MRG Effitas

Efficacy Assessment & Assurance

Webroot SecureAnywhere Business Endpoint Protection Versus ESET Endpoint Security Comparative Analysis 2016 June

Table of Contents

1 Introduction

MRG Effitas is a testing and research organization that specializes in specific tests. For example, our Online Banking/Browser Security tests focus on all the dangers users face when they conduct online purchases or online banking transactions.

MRG Effitas has also developed a 360 Protection Test that utilizes a unique testing scenario where we do not only focus on detection capabilities of security products but also on the time needed to detect and neutralize samples that were capable of bypassing them.

MRG Effitas also conducts Exploit protection testing, APT protection testing and performance testing.

MRG Effitas was commissioned by Webroot to conduct a comparative assessment. Webroot commissioned MRG Effitas for a comparative analysis of its Webroot SecureAnywhere Business Endpoint Protection product, and ESET Endpoint Security.

1.1 Webroot SecureAnywhere Business Endpoint Protection

Webroot SecureAnywhere Business Endpoint Protection (WSAB) is next-generation endpoint protection software, which includes antivirus (signature-less), antispyware, anti-phishing, personal firewall, password and identity protection for Microsoft Windows. It also includes cloud management capabilities.

1.2 ESET Endpoint Security

ESET Endpoint Security provides cloud-based enterprise endpoint protection, which includes antivirus, anti-phishing, host intrusion prevention, personal firewall, anti-spam and other protective features for Microsoft Windows.

1.3 Executive summary

The purpose of this report is to run a comprehensive comparative assessment of two enterprise endpoint protection products: Webroot SecureAnywhere Business Endpoint Protection and ESET Endpoint Security.

In this assessment we used a wide spectrum of tests to cover all possible threats that any enterprise environment faces. The most important testing metric in this comparative assessment is the Time to Detect.

As endpoints get compromised on an ever greater scale, we cannot run a simple detection test to determine a product's effectiveness. The Time to Detect metric focuses on the time needed for a successfully running threat to be detected and neutralized. In this comparative assessment we used a 24h interval to measure the Time to Detect.

In addition to the Time to Detect metric, we also compared the two products' Phishing protection, Performance/System impact and Feature comparisons.

In 2010, MRG Effitas began reverse engineering financial malware to create simulators that employ the same "Man in the Browser" attacks as the in-the-wild code, and so were for the first time able to determine whether secure browsers were capable of preventing data exfiltration. This was so revolutionary that in 2012 the BBC based a TV programme on our work – BBC Click, "The Man in the Browser".

See http://www.youtube.com/watch?v=DUnZMwXCkyw

Why do we use simulators? We have been asked this question countless times in the past and we always answer such questions with the following:

Simulators are used in every industry and sector, including aerospace, automotive, law enforcement, the military and finance. Nobody questions the validity of using simulators in these sectors as it is a well-known fact that simulators improve performance.

There are two major types of simulators, one that is used to teach students (e.g. pilots) and the other to simulate various types of attacks (e.g. military). This is exactly why MRG Effitas decided to start creating simulators. By developing test tools we try to simulate attacks that may not be as prevalent at present but may become more so in the future (which can be just around the corner). Simulators can point out potential weaknesses in products and even use new types of attacks that can be useful for developers as they can learn about these from a testing lab, rather than from their users when an attack of this type occurs in the wild.

All the attack methods implemented by our simulators are valid and could be used or are being used by certain types of less prevalent malware. It should be noted that high prevalence results if a known type of malware is used in large scale attacks. However, as highlighted before, some malware attacks cannot be used in large scale attacks, but the outcome can be even more lucrative than with the highly prevalent ones. In addition to the simulators, we also tested the proactive protection of these products with real botnet, a recent Zeus sample (ZeusVM/KINS).

When conducting these tests, we tried to simulate normal user behaviour. We are aware that a "Real World" test cannot be conducted by a team of professionals inside a lab because we understand how financial malware works, how it attacks and how such attacks could be prevented. Simulating normal user behaviour means that we paid special attention to all alerts given by security applications. A pass was given only when alerts were straightforward and clearly suggested that malicious action should be blocked.

Final results

Based on the number of different tests, Webroot SecureAnywhere performed better in the time-to-detect test, simulators and botnets, phishing, performance test and at the functionalities compared to ESET Endpoint Security. In the remediation, self-protection and in management and reporting capabilities both products performed the same.

2 Tests employed

It is no secret that when it comes to malware, vendors have a lot of work on their hands. Bad guys use various techniques to evade detection. Luckily, so far AV developers have been able to respond to these "enhancements" swiftly.

All tests were done in a fully patched Windows 7 64-bit.

The tested version of Webroot SecureAnywhere Business Endpoint Protection was 9.0.8.100, and that of ESET Endpoint Security 6.32016.0. The test was carried out between May 15 and June 13, 2016.

2.1 High-level overview of the tests

Webroot SecureAnywhere works like a cloud manageable enterprise endpoint protection, in combination with browser protection (called identity protection), but it uses signature-less protection along with different shield technologies.

ESET Endpoint Security works like a traditional enterprise endpoint protection, which includes antivirus, anti-phishing, host intrusion prevention, personal firewall, anti-spam and other protective features for Microsoft Windows.

In order to gain better insight into the functionalities of Webroot SecureAnywhere and ESET Endpoint Security, we employed a combination of in-the-wild malware test (protection, time-to-detect), phishing, performance, self-protection, reporting capabilities, deployment time, function comparison, etc.

During the tests, we used a default install of the products; Potentially Unwanted Application detection was turned on, on every product and browser extensions were enabled and installed.

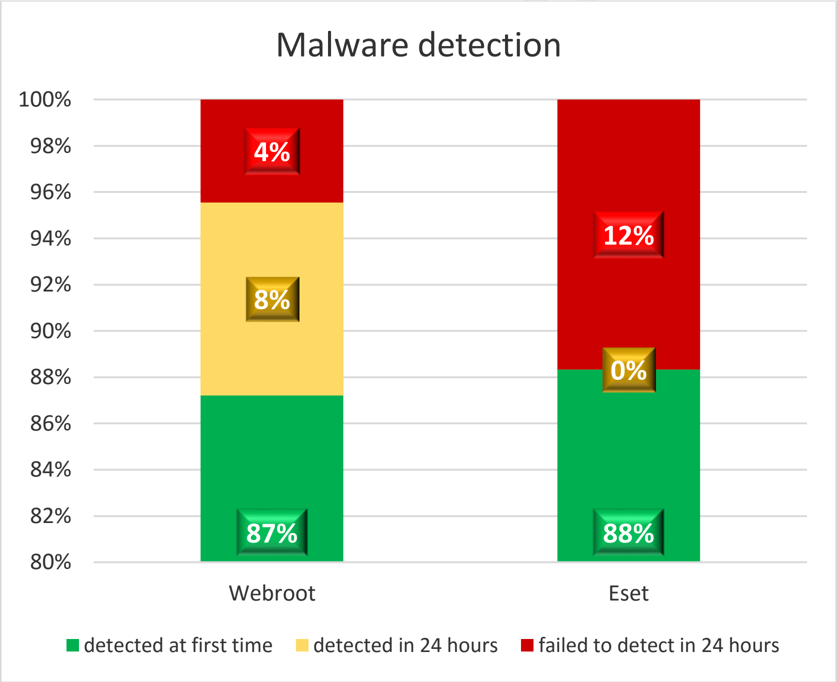
2.2 Malware protection (time-to-protect) test

Sample selection is of fundamental importance to this and all similar tests. All samples used were "live" and "in the wild", by which we mean that they reside at the URLs selected or created by the cybercriminals, and not from a time lagged ITW list. As these are live ITW samples, they represent current zero day-threats that can be an issue with sample verification. There is no effective and reliable way to verify samples before testing that does not introduce possible artificial sample submission or delay, so all verification is conducted after testing. Tests performed using samples that are later proven to be invalid are excluded from the results. The type of samples used is selected by MRG Effitas on the basis of a mixture of criteria, cantering about key relevancies:

1. Prevalence – they are widespread and so represent the most common threats.
2. Growth – they may be few now, but our research shows they are rapidly expanding.
3. Innovation – they employ innovative techniques to counter security measures.

We collected live samples of in-the-wild financial malware, ransomware, PUA, and rootkits, and started the malware on an already protected system. Exe, zip, rar and scr file-types were used for the test. If the malware was not detected at the time of the test, we created a snapshot of this infected system, and checked it for 24 hours. We used 136 samples in total.

Result of the test



Although ESET Endpoint Security was better in initial sample detection, Webroot's performance was better on a 24-hour scale.

