Teachings of Malware Exposed to an Uneducated Society

Christopher Sigouin

CIS-1150 Holland College

**Introduction**

“Know your enemy, know yourself” (Sun Tzu, 2014) is quoted by Sun Tzu, a Chinese general military strategist who believed that if you know yourself and your enemy you can be victorious without worry. In today’s cyberspace there are multitudes of digital enemies lurking the corridors of its information highways. They are commonly known as malware. Defined, malware is short for “malicious software” that is created for the intended purposes of collecting private data, accessing a system without authorization, or just disturbing the normal routines of a computer system. Its name is generalized to cover the many forms that fall under it such as worms, trojans, randsomware, keyloggers and much more. The first personal computer (PC) based piece of malware was called Brain.A and developed by two brothers. Their non-malicious intent was to prove that the PC based system was not immune to software infection. The program’s mission was to copy itself into the boot sector of floppy disks and when used, would in turn infect the PC systems drive and continue to infect any further floppy disks used with it. After its creation, malware with more destructive capabilities began to come into existence (Milošević, 2013)

Malware is growing and changing every day from the mischievous programs we may think they are to full blown crime capable code. The increasing threat has made security and education experts look at alternatives to thwarting the chaos. Much like the beliefs of Sun Tzu, they also believe that understanding the enemy is the key to its demise. Educational institutions that provide courses on computer science pump out students every year with the knowledge to develop proper working, ethical, and memory efficient applications. Yet many institutions fail to deliver any type of course on malware behavior or design. Past convictions for teaching anything malware related was believed as forbidden, much like performing autopsies on cadavers was forbidden before the advancement of medicine (Ledin, 2011). An effective method of dealing with malware is to teach students how to design and implement malware practices. If we submit to ignorance then our enemy will surely grow and overcome any defense we put into place. It is time to fight back.

**Become the Infection**

Courses in computer science get small amounts of coverage on the study of malware. At most these courses get history lessons or personal encounters with infected code. Anything to do with the development or design of malicious software is seldom taught. Students of the trade need to be able to dissect today’s cyber bugs and be able to code their own versions for the same reason that other professions study different areas. To gain insight and knowledge on those aspects. Take for example all of the biological diseases and infections studied in the world of biology. Without research in those areas we would fail to see how they function or behave and therefore, be unable to develop any sort of remedy. Learning to spot and evaluate malware would lead to the removal and immobilization of such pests (Ledin, 2011).

“We cannot afford to wait for the computer equivalent of 9/11 to learn what the bad guys were doing. Not teaching viruses and worms is a prescription for disaster” (Wasp, 2008, p. 16)

Students researching with the philosophies of treacherous software could not do so without any ethical training to support them. Morals can be challenged when dealing with knowledge of this type of programming. While this research is conducted and vulnerabilities are found, students are reminded of the reason why they are proceeding to exploit computer systems. This reinforces the very reasons for learning and is more valuable to them than any list of conduct provided (Sullins, 2014).

Learning how a piece of software grants unauthorized access would give you the knowledge to know how to either build software to defend against the attack or at least provide the know how to better secure the exploit itself. A criminal profiler, by definition, is someone who studies criminal behavior for the purpose of capturing them. The analogy can also work for examining malware in the purpose of building better defensive strategies.

**Build a Better Defense**

Software written by human beings is obviously prove to human error. Flaws and exploits will be found by either the programmer, Quality Assurance (QA) testers, or in the worst case scenario by someone using malware to pinpoint a weak area in its architecture. Take anti-virus (or anti-malware) software for example. An application that is created to prevent, detect, and remove malicious code from the user’s PC (Antivirus, 2014). These applications are developed by humans like any piece of software, thus leaving the possibility that even anti-malware programs can be attacked and/or infected.

In general so much faith is placed into their ability to protect our private data that we don’t realize it can be a target as well. The level of unreliability with an anti-malware application grows with how complicated they become. Dealing with so many file types like exe, zip and com certainly add to this because of the file types own make up. The main categories that are exceptionally vulnerable with anti-malware are local privilege escalation, active-x related control, engine based, and the management interface that is used for overseeing the program. A more specific view on these involves a piece of software called Kaspersky Antivirus and its known design vulnerabilities with ActiveX. One such vulnerability involves a method called “StartUploading()” that is used to bring files from an online source to the clients PC. When exploited it can be used to access files directly from his/her PC and send them to wherever the attacker wishes. Visitors are infected by a website hosted by the assailant that begins the exploit process. Another case made use of a remote management system with the software known as Symantec Antivirus. A simple but manipulated command like “COM\_FOWARD\_LOG“ would be sent to the system running the software and cause what is known as a stack based buffer overflow. This would leak data outside the applications designated areas in memory essentially to be used by the malicious party (Xue, 2008).

