Ben Rain

Dr. Phil Lunsford

ICTN 4040

04/13/2015

Mitigation Techniques and Practical Implementation of EMET

As the size of the internet and number of devices both on the personal and enterprise level increases, so does the number of cyber criminals. With so many devices, and so many vulnerabilities within these networks, how do we keep information safe? The answer is a endless amount of applications, physical devices, intrusion detection systems and the list goes on. Wouldn't it be nice if there was an application already loaded on most operating systems that could give a substantial amount of protection for our vulnerable programs but was not hard to implement?

The answer is EMET. EMET stands for Enhanced Mitigation Experience Toolkit and is a free Windows-based security tool that adds security defenses by using specific mitigation techniques to prevent exploits related to memory corruption. EMET defends against memory corruption vulnerabilities; EMET is easily implemented; and EMET has many practical uses.

The simple answer to how EMET defends against malware is that it stops attackers who use web, email, or other tools to convince a user to open any document such as a PDF, excel sheet, etc. Once the corrupted file is opened, the improperly formatted file causes the computer to run the attacker's malicious code from within the file. EMET comes in by using one of the mitigation techniques that EMET is constantly using per application and inhibits the malicious code from running inside the data file. Before defense is actually needed though there are several impressive specific mitigation techniques that EMET employs.

The first and possibly most used mitigation technique that is used by EMET is called Structured Exception Handler Overwrite Protection (SEHOP). SEHOP protects against the most common technique for exploiting stack overflows in Windows. Without EMET, the attacker can overwrite the handler pointer of an exception record on the stack. Once the exception happens, the operating system will walk the exception chain and call all of the handlers on each exception record. Because the attacker controls the records, the operation system will go to wherever the attacker wants granting the attacker control of the flow of execution. EMET stops this by validating the exception record chain before the operating system can call on any exception handlers. If the chain is in fact corrupted, EMET will terminate the process without calling any handlers. (Microsoft 2014). This feature has been used with EMET since Windows Vista Service Pack 1.

Another common mitigation technique is Data Execution Prevention (DEP). Security Research and Defense describes DEP as "a hardware and software solution for stopping the execution of code from pages of memory that are not explicitly marked as executable." Before DEP, an memory exploit could be run from the memory allocated without memory protection constant set. In other words, if a page of memory were allocated using a memory protection constant and was read-write, the file could still have code executed through it. DEP is capable of functioning in two modes, Software-enforced and Hardware-enforced. Software-enforced DEP is used on machines that cannot support true non-executable pages due to their inadequate hardware support. Hardware-enforced DEP on the other hand is a superset of Software-enforced DEP and is used on hardware that supports marking pages as non-executable (Uninformed- Vol 2). Although it is a great preventative tool, DEP can be maneuvered around by using techniques such as the return-to-libc that exploits the Hardware-enforced mode of DEP.

Two similar mitigation techniques that defend against attackers placing copies of their shellcode in as many memory locations as possible are called Heapspray Allocations and Null Page Allocation. EMET implements Heapspray Allocation so that commonly attackers using these commonly controlled pages will not be able to due to EMET pre-allocating the pages. This will cause making a copy of the shellcode to fail when the attackers are trying to take control of the specific pages. Null Page Allocation works in a similar way to Heapspray but is designed to prevent null dereference issues in user mode. The attacks that Null Page Allocation defends against are done by an attacker intentionally triggering a null pointer dereference. The reason for doing this is to reveal debugging information that could be used for subsequent attacks.

To help defend against return oriented programming exploits, EMET implements a mitigation technique called Mandatory Address Space Layout Randomization (ASLR). As mentioned before. DEP can be bypassed and one of the techniques used to do this is ROP. It predicts the mapping of dynamic link libraries. Sotirov (2008) explains that ASLR fights against these bypass techniques by randomizing the addresses where modules are loaded. This helps prevent the intruder from leveraging data at locations that the attacker may have attempted as one of their predictable locations. All modules must use a compile time flag to opt into this. A related but slightly different technique is Bottom Up ASLR. It uses the base addresses of the bottom-up allocations such as heaps and stacks. This technique will make future memory allocation less predictable along with providing images that have been randomized with ASLR an amount of entropy. A note to make about Mandatory ASLR is that Windows 8 and newer versions of Windows will not use this mitigation if the native enforced ASLR provided by the operating system has already been activated for an application.

