



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

## Interested in learning more?

Check out the list of upcoming events offering  
"Hacker Tools, Techniques, and Incident Handling (Security 504)"  
at <http://www.giac.org/registration/gcih>

# **Application Whitelisting: Panacea or Propaganda?**

## **GIAC GCIH Gold Certification**

Author: Jim Beechey, beechey@northwood.edu  
Advisor: Rodney Caudle

Accepted: December XX, 2010

### **Abstract**

Our endpoints are under attack. Every day, organizations of all sizes struggle to protect their endpoints from a constant barrage of malware. Issues range from the annoying, rogue antivirus for instance, to levels of national security. The common thread is that these systems have become the ideal target for attackers whether they are the end goal or just the initial victim in a larger attack. Traditional defenses such as antivirus and firewalls are no longer effective. Organizations need to consider new approaches to battling malware. One such approach called application whitelisting has been around for a while, but only now gaining momentum. Is whitelisting just another passing fad or a solution to this ever growing problem? Join us as we attempt to answer this question.

## 1. Introduction

Every day, organizations of all sizes struggle to protect their endpoints from a constant barrage of malware. The number of threats continues to increase dramatically each year. Figure 1 shows the number of unique samples in av-test.org's malware collection from 2004 – 2009.

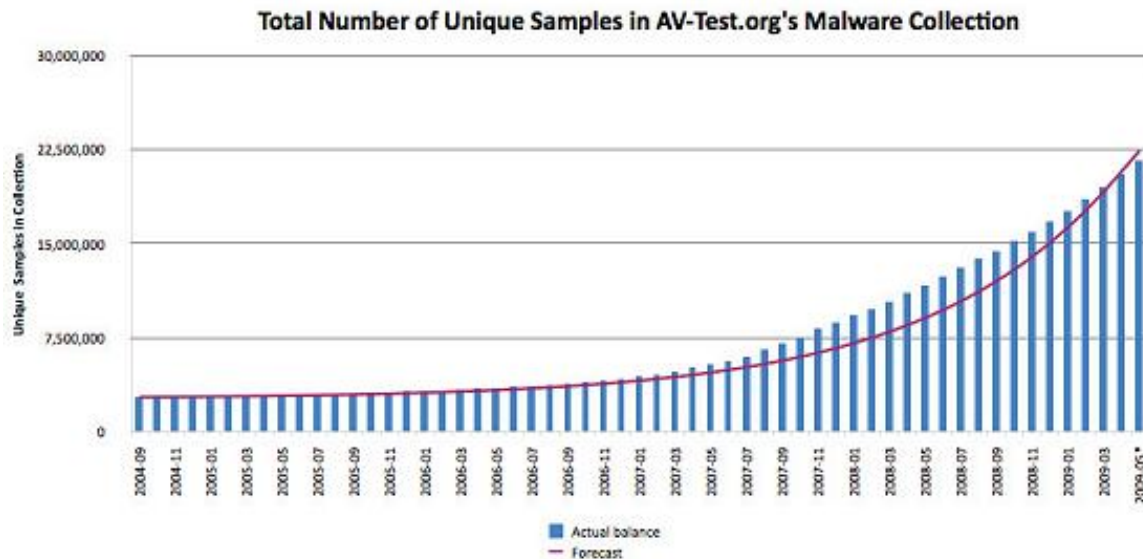


Figure 1

<http://www.sophos.com/blogs/gc/g/2009/07/24/avtestorgs-malware-count-exceeds-22-million/>

Unfortunately, traditional defenses such as firewalls and antivirus software are not effective against many attacks. Firewalls are ineffective as many attacks utilize already allowed ports and services. Clients are often the initial target of a larger attack and can be compromised simply by visiting a website hosting malware. While clients continue to be compromised when users visit inappropriate websites, clients can also be compromised by visiting some of the largest, most trusted sites on the Internet. Sites such as the US Army, Best Buy, CNN and MySpace have all had similar issues. ("Sophos list of," 2010). In addition, many clients continue to be compromised by clicking on infected attachments. Email based attacks have been around for years, but continue to be highly effective. Attackers will use spear phishing attacks to target specific individuals within an organization. Mass email based viruses have been around

for years, but are still effective as shown by the ‘here you have’ virus which was seen in September 2010. “Ram Herkanaidu, security researcher at Kaspersky Lab, said that the email closely resembles the ‘I love you’ virus which caused havoc about ten years ago... He said: “We’ve identified an email worm called VBMania. The interesting thing about it is that it uses very old tactics. In this case, very old means about ten years ago.” (Firth, 2010)

The bottom line is; signature based antivirus vendors simply cannot keep up with the sheer volume of malware. “Security software testing firm **NSS Labs** completed another controversial test of how the major anti-virus products fared in detecting malware pushed by malicious Web sites: Most of the products took an average of more than 45 hours — nearly two days — to detect the latest threats.” (Krebs, 2010) These issues point to the need for new approaches in defending endpoints against malware. One approach which has received a fair amount of attention recently is application whitelisting.

The general concept behind application whitelisting is quite simple. Instead of attempting to block malicious files and activity, application whitelisting will only permit known good files. Essentially, whitelisting flips the antivirus model from a ‘default allow’ to a ‘default deny’ for all executable files. This is accomplished by creating a list of known or approved file hashes and only allowing files with approved hashed to execute.

