MRG EFFITAS EXPLOIT AND POST-EXPLOIT PROTECTION TEST

May 2018

May 24, 2018

1 Contents

2 Introduction

Web browsing is an integral part of both home and corporate internet users' daily activity. The web is almost ubiquitous and people use it for communication, social life, gaming, business, shopping, education etc. People browse the web very often with outdated software (both at home and in the enterprise) and these outdated applications have known vulnerabilities. Some of these vulnerabilities let the attackers run code on the victim's computer, without any warning on the victim's side. After the victim's computer is infected, the attackers can use this malicious code to steal money from their internet banking application, steal credit card data, steal personal information, steal confidential corporate information, or even lock the computer until the victim pays a ransom.

Drive-by download exploits are one of the biggest threats and concerns in an enterprise environment because no user interaction is needed to start the malware on the victim machine. Even traditional, legitimate sites used by enterprises on a daily basis get infected by malware. Browser and Office based exploits are especially popular among organized criminals. Outdated browser and Office environments are very "popular" in enterprise environments because of compatibility issues, lack of proper patch-management, etc.

Exploits and drive-by download attacks are commonly used in Advanced Persistent Threat (APT) attacks as well. Home users and small to medium businesses often lack the knowledge and awareness about exploits, exploit prevention, targeted attacks and the importance of software updates. Big enterprises face the challenge of managing complex IT environments and consequently endure a high probability of becoming a target of exploit and malware-based attacks.

Antivirus systems and Internet Security Suites have had a long journey from traditional signature-based protection to that which is implemented in a modern protection system. Advanced heuristics, sandboxing, intrusion prevention systems, URL filtering, cloud-based reputation systems, Javascript analysers, memory corruption protection and more are now used to combat modern malware threats. In order to fully evaluate an endpoint protection system, one has to test all modules of the protection employed by that system. Also, the test has to be done in a way which emulates standard user behaviour accurately.

One area that is often overlooked in antivirus testing is protection from exploit and post-exploit attack techniques.

The main purpose of this test is to see how security products handle a specific exploitation technique. In order to be able to test this, we developed test cases that simulate the corresponding exploit and post-exploit techniques only. By this method we were able to see which products protect against which techniques.

We were not looking to test the products' ability to avoid exposure to adversaries, to interrupt malware delivery before it reaches the device or to identify malicious files. We wanted to focus explicitly on each product's ability to mitigate each attack technique. **The results are not intended to evaluate the complete efficacy of the products, but rather the products' anti-exploit and anti-post-exploit features in isolation.**

This assessment was commissioned and sponsored by Sophos, to serve as an independent efficacy assessment of its Sophos Intercept X compared with other popular endpoint protection software.

3 Testing methodology

In most test cases, we targeted so-called protected applications like Internet Explorer, Microsoft Office Word, Mozilla Firefox or the operating system itself.

We think that the best way to test exploit protection capabilities of products is to keep them offline and test them against exploit techniques in this state. We think that in exploit mitigation features, cloud functionalities do not provide additional protection; on the other hand, if left online, products would upload test files to the vendors and by this damage further tests by detecting the files.

To keep the picture clean, we restore all virtual machines to the original state after all test cases. This lets us know how the products behave in a certain situation and ensures the previous test case did not have influence on the current test case.

In test cases where we wanted to test how the products recognise memory corruption exploits, we used two techniques to get inside a protected application:

- We used our kernel driver to inject test DLLs to protected applications. We injected the DLL in the early stages of the process, waited until the protected application fully loaded, then triggered the current memory corruption exploit.
- We also used user-mode tools to inject test DLLs into already running protected applications.

In test case 2. - Data Execution Prevention (DEP)

In this test we exploited our own application (called: skeleton_no_dep.exe), to be able to test this protection feature.

In test case 3. - Mandatory Address Space Layout Randomization (ASLR)

To easily test this functionality we used our test application, which prints the EIP to the console.

In test case 21. - Process hollowing

In this test case we used our test application as a non-malicious application and used it in a process of hollowing.

3.1 Test system setup

Microsoft Windows versions used:

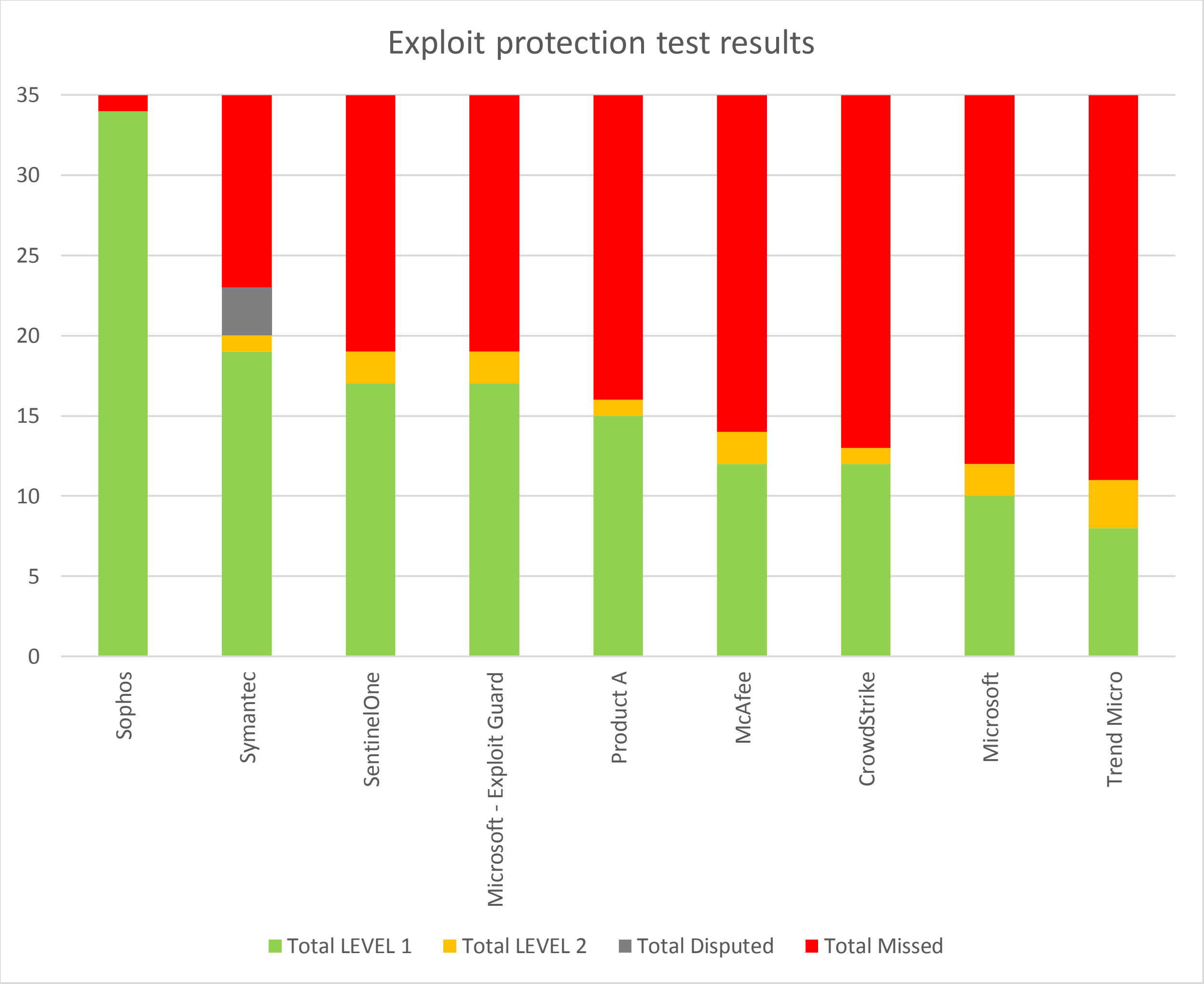
- Microsoft Windows 7 Professional x64 (6.1.7601 Service Pack 1)
- Microsoft Windows 10 Pro (10.0.16299 Fall Creators Update)

3.2 Security Applications Tested

- McAfee Endpoint Security with Threat Protection (version 10.5.3; Threat Protection version: 5.0.6.220)
- Symantec Endpoint Protection (version: 14 [14 UR1 MP2])
- Trend Micro Smart Protection for Endpoints (Agent Version: 6.3.1215/13.1.2054; Scan Engine: 10.000.1043)
- CrowdStrike Falcon Prevent (version: 4.4.6711.0)
- Sophos Intercept X (version: 2.0.2)
- SentinelOne Endpoint Protection (version: 2.1.2.6003)
- Microsoft Windows 10 Professional with Defender Antivirus (Fall Creators Update)
- Microsoft Windows 10 Professional with Defender, Exploit Guard (Fall Creators Update)
- Product A (included anonymously by agreement with the vendor)

4 Test Results

The table below shows the results of the exploit test.



5 Understanding Grade of Pass

- **LEVEL 1**

The product performed as expected to get a positive result in the test. In most cases, this means the product blocked the exploit or attack technique. In false positive tests, it means the product did not block the test sample's execution.

In the case of Microsoft Windows10 configuration, if the attack is not possible on Windows 10 anymore due to hardening steps of Microsoft, we counted these as Level 1 as well.

- **LEVEL 2**

The product blocked the test case before any malicious activity was performed by the sample or before we reached the main part of the test. For example, in some cases test samples were blocked because we used PowerShell or other tools, not because of test-relevant activities or the presence of the exploit protection feature.

- **Disputed**

We used this flag when test was failed but vendor was totally sure about that the certain test case should have been blocked by the product. Maybe the result was influenced some configuration issue.

- **MISSED**

The product did not detect the attack and did not block it. We were able to execute our proof of concept code before the process had been terminated (if it was terminated at all).

6 Test cases

The following paragraphs include the detailed descriptions of the test cases performed. Some descriptions of the exploit techniques and protections are copied from https://secure2.sophos.com/en-us/en-us/medialibrary/Gated-Assets/white-papers/Sophos-Comprehensive-Exploit-Prevention-wpna.pdf?la=en

6.1 False positive test

Test case to see whether our helper tools are working as expected and are not blocked by the specific product. No malicious activity is performed.

Expected result	Operating System	Exploited application
Sample execution is not blocked	*Windows 10 / Windows 7*	-

6.2 Enforce Data Execution Prevention (DEP)

Note: In this test we exploited our own application (called: skeleton_no_dep.exe), to be able to test this protection feature.

Data execution prevention (DEP) is a set of hardware and software technologies that perform additional checks on memory to help prevent buffer overflows. Without DEP, an attacker can attempt to exploit a software vulnerability by jumping to malicious code (shellcode) at a memory location where attacker-controlled data resides, such as the heap or stack. Without DEP, these regions are normally marked as executable, so malicious code will be able to run.

DEP is an opt-in option for Windows XP and above that must be set by the software vendor when building an application. Furthermore, attacks are available for bypassing built-in DEP protection and, as such, dependence on the operating system implementation is not recommended.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Custom app

6.3 Mandatory Address Space Layout Randomization (ASLR)

Some exploits work by targeting memory locations known to be associated with particular processes. In older versions of Windows (including Windows XP), core processes tended to be loaded into predictable memory locations upon system startup. Address space layout randomization (ASLR) randomizes the memory locations used by system files and other programs, making it much harder for an attacker to correctly guess the location of a given process, including the base of the executable and the positions of the stack, heap and libraries.

ASLR is only available on Windows Vista and above and, like DEP, must be set by the software vendor when building an application. And like DEP, attacks are available for bypassing built-in ASLR protection and, as such, dependence on the operating system implementation is not recommended.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	To easily test this functionality we used our test application which prints the EIP to the console. If ASLR works, this address should change each time the application started.