Export Address Table Access Filtering (EAF) is used when malicious shellcode wants to actually do something to whatever system it is on. Vlaszaty & Rohani (2014) say in order for the attacker to do something useful on the system, the code needs to access Window's APIs, which stands for application programming interfaces. For an attacker to get to an API, however, the shellcode must find the address where the API is located. Attackers will then have their malicious code run through the Export-Address-Table (EAT) of all loaded modules and look for specific modules that contain useful APIs such as kernel32.dll, ntdll.dll, or kernelbase.dll. EMET protects against this by filtering access to Export-Address-Table so the lookups for modules can be blocked. EMET also offers Export Address Table Access Filtering Plus (EAF+), which is an extension of EAF that can be used independently or in tandem with EAF.

EMET can also specifically block modules or plugins within target applications using Attack Surface Reduction (ASR). ASR can be configured to prevent everything from flash plugin being run on a Microsoft suite application to java not being able to run on an internet browser. ASR prevents these actions by not letting dll load on a per-process basis.

One of the last big things EMET offers is Certificate Trust otherwise known as configurable certificate pinning. Microsoft's EMET user guide (2014) explains that this feature adds additional checks during the certificate chain trust validation process. The goal is to detect man-in-the-middle attacks. Every time a certificate chain trust is built by a browser for a SSL certificate while browsing an HTTPS website, EMET validates the end-entity SSL certificate and the root Certificate Authority that issued the certificate against the pinning rule that was configured by the user through EMET. The Certificate Trust feature can detect variations of the ROOT CA for SSL certificates. EMET does not stop these connections by default but EMET can be configured to close the connection. Exceptions can be added in EMET for each specific pinning rule.

With all of EMET's big mitigation techniques, EMET may sound complicated and confusing to implement. That is not the case at all. EMET is a free downloadable application that allows the user or system admin to opt in applications with command-line utility as well as a GUI interface if the user is not comfortable in command line. One of the great things about being able to add applications so easily is that software may have been written before the attack or vulnerability was discovered or made. What this means is that legacy applications can have an updated defense through EMET. Another great reason to implement EMET is that as well as adding whichever applications the user would like, the user can also configure the application on a per process basis. The reason for this is compatibility issues. With increased security comes the risk of compatibility issues since Windows cannot design EMET for every application ever written. Therefore, EMET provides the user with the ability to turn specific mitigation techniques on and off per application. Updates and new versions come out every year that offer new techniques and tools which can be found at support.microsoft.com. EMET can also be configured for group policy deployment making it an ideal tool to have in any size of enterprise network. Companies may pay for great intrusion prevention systems and other tools, but EMET at no cost is a great, effective and responsible way to keep corporate machines safe from memory corruption and application attacks. Although EMET was only deployed six years ago in October 2009, EMET has patched and evolved as the attacks have evolved.

In summary, EMET is an excellent mitigation tool that can be implemented on anything from a home computer to an enterprise group policy. EMET cannot stop all attacks but no piece of mitigation software ever can. Computer security is always going to be about accepting risk and making the risk as small as possible with the best tools available. Security will never be able to get rid of risk completely. EMET, although not perfect, is a respectable hurdle that makes attackers jump, dive, and crawl around it. EMET is worth it.

References

The Enhanced Mitigation Experience Toolkit. (n.d.). Retrieved April 13, 2015, from https://support.microsoft.com/en-us/kb/2458544/

(n.d.). Retrieved April 13, 2015, from https://www.nsa.gov/ia/_files/os/Win_EMET/EMET_FAQ_v3.1.pdf

Krebs on Security. (n.d.). Retrieved April 13, 2015, from http://krebsonsecurity.com/2013/06/windows-security-101-emet-4-0/#more-20368

Security Research & Defense. (n.d.). Retrieved April 13, 2015, from http://blogs.technet.com/b/srd/archive/2009/06/12/understanding-dep-as-a-mitigation-technology-part-1.aspx

Uninformed - vol 2. (n.d.). Retrieved April 13, 2015, from http://www.uninformed.org/?v=2&a=4&t=txt

Vlaszaty, B., & Rohani, H. (2014). Test the Effectiveness of the Enhanced Mitigation Experience Toolkit Using Well-known Attacks on Well-known Binaries. *

Sotirov, A., & Dowd, M. (2008). Bypassing browser memory protections in Windows Vista. *Blackhat USA.**

Enhanced Mitigation Experience Toolkit 5.1 User Guide. (November 2014). Retrieved April 13, 2015, from support.microsoft.com/.../EMET%205.1%20User%20Guide.pdf