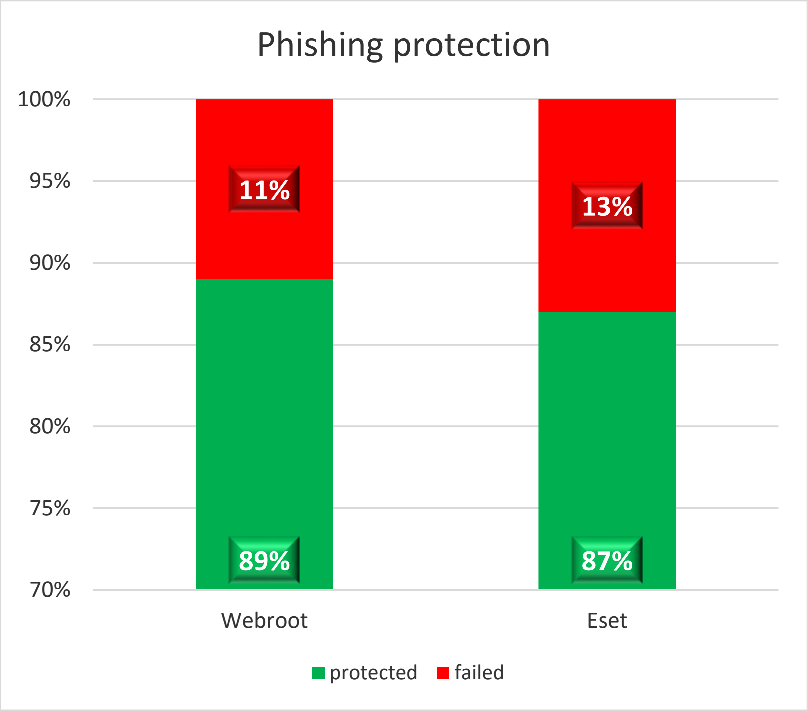
2.3 Phishing protection

We gathered 100 different, active phishing sites (financial and e-mail related), and navigated the browser to those sites. We waited for the webpages to load and if the fake login screen was not visible, we clicked on the page to show the fake login page, and finally we waited 10 seconds before entering credentials. If the credentials were sent to the phishing sites, the protection suite failed the test.

Test results

Webroot SecureAnywhere failed to block 11 of the phishing sites from stealing the username and password.

ESET Endpoint Security failed to block 13 of the phishing sites from stealing the username and password.



Webroot SecureAnywhere performed better in this test.

2.4 Self protection

We applied the "Advanced Process Termination v4.2 – DiamondCS" tool, the Sysinternals Process Explorer tool and the OSR Driver Loader to test self-protection of the protection system. 12 user-mode kill methods, 2 kernel-mode kill methods, 2 crash methods and 2 suspend methods and one driver unloading method were used. The methods included: TerminateProcess, WM_Close, WM_Quit, SC_Close, TerminateThread, CreateRemoteThread -> Exitprocess, Endtask, DebugActiveProcess, EIP modification -> Exitprocess, and DLL injection + Exitprocess, used accomplice process as terminator, ZwTerminateThread, ZwTerminateProcess, VirtualProtectEx crash, WriteProcessMemory crash, suspended all threads and NTSuspendProcess. In our tests, we first tried to suspend all processes, then kill all processes by all the listed methods and unload all kernel drivers. After this iteration we downloaded a previously detected malware test file and opened phishing site then checked whether the test file was still detected and the site blocked or not.

Test results

Webroot SecureAnywhere failed to protect the process that is running under the actual user (it was terminated), but the process restarted immediately. Moreover the wrUrlFlt driver can be unloaded, but it did not affect any of the protection capabilities so we marked all the tests as passed.

Eset failed to protect the egui.exe process that provides the GUI for the user, but it is not play any role in the protection of the system. No driver or services could be stopped or unloaded.

Both products performed the same during this test.

2.5 Simulator and botnet tests

We tested the endpoint protection systems against the following simulators and real botnets.

KINS/ZeusVM Real Botnet Test

MRG Effitas is proud to present the world's first real, public botnet test. In this test, we acquired leaked builders from real financial malware (ZeusVM/KINS), created the droppers and configured the C&C servers in a safe SoftLayer environment. Because this test uses real financial malware, where data exfiltration can be tested as it happens in the wild, the test efficiently maps the real-world threats users face today. These builders and droppers are available to everyone for free, thus the threats provide an entry level for criminals and are common threats in the wild.

Reflective injection + inline hooking HTTPSendRequestW simulator test

Financial malware developers always find new ways to bypass current protection technologies. One of the oldest techniques is to inject the attacker supplied DLL into Internet Explorer, then hook (redirect) the API calls, where the password can be found in a buffer passed to the function as a parameter. In this test, we used reflective DLL injection technique for the DLL injection step, and hooked either the HTTPSendrequestW function, via inline hooking.

Internet Explorer BHO simulator test

We created proprietary browser helper add-on (BHO). The BHO is installed to the browser before any security solution is installed. The BHO is able to steal POST data contents (which contain usernames, passwords), and send this to a server operated by MRG Effitas.

Injection via context switch simulator test

The Context Switch method uses standard Windows functions to allocate memory in the target process and find a running remote thread to hijack in the target process. It saves the current EIP and sets it to the address of the LoadLibrary function and writes the function and parameters (injected DLL name) in the remote process; the hijacked thread executes the LoadLibrary call, and finally the (malicious) functions in the DLL are executed because DLL_PROCESS_ATTACH is triggered.

Keylogger GetKeyState simulator test

We tested a common keylogger technique, GetKeyState: "This API returns the current key state for a given key. This method is less reliable than a global hook, but is stealthier and does not require administrator privileges."

A note on simulators

After a successful attack, the attacker can either extract passwords, session cookies and/or credit card/CVV numbers from the web sessions, or inject html forms into the web sessions (e.g. credit card number and CVC/CVV code), because SSL encryption takes place after the API calls. The purpose of testing with simulators is that the simulator is unknown to the security solution and thus it will not detect the simulator using traditional AV methods, which are known to be bypassed easily. This test measures the protection capabilities against zero day threats.

Protected browsers

Last but not least, we checked which browsers are protected by advanced behavior based browser protections. We marked it as a pass when a hardened browser was part of the product, and we marked it as a warning when some of the behavior based protections of the endpoint protection could block a financial malware attack.

Test result

Type of simulator	Webroot	ESET
KINS/ZeusVM real botnet	✓	✓
reflective injection + inline hooking HTTPSendRequestW	✓	✗
Internet Explorer BHO	✓	✗
injection via context switch	✓	✗
keylogger GetKeyState test	✓	✓
protect IE	✓	✗
protect Firefox	✓	✗
protect Chrome	✓	✗

✓	The application blocked the simulator
⚠	The application protects the browser, but it is not a full hardened safe browser
✗	The application failed to block the simulator

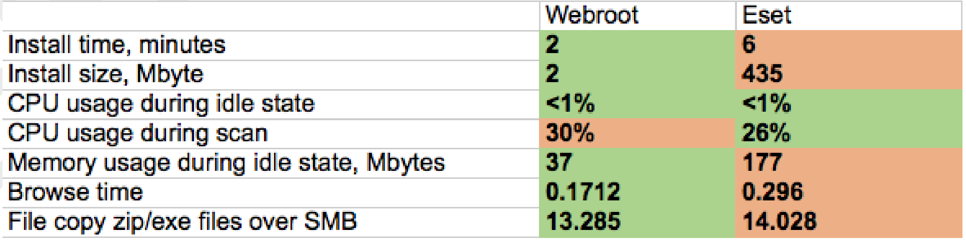
2.6 Performance

We measured the following parameters:

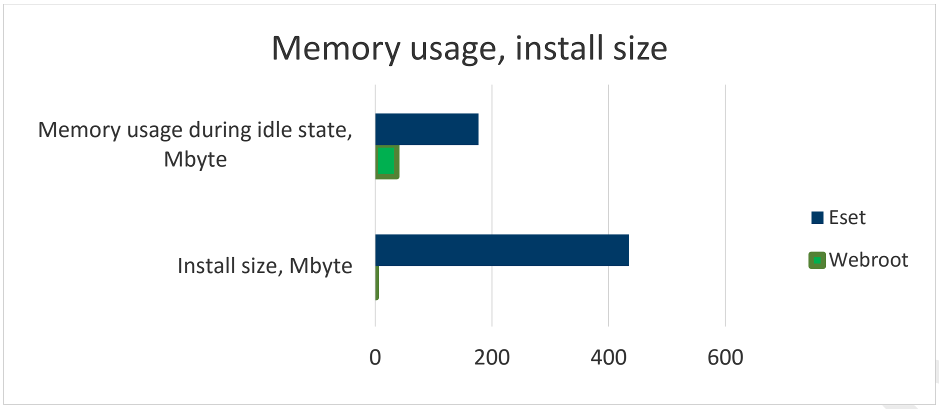
- Install time (in minutes). The install time includes finding the install webpage, registering, accessing the web console, downloading and installing the agent. The test is finished when the protection on the OS is updated, up and running.
- Install size on the disk (Program files (x86), Program files, Program data), in Mbyte
- CPU utilization during idle state
- CPU utilization during quick scan
- Memory usage during idle state (private bytes)
- Open Internet Explorer with a webpage on the local network, 100 times, measure average
- Copy files from SMB to localhost (ZIP file containing executables and executables)