6.4 Null Page (Null Deference)

Starting with Windows 8 and onwards, Microsoft denies programs the ability to allocate and/or map the "NULL page" (memory residing at virtual address 0x00000000 in the address space). By doing this, Microsoft successfully mitigates the direct exploitation of a whole class of vulnerabilities called "NULL pointer dereference" vulnerabilities.

On Windows XP, Windows Vista, and Windows 7, the exploitation of such a flaw would allow the attacker to execute code in the context of the kernel (under the ring0 CPU privilege level), resulting in privilege escalation to one of the highest levels. Such vulnerabilities give attackers access to virtually all parts of the operating system.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Operating System

6.5 Heap Spray Pre-Allocation

A heap spray is a technique that does not actually exploit vulnerabilities but is used to make a vulnerability easier to exploit. Using a technique called Heap Feng Shui1 an attacker is able to reliably position intended data structures or shellcode on the heap, thus facilitating a reliable exploitation of a software vulnerability.

A typical heap spray mitigation involves reserving or pre-allocating commonly used memory addresses, so they cannot be used to house payloads. More creative attackers are aware of these addresses so in a real-world attack scenario this mitigation has little effect. Also known as Anti-HeapSpray Enforcement or Shellcode Preallocation, the heap spray pre-allocation is typically effective against default exploits used by testing organizations.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

6.6 Dynamic Heap Spray

Compared to the static Heap Spray Pre-Allocation, the Dynamic Heap Spray mitigation is typically triggered by a sudden increase in memory consumption.

The dynamic heap spray mitigation actually analyzes the contents of recent memory allocations to detect patterns that indicate heap sprays containing NOP sleds, polymorphic NOP sleds, JavaScript arrays, and other suspicious sequences that are placed on the heap to facilitate exploit attacks.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

6.7 Stack Pivot

The stack of an application is a memory area that contains, among other things, a list of memory address locations (so-called return addresses). These locations contain the actual code that the processor needs to execute in the near future.

Stack pivoting is widely used by vulnerability exploits to bypass protections like DEP, for example by chaining ROP gadgets in a return-oriented programming attack. With stack pivoting, attacks can pivot from the real stack to a new fake stack, which could be an attacker-controlled buffer such as the heap, from which attackers can control the future flow of program execution.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

6.8 Stack Exec

Under normal circumstances, the stack contains data and addresses pointing to code for the processor to execute in the near future. Using a stack buffer overflow,it is possible for attackers to overwrite the stack with arbitrary code. In order to make this code run on the processor, the memory area of the stack must be made executable to circumvent DEP. Once the stack-memory is executable, it is very easy for an attacker to supply and run program code.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

6.9 Return Oriented Programming (ROP)

To execute malicious code in the presence of security defences like data execution prevention (DEP), address space layout randomization (ASLR) and code signing, attackers typically resort to hijacking control-flow of vulnerable internet-facing applications. Such in-memory attacks are often invisible to antivirus and other defenses as there are no malicious files involved. Instead, the attack is constructed at run time by combining short pieces of benign code that are part of existing applications like Internet Explorer and Adobe Flash Player – a so-called code-reuse or Return-Oriented Programming (ROP) attack.

During normal control-flow, sensitive API functions – like VirtualAlloc and CreateProcess – are invoked by the CALL instruction. Upon invoking a sensitive API, typical stack-based ROP defenses stop code execution to determine the API invoking address, using the 'return' address which is located on top of the stack. If the instruction of the API invoking address is not a CALL, the process is terminated.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

6.10 Return Oriented Programming (ROP) with CALL-preceded ROP gadget

Stack-based defenses against return-oriented programming (ROP) are coarse-grained and more manipulation-prone. For example, stack-based ROP defenses can be bypassed if an able attacker can find and use a so-called CALL-preceded ROP gadget to access that calls a sensitive API function. This more sophisticated use of ROP is in-the-wild since at least 2015.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Firefox

References:

- https://www.rsaconference.com/writable/presentations/file_upload/spo3-t11_how-nation-states-andcriminal-syndicates-use-exploits-to-bypass-security.pdf (slide 33)

6.11 Structured Exception Handler Overwrite Protection (SEHOP)

An attacker can overwrite, with a controlled value, the handler pointer of an exception record on the stack. Once an exception happens, the operating system will walk the exception record chain and call all the handlers on each exception record.

Since the attacker controls one of the records, the operating system will jump to wherever the attacker wants, giving the attacker control over the flow of execution.

SEHOP is an opt-in option on Windows Vista and above and must be set by the software vendor when building the application. Attacks are available for bypassing built-in SEHOP protection and, as such, dependence on the operating system implementation is not recommended.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Internet Explorer 11

6.12 Import Address Table Filtering

An attacker eventually needs the addresses of specific system functions (e.g. kernel32!VirtualProtect) to be able to perform malicious activities.

These addresses can be retrieved from different sources, one of which is the import address table (IAT) of a loaded module. The IAT is used as a lookup table when an application calls a function in a different module. Because a compiled program cannot know the memory location of the libraries it depends upon, an indirect jump is required whenever an API call is made. As the dynamic linker loads modules and joins them together, it writes actual addresses into the IAT slots so that they point to the memory locations of the corresponding library functions.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Internet Explorer 11

6.13 Load Library - Loading a DLL from a remote server using an UNC path

Attackers can attempt to load malicious libraries by placing them on UNC paths. Monitoring of all calls to the LoadLibrary API can be used to prevent this type of library loading.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Internet Explorer 11

6.14 Reflective DLL Injection

Normally when you load a DLL in Windows, you call the API function LoadLibrary. LoadLibrary takes the file path of a DLL as input and loads it into memory.

Reflective DLL loading refers to loading a DLL from memory rather than from disk. Windows doesn't have a LoadLibrary function that supports this, so to get this functionality you have to write your own. One benefit to writing your own function is that you can omit some of the things Windows normally does, such as registering the DLL as a loaded module in the process, which makes the reflective loader sneakier when being investigated. Meterpreter is an example of a tool that uses reflective loading to hide itself. Mitigation is performed by analyzing if a DLL is reflectively loaded inside memory.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Internet Explorer 11

References:

- https://github.com/stephenfewer/ReflectiveDLLInjection

6.15 VBScript God Mode

On Windows, VBScript can be used in browsers or the local shell. When used in the browser, the abilities of VBScript are restricted for security reasons. This restriction is controlled by the safemode flag. If this flag is modified, VBScript in HTML can do everything as though it's in the local shell. Consequently, attackers can easily write malicious code in VBScript. Manipulating the safemode flag on VBScript in the web browser is known as God Mode.

As an example, an attacker can modify the safemode flag value by leveraging the CVE-2014-6332 vulnerability, a bug caused by improper handling while resizing an array in the Internet Explorer VBScript engine. In God Mode, arbitrary code written in VBScript can break out of the browser sandbox. Thanks to God Mode, data execution prevention (DEP), address space layout randomization (ASLR), and control- flow guard (CFG) protections are not in play.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Internet Explorer 11

References:

- https://www.rapid7.com/db/modules/exploit/windows/browser/ms14_064_ole_code_execution

6.16 WoW64

Microsoft provides backward-compatibility for 32-bit software on 64-bit editions of Windows through the "Windows on Windows" (WoW) layer. Aspects of the WoW implementation provide interesting avenues for attackers to complicate dynamic analysis, binary unpacking, and to bypass exploit mitigations.

The behavior of a 32-bit application under the WoW64 environment is different in many ways from a true 32-bit system. The ability to switch between execution modes at runtime can provide an attacker methods for exploitation, obfuscation, and anti-emulation such as:

- Additional ROP gadgets not present in 32-bit code
- Mixed execution mode payload encoders
- Execution environment features that may render mitigations less effective
- Bypassing hooks inserted by security software, only in 32-bit user space

Most endpoint protection software will only hook sensitive API functions in the 32-bit user memory space if a process is running under WoW64. If an attacker is able to switch to 64-bit mode, access is gained to unhooked 64-bit versions of the sensitive API functions that are hooked in 32-bit mode.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Microsoft Office Word 2010

References:

- https://duo.com/blog/wow64-and-so-can-you
- https://duo.com/assets/pdf/WoW64-Bypassing-EMET.pdf

6.17 Syscall

A syscall (or system call) is the programmatic way in which a computer program requests a service from the kernel of the operating system. This includes hardware-related services like accessing the local disk and creation and execution of new processes.

Generally, the operating system provides a generic application programming interface (API) that sits between normal programs and the operating system. Under normal circumstances, an application will always call an API to request a specific task from the kernel. Security software places hooks on sensitive API functions to intercept and perform checks like antivirus scanning, before it allows the kernel to service the request.

An attacker can take advantage of the fact that:

- Not all API functions are hooked by security software; only sensitive functions.
- The stubs that are used to call kernel functions are very similar; only the function index is unique.

By calling an unmonitored non-sensitive function stub at an offset (to intentionally address a sensitive kernel service instead) an attacker can effectively evade most security software or sandbox analysis.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Firefox

References:

- https://www.evilsocket.net/2014/02/11/on-windows-syscall-mechanism-and-syscall-numbers-extraction-methods/
- https://breakdev.org/defeating-antivirus-real-time-protection-from-the-inside/

6.18 Lockdown - an Office application that drops a file to disk and executes it

This test drops a file from an Office application and executes it. This chain of events can be observed in attacks that use for example a crafted (malicious) macro in an Office document, attached to a (spear) phishing email.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Microsoft Office Word 2016

6.19 Lockdown - Word document running a macro that spawns existing Windows Calculator

False positive test: Word document running a macro that spawns existing Windows Calculator.

Expected result	Operating System	Exploited application
Sample execution is not blocked	*Windows 10*	Microsoft Office Word 2016

6.20 Sticky Key

Setting the cmd.exe as Debugger to sethc.exe under Image File Execution options in registry. This can provide a backdoor functionality where attackers can bypass the login screen without providing a password.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Operating System

References:

- https://www.rapid7.com/db/modules/post/windows/manage/sticky_keys

6.21 Process hollowing

Process hollowing is a technique in which a trusted application – like explorer.exe or svchost.exe – is loaded on the system solely to act as a container for hostile code.

A hollow process is typically created in a suspended state then its memory is unmapped and replaced with malicious code. Similar to code injection, execution of the malicious code is masked under a legitimate process and may evade defenses and detection analysis.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Operating System

References:

- https://github.com/m0n0ph1/Process-Hollowing

6.22 DLL hijacking via web browser

Due to a vulnerability commonly known as DLL hijacking, DLL spoofing, DLL preloading or binary planting, many programs will load and execute a malicious DLL contained in the same folder as a data file opened by these programs.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Operating System

References:

- https://textslashplain.com/2015/12/18/dll-hijacking-just-wont-die/

6.23 Credential theft

Mimikatz is an open-source tool built to gather and exploit Windows credentials. Since its introduction in 2011 by author Benjamin Delpy, the attacks that Mimikatz is capable of have continued to grow. Also, the ways in which Mimikatz can be packaged and deployed have become even more creative and difficult to detect by security professionals.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Operating System

References:

- https://github.com/gentilkiwi/mimikatz

6.24 Backdoor injected by ShellterPro

Code cave is a technique used by adversaries where they modify what is likely legitimate software so that it contains an additional application. This additional application is inserted into what is called a code cave, a section of the target application's file that is unused by the program. Code caves exist in most applications and adding code to these sections should not break the behavior of the primary application.

Often the execution code inserted into a code cave is simply a remote shell launcher or backdoor; these can be very small and simply grant the adversary access to the endpoint where they can perform other actions. This type of attack requires the attacker to have established a presence on the endpoint so they can deploy the back doored application or to trick the user to download and install an application that has the code cave already exploited.

One of the primary reasons adversaries use code caves is to hide from detection by the general user and administrators. The expected application still works fine, but the inserted application is also running.

