Anti-Virus Comparative

Exploit Test for SentinelOne

Language: English
December 2016
Last revision: 14th December 2016

http://www.av-comparatives.org
http://www.mrg-effitas.com

Commissioned by SentinelOne

Content

Introduction

This exploit test has been commissioned by SentinelOne. SentinelOne updated and configured their product for optimal exploit protection. The test, which consisted of 20 exploits, has been performed in December 2016.

Tested Products

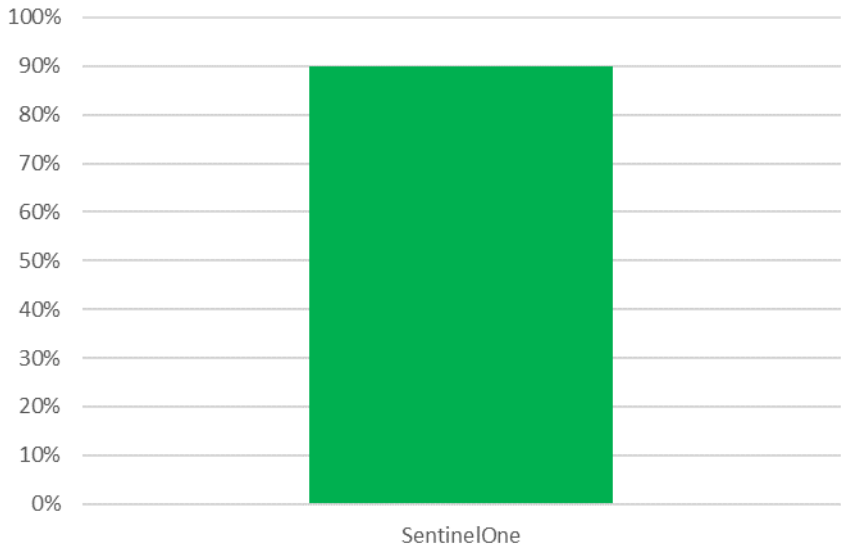
The following product has been under Windows 10 64-bit:

Vendor	Product	Version
SentinelOne	Endpoint Protection Platform	1.8.3.5028

Results

Exploit Protection Test

Below are the results achieved by Sentinel One in the exploit test:



	Protected	Detected
Sentinel One	90%	90%

Scoring / Calculation of Results

Scoring Method for Exploit Protection/Detection

We have defined the following stages at which the functioning of the exploit can be prevented by the endpoint protection system:

1. Blocking any relevant URL (infected URL, exploit kit URL, redirection URL, malware URL) by the URL database (local or cloud). For example, a typical result is the browser displaying a "Site has been blocked" message from the endpoint protection. The sooner the threat is detected in the exploit chain, the easier it is to remove the malicious files from the system, the less information can be gathered from the system by the attackers, and the lower the risk of an attack targeting the particular security solution on the endpoint.

2. Analysing and blocking a web page containing a malicious HTML code, JavaScripts (redirects, iframes, obfuscated JavaScripts, etc.), or Flash files.

3. Blocking the exploit before the shellcode is executed.

4. Blocking the downloaded payload by analysing the malware before it is started. For example, the malware payload download (either the clear-text binary or the encrypted/encoded binary) can be seen in the proxy traffic, but no malware process starts.

5. The malware execution is blocked (no process created or library loaded).

6. There was a successful start by the dropped malware.

7. There was a successful start by the dropped malware, but after some time, all dropped malware was terminated and deleted ("malware starts, but blocked later").

The "protection" scoring of the results was calculated as the followings:

- If no malicious untrusted code was able to run on the endpoint, 5 points were given to the product. This can be achieved via blocking the exploit in any of steps 1, 2 or 3.
- If malicious untrusted code (exploit shellcode, downloader code) was able to run on the system, but the final malware was not able start, 4 points were given to the product. This can be achieved via blocking the exploit in either of steps 4 or 5.
- If both the exploit shellcode (or downloader code) and the final malware were able to run, 0 points were given to the product.

The "detection" scoring of the results was calculated as follows:

- If at any stage of the infection process, a medium or high severity alert was generated (even if the infection was not prevented), 1 point was given to the product.

We used this scoring for the following reasons:

- The scope of the test was exploit prevention and not the detection of malware running on the system.
- It is not possible to determine what kind of commands were executed or what information was exfiltrated by the malware. Data exfiltration cannot be undone or remediated.
- It cannot be determined if the malware exited because the endpoint protection system blocked it, or if the malware quit because it detected e.g. monitor processes or virtualization, or because it did not find its target environment.
- Checking for malware remediation can be too time-consuming, and remediation scoring very difficult, in an enterprise environment. For example, in recent years we have experienced several alerts stating that the endpoint protection system blocked a URL/page/exploit/malware, but the malware was still able to execute and run on the system. On other occasions, the malware code was deleted from the disk by the endpoint protection system, but the malware process was still running, or some parts of the malware were detected and killed, while others were not.
- In a complex enterprise environment, multiple network and endpoint products protect the endpoints. If one network product alerts that malicious binary has been downloaded to the endpoint, administrators have to cross-check the alerts with the endpoint protection alerts, or do a full forensic investigation to be sure that no malware was running on the endpoint. This process can be time/resource consuming, which is why it is better to block the exploit before the shellcode starts.
- Usually the exploit shellcode downloads and executes a new piece of malware in one step, but in targeted attacks, the exploit shellcode can be more complex.