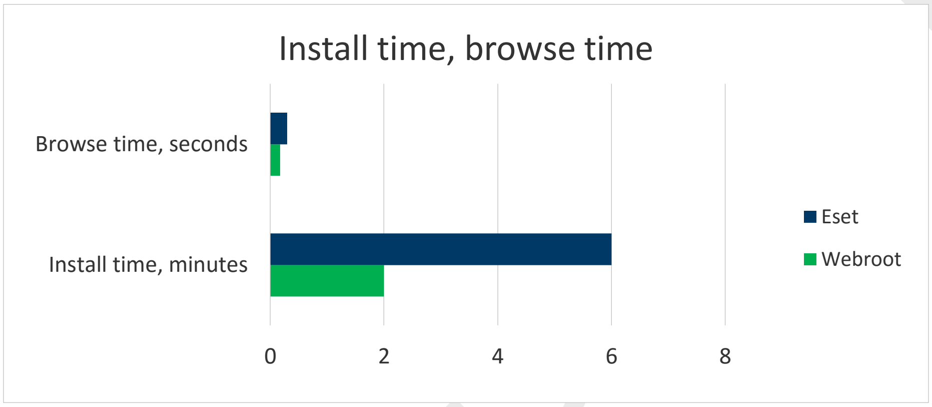
Test results

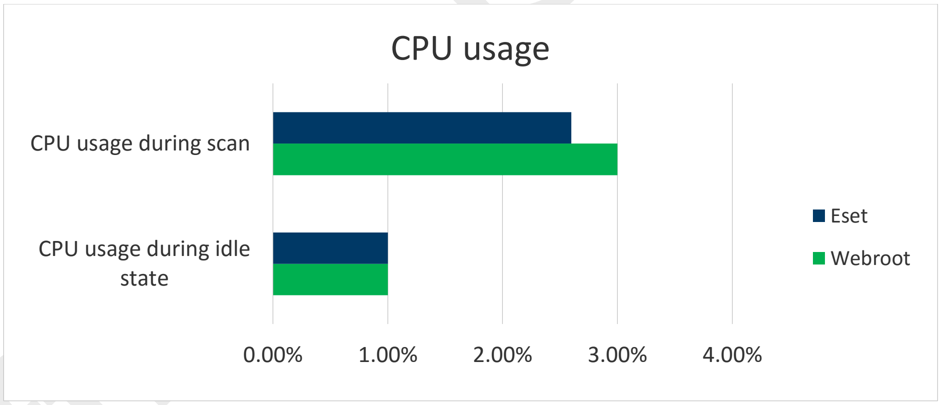
In the following table, the colour green means that the product performed better and the colour red that it performed worse than the competitor.

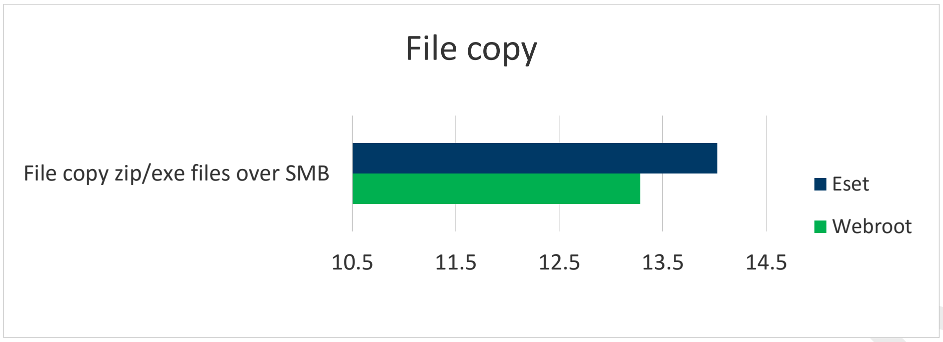


	Webroot	Eset
Install time, minutes	2	6
Install size, Mbyte	2	435
CPU usage during idle state	<1%	<1%
CPU usage during scan	30%	26%
Memory usage during idle state, Mbytes	37	177
Browse time	0.1712	0.296
File copy zip/exe files over SMB	13.285	14.028


Memory usage, install size
Memory usage during idle state, Mbyte
Install size, Mbyte
0
200
400
600
Eset
Webroot


Install time, browse time
Browse time, seconds
Install time, minutes
0
2
4
6
8
Eset
Webroot


CPU usage
CPU usage during scan
CPU usage during idle state
0.00%
1.00%
2.00%
3.00%
4.00%
Eset
Webroot



Webroot SecureAnywhere performed better in this test. The install size of Webroot is so low that it is not really comparable.

2.7 Remediation test with in-the-wild malware

In this test case we carefully selected 10 in-the-wild malware (1 rootkit, 3 backdoors, 6 financial malware) and infected a clean system. After we confirmed that the malware was working, we started to analyze the results. Our samples included a rootkit, backdoors and financial malware as well.

With the knowledge that we gained from monitoring the malware, we started the remediation test with WSA and Eset. We turned off most of the features that can block malware execution and interfere with the harm it was doing. The reason why we did this was to emulate an environment where the malware is not yet known, so it can do anything on the system without the AV blocking it.

After infection and one hour of running time, we rebooted the machine and turned all the features on. In case the product found the infection in less than one hour or managed to remove all infected files and the infection itself, we marked it as a success. If the product did not find or was not able to remove the infection, we marked that as a failure.

We tested 10 in-the-wild malware, which were selected manually. The samples included 1 rootkit, 3 backdoor, 6 financial malware.

2.7.1 Win32/Poweliks.B

Behavior: This rootkit modified the permission of multiple registry keys and directories, made itself persistent and executed several infected processes (dllhost.exe, ctfmon.exe, systray.exe)

Eset: Found the threat right after enabling all features, removed the binary and rebooted the system, but the infection was not fully removed

WSA: It was unable to find the infection and restore the system to its original state.

Conclusion: Both products failed in this test case.

2.7.2 Trojan.Zbot.Agent.U

Behavior: This dropper created a file in a random directory and executed it. The executed process was a financial malware that checks the system and crawls all e-mails and certificates.

Eset: Found the threat right after enabling all features and removed the threat and all infections (including binaries, registry)

WSA: After a manual scan, it found the threat and removed it properly.

Conclusion: Both products remediated the infection.

2.7.3 Trojan.Cridex

Behavior: It connects to a C&C server and waits for commands. It tries to access the server through several proxies that are configured on the system.

Eset: After turning on all features, it removed all threats.

WSA: It found the malware and removed all threats.

Conclusion: Both products remediated the infection.

2.7.4 Backdoor.MSIL.NanoBot.hja

Behavior: It creates several Monitor directories in Program Files and drops a binary there. In the meantime, it connects to the C&C server.

Eset: After turning on all features, it removed all threats.

WSA: It found the threat and disinfected the system.

Conclusion: Both products remediated the infection.

2.7.5 Trojan/Win32.Bublik

Behavior: This malware is a file infector; it copies itself into a user profile directory and modifies the registry to make itself persistent. It collects information about the operating system (MachineGuid, DigitalProductId, SystemBiosDate), then starts svchost.exe and cmd.exe and injects code into these processes.

Eset: After protection was turned on, it found the threat right away. Disinfected the system and asked for reboot.

WSA: It found the threat and disinfected the system after all protection was turned on.

Conclusion: Both products remediated the infection.

2.7.6 Trojan.Kryptik!AlBozWD+q+I

Behavior: Creates a random directory and copies itself into it. Tries to connect to the proxy configured on the system (checking Internet connectivity) and creates some mutexes that are usually used by Zeus Trojan. Persistence is achieved via the Windows startup registry key.

Eset: Found the threat right away when all protection was turned on.

WSA: The malware crashed a few seconds after it started. It most probably does not handle an exception related to the WSA self-protection mechanism.

Conclusion: Both products remediated the infection.

2.7.7 Trojan.Nitol.A

Behavior: Creates a file in the %WINDOWS%\syswow64 directory, renames itself as software.log and puts the binary into the temp directory. After infection copies the malware into several directories under the Program Files.

Eset: Removed the threat after turning on the protection mechanism asked for reboot twice.

WSA: It found the threat and disinfected the system after all protection was turned on.

Conclusion: Both products remediated the infection.

2.7.8 W32/Dyre.A!tr

Behavior: Moved itself to the appdata\local\ directory under the name of googleupdaterr.exe and connected to the Internet.

Eset: Found the threat right away when all protection was turned on and asked for reboot. All threats were removed.

WSA: It found the threat and disinfected the system after all protection was turned on.

Conclusion: Both products remediated the infection.

2.7.9 Trojan.Win32.Garrun.anf

Behavior: Started an explorer.exe process and injected into that process. Then it connected to the C&C server and put the malware into the recyclebin directory.

Eset: Found the threat right away when all protection was turned on.

WSA: It found the threat and disinfected the system after all protection was turned on.

Conclusion: Both products remediated the infection.

2.7.10 Zbot.FC

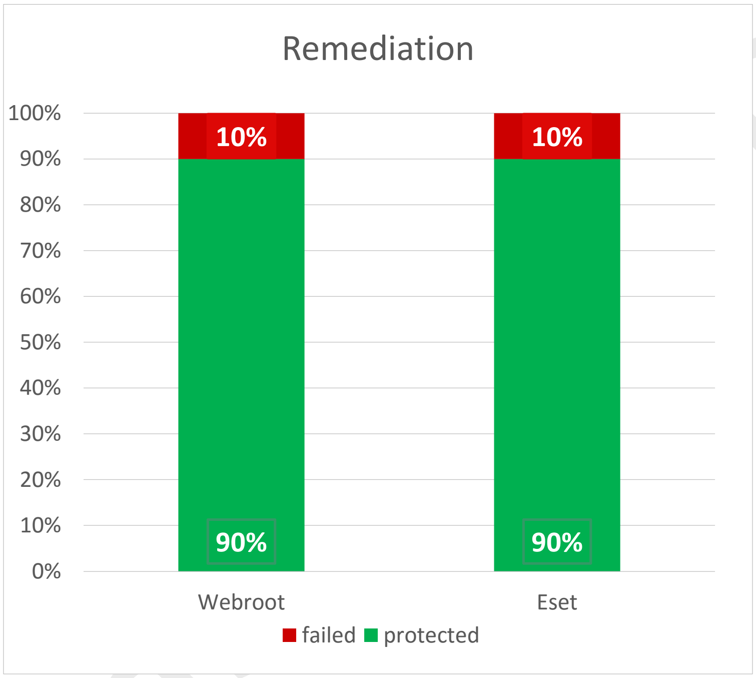
Behavior: Installs tor and connects to the tor network and waits for instructions.