If the application that has been modified is a legitimate business tool that the administrator expects to be on the device they are less likely to consider it malware if traditional antivirus detects a problem. Administrators may simply add it to the exemption list, assuming the antivirus engine has generated a false positive. In this way, the adversary establishes persistence on the endpoint and may have even tricked the admin to allow their inserted application to run.

ShellterPro is a tool to inject code into a legitimate application (Sysinternals – Process Explorer).

Expected result	Operating System	Exploited application

Sample execution is blocked	*Windows 10*	Sysinternals – Process Explorer was infected with our DLL.

References:

- https://www.shellterproject.com

6.25 Backdoor injected by Backdoor Factory

Backdoor Factory is a tool to inject code into a legitimate application (Sysinternals – Process Explorer).

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Sysinternals – Process Explorer was infected with our DLL.

References:

- https://github.com/secretsquirrel/the-backdoor-factory

6.26 Backdoor injected by InfectPE

InfectPE is a tool to inject code to a legitimate application (Sysinternals – Process Explorer).

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Sysinternals – Process Explorer was infected with our DLL.

References:

- https://github.com/secrary/InfectPE

6.27 DoublePulsar code-injection

DoublePulsar was originally a backdoor implant tool developed by the U.S. National Security Agency's (NSA) Equation Group that was leaked by The Shadow Brokers in early 2017. The implant contains a novel injection technique that is part of several NSA exploits, including EternalBlue and EternalRomance. These exploits were also used for the self-spreading worm component in the WannaCry and NotPetya outbreaks.

The DoublePulsar code injection technique employs an Asynchronous Procedure Call (APC) to run arbitrary code (shellcode) inside a regular trusted process.

Expected result	Operating System	Exploited application

Sample execution is blocked	*Windows 10*	Internet Explorer 11

References:

- https://en.wikipedia.org/wiki/DoublePulsar
- https://github.com/countercept/doublepulsar-usermode-injector

6.28 AtomBombing code-injection

Asynchronous Procedure Call (APC) injection involves attaching malicious code to the APC queue of a process's thread. Queued APC functions are executed when the thread enters an alterable state. AtomBombing is a variation that utilizes APCs to invoke malicious code previously written to the global atom table.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Internet Explorer 11

References:

- https://github.com/BreakingMalwareResearch/atom-bombing

6.29 Privilege escalation: stealing Windows access token

CVE-2014-4113 allows for the elevation of privileges when exploited successfully.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 7*	Operating System

References:

- https://blog.trendmicro.com/trendlabs-security-intelligence/an-analysis-of-a-windows-kernel-mode-vulnerability-cve-2014-4113/

6.30 Detection of financial malware manipulating the web browser

Hooking relevant APIs (HttpSendRequest, EncryptMessage, DecryptMessage, CryptEncrypt, CryptDecrypt…) in a web-browser allows attackers to steal sensitive user data. In this test we used our internal tool to test hook detection capability of the products.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Internet Explorer 11

References:

- https://en.wikipedia.org/wiki/Man-in-the-browser
- https://www.mrg-effitas.com/wp-content/uploads/2017/05/MRG-Effitas-Online-Banking-Certification_Q1_2017_Level_1_wm.pdf

6.31 Encryption or other unauthorized modification of the master boot record and/or volume boot record

The Master Boot Record is a critical part of a computer; the MBR contains the Partition Table, and tells the computer what partition to boot into. The Partition Table contains documented information about each partition on the hard drive [Volume, Type, Format etc.] The Master Boot Record is so vital in a computer, it is obviously a target for viruses, especially trojans and ransomware.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	Operating System

References:

- https://www.fortinet.com/blog/threat-research/petya-s-master-boot-record-infection.html

6.32 Unauthorized in-place encryption of Word documents (rename file)

Ransomware is a type of malicious software from cryptovirology that threatens to publish the victim's data or perpetually block access to it unless a ransom is paid.

We created a proof-of-concept test application to simulate ransomware activities and encrypting. The product should have to block the application before it finishes its activities.

The test malware first encrypts the victim’s file in place than appends the extension “.mrg”

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	-

6.33 Unauthorized encryption of documents by creating new encrypted file and deleting original

We created a proof-of-concept test application to simulate ransomware activities and encrypting. The product should have to block the application before it finishes its activities.

The test malware encrypts the user documents by creating a new encrypted file and overwriting the original. Newly created filename will have “.mrg” appended.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	**-**

6.34 [Network version] Unauthorized in-place encryption of Word documents

The tester app is the same used in Test Case 32. Targeted folder is located on a network share. Ransomware is running on the protected machine and is attacking files on an unprotected network share.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	**-**

6.35 [Network version] Unauthorized encryption of documents by creating new encrypted file and deleting original

The tester app is the same used in Test Case 33. Targeted folder is located on a network share.

Expected result	Operating System	Exploited application
Sample execution is blocked	*Windows 10*	**-**

7 Detailed test results

The following table represents the detailed test results. For the colour decode, please refer to chapter 5 Understanding Grade of Pass:

ID	Test cases	Sophos	Symantec	SentinelOne	Microsoft - Exploit Guard	Product A	McAfee	CrowdStrike	Microsoft	Trend Micro
1	False positive test	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
2	Data Execution Prevention (DEP)	🟩	🟩	🟥	🟥	🟥	🟥	🟥	🟥	🟥
3	Mandatory Address Space Layout Randomization (ASLR)	🟩	🟩	🟩	🟥	🟥	🟩	🟩	🟥	🟥
4	Null Page (Null Deference)	🟩	🟩	🟩	🟩	🟩	🟥	🟥	🟩	🟩
5	Heap Spray Pre-Allocation	🟩	⬛	🟥	🟥	🟥	🟥	🟥	🟥	🟥
6	Dynamic Heap Spray	🟩	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
7	Stack Pivot	🟩	🟩	🟩	🟥	🟩	🟩	🟩	🟥	🟥
8	Stack Exec	🟩	🟩	🟩	🟥	🟥	🟥	🟥	🟥	🟥
9	Return Oriented Programming (ROP)	🟩	⬛	🟥	🟩	🟩	🟩	🟥	🟥	🟥
10	Return Oriented Programming (ROP) with CALL-preceded ROP gadget	🟩	⬛	🟥	🟥	🟥	🟥	🟥	🟥	🟥
11	SEHOP	🟩	🟩	🟥	🟩	🟩	🟥	🟥	🟥	🟥
12	Import Address Table Filtering	🟩	🟥	🟥	🟥	🟩	🟥	🟥	🟩	🟥
13	Load Library - Loading a DLL from a remote server using an UNC path.	🟩	🟩	🟩	🟨	🟥	🟥	🟩	🟨	🟥
14	Reflective DLL Injection - wm_simulator	🟩	🟩	🟥	🟩	🟩	🟩	🟩	🟥	🟥
15	VBScript God Mode	🟩	🟩	🟥	🟩	🟩	🟩	🟩	🟩	🟩
16	WoW64	🟩	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
17	Syscall	🟩	🟩	🟩	🟩	🟩	🟩	🟥	🟨	🟩
18	Lockdown - an Office application that drops a file to disk and executes it	🟩	🟥	🟥	🟩	🟩	🟩	🟥	🟥	🟥
19	Lockdown - Word document running a macro that spawns Calculator	🟩	🟩	🟩	🟩	🟥	🟩	🟩	🟩	🟥

20	Sticky Key - Debug Process (Registry hack)	🟩	🟥	🟩	🟥	🟩	🟥	🟩	🟥	🟥
21	Process hollowing	🟩	🟨	🟨	🟨	🟥	🟨	🟥	🟩	🟩
22	DLL hijacking via web browser	🟩	🟥	🟨	🟥	🟥	🟨	🟥	🟩	🟥
23	Credential theft - Straight from LSASS (e.g. Mimikatz sekurlsa::logonpasswords)	🟩	🟩	🟩	🟥	🟩	🟩	🟨	🟩	🟩
24	Stealth backdoor injected by ShellterPro	🟩	🟥	🟩	🟩	🟥	🟥	🟥	🟥	🟥
25	Backdoor injected by Backdoor Factory	🟩	🟩	🟩	🟥	🟥	🟩	🟩	🟥	🟥
26	Optional: other tool that utilizes a code cave / injects backdoor	🟩	🟥	🟩	🟥	🟩	🟥	🟩	🟥	🟥
27	DoublePulsar code-injection	🟩	🟥	🟩	🟩	🟩	🟩	🟥	🟥	🟥
28	AtomBombing code-injection	🟩	🟩	🟩	🟩	🟩	🟥	🟩	🟥	🟥
29	Privilege escalation: stealing Windows access token	🟩	🟩	🟩	🟩	🟥	🟥	🟩	🟩	🟥
30	Detection of financial malware manipulating the web browser	🟥	🟥	🟥	🟥	🟨	🟥	🟥	🟩	🟩
31	Encryption of the master boot record and/or volume boot record	🟩	🟩	🟩	🟥	🟥	🟥	🟥	🟥	🟩
32	Unauthorized in-place encryption of Word documents	🟩	🟩	🟥	🟩	🟥	🟥	🟥	🟥	🟨
33	Unauthorized encryption of documents by creating new encrypted file	🟩	🟥	🟥	🟩	🟥	🟥	🟥	🟥	🟨
34	[Network] Unauthorized in-place encryption of documents	🟩	🟥	🟥	🟩	🟥	🟥	🟥	🟥	🟨
35	[Network] Unauthorized encryption of documents	🟩	🟥	🟥	🟩	🟥	🟥	🟥	🟥	🟥
	Total LEVEL 1	34	22	17	17	15	12	12	10	8
	Total LEVEL 2	0	1	2	2	1	2	1	2	3
	Total Disputed	0	3	0	0	0	0	0	0	0
	Total Missed	1	12	16	16	19	21	22	23	24

🟩	LEVEL 1
🟨	LEVEL 2
▨	Disputed
🟥	Missed

By default, Microsoft Office 2016 on Windows 10 Fall Creators update does not block the execution of macro code, either malicious or benign. But there are many options how users can protect against these threats. There is an option to block macro execution on documents which were downloaded from the Internet either via a browser or an email client. Also by configuring Exploit Guard and Application Surface Reduction (ASR), it is possible to block different stages of the macro code execution. For Office apps, ASR can:

- Block Office apps from creating executable content
- Block Office apps from launching child process
- Block Office apps from injecting into process
- Block Win32 imports from macro code in Office
- Block obfuscated macro code

Also, Exploit Guard can be configured to protect with/against:

- DEP
- Mandatory ASLR
- Heap Spray PreAllocation
- StackPivot
- Import Address Table Filtering
- Return Oriented Programming (ROP) software-only (stack-based)
- SEHOP

When it comes to ransomware activity, Windows 10 has a feature called Controlled Folder access. By turning this option on, files on the local machine and on a network share can be protected. This feature is not on by default and must also be configured (and kept manually up-to-date) to protect the specific folders on the local machine and network share where users store their files in. Other files and folders are not protected.

Regarding the "Detection of financial malware manipulating the web browser" test case, the recommended browser by Microsoft is Edge, and it is not possible to inject code into Edge the way it was possible to do it into Internet Explorer.

One problem with non-default configurations like this is that one has to spend resources on figuring out a configuration which both protects the application but does not create false positives.

8 Vendor feedback

Before this assessment was started, all the vendors in the original cohort were contacted and notified that their product had been proposed to be included.

Voluntary participants had the opportunity to review and challenge their results prior to publication. Voluntary participants also had the chance to change default configurations of their products. MRG Effitas also changed the default protections of some products whenever it was needed. All these changes are documented in section 10.1.

Of all the vendors contacted, the following agreed to be voluntary participants:

- Sophos
- Symantec

Wherever possible, Effitas still included vendors who requested not to be active participants. However, we had to remove certain vendors from the cohort for reasons that Effitas is not able to disclose due to business, legal, or contractual considerations.

