SOPHOS
Security made simple.

Exploits Explained:

Comprehensive Exploit Prevention

Exploits take advantage of weaknesses in legitimate software products like Adobe Flash and Microsoft Office to infect computers for criminal purposes. They're commonly leveraged by cybercriminals in order to penetrate organizations' defenses. The objectives of these criminals are diverse: stealing data or holding it for ransom, performing reconnaissance, or simply as a means to deploy more traditional malware.

It's common to find exploits used as part of cyber attacks: upwards of 90% of reported data breaches find that an exploit is used at one or more points in the attack chain. Including exploit prevention as part of a comprehensive lineup of security defenses is clearly valuable.

Exploits have been around for more than 30 years, so it should come as no surprise that almost every major security vendor can claim some level of exploit prevention. However, the breadth and depth of that protection varies significantly between vendors. For some, it's a box to tick; for others, it's a major focal point. Read this paper to learn more about exploits and the various levels of exploit prevention found in prominent security products.

A Sophos Whitepaper January 2018

Contents

The Exploit Industry: Crimeware as a Service

Thanks to exploit kits, malware authors don't need to worry about how to find bugs in Java or Silverlight or Flash; how to build those bugs into working exploits; how to find insecure web servers to host the exploits; or how to entice prospective victims to the booby-trapped web pages.

Likewise, exploit kit authors don't have to worry about writing full-blown malware; they don't have to run servers to keep track of infected computers or to collect money from individual victims; they don't have to get involved in the exfiltration of stolen data or selling that data.

With cybercrime now a multi-billion-dollar industry that is projected to cause nearly $2 trillion in damages by 2019, each aspect of an attack has been industrialized. Criminals have the luxury of being able to specialize in one or more parts of the threat landscape in what's become known jokingly as CaaS – or "Crimeware as a Service."

In this now-lucrative industry, exploit brokers have emerged: they buy exploits from people who discover them and sell them to people who want to make use of them, whether government agencies or nefarious hackers.

Buyers invariably keep their purposes to themselves. As Kevin Mitnick, founder of Mitnick's Absolute Zero Day Exploit Exchange, explained to Wired, "When we have a client that wants a zero-day vulnerability for whatever reason, we don't ask, and in fact they wouldn't tell us. Researchers find them, they sell them to us for X, we sell them to clients for Y and make the margin in between."

"When we have a client that wants a zero-day vulnerability for whatever reason, we don't ask, and in fact they wouldn't tell us. Researchers find them, they sell them to us for X, we sell them to clients for Y and make the margin in between."

Kevin Mitnick

Exploit Mitigation Techniques

With more than 400,000 unique malware samples created each day and thousands of new vulnerabilities discovered each year, the challenge of preventing malicious attacks is daunting. This explosion of growth in malware variants requires new and innovative approaches when it comes to defending against cybercriminals.

A careful examination of the modern cybercrime industry shows an opportunity for asymmetric defense. As it turns out, despite the seemingly endless parade of new attacks, there are only about two dozen techniques that can be used to exploit software. So an approach that's able to counteract this handful of exploit techniques – instead of targeting each and every exploit – is extremely powerful.

What's more: depending on the vulnerability, attackers often end up having to chain a handful of exploit techniques together to get to the stage where they can deliver malware. These techniques don't change much from year to year: perhaps one or two new tricks are added to the list of available techniques.

When evaluating major security products, the absence of significant exploit technique mitigation can be surprising. And while some of the newer vendors who claim to offer next-generation technology have broader support for exploit mitigation, even here the coverage is spotty.

Below is a list of exploit mitigations that are aimed to eliminate entire classes or vulnerabilities and break the exploit techniques that are used by cybercriminals and nation-states. Mitigations for each technique will vary by vendor. It is important to know that when a vendor claims to prevent exploits, most vendors simply protect against a fraction of the commonly used exploit methods and are often not in effect on 64-bit applications. Only Sophos provides truly comprehensive exploit prevention.

Enforce Data Execution Prevention (DEP)

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.

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.

Bottom-up ASLR

If enabled, Bottom-Up ASLR improves the entropy or randomness of mandatory ASLR.

The main advantage of Mandatory ASLR and Bottom-Up ASLR in Sophos Intercept X is that base addresses of applications are not only randomized on each reboot, but also each time the protected application is started.

Null Page (Null Dereference Protection)

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.

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 Shui[1] 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.

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.

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.

1 https://cansecwest.com/slides/2014/The%20Art%20of%20Leaks%20-%20read%20version%20-%20Yoyo.pdf

Stack Exec (MemProt)

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[2], 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.

Stack-based ROP Mitigation (Caller)

To defeat security technologies like data execution prevention (DEP) and address space layout randomization (ASLR), attackers typically resort to hijacking control-flow of vulnerable internet-facing applications. Such in-memory attacks are invisible to antivirus, most "next-gen" products, and other cyber 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 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.

