is grossly behind the curve in regards to security, exploits will have a larger audience to choose from. The telecommuter is now a common employee within a corporation. More and more people are working from home. And here lies the extreme vulnerability that this represents. While Joseph Blade is taking care of a last minute marketing pitch on-line through a VPN connection, little Joey has left IM running with a Sub-Seven Trojan running in the background. Or the user loves music, and downloads mp3's from a web site that is being run by Hfury or perfect.br , without ever knowing malicious code is being delivered with it. The future runaway worm will have the luxary of being implemented without user intervention or knowledge that it is even present. With this breeding ground to launch more malicious attacks, the cracker will have the ability to hide within the confines of a private individuals environment as opposed to the corporate one he/she now must penetrate. This will allow for the access to the resources needed to make the worm, possible. It also will have the ability to propagate without detection due to its size and speed of execution. The challenge in implementing this worm successfully will be in the way the scanning package can be pre–distributed to the hundreds of thousands of systems that the worm will need to run optimally. What this means is that it will be written in assembly language, but it cannot be run or recompiled on different types of computers. The key will be in how these different CPU based micro programs or interpreters within the machine instruction sets can execute the code. A thought that keeps entering my head is in firmware or microcode updates that hardware manufacturers routinely asks their customers to run against their hardware platforms. How many people religiously run md5 and an A/V program against everything that they load into their computers? What better way to get access to hundreds of thousands of both servers and desktops, then from an update from a trusted manufacturer? How about the millions of freely distributed cd of AOL that is in circulation? This could possibly become the most efficient avenue of delivery in future.

Mommy I'm scared

Worms that are active will be able to "learn" and change characteristics as they spread from system to system. We are already seeing smart phones, these are cellular phones that have both PDA and telco capabilities, being used. Smart appliances are just now starting to be seen on the horizon. It will not be to far into the future, when you will be able to start a load of laundry, plan and prepare supper for the family and have your refrigerator verify your grocery list, through the Internet. Although the address space range of IPV6 will make this more difficult, it will provide the attacker with the numbers of additional compromised hosts it will need to do the upfront scanning of networks. With this next generation of IP devices also allows for attacks that can become multi-threaded which will address the issue of the Runaway worm becoming saturated, our in other words, trying to infect and scan an already compromised network. With the millions of new zombies to chose from, and the continued growth of both personal and business web servers, the pre-scanning and recon work can all be done as a first stage. With the ability to have a pre-compiled vulnerability list up front, this will allow the worm

itself to be a more stream lined and malicious one. If the Runaway worm can be broken down and staged, the worm can then carry a pre-disposed vulnerability list as part of its payload as it moves throughout the Internet. This will allow child segments to be left behind as processes within the breached networks. Infection rates now will be a matter of minutes with the goal of seconds very attainable as the quantity just as much as the quality of the worm is developed. If you venture out to some of the more select malware web sites, you will see the so-called blueprints of a multi-pronged worm attack just waiting to be unleashed. It is aimed solely at causing the most destruction in the shortest amount of time.