8.1 Comments received from Crowdstrike

CrowdStrike participates in third-party testing to validate our capabilities and also to continually improve our product. Exploit protection is important and we strive to deliver both strong exploit protection as well as detection and prevention of post-exploitation activity. While this test did not offer an opportunity to show our ability to detect and prevent post-exploit malicious activity, we are committed to working with MRG-Effitas and other testing organizations to deliver testing results that span the entire cyber kill chain so that security teams can understand a product's ability to go beyond malware and exploit protection to stop an attacker from achieving their ultimate goal. For more information on CrowdStrike testing results please visit

https://www.crowdstrike.com/products/third-party-testing-evaluations-results/

8.2 Comments received from Symantec

Symantec disputes the following test results as according to them the product implements these protection features, although the data from our test does not support this:

- Heap Spray Pre-Allocation
- Return Oriented Programming (ROP)
- Return Oriented Programming (ROP) with CALL-preceded ROP gadget

9 Conclusion

Based on the exploit and anti-exploit protection tests, Sophos Intercept X performed the best.

10 Appendix

10.1 Non-default configurations used

The following Endpoint Protections were used in a non-default configuration, after reaching a consensus with the vendors:

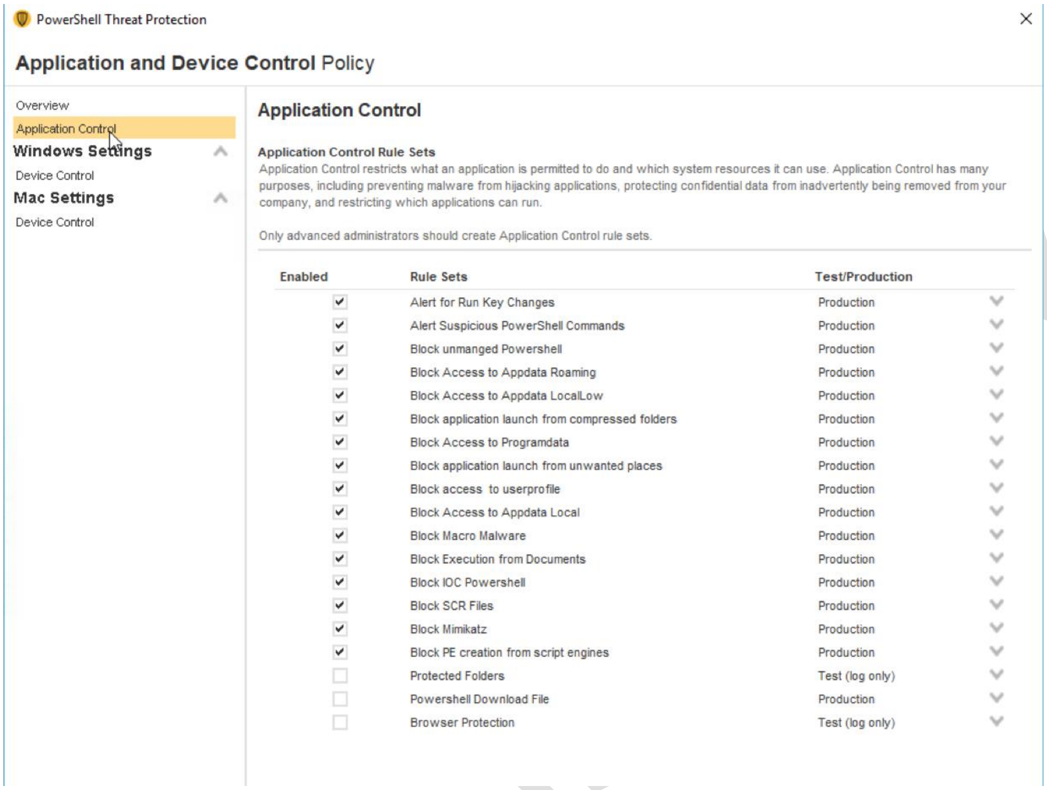
- Symantec Endpoint Protection

The following Endpoint Protections were used in a non-default configuration, based on our best practices:

- Trend Micro Smart Protection for Endpoints
- Crowdstrike Falcon Prevent
- Windows Defender with Exploit Guard configured

The difference from the default configuration can be found below.

10.1.1 Symantec non-default configuration



Application and Device Control Policy

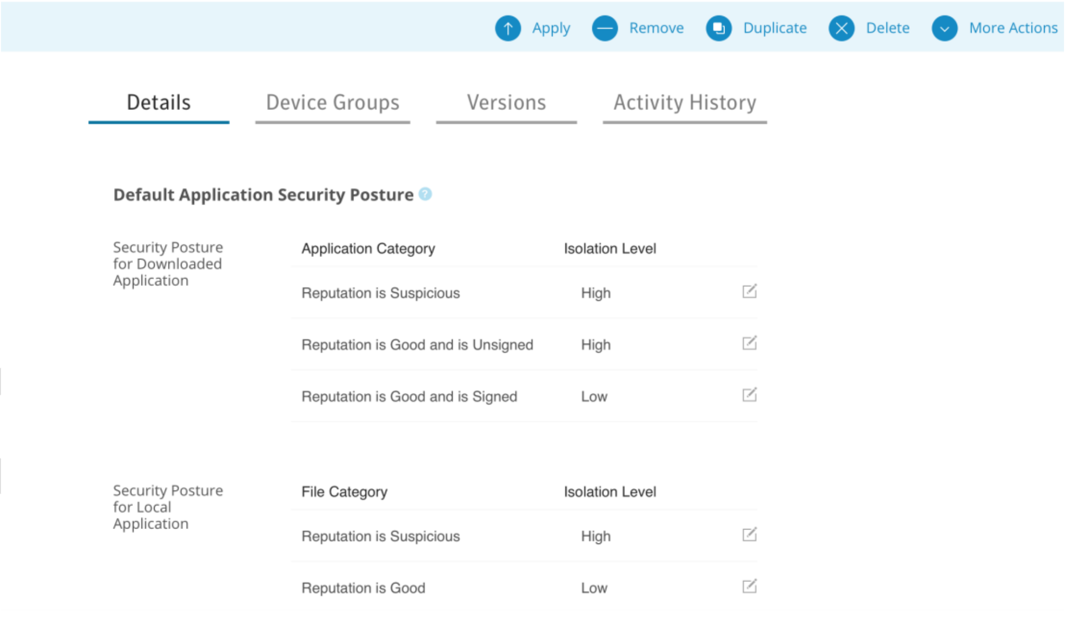
Application Control

Application Control Rule Sets

Application Control restricts what an application is permitted to do and which system resources it can use. Application Control has many purposes, including preventing malware from hijacking applications, protecting confidential data from inadvertently being removed from your company, and restricting which applications can run.

Only advanced administrators should create Application Control rule sets.

Enabled	Rule Sets	Test/Production
☑	Alert for Run Key Changes	Production
☑	Alert Suspicious PowerShell Commands	Production
☑	Block unmanged Powershell	Production
☑	Block Access to Appdata Roaming	Production
☑	Block Access to Appdata LocalLow	Production
☑	Block application launch from compressed folders	Production
☑	Block Access to Programdata	Production
☑	Block application launch from unwanted places	Production
☑	Block access to userprofile	Production
☑	Block Access to Appdata Local	Production
☑	Block Macro Malware	Production
☑	Block Execution from Documents	Production
☑	Block IOC Powershell	Production
☑	Block SCR Files	Production
☑	Block Mimikatz	Production
☑	Block PE creation from script engines	Production
□	Protected Folders	Test (log only)
□	Powershell Download File	Production
□	Browser Protection	Test (log only)



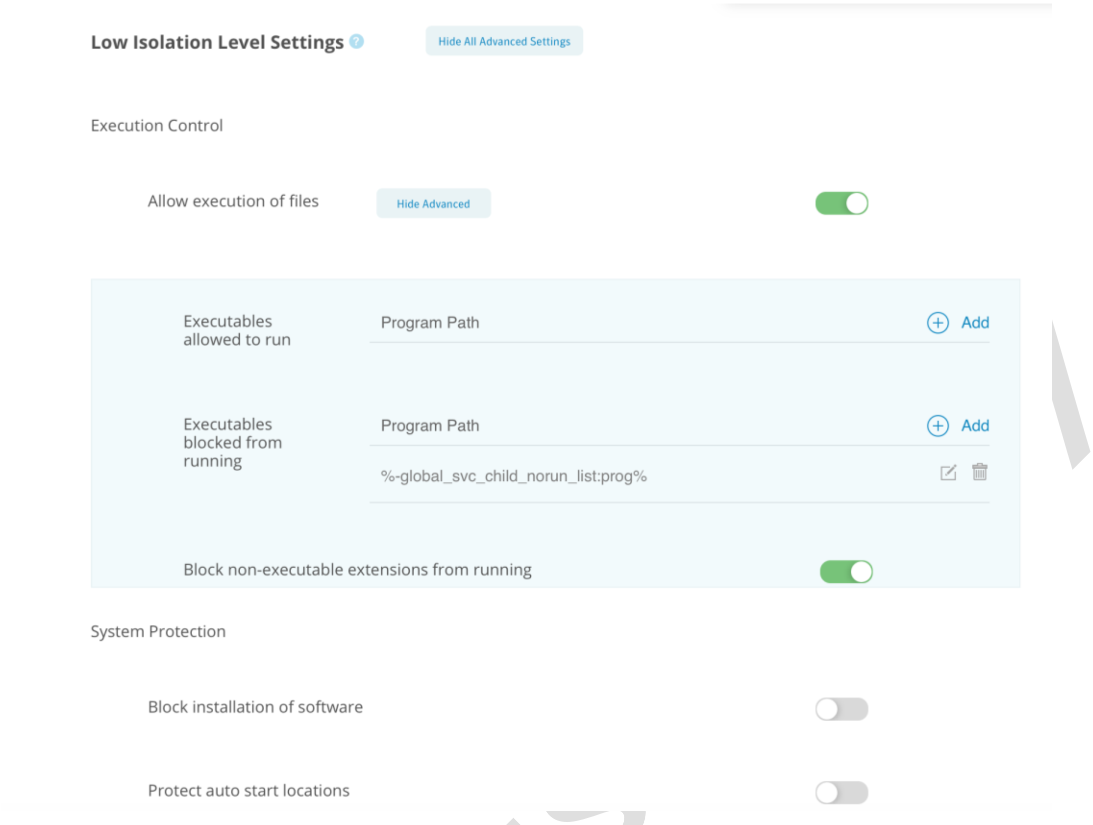
Default Application Security Posture

Security Posture for Downloaded Application

Application Category	Isolation Level
Reputation is Suspicious	High
Reputation is Good and is Unsigned	High
Reputation is Good and is Signed	Low

Security Posture for Local Application

File Category	Isolation Level
Reputation is Suspicious	High
Reputation is Good	Low