These types of abuse used on anti-malware software can be avoided or at the very least included into the design phase of victimized applications. Developers with knowledge on how an attack could cripple systems wield great power. When used properly it could end our feuds and inspire generations.

**Know how to Deal with Malware**

Not to say that spreading practices of malware design is the final solution, but it can certainly be a great start with one. The goal would be for the average learner to respond with an increased efficiency in the face of threat. Education is more useful when it is explained why you are learning about the subject, familiar examples, and of course the lessons that need to be addressed (Abrams, 2008).

Some systems exist that are vitally important and contain no protection what so ever. Because of this, it is with great importance that malware is studied as soon as it is identified. A lot of the behavior can be studied under a secure environment that is specifically designed to analyze the way malware works. Though some more theory based sections like the removal procedure and what kind of actions it is capable of need to be studied by a living being to tap into its true potential. A certain genre of malicious code carries a unique ability to leave an identifier on computer systems. It does this to notify itself that it has already compromised that system and does not need to execute its primary function. Knowing this behavior of the malware suggests that an identifier could be developed and essentially be placed on a non-infected system to portray a false positive. The malware seeing its marker would obviously not infect the identified system (Wichmann & Elmar, 2012).

It is a wholly new thing—a smart virus with a grudge—evolving, self-aware, self-educating, craftily fulﬁlling its mission (Ledin, 2011).

With ethical research and development of malware at the fore front, it can be understood that we stand a better chance at protecting ourselves. Utilizing strengths and weaknesses derived from malware constructs the tools required to deal with them. At the University of Calgary students are provided assignments to develop spyware that is designed to key log a user’s PC. It would begin with the spyware using a startup hook that causes the malware to load what the operating system of the PC turns on. At that point it would direct the user’s browser to a different homepage and initiate the keylogging. Private data pertaining to the affected user would be then gathered and sent back to the bug creators. After the development of these spyware applications students would trade projects and write anti-malware designs to counteract their effects (Aycock, 2006). Future generations of computer science graduates acquiring these skills is absolutely necessary. It is a duty of higher learning institutions to spread this knowledge so that we can better protect ourselves.

**Conclusion**

They are dangerous and unforgiving designs to teach. In the wrong hands can potentially cause more unnecessary criminal action and harm that society does not need. These are reasons why education with malware is not as widespread as it should be. Fear of what may come doubts the minds of individuals who have the power to start the journey. Do we continue to allow software development to leave its pre-code stages without reviewing alternative possibilities for causes of infection by an outside force? We provide knowledge on topics like chemistry and physics to all who are willing to learn. Can they not utilize skills from those fields of study and forge a dark force to unleash it onto the world? Designers of malicious code are becoming better and smarter daily. Anti-malware products are barely holding the front lines. We have to fight fire with fire.

“Knowledge is Power” (Bacon, 2014)

**References**

Abrams, R., & Harley, D. (2008). People patching: is User Education of any Use at all. *In AVAR Conference Proceedings.*

Antivirus Software. (n.d.). In Wikipedia. Retrieved October 23, 2014 from http://en.wikipedia.org/wiki/Antivirus\_software

Aycock, J. (2006). Teaching spam and spyware at the university of c@ 1g4ry. *In CEAS.*

Bacon F. (n.d.). In Wikiquote. Retrieved October 23, 2014 from <http://en.wikiquote.org/wiki/Francis_Bacon>

Ledin, George Jr. (2005). Not teaching viruses and worms is harmful. *Communications of the ACM, 48*(1), 144.

Ledin, George Jr. (2011). Catch me if you can. *Communications of the ACM, 54*(3), 112-113

Ledin, George Jr. (2011). The growing harm of not teaching malware. *Communications of the ACM, 54*(2), 32-34.

Milošević, N. (2013). History of malware. *arXiv preprint arXiv*:1302.5392.

Sullins, J. P. (2014). A Case Study in Malware Research Ethics Education.

Sun Tzu. (n.d.). In Wikiquote. Retrieved October 23, 2014 from http://en.wikiquote.org/wiki/Sun\_Tzu

Wasp, J. (2008). The dark world of computer security. *Sonoma Insights, Fall*, 14-16.

Wichmann, Andre & Elmar, Gerhards-Padilla. (2012). Using infection markers as a vaccine against malware attacks. *In: IEEE International Conference on Green Computing and Communications (GreenCom)*, 737-742, 20-23

Xue, F. (2008). Attacking antivirus. *In Black Hat Europe Conference*.