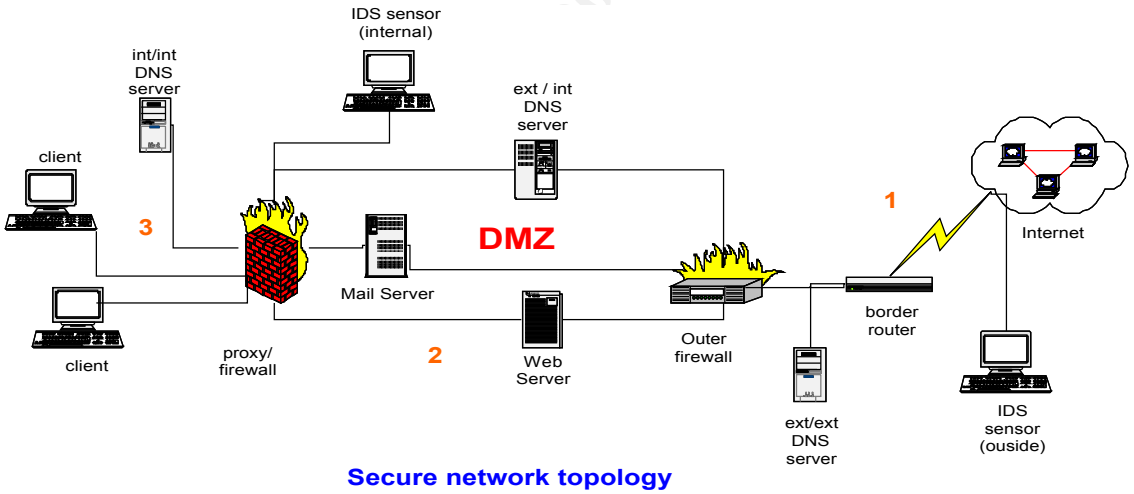
Now that the delivery base has been substantially expanded, here comes the truly devious portion of our exploit. The Runaway will have the ability to be executed in a multi-plat formed environment with all vendors invited to the party. By finding a vulnerability that is not published about a DNS buffer overflow or finding an almost daily new format string attack and writing the major component of the worm to exploit this, you will be giving the security community its worse possible nightmare. A true **zero-day** worm. Using K2's tool, ADMutate to continuously change the attack characteristics of the Runaway worm, it could be days or weeks before it is even discovered. By mutating, it will avoid IDS's pattern matching or predefined signatures and would generate new and unrecognizable signatures as it spreads throughout the network. K2 claims that this technique, polymorphism coding, will in all intents and purposes allow the worm to infiltrate and bypass existing IDS sensors, because the sensors will be trying to trap on certain shell code or string that matches a pattern of malicious behavior. It also will be encapsulated so as to provide another layer of concealment. This ability to evolve will allow the exploit to expose individual vulnerabilities within that server's operating system in that particular environment. Since the most productive way to spread the worm is by infiltrating the corporate email server infrastructure, it shall use one of a number of Exchange mail server vulnerabilities, to spread the exploit in the quickest way possible. The Runaway worm could then use a IIS Unicode exploit to gain access to the web server, then run a microcode attack to crash the IBM z900 database mainframe, then start a sadmin rpc exploit to go after the Solaris web apps server and then finally attack the Red Hat DNS servers to Ramen through rpc.statd or lprd. It will also leave a copy of tini, a port listener that lives in the data stack on *nix flavored servers as to allow future access through a backdoor. Also the worm, if discovered and tampered with, will launch a script that will format or dd the fixed disks and a script to re-flash the BIOS, rendering the server incapacitated. The techniques to hide this type of hack are present today, and do not take much skill to use. There are scripts and kits that can be found on both white and black hat sites throughout the Internet, to both gain access to networks, and to completely hide ones tracks while in that network.

By making the exploit multi-dimensional, this will prevent any one defense mechanism to thwart its functionality. If one door is closed, it will find a way to open a window. This is what causes the security community to stay up at night, thinking about the numerous ways that one needs to defend this sort of attack. This also opens the door to the

possibility of true data altering, where as internal sensors within the network structure do not register that the network has been compromised. More so then removing or destroying information, altering data would be what would frighten me the most. The consequences of this could be limitless. Imagine the effect this would have on consumer confidence, which in turn could have a devastating effect on the economy. When is the last time you questioned a bank statement? Now consider multiplying this by millions, while also taking into account, possibly your electric bill or that direct deposit pay statement that was emailed to you. If a society that depends on information technology cannot trust this information, it will start to collapse from within Code Red and Nimda were just the beginning or tip of the iceberg, compared to what is being discussed within the community. The ability for this to be unleashed currently is strictly due to the skill sets needed to use ADMutate to its potential. Once the ability to mutate worms within an environment is attained, the gloves will truly be off.