We believe that such zero-tolerance scoring helps enterprises to choose the best products, using simple metrics. Manually verifying the successful remediation of the malware in an enterprise environment is a very resource-intensive process and costs a lot of money. In our view, malware needs to be blocked before it has a chance to run, and no exploit shellcode should be able to run.

Test Procedure / Methodology

Exploit Test Setup

Testing Cycle for Each Test Case

1) One default-install Windows 10 x64 virtual machine (VirtualBox) endpoint was created. The default HTTP/HTTPS proxy was configured to point to a proxy running on a different machine. SSL/TLS traffic was not intercepted on the proxy.

2) The security of the OS was weakened by the following actions:
   a) Microsoft Defender was disabled
   b) Internet Explorer SmartScreen was disabled
   c) Vulnerable software was installed, see "Software Installed" for details.
   d) Windows Update was disabled

3) At this point, different snapshots of the virtual machine were created, one each for every endpoint protection product, and one with none. This procedure ensured that the base system was exactly the same in all test systems.

   The following endpoint security suites, with the following configuration, were defined for this test:

   a) No additional protection; this snapshot was used to infect the OS and to verify the exploit replay
   b) Product 1 installed
   c) Product 2 installed
   d) …

The endpoint protection products were installed with their respective default configurations, removal of potentially unwanted software was enabled, and if it was an option during installation, cloud/community participation was enabled.

4) The exploit sources can be divided into two categories. In-the-wild threats and Metasploit. VBScript-based downloaders and Microsoft Office documents containing macros were also in the scope, as these threats are usually not included in other test scenarios.

5) The virtual machine was reverted to a clean state and traffic was replayed by the proxy server. The replay meant that the browser was used as before, but instead of the original webservers, the proxy server answered the requests based on the recorded traffic. When the "replayed exploit" was able to infect the OS, the exploit traffic was marked as a source for the tests. This method guarantees that exactly the same traffic will be seen by the endpoint protection systems, even if the original exploit kit goes down during the tests. This exploit replay is NOT to be confused with tcpreplay-type replay.

6) After new exploit traffic was approved, the endpoint protection systems were tested. Before the exploit site was tested, it was verified that the endpoint protection had been updated to the latest version with the latest signatures, and that all cloud connections were working. If there was a need to restart the system, it was restarted. In the proxy setup, unmatched requests were allowed to pass through, and SSL/TLS was not decrypted to ensure AV connectivity. VPN was used during the test on

the host machine. When user interaction was needed from the endpoint protection (e.g. site visit not recommended, etc.), the block/deny action was chosen. When user interaction was needed from Windows, we chose the run/allow options. No other processes were running on the system, except the Process Monitor/Process Explorer from SysInternals and Wireshark (both installed to non-default directories).

7) After navigating to the exploit site, the system was monitored to check for new processes, loaded DLLs or C&C traffic.

8) The process went back to step 5, until all exploit site test cases had been tested.

The following hardware was dedicated to the virtual machine:

- 4 GB RAM memory
- 2 processors dedicated from AMD FX 8370E CPU
- 65 GB free space
- 1 network interface
- SSD

The VirtualBox host and guest system for the exploit test has been hardened in such a way that common virtualization and sandbox detection techniques cannot detect the system as an analysis system.

Analysis of the Exploit Kits Used in the Exploit Test

Unfortunately, the timing of the test and OS configuration were not in our favour. At the time of the test, the Internet was dominated by just two exploit kits, Sundown and RIG. That is why we felt it was important to test with Metasploit, to diversify the exploit types used in the test.

We also used three samples, that are not in themselves exploits, but rather non-PE downloaders, such as Microsoft Office macros and OLE downloaders. We added these into the mix because these "exotic" file types are often excluded from real-world tests, but remain prevalent in-the-wild.

A total of 20 test cases were used.

- 10 RIG EK (Rig-E, Rig-V, Rig standard)
- 2 Sundown EK
- 3 Metasploit
- 1 Powershell Empire
- 1 Metasploit Macro
- 1 Locky macro
- 1 Dridex macro
- 1 Dridex OLE

These exploit kits targeted Adobe Flash, Internet Explorer, Microsoft Office (macro), Silverlight, Firefox, Java.

Software Installed

For the exploit test part, the following vulnerable software was installed:

Vendor	Product	Version	Vendor	Product	Version
Adobe	Flash Player ActiveX - built-in	18.0.0.160	Microsoft	SilverLight	5.1.10411.0
AutoIT	AutoIT	3.3.12.0	Mozilla	Firefox	31.0
Microsoft	Internet Explorer	11.162.10586	Oracle	Java	1.7.0.17
Microsoft	Office	2016			

Copyright and Disclaimer