Low Isolation Level Settings ? Hide All Advanced Settings

Execution Control

Allow execution of files Hide Advanced ☑

Executables allowed to run — Program Path ⊕ Add

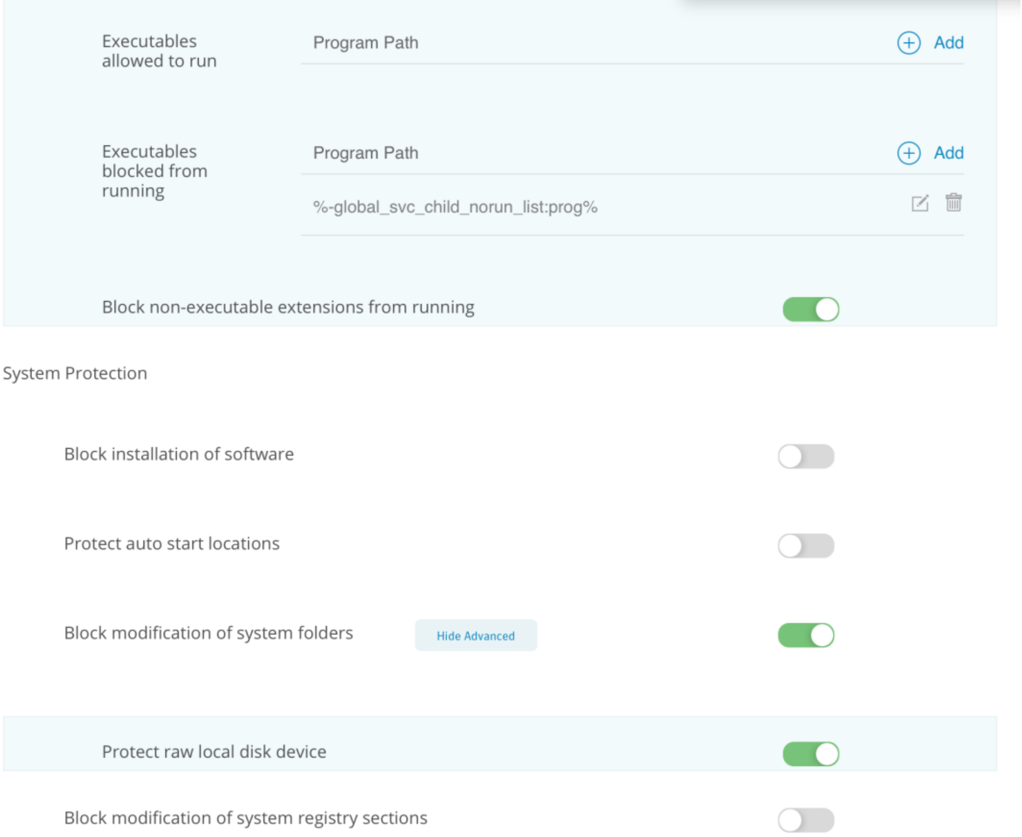
Executables blocked from running — Program Path ⊕ Add
%-global_svc_child_norun_list:prog%

Block non-executable extensions from running ☑

System Protection

Block installation of software ☐

Protect auto start locations ☐



Executables allowed to run — Program Path ⊕ Add

Executables blocked from running — Program Path ⊕ Add
%-global_svc_child_norun_list:prog%

Block non-executable extensions from running ☑

System Protection

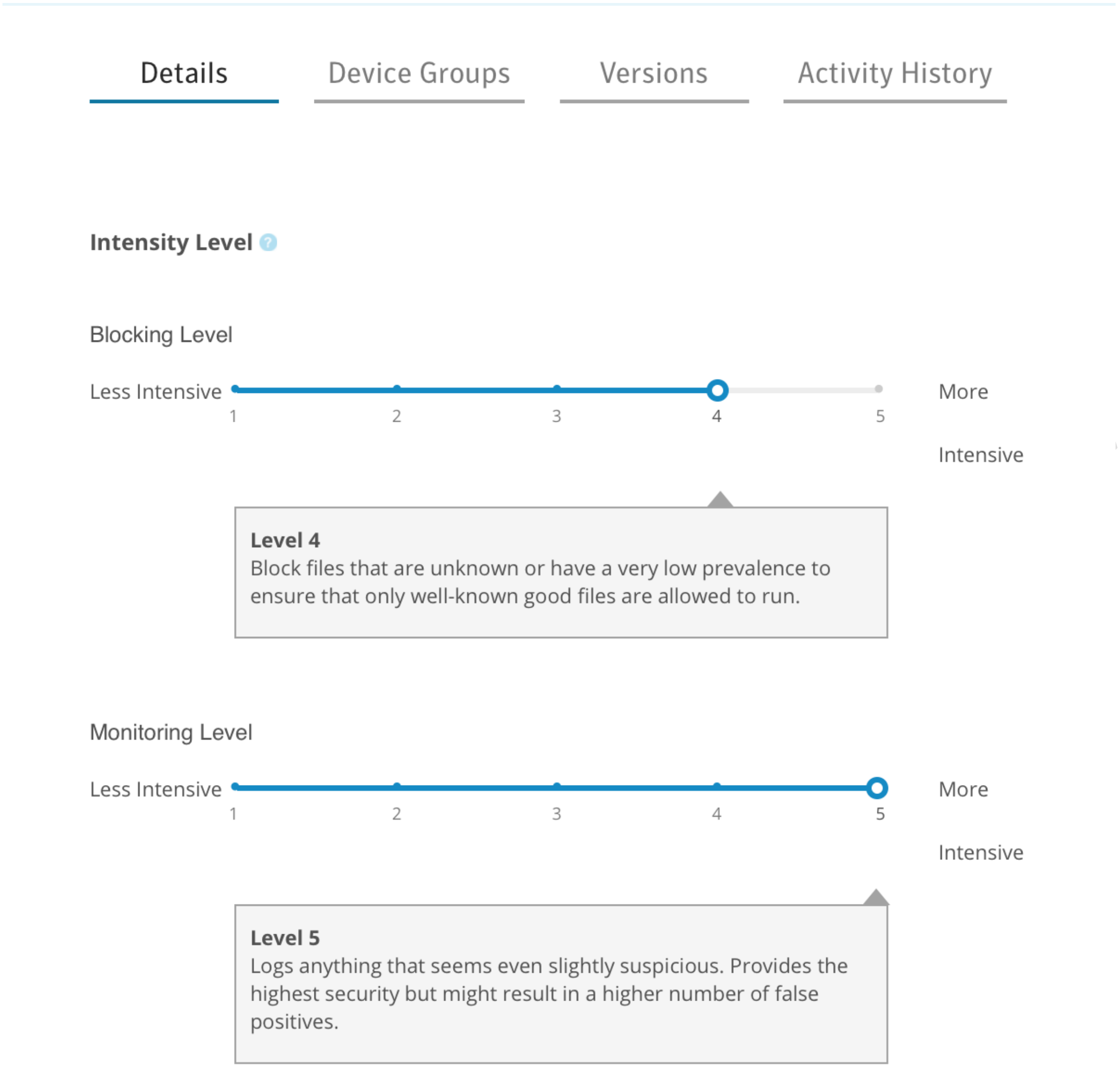
Block installation of software ☐

Protect auto start locations ☐

Block modification of system folders Hide Advanced ☑

Protect raw local disk device ☑

Block modification of system registry sections ☐



Intensity Level

Blocking Level

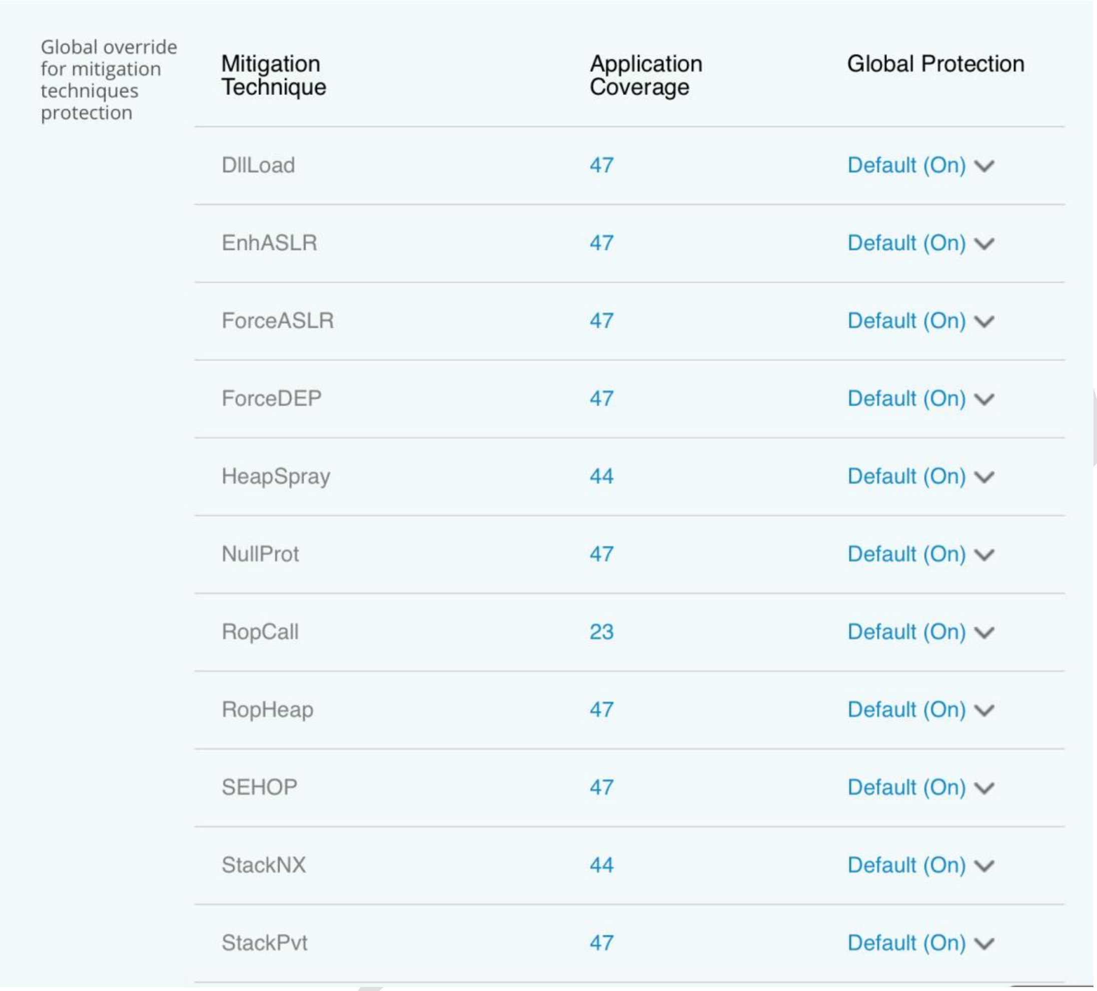
Less Intensive 1 2 3 **4** 5 More Intensive

Level 4
Block files that are unknown or have a very low prevalence to ensure that only well-known good files are allowed to run.