Eset: Found the threat right away when all protection was turned on.

WSA: It found the threat and disinfected the system after all protection was turned on.

Conclusion: Both products remediated the infection.

Both products remediated 9 infections out of 10. The only failed test case was the Poweliks rootkit where none of the products managed to disinfect the system and restore to the original state.



The products protected against the malware the same level.

2.8 Product feature comparison

We compiled important feature lists common in endpoint protection systems, and tested these features on both protection systems.

	Webroot	ESET
Active directory integration	**yes**	**yes**
Centralized cloud management	**yes**	**no**
Cloud reputation	**yes**	**yes**
Desktop policy	**yes**	**yes**
Exploit protection	**no**	**yes**
Host intrusion prevention	**yes**	**yes**
Mac (OSX) management	**yes**	**yes**
Mobile management	**yes, optional**	**yes**
On access scan	**yes**	**yes**
Protect browser from malware	**yes**	**no, only in home version**
Removable media control	**no**	**yes**
Journaling, monitoring and rollback	**yes**	**no**
Scheduled scans	**yes**	**yes**
Server policy	**yes**	**yes**
Software Firewall	**yes**	**yes**
Web filter	**yes**	**yes**
Windows 10 support	**yes**	**yes**

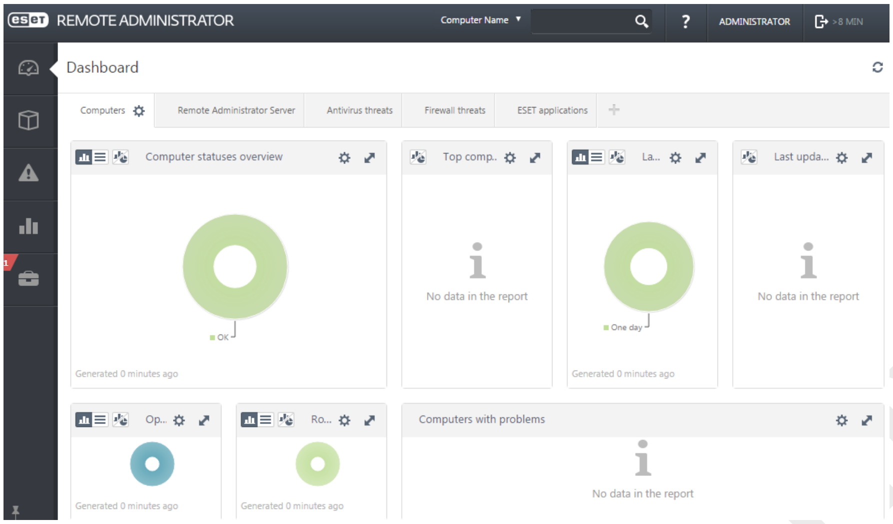
This test is subjective, as every company has different features as priorities.

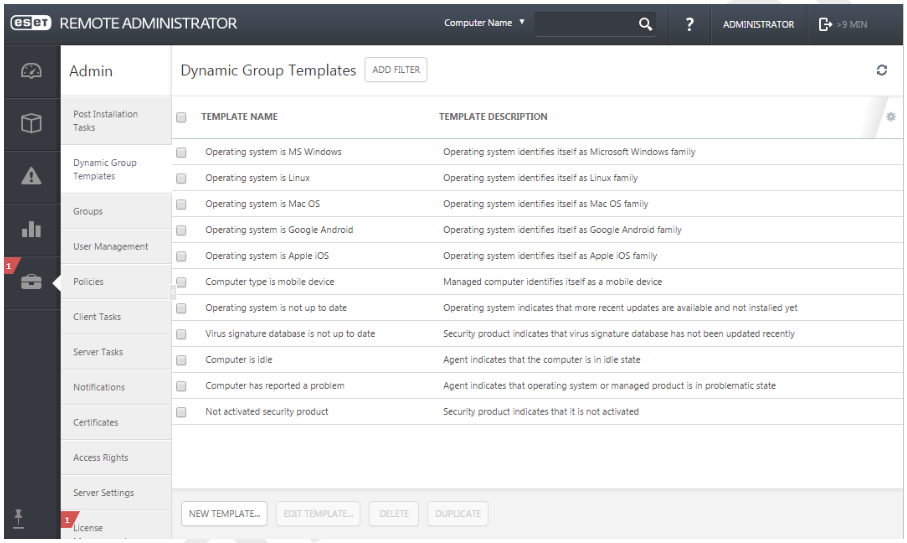
Although there were only slight differences, Webroot performed better during this test.

2.9 Management interface, reporting capabilities

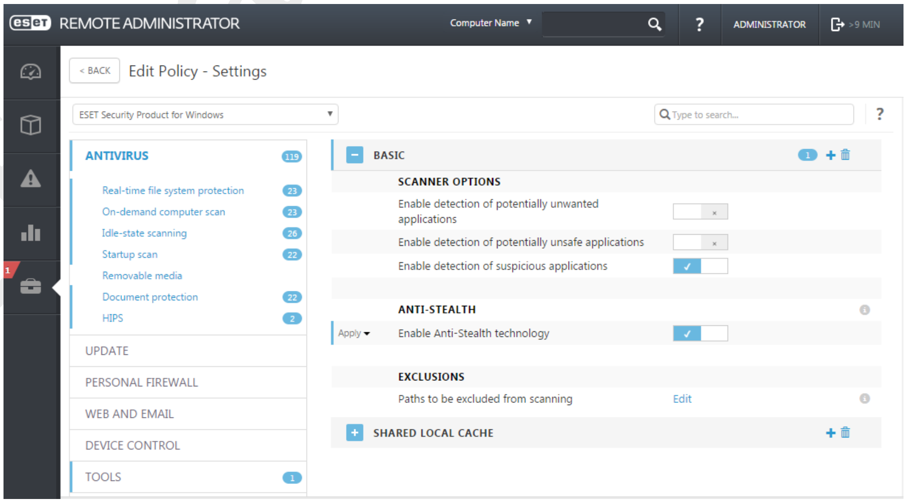
It is not an easy task to compare the overall management interface and reporting capabilities of two different (security…) products. We made every attempt to remain as impartial as possible. Following is a general assessment from our perspective.

Following screenshots are samples from the management interface of ESET Endpoint Security.





TEMPLATE NAME	TEMPLATE DESCRIPTION
Operating system is MS Windows	Operating system identifies itself as Microsoft Windows family
Operating system is Linux	Operating system identifies itself as Linux family
Operating system is Mac OS	Operating system identifies itself as Mac OS family
Operating system is Google Android	Operating system identifies itself as Google Android family
Operating system is Apple iOS	Operating system identifies itself as Apple iOS family
Computer type is mobile device	Managed computer identifies itself as a mobile device
Operating system is not up to date	Operating system indicates that more recent updates are available and not installed yet
Virus signature database is not up to date	Security product indicates that virus signature database has not been updated recently
Computer is idle	Agent indicates that the computer is in idle state
Computer has reported a problem	Agent indicates that operating system or managed product is in problematic state
Not activated security product	Security product indicates that it is not activated



SCANNER OPTIONS

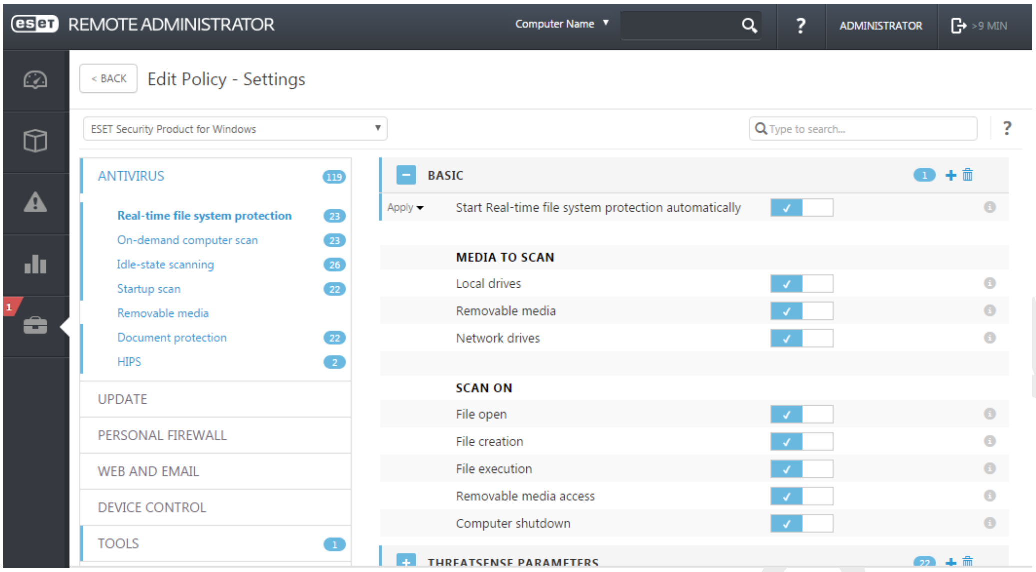
- Enable detection of potentially unwanted applications ✗
- Enable detection of potentially unsafe applications ✗
- Enable detection of suspicious applications ✓