Since the contents of the stack are writable, an attacker can write specific values on the stack to mislead the analysis of the stack-based ROP defense. The stack-based ROP defense cannot determine if the contents of the stack are benign or manipulated by an attacker.

Branch-based ROP Mitigation (Hardware Augmented Control-Flow Integrity)

As previously explained, stack-based defenses against return-oriented programming (ROP) are coarse-grained and manipulation-prone. To improve this, defenders would more fine-grained and manipulation-resistant data to analyze at run-time.

Sophos Intercept X introduces hardware augmented Control-Flow Integrity (CFI) by leveraging an unused hardware feature available in mainstream Intel® processors (from 2008 and newer). The processor hardware itself offers read-only data to augment the detection of sophisticated exploit attacks at run-time. Employing hardware-traced (branch) records has a significant security benefit over software stack-based approaches. The branch information that can be retrieved from these records not only identifies the target of the branch, but also the source. So, it actually shows where the change in control-flow originated from. This specific information cannot be obtained with the same level of confidence using a stack-based solution like Microsoft EMET or Palo Alto Networks Traps.

2 https://en.wikipedia.org/wiki/Stack_buffer_overflow

Branch information in the hardware-traced records cannot be manipulated; there's no way for it to be overwritten with controlled data by an attacker. Stack-based approaches rely on stack data, which is – especially in case of a ROP attack – under control of the attacker, who in turn can mislead the defender. In contrast, the hardware-traced data examined by Sophos Intercept X is more reliable and tamper resistant.

An alternative hardware-assisted control-flow integrity implementation (HA-CFI) from Endgame is based on training regular control-flow to detect deviation from the programmer intended code path. It must be continuously trained to build a whitelist of valid code pointer addresses reflecting all possible functionality and versions of the protected application. Sophos Intercept X does not require training and also functions correctly during thread context switches and dynamic frequency scaling.

Sophos Intercept X will automatically employ hardware augmented control-flow tracing when it detects an Intel® Core™ i3, i5, or i7 processor (CPU). If supported processor hardware is not detected, Sophos Intercept X will automatically fall back on software-only stack-based integrity checks.

Sophos Intercept X does not only leverage hardware-traced records to augment ROP detection. It is also used for Import Address Filtering (IAF), to protect the import address table of protected applications.

Note: The patches to fix the Spectre vulnerabilities regarding the branch predictor inside Intel CPU hardware do not affect the correct functioning of Sophos Intercept X.

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.

Import Address Table Access Filtering (IAF)

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.

Sophos Intercept X introduces hardware augmented Import Address Table Access Filtering by leveraging hardware features available in mainstream Intel® processors (from 2008 and newer). In addition to the hardware-traced branch records to enforce Control-Flow Integrity, Sophos Intercept X also leverages the hardware branch prediction to further enhance the protection of the import address table.

Note: The patches to address the Spectre vulnerabilities regarding the branch predictor inside Intel CPU hardware do not affect the correct functioning of Sophos Intercept X.

Load Library

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.

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.

Shellcode

A shellcode is a small piece of machine code used as the payload in the exploitation of a software vulnerability. It is called 'shellcode' because historically it would start a command shell from which the attacker can control the compromised machine, but any piece of code that performs a similar task can be called shellcode.

An exploit will commonly inject a shellcode into the target process before or at the same time as it exploits a vulnerability to gain control over the processor instruction pointer (EIP/RIP). The instruction pointer is adjusted to point to the shellcode, after which it gets executed and performs its task.

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[3].

As an example, an attacker can modify the safemode flag value by leveraging the CVE-2014-6332 vulnerability[4], 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.

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

3 https://en.wikipedia.org/wiki/Glossary_of_video_game_terms#God_mode
4 https://www.rapid7.com/db/modules/exploit/windows/browser/ms14_064_ole_code_execution

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.

On 64-bit editions of Windows, Sophos Intercept X prohibits the program code from directly switching from 32-bit to 64-bit mode (e.g. using ROP), while still enabling the WoW64 layer to perform this transition.

For more information about abusing WoW64, see research from Duo Security: WoW64 and So Can You[5] and Mitigating Wow64 Exploit Attacks[6].

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.

Sophos Intercept X includes a novel approach to prevent attackers from addressing sensitive kernel functions via API functions that are unprotected.

For more information about abusing syscalls, see the BreakDev.org blog entry titled Defeating Antivirus Real-time Protection From The Inside[7].

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.