Monitoring Level

Less Intensive 1 2 3 4 **5** More Intensive

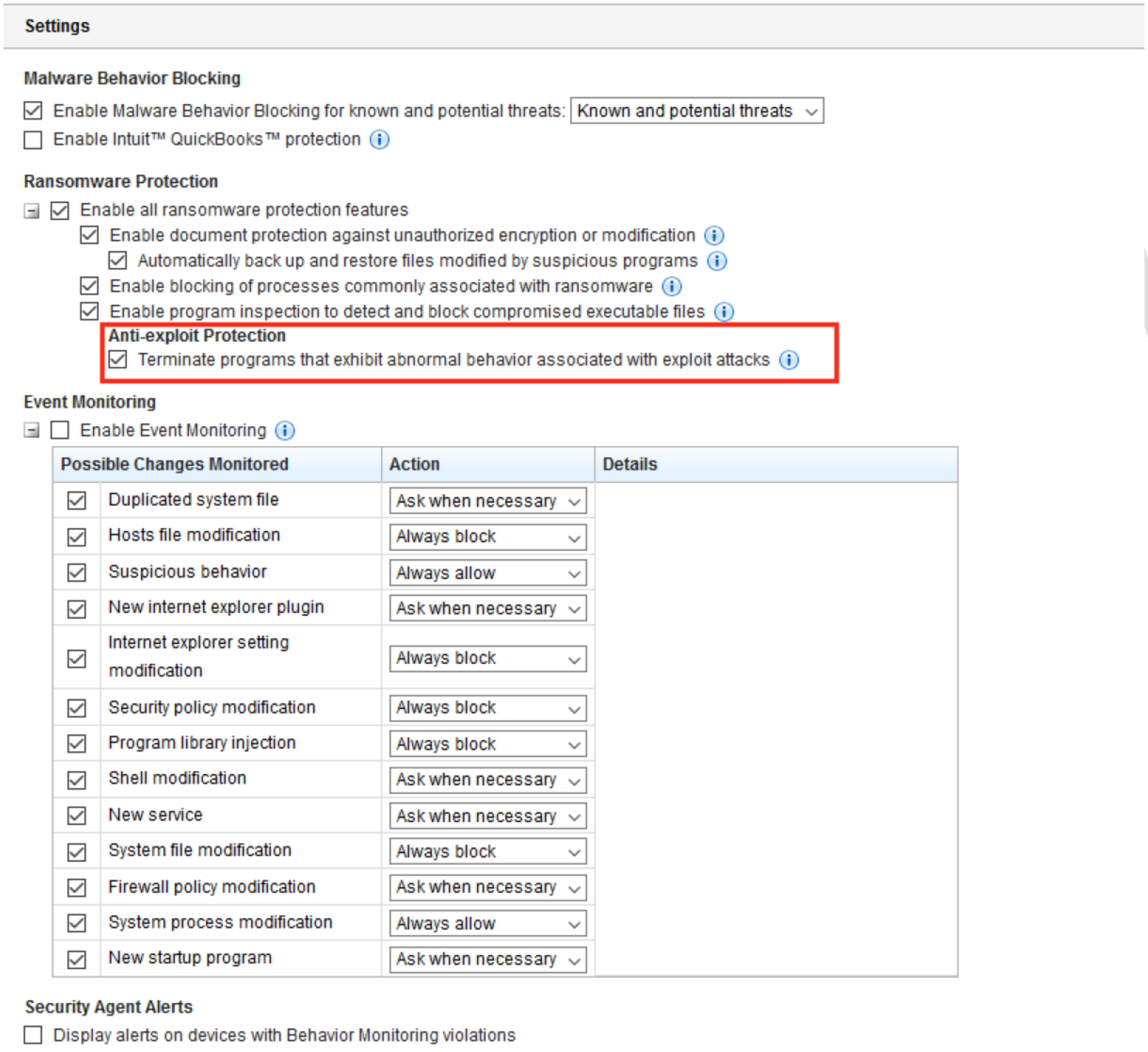
Level 5
Logs anything that seems even slightly suspicious. Provides the highest security but might result in a higher number of false positives.



Global override for mitigation techniques protection

Mitigation Technique	Application Coverage	Global Protection
DllLoad	47	Default (On) ⌄
EnhASLR	47	Default (On) ⌄
ForceASLR	47	Default (On) ⌄
ForceDEP	47	Default (On) ⌄
HeapSpray	44	Default (On) ⌄
NullProt	47	Default (On) ⌄
RopCall	23	Default (On) ⌄
RopHeap	47	Default (On) ⌄
SEHOP	47	Default (On) ⌄
StackNX	44	Default (On) ⌄
StackPvt	47	Default (On) ⌄

10.1.2 Trend Micro non-default configuration



Settings

Malware Behavior Blocking

☑ Enable Malware Behavior Blocking for known and potential threats: Known and potential threats

□ Enable Intuit™ QuickBooks™ protection

Ransomware Protection

☑ Enable all ransomware protection features

- ☑ Enable document protection against unauthorized encryption or modification
  - ☑ Automatically back up and restore files modified by suspicious programs
- ☑ Enable blocking of processes commonly associated with ransomware
- ☑ Enable program inspection to detect and block compromised executable files

Anti-exploit Protection

☑ Terminate programs that exhibit abnormal behavior associated with exploit attacks

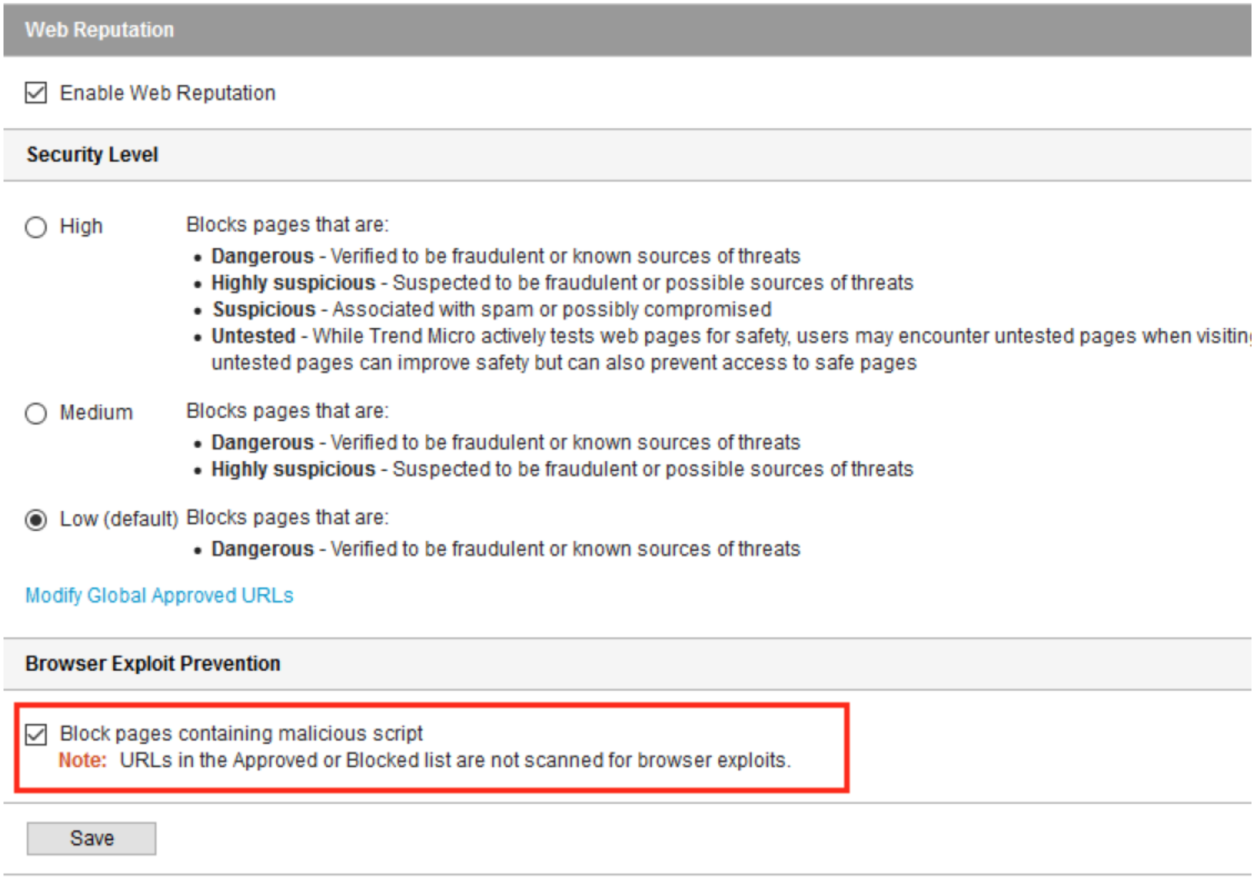
Event Monitoring

□ Enable Event Monitoring

	Possible Changes Monitored	Action	Details
☑	Duplicated system file	Ask when necessary	
☑	Hosts file modification	Always block	
☑	Suspicious behavior	Always allow	
☑	New internet explorer plugin	Ask when necessary	
☑	Internet explorer setting modification	Always block	
☑	Security policy modification	Always block	
☑	Program library injection	Always block	
☑	Shell modification	Ask when necessary	
☑	New service	Ask when necessary	
☑	System file modification	Always block	
☑	Firewall policy modification	Ask when necessary	
☑	System process modification	Always allow	
☑	New startup program	Ask when necessary	

Security Agent Alerts

□ Display alerts on devices with Behavior Monitoring violations



Web Reputation

☑ Enable Web Reputation

Security Level

○ High Blocks pages that are:

- **Dangerous** - Verified to be fraudulent or known sources of threats
- **Highly suspicious** - Suspected to be fraudulent or possible sources of threats
- **Suspicious** - Associated with spam or possibly compromised
- **Untested** - While Trend Micro actively tests web pages for safety, users may encounter untested pages when visitin untested pages can improve safety but can also prevent access to safe pages

○ Medium Blocks pages that are:

- **Dangerous** - Verified to be fraudulent or known sources of threats
- **Highly suspicious** - Suspected to be fraudulent or possible sources of threats

● Low (default) Blocks pages that are:

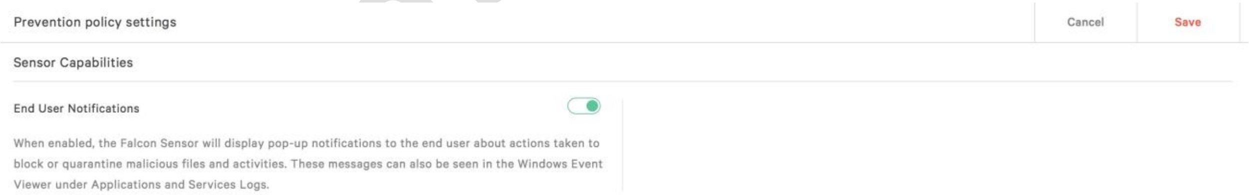
- **Dangerous** - Verified to be fraudulent or known sources of threats

Modify Global Approved URLs

Browser Exploit Prevention

☑ Block pages containing malicious script
Note: URLs in the Approved or Blocked list are not scanned for browser exploits.

10.1.3 CrowdStrike non default configuration

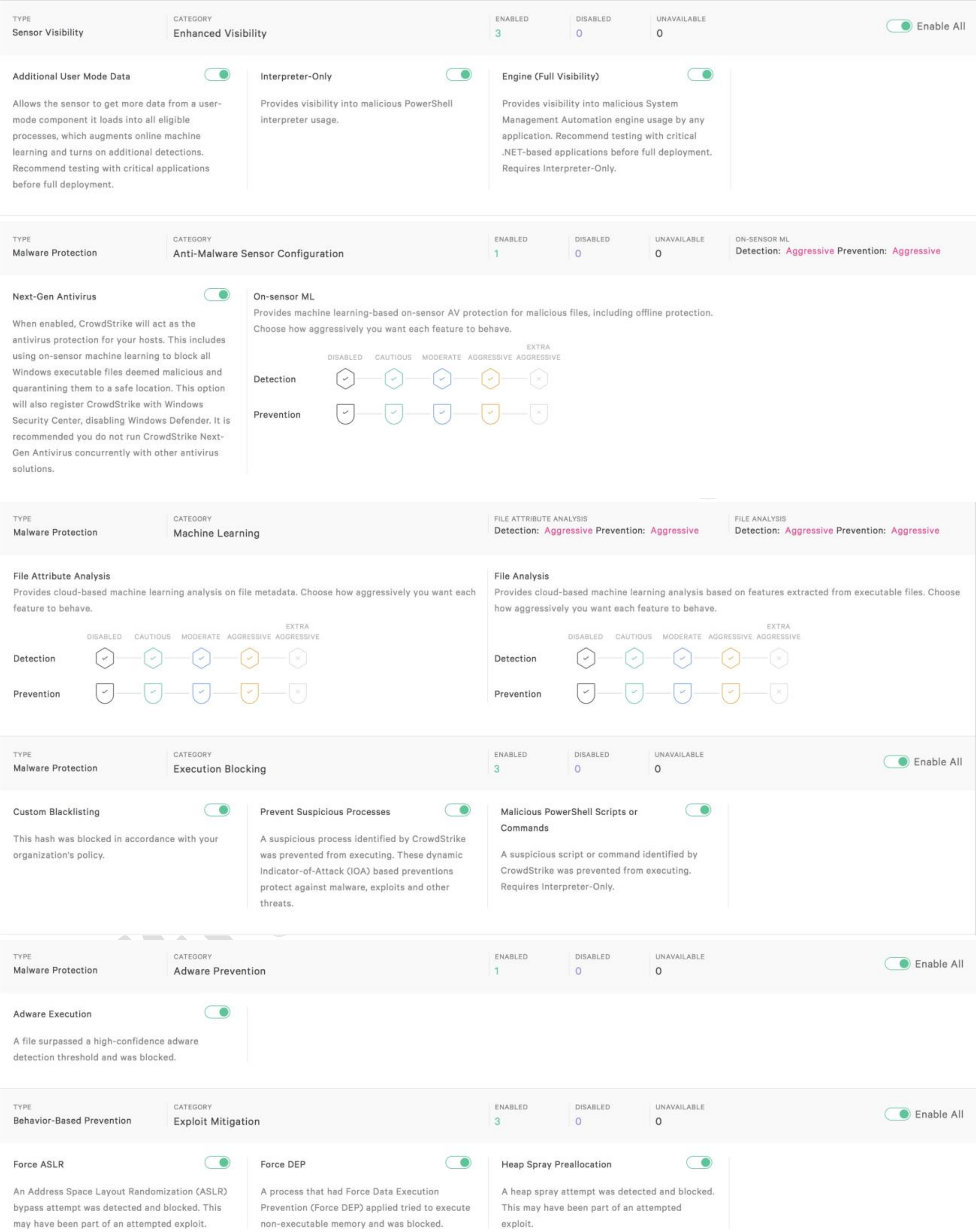


Prevention policy settings

Sensor Capabilities

End User Notifications

When enabled, the Falcon Sensor will display pop-up notifications to the end user about actions taken to block or quarantine malicious files and activities. These messages can also be seen in the Windows Event Viewer under Applications and Services Logs.