ANTI-STEALTH

- Apply ▾ Enable Anti-Stealth technology ✓

EXCLUSIONS

- Paths to be excluded from scanning Edit



Edit Policy - Settings

BASIC

Start Real-time file system protection automatically ✓

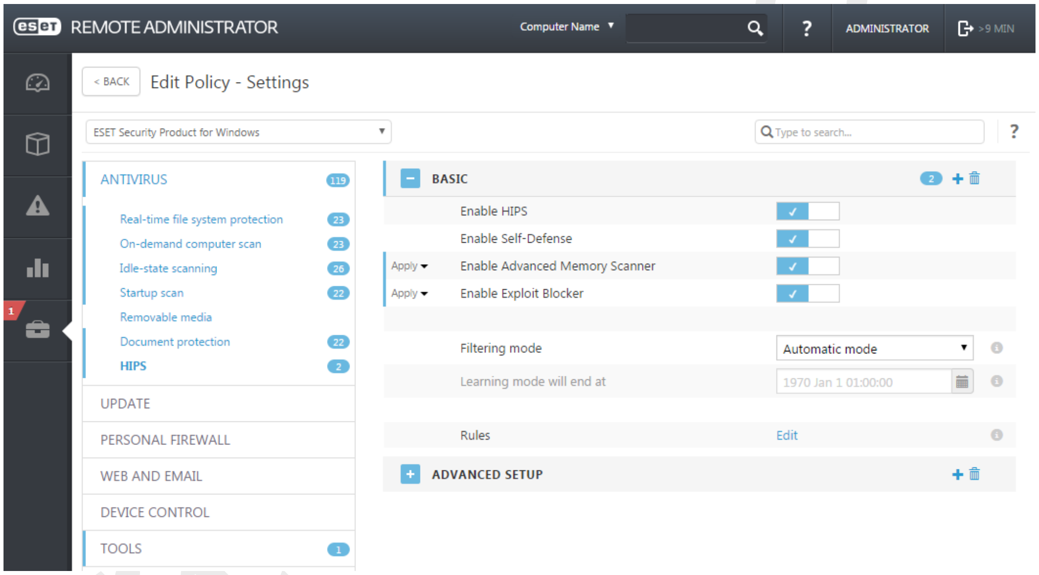
MEDIA TO SCAN

Local drives ✓
Removable media ✓
Network drives ✓

SCAN ON

File open ✓
File creation ✓
File execution ✓
Removable media access ✓
Computer shutdown ✓

THREATSENSE PARAMETERS



Edit Policy - Settings

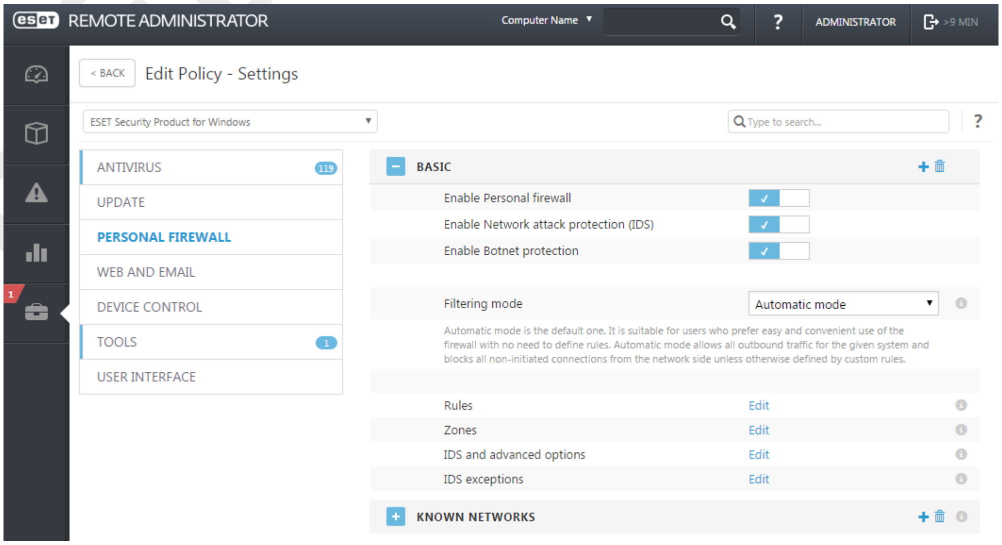
BASIC

Enable HIPS ✓
Enable Self-Defense ✓
Enable Advanced Memory Scanner ✓
Enable Exploit Blocker ✓

Filtering mode: Automatic mode
Learning mode will end at: 1970 Jan 1 01:00:00

Rules: Edit

ADVANCED SETUP



Edit Policy - Settings

BASIC

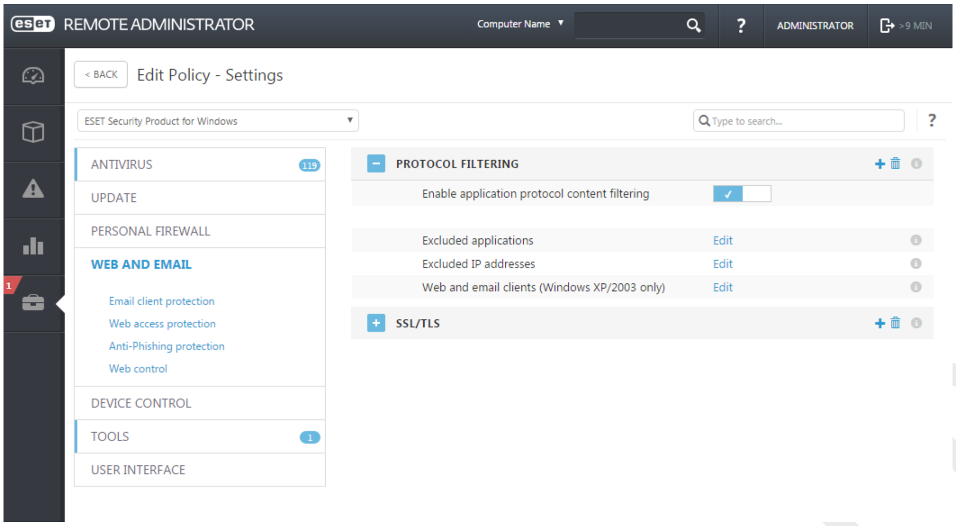
Enable Personal firewall ✓
Enable Network attack protection (IDS) ✓
Enable Botnet protection ✓

Filtering mode: Automatic mode

Automatic mode is the default one. It is suitable for users who prefer easy and convenient use of the firewall with no need to define rules. Automatic mode allows all outbound traffic for the given system and blocks all non-initiated connections from the network side unless otherwise defined by custom rules.

Rules: Edit
Zones: Edit
IDS and advanced options: Edit
IDS exceptions: Edit

KNOWN NETWORKS



Edit Policy - Settings

PROTOCOL FILTERING

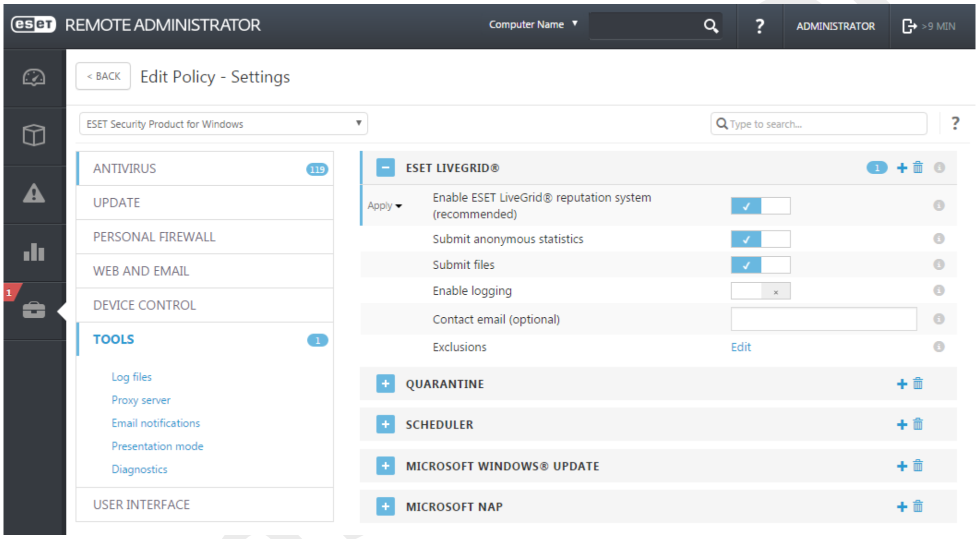
Enable application protocol content filtering ✓

Excluded applications Edit

Excluded IP addresses Edit

Web and email clients (Windows XP/2003 only) Edit

SSL/TLS



Edit Policy - Settings

ESET LIVEGRID®

Enable ESET LiveGrid® reputation system (recommended) ✓

Submit anonymous statistics ✓

Submit files ✓

Enable logging ×

Contact email (optional)

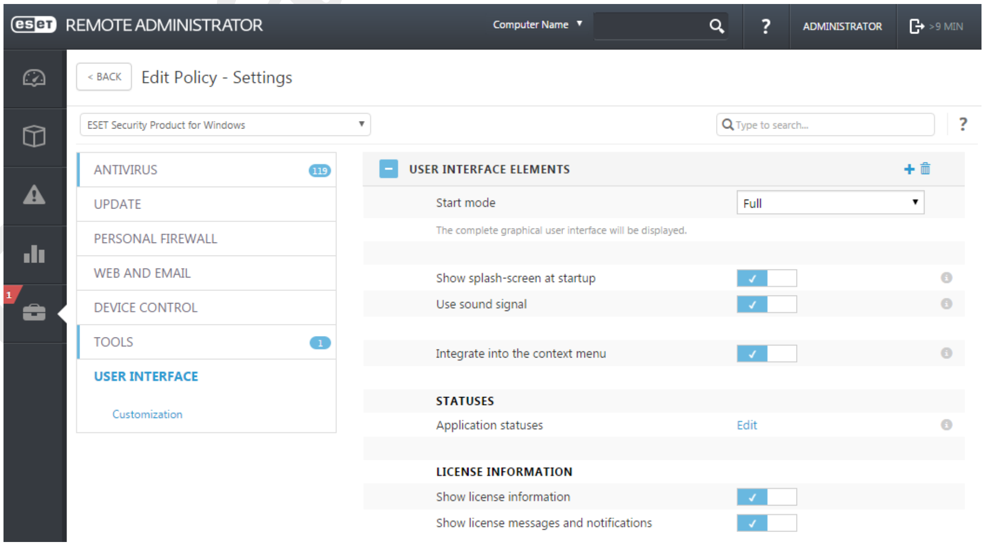
Exclusions Edit

QUARANTINE

SCHEDULER

MICROSOFT WINDOWS® UPDATE

MICROSOFT NAP



Edit Policy - Settings

USER INTERFACE ELEMENTS

Start mode Full ▾

The complete graphical user interface will be displayed.