5 https://duo.com/blog/wow64-and-so-can-you
6 https://hitmanpro.wordpress.com/2015/11/10/mitigating-wow64-exploit-attacks
7 https://breakdev.org/defeating-antivirus-real-time-protection-from-the-inside/

Process Doppelgänging

Most Windows computers use the NTFS file system. In 2007 Microsoft introduced a new feature called Transactional NTFS (TxF). This feature allows multiple file operations to be treated as a whole: they can either succeed as a whole – and be committed, or fail as a whole – and be rolled back. This way, an application could make many changes to many files on disk and roll back all files to the original state if an error is detected. The most common use of TxF is during installations of Windows updates.

Process Doppelgänging exploits the TxF mechanism to hide malware. It chooses an innocent file, overwrites it and runs the malware via a low-level API to e.g. impersonate a trusted file (similar to process hollowing). Just before letting the malware run – it rejects or rolls back all changes thus preventing antivirus software from scanning the file content that is really being executed. If opened, the file on disk will contain no suspicious content. Moreover, this file can be a well-known, digitally signed application.

DLL Hijacking

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.

Dynamic Data Exchange (DDE)

Windows Dynamic Data Exchange (DDE) is a client-server protocol for inter-process communication (IPC) between applications. Attackers may use DDE to execute arbitrary commands. For example, Microsoft Office documents can be poisoned with DDEAUTO commands and used to deliver execution of PowerShell commands via spear phishing campaigns or hosted Web content, avoiding the use of Visual Basic for Applications (VBA) macros. It is also possible to embed DDEAUTO commands in the message body of emails or meeting requests, which are executed when replying or accepting them in Microsoft Outlook.

Thanks to the design of the Application Lockdown mitigation, Sophos Intercept X inherently prevents malicious code execution via Dynamic Data Exchange too.

Application Lockdown

In the event an attacker successfully exploits and bypasses all memory and code mitigations, Sophos Intercept X limits an attacker's abilities. This feature, called Application Lockdown, aims to prevent attackers from introducing unwanted code.

Application Lockdown stops attacks that do not typically rely on software bugs in applications. Such an attack could be the use of a crafted (malicious) macro in an office document attached to a (spear) phishing email, for example. Macros in documents are potentially dangerous as they are created in the Visual Basic for Applications (VBA) programming language, which includes the ability to download and run binaries from the web and also allows the use of PowerShell and other trusted applications.

This unexpected feature (or logic-flaw exploit) offers attackers an obvious advantage as they do not need to exploit a software bug or find a way to bypass code and memory defenses in order to infect computers. They simply abuse standard functionality offered by a widely-used trusted application and only need to use social engineering to persuade the victim to open the specially crafted document.

Without the need to maintain a blacklist of folders, Sophos Intercept X will automatically terminate a protected application based on its behavior; for example, when an office application is leveraged to launch PowerShell, access the WMI, run a macro to install arbitrary code or manipulate critical system areas, Sophos Intercept X will block the malicious action – even when the attack doesn't spawn a child process.

Java Lockdown

Previously, exploit kits were key enablers of drive-by-downloads of malware. They leveraged vulnerabilities in the Java Runtime Environment (JRE) to drop a Windows PE payload. JRE is loaded as a plug-in or add-on in mainstream browsers.

Sophos Intercept X prevents JRE from running non-Java applications. For example, Sophos Intercept X will terminate a Java application when it attempts to introduce and run a Windows PE binary. In addition, attackers cannot abuse Java to manipulate auto-starting locations, including the startup folder, Run, RunOnce, and other Registry keys.

Note: With the introduction of Java 8 update 20 in 2014, the security level for Java applications is default set to High. This made it increasingly more difficult for attackers to run Java exploits with sufficient permissions to infect the endpoint. Consequently, Java exploits are no longer a favorite with exploit kits, which makes the Java Lockdown mitigation somewhat deprecated.

Code Cave

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.

In a so-called supply chain attack, an attacker could also breach the software update servers on which an update can be laced with malicious code to silently infect its customers with e.g. ransomware or wiper malware.

Sophos Intercept X automatically blocks execution of applications that are laced with a backdoor. It even detects the added shellcode when code execution doesn't flow to a code cave or to an added section in the infected PE file. It offers broad protection against shellcode injection tools like Shellter and Backdoor Factory.

Process Migration – remote reflective DLL injection

Process migration is a common technique performed by an adversary when they first establish their presence on a device and want to move to another process to either escalate privileges or gain more enduring access. The adversary does not want to lose control when the end user simply closes their browser or terminates a process that has been compromised, so migrating to a system process is desired.

A remote reflective DLL attack is similar to process migration. The adversary has already compromised one process and from there they are manipulating another process to load DLLs, and run arbitrary code.

Local Privilege Escalation (LPE)

Sophos Intercept X prevents a low-privileged process from being escalated via a token stolen from a process with greater privileges. This technique is often used in tandem with another vulnerability to successfully deliver and run an attacker's malicious code with system permissions.