TYPE
Sensor Visibility

CATEGORY
Enhanced Visibility

ENABLED	DISABLED	UNAVAILABLE
3	0	0

Enable All

Additional User Mode Data

Allows the sensor to get more data from a user-mode component it loads into all eligible processes, which augments online machine learning and turns on additional detections. Recommend testing with critical applications before full deployment.

Interpreter-Only

Provides visibility into malicious PowerShell interpreter usage.

Engine (Full Visibility)

Provides visibility into malicious System Management Automation engine usage by any application. Recommend testing with critical .NET-based applications before full deployment. Requires Interpreter-Only.

TYPE
Malware Protection

CATEGORY
Anti-Malware Sensor Configuration

ENABLED	DISABLED	UNAVAILABLE
1	0	0

ON-SENSOR ML
Detection: Aggressive Prevention: Aggressive

Next-Gen Antivirus

When enabled, CrowdStrike will act as the antivirus protection for your hosts. This includes using on-sensor machine learning to block all Windows executable files deemed malicious and quarantining them to a safe location. This option will also register CrowdStrike with Windows Security Center, disabling Windows Defender. It is recommended you do not run CrowdStrike Next-Gen Antivirus concurrently with other antivirus solutions.

On-sensor ML

Provides machine learning-based on-sensor AV protection for malicious files, including offline protection. Choose how aggressively you want each feature to behave.

TYPE
Malware Protection

CATEGORY
Machine Learning

FILE ATTRIBUTE ANALYSIS
Detection: Aggressive Prevention: Aggressive

FILE ANALYSIS
Detection: Aggressive Prevention: Aggressive

File Attribute Analysis

Provides cloud-based machine learning analysis on file metadata. Choose how aggressively you want each feature to behave.

File Analysis

Provides cloud-based machine learning analysis based on features extracted from executable files. Choose how aggressively you want each feature to behave.

TYPE
Malware Protection

CATEGORY
Execution Blocking

ENABLED	DISABLED	UNAVAILABLE
3	0	0

Enable All

Custom Blacklisting

This hash was blocked in accordance with your organization's policy.

Prevent Suspicious Processes

A suspicious process identified by CrowdStrike was prevented from executing. These dynamic Indicator-of-Attack (IOA) based preventions protect against malware, exploits and other threats.

Malicious PowerShell Scripts or Commands

A suspicious script or command identified by CrowdStrike was prevented from executing. Requires Interpreter-Only.

TYPE
Malware Protection

CATEGORY
Adware Prevention

ENABLED	DISABLED	UNAVAILABLE
1	0	0

Enable All

Adware Execution

A file surpassed a high-confidence adware detection threshold and was blocked.

TYPE
Behavior-Based Prevention

CATEGORY
Exploit Mitigation

ENABLED	DISABLED	UNAVAILABLE
3	0	0

Enable All

Force ASLR

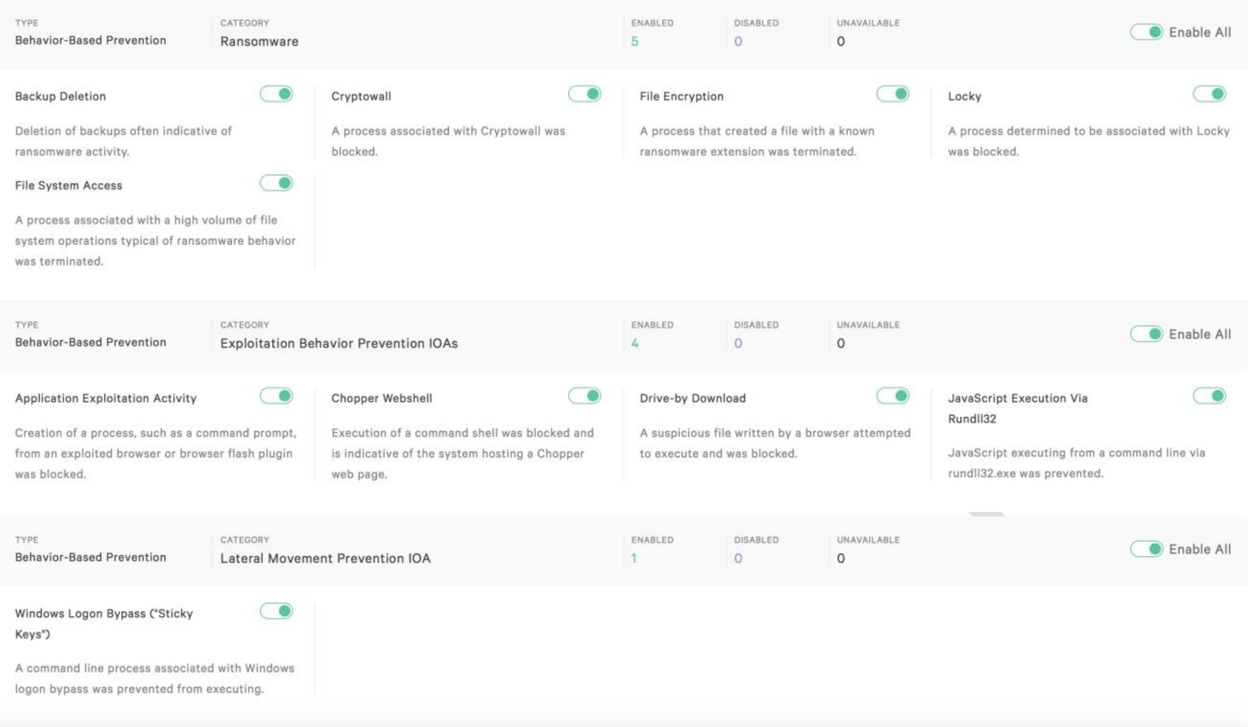
An Address Space Layout Randomization (ASLR) bypass attempt was detected and blocked. This may have been part of an attempted exploit.

Force DEP

A process that had Force Data Execution Prevention (Force DEP) applied tried to execute non-executable memory and was blocked.

Heap Spray Preallocation

A heap spray attempt was detected and blocked. This may have been part of an attempted exploit.



TYPE
Behavior-Based Prevention

CATEGORY
Ransomware

ENABLED	DISABLED	UNAVAILABLE
5	0	0

Backup Deletion

Deletion of backups often indicative of ransomware activity.

Cryptowall

A process associated with Cryptowall was blocked.

File Encryption

A process that created a file with a known ransomware extension was terminated.

Locky

A process determined to be associated with Locky was blocked.

File System Access

A process associated with a high volume of file system operations typical of ransomware behavior was terminated.

TYPE
Behavior-Based Prevention

CATEGORY
Exploitation Behavior Prevention IOAs

ENABLED	DISABLED	UNAVAILABLE
4	0	0

Application Exploitation Activity

Creation of a process, such as a command prompt, from an exploited browser or browser flash plugin was blocked.

Chopper Webshell

Execution of a command shell was blocked and is indicative of the system hosting a Chopper web page.

Drive-by Download

A suspicious file written by a browser attempted to execute and was blocked.

JavaScript Execution Via Rundll32

JavaScript executing from a command line via rundll32.exe was prevented.

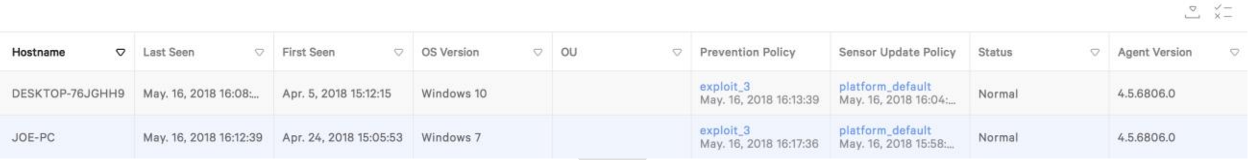
TYPE
Behavior-Based Prevention

CATEGORY
Lateral Movement Prevention IOA

ENABLED	DISABLED	UNAVAILABLE
1	0	0

Windows Logon Bypass ("Sticky Keys")

A command line process associated with Windows logon bypass was prevented from executing.

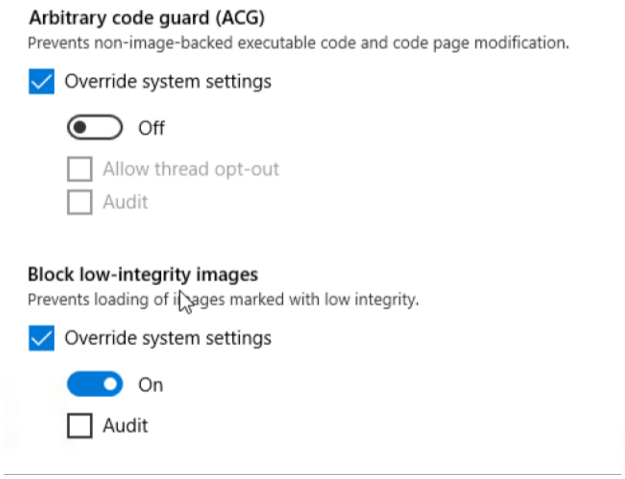


Hostname	Last Seen	First Seen	OS Version	OU	Prevention Policy	Sensor Update Policy	Status	Agent Version
DESKTOP-76JGHH9	May. 16, 2018 16:08:...	Apr. 5, 2018 15:12:15	Windows 10		exploit_3 May. 16, 2018 16:13:39	platform_default May. 16, 2018 16:04:...	Normal	4.5.6806.0
JOE-PC	May. 16, 2018 16:12:39	Apr. 24, 2018 15:05:53	Windows 7		exploit_3 May. 16, 2018 16:17:36	platform_default May. 16, 2018 15:58:...	Normal	4.5.6806.0

10.1.4 Windows 10 Defender Security Center - Exploit protection mitigations

Note: To be able to execute Internet Explorer 11, Office Word 2016, and Firefox 31.0 we could not turn on all protection mechanisms. By turning on everything applications necessary to the test were not able to start.

- Control flow guard (CFG)
- Code integrity guard
- Disable Win32k system calls
- Do not allow child processes



Arbitrary code guard (ACG)
Prevents non-image-backed executable code and code page modification.