Show splash-screen at startup ✓

Use sound signal ✓

Integrate into the context menu ✓

STATUSES

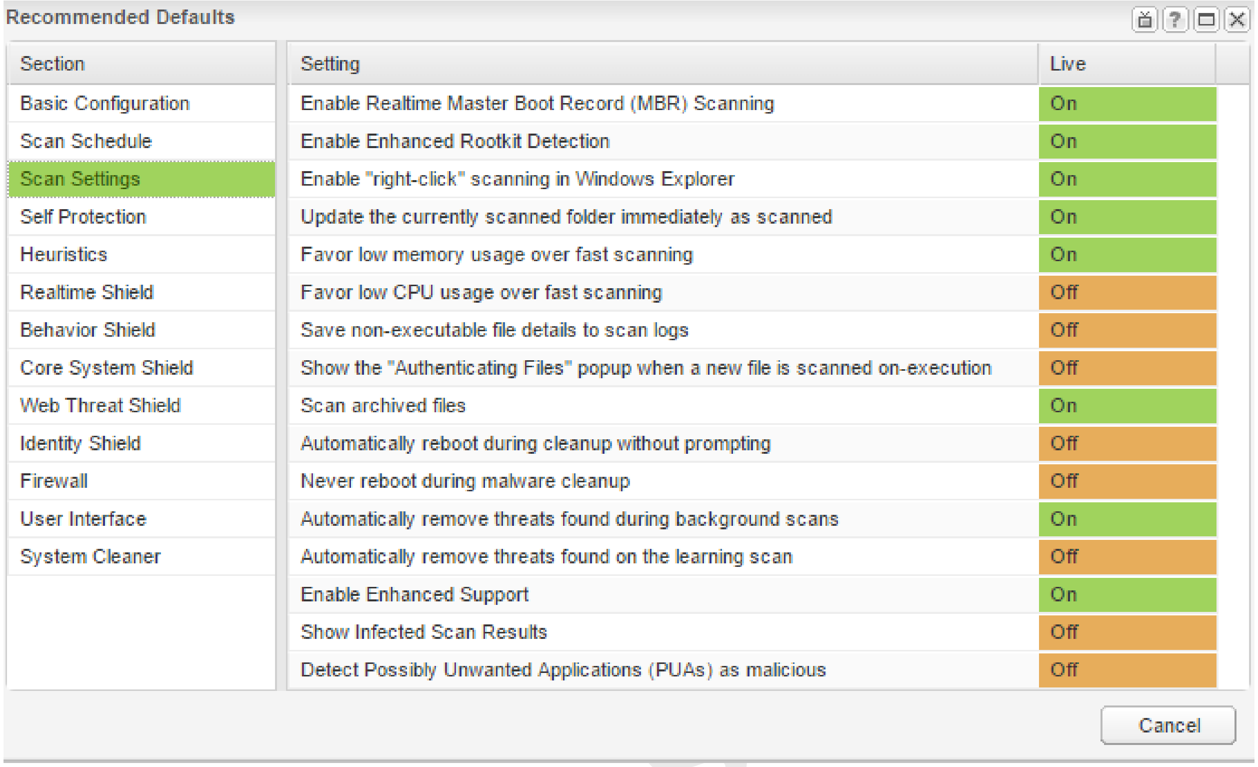
Application statuses Edit

LICENSE INFORMATION

Show license information ✓

Show license messages and notifications ✓

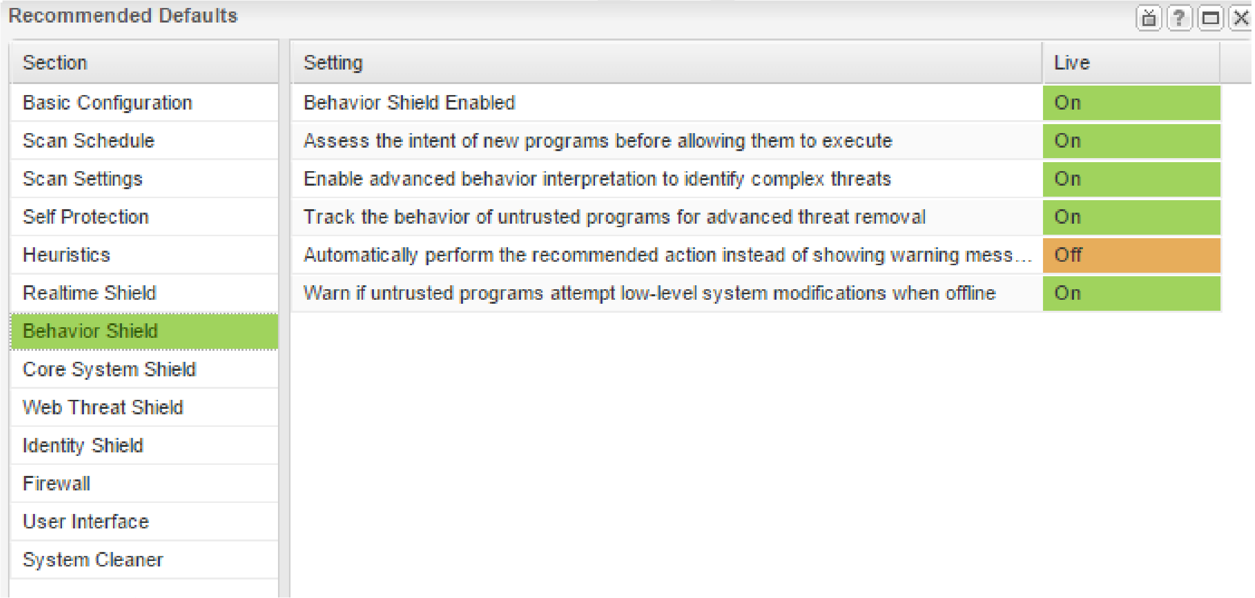
The following screenshots are samples from the management interface of Webroot.



Recommended Defaults

Section
Basic Configuration
Scan Schedule
Scan Settings
Self Protection
Heuristics
Realtime Shield
Behavior Shield
Core System Shield
Web Threat Shield
Identity Shield
Firewall
User Interface
System Cleaner

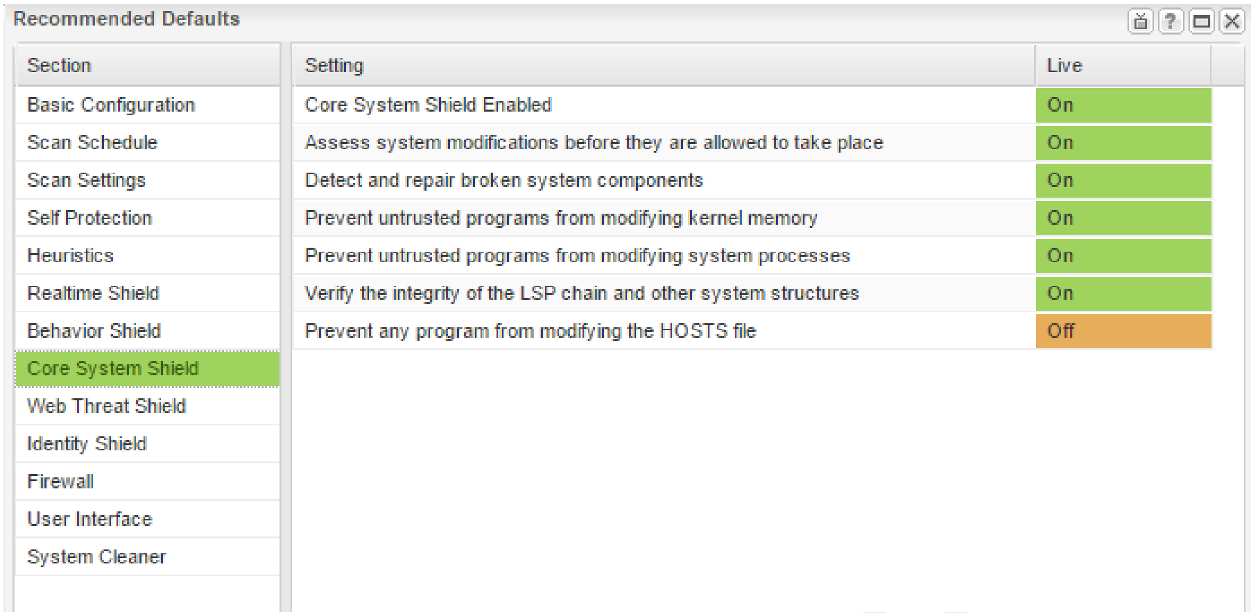
Setting	Live
Enable Realtime Master Boot Record (MBR) Scanning	On
Enable Enhanced Rootkit Detection	On
Enable "right-click" scanning in Windows Explorer	On
Update the currently scanned folder immediately as scanned	On
Favor low memory usage over fast scanning	On
Favor low CPU usage over fast scanning	Off
Save non-executable file details to scan logs	Off
Show the "Authenticating Files" popup when a new file is scanned on-execution	Off
Scan archived files	On
Automatically reboot during cleanup without prompting	Off
Never reboot during malware cleanup	Off
Automatically remove threats found during background scans	On
Automatically remove threats found on the learning scan	Off
Enable Enhanced Support	On
Show Infected Scan Results	Off
Detect Possibly Unwanted Applications (PUAs) as malicious	Off