So what now Batman?

There are counter-measures that we can take that will greatly reduce our risks. Here is an example of a secure network and its hardware components. I have broken it down to 3 sections that will allow me to better explain some of the ways we can provide more thorough levels of security.



Secure network topology

What this diagram is in effect saying is to partition or segment your network. That to better protect the private internal network, issues like a internal DNS server should never be able to do zone transfers to the outside world or and if possible their should be an internal application proxy server to break the connection to the Internet. In section 1, the ability for border routers to do a more efficient job of egress filtering is one of those. While IP checking and fragment offset is a fairly common practice, it still is not possible

to analyze and control data with filtering due to its limitations in understanding the contents of a particular service. Since the border routers are on the front lines, proper ACL's offer a significant improvement in preemptive detection. We also want to keep DNS queries from the Internet from entering our trusted network, which in turn will prevent DNS cache poisoning attacks. We also must do a better job of configuring the outer firewalls. Because firewalls now come with nice, pretty GUI's end users are more apt to implement the configuration of these said firewalls. That is not the problem. The problem lies in the use of the defaults that do not seem to follow known logic. Even a market leader like Checkpoint FireWall-1 leaves port 53(DNS) in both UDP/TCP protocols, open by default. They along with other firewall vendors also allow by default, ACK bit enabled TCP scans through them. This is not to say, that firewalls are not an integral part of security infrastructures, just that we should not depend on them to be a catchall. The idea that any one these defensive mechanisms will protect a network, is a battle we seem to be losing at the moment. I have had too many personal experiences to name that start out with, " But, we have a firewall…." Also we have to better define logging of these devices, simply to make them log to a more centralized console. Using the path to least privilege theory, as a default rule, we must make sure to deny all as the blanket ruleset, and allow as needed. Also in regards to firewalls, how if these future worms will exploit web servers and web browsers, will a network firewall detect outbound http requests from its internal authorized clients applications as malicious behavior? There is a great article on this by Bob Sundling as far as the false sense that personal firewalls give, at the following web site: http://tooleaky.zensoft.com/. Crackers rely on unlatched software and bad configurations to spread the malware that they produce. We also want to know who or what is banging on our door. A nice preemptive first indicator is the use of an IDS (Intrusion Detection System), like Snort on the WAN interface of our router. This could let us know if port scanning activity to our network, is on the increase. The short comings of this right now is IDS signatures number in the hundreds, as opposed to A/V engines which have over 50,000 signatures at its disposal. As more and more signatures get written, the sensors themselves will become more sensitive to detecting these malicious exploits. In regards to section 2, the business critical servers must be hardened. The DMZ and its servers must be built with a so-called "**golden image**" or clean operating system load. This step alone would provide the biggest bang for the buck, as far as exposure to known exploits. By taking out the training wheels niche, you will be better suited to concentrate your efforts on the true professional cracker. Most exploits happen using known vulnerabilities that are 1 to 3 years old. Before it is even given an IP and wired, it should have a battery of tests that it must go through. The image load itself should be a minimal install to support that application. Not only should the O/S be up to date, as far as patches are concerned, but the services or ports open for that server should be strictly limited. As an example, on the web server port 80 for http and port 8080 for https should be the only ones that are available to the outside world. Having a switched network for these critical servers also allows for a better defense against packet sniffers. In parallel with the IDS sensor on the Internet side, the Intranet sensor will provide information as to any destructive behavior in the DMZ. The use of HIDS or a host-based intrusion detention system is also a necessary