☑ Override system settings

Off

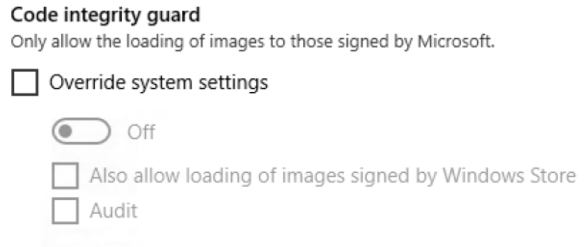
☐ Allow thread opt-out
☐ Audit

Block low-integrity images
Prevents loading of images marked with low integrity.

☑ Override system settings

On

☐ Audit

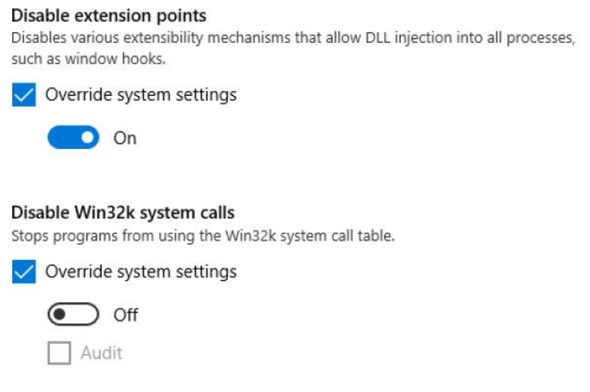


Code integrity guard
Only allow the loading of images to those signed by Microsoft.

☐ Override system settings

Off

☐ Also allow loading of images signed by Windows Store
☐ Audit



Disable extension points
Disables various extensibility mechanisms that allow DLL injection into all processes, such as window hooks.

☑ Override system settings

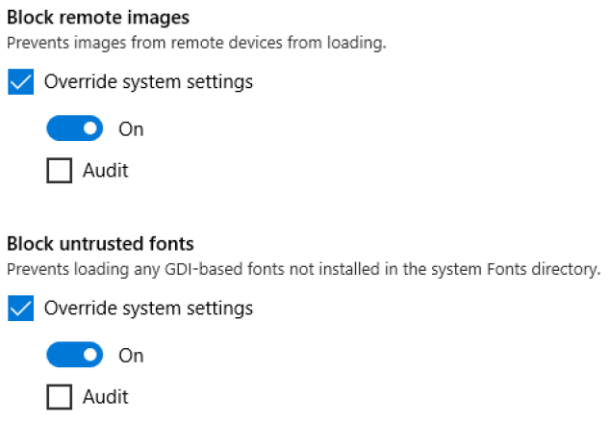
On

Disable Win32k system calls
Stops programs from using the Win32k system call table.

☑ Override system settings

Off

☐ Audit



Block remote images
Prevents images from remote devices from loading.

☑ Override system settings

On

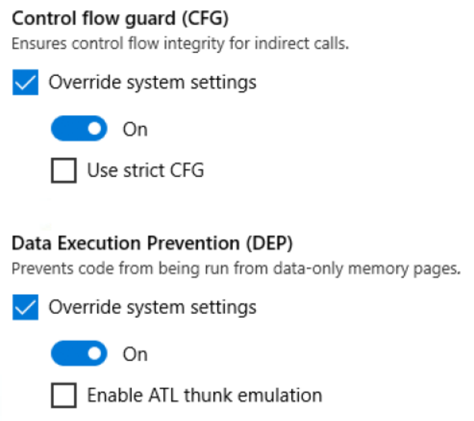
☐ Audit

Block untrusted fonts
Prevents loading any GDI-based fonts not installed in the system Fonts directory.

☑ Override system settings

On

☐ Audit



Control flow guard (CFG)
Ensures control flow integrity for indirect calls.

☑ Override system settings

On

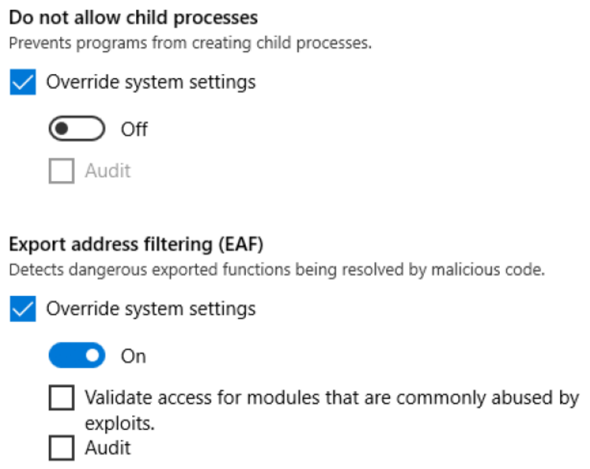
☐ Use strict CFG

Data Execution Prevention (DEP)
Prevents code from being run from data-only memory pages.

☑ Override system settings

On

☐ Enable ATL thunk emulation



Do not allow child processes
Prevents programs from creating child processes.

☑ Override system settings

Off

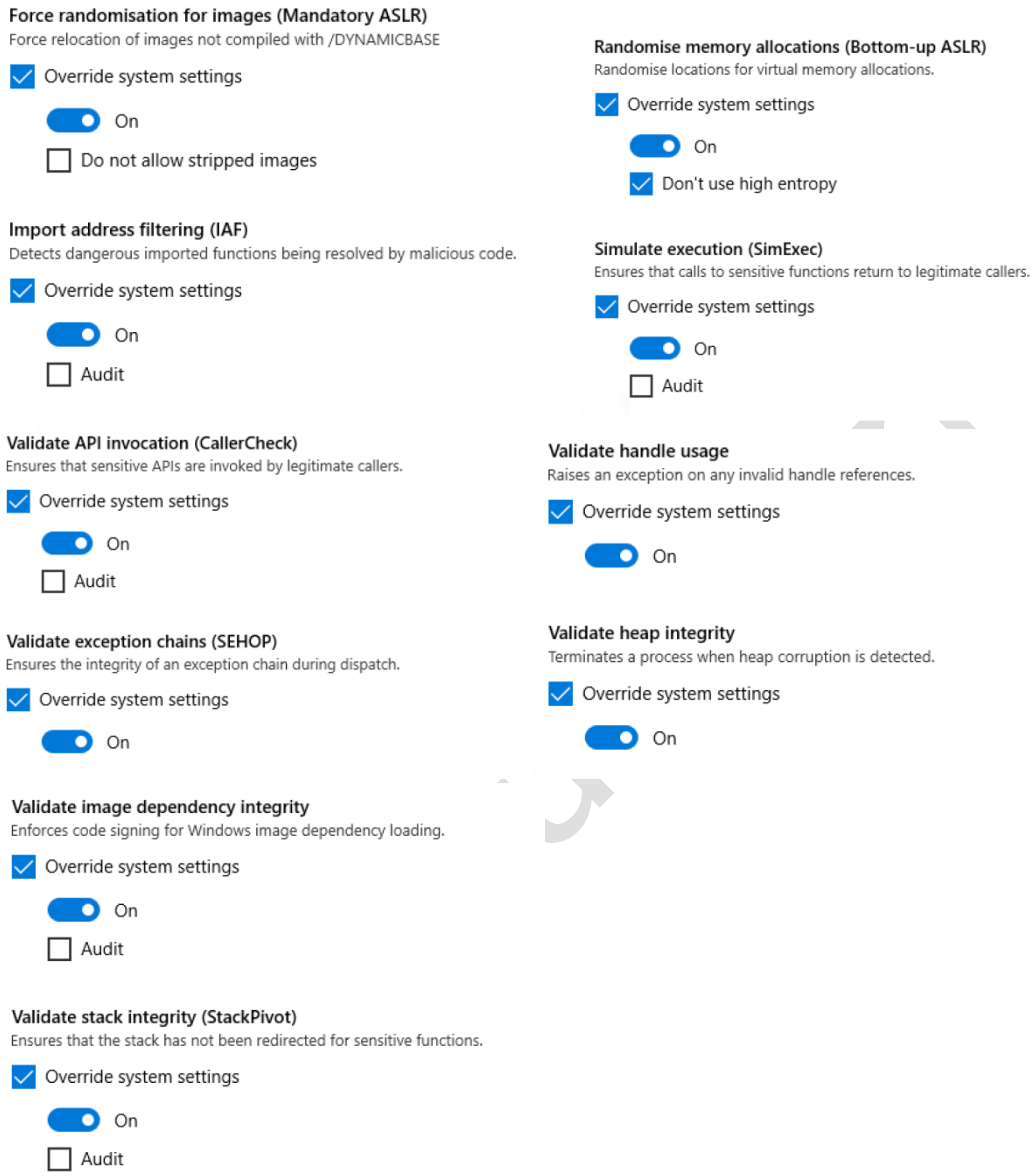
☐ Audit

Export address filtering (EAF)
Detects dangerous exported functions being resolved by malicious code.

☑ Override system settings

On

☐ Validate access for modules that are commonly abused by exploits.
☐ Audit



Force randomisation for images (Mandatory ASLR)

Force relocation of images not compiled with /DYNAMICBASE

☑ Override system settings

On

☐ Do not allow stripped images

Import address filtering (IAF)

Detects dangerous imported functions being resolved by malicious code.

☑ Override system settings

On

☐ Audit

Validate API invocation (CallerCheck)

Ensures that sensitive APIs are invoked by legitimate callers.

☑ Override system settings

On

☐ Audit

Validate exception chains (SEHOP)

Ensures the integrity of an exception chain during dispatch.

☑ Override system settings

On

Validate image dependency integrity

Enforces code signing for Windows image dependency loading.

☑ Override system settings

On

☐ Audit

Validate stack integrity (StackPivot)

Ensures that the stack has not been redirected for sensitive functions.

☑ Override system settings

On

☐ Audit

Randomise memory allocations (Bottom-up ASLR)

Randomise locations for virtual memory allocations.

☑ Override system settings

On

☑ Don't use high entropy

Simulate execution (SimExec)

Ensures that calls to sensitive functions return to legitimate callers.

☑ Override system settings

On

☐ Audit

Validate handle usage

Raises an exception on any invalid handle references.

☑ Override system settings

On

Validate heap integrity

Terminates a process when heap corruption is detected.

☑ Override system settings

On

10.2 About MRG Effitas

MRG Effitas is a UK-based, independent IT security research organisation that focuses on providing cutting-edge efficacy assessment and assurance services, the supply of malware samples to vendors and the latest news concerning new threats and other information in the field of IT security.

MRG Effitas' origin dates back to 2009 when Sveta Miladinov, an independent security researcher and consultant, formed the Malware Research Group. Chris Pickard joined in June 2009, bringing expertise in process and methodology design, gained in the business process outsourcing market.

The Malware Research Group rapidly gained a reputation as the leading efficacy assessor in the browser and online banking space and, due to increasing demand for its services, was restructured in 2011 when it became MRG Effitas, with the parent company Effitas.

Today, MRG Effitas has a team of analysts, researchers and associates across EMEA, UATP and China, ensuring a truly global presence.

Since its inception, MRG Effitas has focused on providing ground-breaking testing processes and realistically modelling real-world environments in order to generate the most accurate efficacy assessments possible.

MRG Effitas is recognised by several leading security vendors as the leading testing and assessment organisation in the online banking, browser security and cloud security spaces and has become their partner of choice.

Our analysts have the following technical certificates:

Offensive Security Certified Expert (OSCE), Offensive Security Certified Professional (OSCP), Malware Analysis (Deloitte NL), Certified Information Systems Security Professional (CISSP), SecurityTube Linux Assembly Expert, SecurityTube Python Scripting Expert, Powershell for Penetration Testers, Certified Penetration Testing Specialist (CPTS), Computer Hacking Forensics Investigator (CHFI), and Microsoft Certified Professional (MCP).