Recommended Defaults

Section
Basic Configuration
Scan Schedule
Scan Settings
Self Protection
Heuristics
Realtime Shield
Behavior Shield
Core System Shield
Web Threat Shield
Identity Shield
Firewall
User Interface
System Cleaner

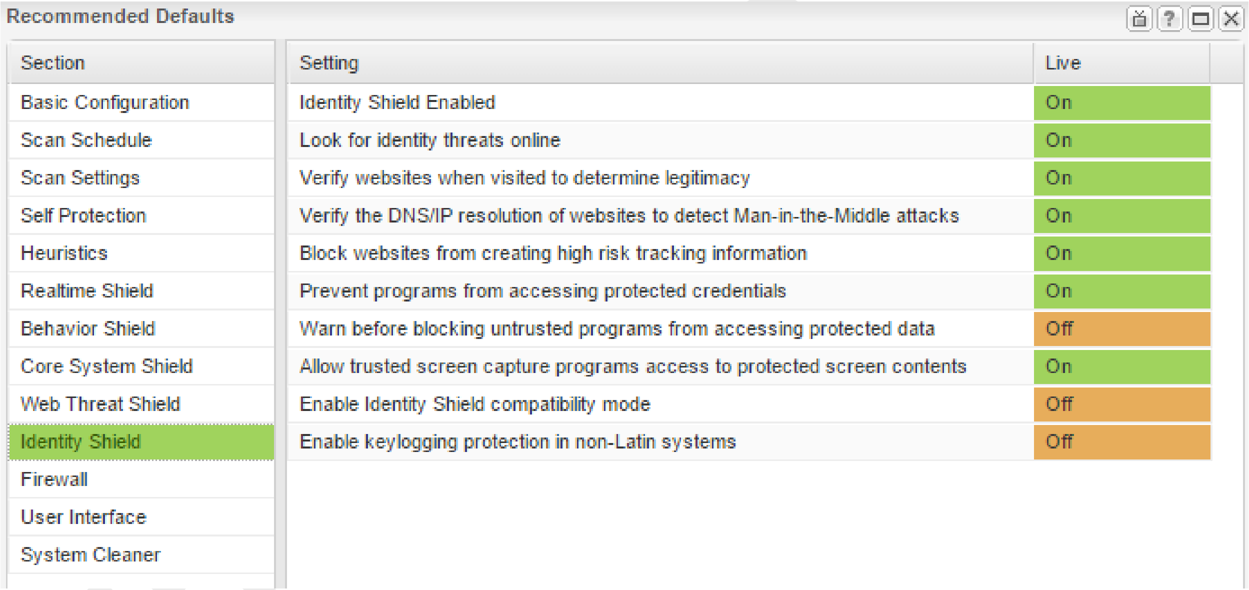
Setting	Live
Behavior Shield Enabled	On
Assess the intent of new programs before allowing them to execute	On
Enable advanced behavior interpretation to identify complex threats	On
Track the behavior of untrusted programs for advanced threat removal	On
Automatically perform the recommended action instead of showing warning mess…	Off
Warn if untrusted programs attempt low-level system modifications when offline	On



Recommended Defaults

Section
Basic Configuration
Scan Schedule
Scan Settings
Self Protection
Heuristics
Realtime Shield
Behavior Shield
Core System Shield
Web Threat Shield
Identity Shield
Firewall
User Interface
System Cleaner

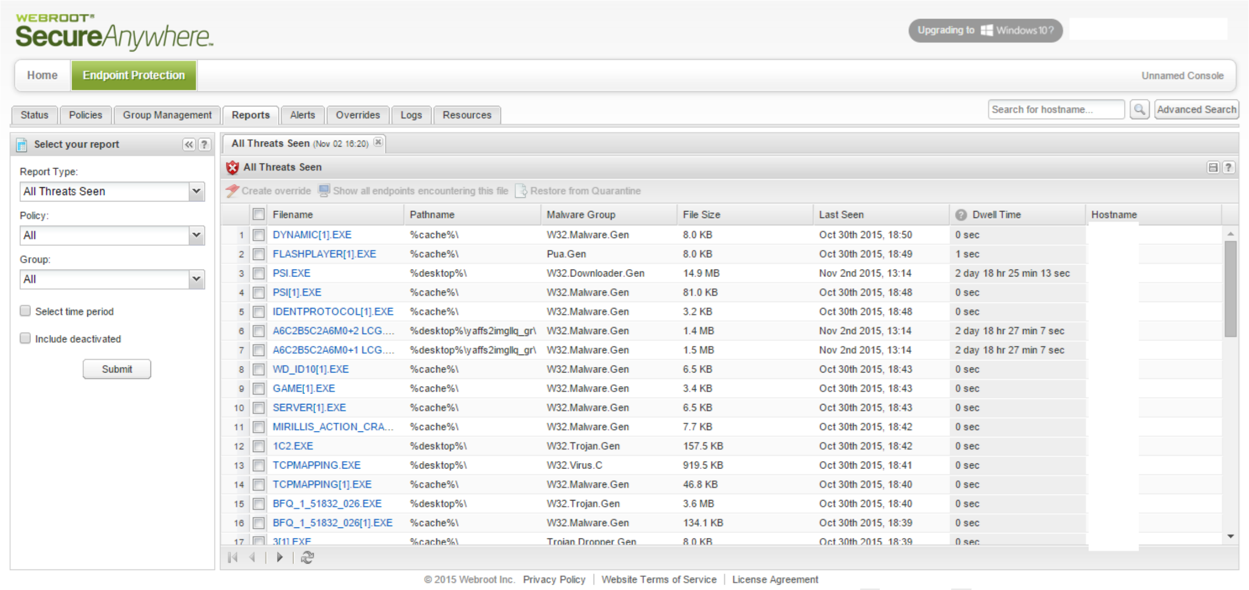
Setting	Live
Core System Shield Enabled	On
Assess system modifications before they are allowed to take place	On
Detect and repair broken system components	On
Prevent untrusted programs from modifying kernel memory	On
Prevent untrusted programs from modifying system processes	On
Verify the integrity of the LSP chain and other system structures	On
Prevent any program from modifying the HOSTS file	Off



Recommended Defaults

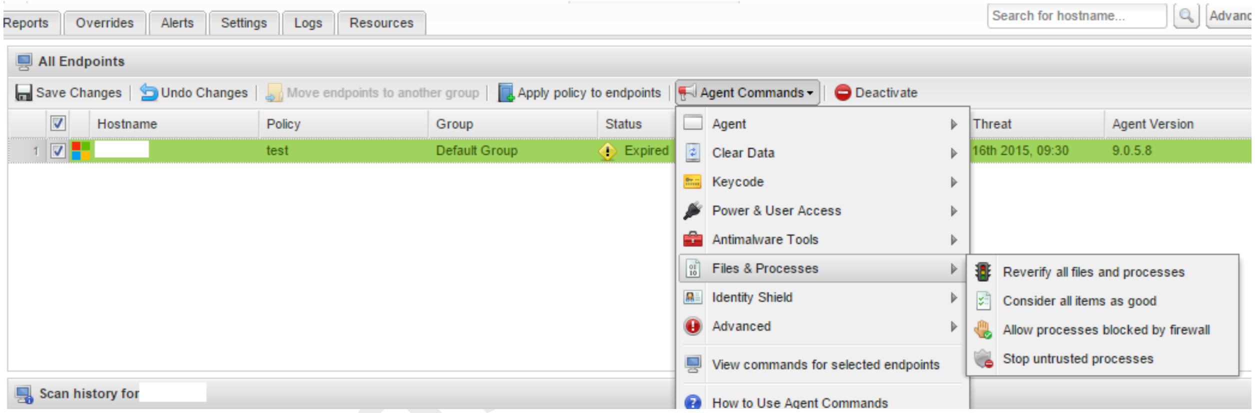
Section
Basic Configuration
Scan Schedule
Scan Settings
Self Protection
Heuristics
Realtime Shield
Behavior Shield
Core System Shield
Web Threat Shield
Identity Shield
Firewall
User Interface
System Cleaner

Setting	Live
Identity Shield Enabled	On
Look for identity threats online	On
Verify websites when visited to determine legitimacy	On
Verify the DNS/IP resolution of websites to detect Man-in-the-Middle attacks	On
Block websites from creating high risk tracking information	On
Prevent programs from accessing protected credentials	On
Warn before blocking untrusted programs from accessing protected data	Off
Allow trusted screen capture programs access to protected screen contents	On
Enable Identity Shield compatibility mode	Off
Enable keylogging protection in non-Latin systems	Off