step. In the case of AIDE (Advanced Intrusion Detection Environment), this provides not only host based intrusion package, but more importantly, a file integrity database engine. The reason why this is a more comprehensive bundle is that it allows for an array of different cryptographic checksum algorithms to be used to verify files. When having to use simple commands like ls or netstat or who, it can come handy knowing that AIDE can provide clean binaries. Also these same binaries should be built into a floppy or CD/R incident response kit. As in the case of sending syslog files to a remote server, AIDE's binaries and database should be either written to an external server or media. In section 3, one of the most important that users can do is encryption. I am a PGP disciple and strongly advise the encryption of not only email, but of hard drive file encryption. Protocols such as rlogin and telnet should be removed and replaced by current versions of ssh. The end user community has to know that there are limitations on the network. This plays into the creation and implementation of a sound security policy and procedures document. Up to the minute A/V client and server signatures files uploads are also extremely critical. Also the end users or clients must be provided with accurate and interactive training that will allow them to become a willing participant in the success of this policy.

Lights out kids

So there you have it, a look into the future, and it's not a bright one either. Once again, due diligence is the key. If we demand that the software manufacturers provide us with secure applications and operating systems instead of products riddled with security holes, this would help us immensely in obtaining our common goal. Until issues, like *nix operating systems coming with hundreds of ports and services enabled or Microsoft's IIS Unicode directory traversal exploits are addressed, base system image installations will be at the core of our problem. It must start with the software companies being more aware of security when it releases new code. Ease of use and secure functionality has to be weighted in a more balanced way. System administrators and more importantly corporate management, has to start prioritizing security as a strategic goal. The current, " Get the application into production on time no matter what", mentality must end. This includes a more uniformed way of patching the operating systems, and with developing applications with non-executable data stacks. The IT infrastructure must be built on security by layers with a daily administrative procedure to review updates and audit logs. Most companies still try to have their security handled by overworked system administrators, instead of establishing a separate department that would make security it's only responsibility. The ability for runaways to replica will be greatly reduced if some of these steps are taken. Until security is viewed as a necessity and not a value added package, we will be fighting a losing battle. The existing threats that are out there are to be taken very seriously, if not, our future will be filled with pain and phone calls that end, " I'll be home sometime tomorrow…." We are so dependant on technology, that we must take all allowable precautions to safeguard it. Not only is a poorly built and maintained network a liability to that corporation, but also it is a liability to the Internet as a whole. The name Runaway worm was derived from the notion that upon its release, it will become this chaotic, destructible exploit, which will only stop when it has reached its

logical apex. Lets hope that we are prepared for its arrival.

References

Stuart Staniford, Gary Grim and Roelof Jonkman " Flash Worms: Thirty Seconds to Infect the Internet."
URL: http://www.silicondefense.com/flash/

Nicholas C. Weaver " Warhol Worms: The Potential for Very Fast Internet Plagues."
URL: http://www.cs.berkeley.edu/~nweaver/warhol.html

Edward Skoudis, On the Cutting Edge " The Year of the Worm."
URL: http://www.infosecuritymag.com/articles/september01/departments_news.shtml

SANS Institute Incidents.org
URL: http://incidents.org/

Bob Sundling " Why your firewall sucks :-)."
URL: http://tooleaky.zensoft.com

Jeff Duntemann " Assembly Language Step by Step, Second Edition, Programming with DOS and Linux." Wiley Computer Publishing 2000

Security Wire Digest " Dangerous I.e. Vulnerability Could Open PC's To Malware."
URL: www.infosecuritymag.com/digest/2001/04-02-01.shtml

Chris Quirke " About Malware."
URL: http://users.iafrica.com/c/cq/cquirke

Jay Lyman " Why Are There No Rich Hackers."
URL: http://newsfactor.com/perl/story/14460.html

Martin W Murhammer, Kok-Keong Lee, Payam Motallebi, Paolo Borghi and Karl Wozabal " IP Network Design Guide."
URL: http://www.redbooks.ibm.com/

Rik Farrow " musings."
URL: http://www.usenix.org/publications/login/2000-6/features/musings.html

Nick Farrell " IBM bottles up mainframe bug"
URL: http://vnunet.com/News/1126893