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. Sophos Intercept X inherently breaks the fundamental method used by DoublePulsar and therefor also stops attacks that rely on the same technique for code injection.

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.

DoubleAgent Code Injection

DoubleAgent exploits a legitimate tool of Windows called Microsoft Application Verifier. This tool is included in all versions of Microsoft Windows and is used as a runtime verification tool in order to discover and fix bugs in applications. The Application Verifier can be set to load any library from the disk, thus opening the possibility of loading a malicious library that will be given the permissions of the victim process.

DoubleAgent is coined as a vulnerability and zero-day attack on antivirus products but in reality, the purpose of Application Verifier is to load arbitrary code into any chosen application, including trusted productivity and Windows processes.

Sophos Intercept X prevents code injection via abuse of the Application Verifier.

Intercept X features

Features	
EXPLOIT PREVENTION	
Enforce Data Execution Prevention	✓
Mandatory Address Space Layout Randomization	✓
Bottom-up ASLR	✓
Null Page (Null Deference Protection)	✓
Heap Spray Allocation	✓
Dynamic Heap Spray	✓
Stack Pivot	✓
Stack Exec (MemProt)	✓
Stack-based ROP Mitigations (Caller)	✓
Branch-based ROP Mitigations (Hardware Assisted)	✓
Structured Exception Handler Overwrite (SEHOP)	✓
Import Address Table Filtering (IAF)	✓
Load Library	✓
Reflective DLL Injection	✓
Shellcode	✓
VBScript God Mode	✓
Wow64	✓
Syscall	✓
Hollow Process	✓
DLL Hijacking	✓
Squiblydoo Applocker Bypass	✓
APC Protection (Double Pulsar / AtomBombing)	✓
Process Privilege Escalation	✓
ACTIVE ADVERSARY MITIGATIONS	
Credential Theft Protection	✓
Code Cave Mitigation	✓
Man-in-the-Browser Protection (Safe Browsing)	✓
Malicious Traffic Detection	✓
Meterpreter Shell Detection	✓

Features	
ANTI-RANSOMWARE PREVENTION	
Ransomware File Protection (CryptoGuard)	✓
Automatic file recovery (CryptoGuard)	✓
Disk and Boot Record Protection (WipeGuard)	✓
APPLICATION LOCKDOWN	
Web Browsers (including HTA)	✓
Web Browser Plugins	✓
Java	✓
Media Applications	✓
Office Applications	✓
DEEP LEARNING	
Deep Learning Malware Detection	✓
Deep Learning Potentially Unwanted Applications (PUA) Blocking	✓
False Positive Suppression	✓
Live Protection	✓
RESPOND INVESTIGATE REMOVE	
Root Cause Analysis	✓
Sophos Clean	✓
Synchronized Security Heartbeat	✓
DEPLOYMENT	
Can run as standalone agent	✓
Can run alongside existing antivirus	✓
Can run as component of existing Sophos Endpoint agent	✓
Windows 7	✓
Windows 8	✓
Windows 8.1	✓
Windows 10	✓
macOS*	✓

* features supported CryptoGuard, Malicious Traffic Detection, Synchronized Security Heartbeat, Root Cause Analysis

Try Sophos Intercept X for free

at sophos.com/intercept-x

Statements contained in this document are based on publicly available information as of November 30, 2016. This document has been prepared by Sophos and not the other listed vendors. The features or characteristics of the products under comparison, which may directly impact the accuracy or validity of this comparison, are subject to change. The information contained in this comparison is intended to provide broad understanding and knowledge of factual information of various products and may not be exhaustive. Anyone using the document should make their own purchasing decision based on their requirements, and should also research original sources of information and not rely only on this comparison while selecting a product. Sophos makes no warranty as to the reliability, accuracy, usefulness, or completeness of this document. The information in this document is provided "as is" and without warranties of any kind, either expressed or implied. Sophos retains the right to modify or withdraw this document at any time.

United Kingdom and Worldwide Sales
Tel: +44 (0)8447 671131
Email: sales@sophos.com

North American Sales
Toll Free: 1-866-866-2802
Email: nasales@sophos.com

Australia and New Zealand Sales
Tel: +61 2 9409 9100
Email: sales@sophos.com.au

Asia Sales
Tel: +65 62244168
Email: salesasia@sophos.com

© Copyright 2018. Sophos Ltd. All rights reserved.
Registered in England and Wales No. 2096520, The Pentagon, Abingdon Science Park, Abingdon, OX14 3YP, UK
Sophos is the registered trademark of Sophos Ltd. All other product and company names mentioned are trademarks or registered trademarks of their respective owners.

18-01-05 WP-NA (2899-DD)

SOPHOS