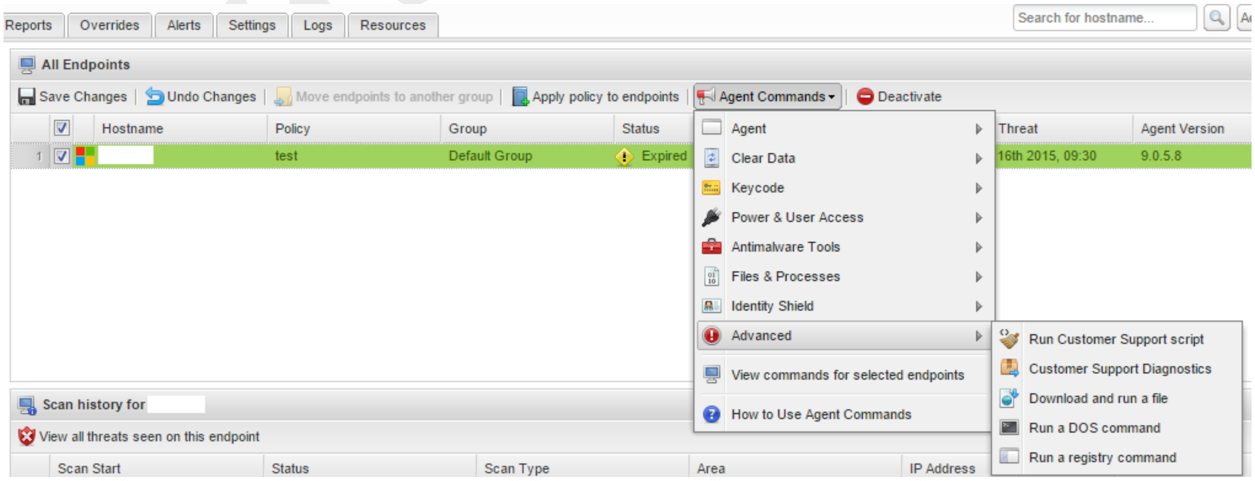


	Filename	Pathname	Malware Group	File Size	Last Seen	Dwell Time	Hostname
1	DYNAMIC[1].EXE	%cache%\	W32.Malware.Gen	8.0 KB	Oct 30th 2015, 18:50	0 sec	
2	FLASHPLAYER[1].EXE	%cache%\	Pua.Gen	8.0 KB	Oct 30th 2015, 18:49	1 sec	
3	PSI.EXE	%desktop%\	W32.Downloader.Gen	14.9 MB	Nov 2nd 2015, 13:14	2 day 18 hr 25 min 13 sec	
4	PSI[1].EXE	%cache%\	W32.Malware.Gen	81.0 KB	Oct 30th 2015, 18:48	0 sec	
5	IDENTPROTOCOL[1].EXE	%cache%\	W32.Malware.Gen	3.2 KB	Oct 30th 2015, 18:48	0 sec	
6	A6C2B5C2A6M0+2 LCG....	%desktop%\yaffs2imgllq_gr\	W32.Malware.Gen	1.4 MB	Nov 2nd 2015, 13:14	2 day 18 hr 27 min 7 sec	
7	A6C2B5C2A6M0+1 LCG....	%desktop%\yaffs2imgllq_gr\	W32.Malware.Gen	1.5 MB	Nov 2nd 2015, 13:14	2 day 18 hr 27 min 7 sec	
8	WD_ID10[1].EXE	%cache%\	W32.Malware.Gen	6.5 KB	Oct 30th 2015, 18:43	0 sec	
9	GAME[1].EXE	%cache%\	W32.Malware.Gen	3.4 KB	Oct 30th 2015, 18:43	0 sec	
10	SERVER[1].EXE	%cache%\	W32.Malware.Gen	6.5 KB	Oct 30th 2015, 18:43	0 sec	
11	MIRILLIS_ACTION_CRA...	%cache%\	W32.Malware.Gen	7.7 KB	Oct 30th 2015, 18:42	0 sec	
12	1C2.EXE	%desktop%\	W32.Trojan.Gen	157.5 KB	Oct 30th 2015, 18:42	0 sec	
13	TCPMAPPING.EXE	%desktop%\	W32.Virus.C	919.5 KB	Oct 30th 2015, 18:41	0 sec	
14	TCPMAPPING[1].EXE	%cache%\	W32.Malware.Gen	46.8 KB	Oct 30th 2015, 18:40	0 sec	
15	BFQ_1_51832_026.EXE	%desktop%\	W32.Trojan.Gen	3.6 MB	Oct 30th 2015, 18:40	0 sec	
16	BFQ_1_51832_026[1].EXE	%cache%\	W32.Malware.Gen	134.1 KB	Oct 30th 2015, 18:39	0 sec	
17	3[1].EXE	%cache%\	Trojan.Dropper.Gen	8.0 KB	Oct 30th 2015, 18:39	0 sec	

© 2015 Webroot Inc. Privacy Policy | Website Terms of Service | License Agreement



	☑	Hostname	Policy	Group	Status	Threat	Agent Version
1	☑		test	Default Group	Expired	16th 2015, 09:30	9.0.5.8



	☑	Hostname	Policy	Group	Status	Threat	Agent Version
1	☑		test	Default Group	Expired	16th 2015, 09:30	9.0.5.8

Both administrative interface is very modern. Both Endpoint Protection performed well in this test, based on our subjective opinion.

3 Conclusion

Based on the number of different tests, Webroot SecureAnywhere performed better in the time-to-detect test, simulators and botnets, phishing, performance test and at the functionalities compared to ESET Endpoint Security. In the remediation, self-protection and in management and reporting capabilities both products performed the same.

4 Appendix

4.1 Methodology Used in the “Simulator Test“

1. Windows 7 64 bit operating system is installed on a virtual machine, all updates are applied and third party applications installed and updated according to our “Average Endpoint Specification”.
2. An image of the operating system is created.
3. A clone of the imaged systems is made for each of the security applications to be used in the test.
4. An individual security application is installed using default settings on each of the systems created in 3 and then, where applicable, it is updated. If restart is recommended by the application (visible to the user), the system is restarted. If the installer has the option to participate in cloud protection, or PUA protection, all of these are enabled.
5. A clone of the system as it is at the end of 4 is created, and the system is started.
6. The simulator is started onto the clean systems with protection installed.
7. Each simulator test is conducted by:
    a. Starting a new instance of Internet Explorer (or the safe browser) and navigating to a financial website. Where the security application offers a secured or dedicated banking browser, this is used. If the security application is designed to protect Internet Explorer, only that component is going to be tested.
    b. Trying to inject the simulator into the browser process.
    c. Text is entered into the Account login page of the financial website using the keyboard, or using a virtual keyboard if the application under test provides such functionality, and then the “log in” button is pressed.
8. A test is deemed to have been passed (marked as a green checkbox) based on the following criteria:
    a. The security application detects the malware simulator when it is executed according to the following criteria:
        i. It identifies the simulator as being malicious and either automatically blocks it or postpones its execution, warns the user that the file is malicious and awaits user input.
        ii. It identifies the simulator as suspicious or unknown and gives the option to run in a sandbox or safe restricted mode, and, when run in this mode, it meets the criterion c below.
    b. The security application prevents the simulator from injecting itself into the browser process.
    c. The security application does not allow the hooking/redirection of the API calls, or even with successful hooking, the password cannot be captured from the browser.
9. A test is deemed to have been failed (marked as a yellow warning) based on the following criteria:
    a. The security application fails to detect the simulator and then:
        i. The security application fails to prevent the simulator from injecting itself into the browser process, and gives no alert or provides informational alerts only.
        ii. The security application allows the hooking/redirection of the API calls, and the password can be captured from the browser.
    b. The security application identifies the simulator as malware or unknown and gives the option to run in a sandbox or safe restricted mode, and, when run in this mode, it:
        i. Fails to prevent the simulator from injecting itself into the browser process, and gives no alert or provides informational alerts only.
        ii. The security application allows the hooking/redirection of the API calls, and the password can be captured from the browser.
10. Testing is conducted with all systems having internet access.
11. Each individual test for each security application is conducted from a unique IP address.

12. All security applications are fully-functional unregistered versions or versions registered anonymously, with no connection to MRG Effitas.

4.2 Methodology Used in the “In the Wild Test”

1. Windows 7 Ultimate Service Pack 1 64 bit operating system is installed on a virtual machine, all updates are applied and third party applications installed and updated according to our “Average Endpoint Specification”.
2. An image of the operating system is created.
3. A clone of the imaged systems is made for each of the security applications to be used in the test.
4. An individual security application is installed using default settings on each of the systems created in 5 and then, where applicable, it is updated and shut down. If the installer has the option to participate in cloud protection, or PUA protection, all of these are enabled.
5. Testing is conducted by:
   a. Downloading the sample using Internet Explorer to the desktop, the browser is kept running, conducting a context menu scan or, where unavailable, a system scan, and then executing the sample.
6. A test is deemed to have been passed based on the following criteria:
   a. The security application blocks the URL where the sample is located, thus preventing its download.
   b. The security application detects the sample whilst it is being downloaded to the desktop.
   c. The security application detects the sample during the context or system scan.
   d. The security application detects the sample when it is executed according to the following criteria:
      i. It identifies the sample as being malicious and either automatically blocks it or pauses its execution, advises the user not to execute it and awaits user input.
7. A test is deemed to have been failed based on the following criterion:
   a. The security application fails to detect the sample under condition 6a, 6b, 6c or 6d.
8. Testing is conducted with all systems having internet access.
9. Each individual test for each security application is performed from a unique IP address. All security applications are fully-functional unregistered versions or versions registered anonymously, with no connection to MRG Effitas.