While the general concept of whitelisting is simple, the application of these principles can be anything but. Consider the operational and political challenges of a default deny model on desktop computers. Whitelisting provides the ultimate level of control over end user systems. This can be a great thing for security, but a big challenge in today’s world of open, creative workplaces. “Cultural issues and policies are still the single biggest obstacles. Users are accustomed to having control over their own PCs, and taking away some of their ability to make changes is more a cultural than a technical change. Organizations must build support for a continuum of control solutions and should never refer to such projects using the term “lockdown.” (MacDonald, & Silver, 2008)

Jim Beechey, beechey@northwood.edu

Even if one can navigate the politics of system lockdown, you still have to manage the whitelist. Manageability is the key technical issue when considering adopting a whitelisting approach. This is also the key differentiator between most vendor solutions. “Although a large number of vendors offer application control solutions, enforcing whitelists and blacklists of applications is a commodity that provides little more than what can be done with Windows GPO-based Software Restriction Policies (SRPs). We continue to advise organizations adopting application control solutions that the key to successful tool selection and implementation is the capability to automate the exception management process and to automate list management.” (MacDonald, & Silver, 2008)

Preventing attacks is the ultimate goal of locking down systems using application whitelisting. However, organizations which cannot lockdown systems due to political or operational issues are still benefitting from the technology. Many organizations deploy whitelisting technology in a monitor only mode. This provides visibility into the executables which are running on end user systems and can be used to detect, confirm and respond to attacks.

While the overall benefits of application whitelisting are clear, no technology can provide complete security. Identifying the strengths and weaknesses of the technology is essential to ensuring appropriate prevention and detection controls are in place. The following sections will discuss various commercial application whitelisting solutions, strategies for managing the whitelist, strengths and weaknesses of the technology including how application whitelisting stands up to today’s malware, complimentary security technologies and possible methods for attacking whitelisting solutions. The goal is provide a comprehensive evaluation of the technology so organizations can best understand how and why to consider deploying the technology.

## 2. Commercial Solutions

The application whitelisting space is made up of a combination of pure-play vendors and larger security companies looking to provide whitelisting as a component of existing solutions or frameworks. Products come from both venture capital funded

startups and well established security companies. As mentioned previously, whitelisting technology itself is a commodity. Vendors differentiate themselves primarily in their management approaches, but also their ability to compensate for weaknesses in the whitelisting model. The following provides a brief look at the major players in the application whitelisting space focusing on the key differentiators in the product.

## 2.1. Bit9 Parity

Bit9 is a pure-play application whitelisting vendor which was born out of a \$2 million dollar research grant from the National Institute of Standards and Technology in 2003. Bit9 is best known for their Global Software Registry which is a collection of file hashes and other pieces of file metadata such as product, publisher and trust level. “Now at over 6 billion records, the Bit9 Global Software Registry is growing at a rate of up to 20 million files each day.” (“Identify, authenticate and,”)

Bit9 Parity, their whitelisting product, uses the software registry, a locally installed management server and the Parity client to enforce software policies throughout the enterprise. Bit9 can be found at <http://www.bit9.com>.

## 2.2. Coretrace Bouncer

Coretrace and their Bouncer product, is another pure-play application whitelisting vendor. Coretrace is best known for their innovation surrounding memory protection within a whitelisting solution. Coretrace has the ability to provide additional protection against buffer overflows in addition to standard file based whitelisting protection. Coretrace can be found at <http://www.coretrace.com>.

## 2.3. Faronics Anti-Executable

Faronics Anti-Executable is one of the new entrants into the whitelisting space. Anti-Exe is a “Product Loadin” which fits into Faronics Core agent. Faronics is probably best known for their Deep Freeze produce which is another module which can be put into the Core agent. Deep Freeze is a solution which returns a computer to a known good state after each and every reboot. Faronics also offers antivirs, power management and configuration management as part of their solution. Faronics can be found at <http://www.faronics.com>.

## 2.4. Lumension Application Control

Lumension Application Control is part of the company's Endpoint Management and Security Suite. Lumension's product suite is aimed at providing customers with one stop shopping for various endpoint security and management needs. These include antivirus, application control (whitelisting), device control, vulnerability, patch and configuration management. Lumension can be found at <http://www.lumension.com>.

## 2.5. McAfee Application Control

McAfee entered the application whitelisting marketplace by acquiring whitelisting vendor SolidCore in 2009. McAfee Application Control is a strong candidate for existing McAfee antivirus customers and those who utilize their EPO management solution. McAfee can be found at <http://www.mcafee.com>.

## 2.6. Microsoft AppLocker

Microsoft AppLocker is one of the more talked about solutions in the industry because Microsoft is providing it at no cost as part of Windows 7. Therefore, many vendors have been working to differentiate their solutions. AppLocker replaces Software Restriction Policies (SRP) which was part of Windows XP and Vista. Unfortunately, AppLocker is limited to Windows 7 only and does lack some of the features and manageability of other solutions. Information on AppLocker can be found at [http://technet.microsoft.com/en-us/library/dd560656\(WS.10\).aspx](http://technet.microsoft.com/en-us/library/dd560656(WS.10).aspx).

## 2.7. Savant Protection

Savant Protection is a pure-play application whitelisting vendor. Savant differentiates themselves by creating unique whitelists on each client system instead of in a centralized database. According to Savant, "This eliminates the need to query a central server or reference database, improves system performance, and eliminates a central point of attack." ("Savant protection -,")

### 3. Managing the Whitelist

One of the biggest decision points and discussions regarding the implementation of a whitelisting solution will be how to manage the whitelist. Vendors offer a variety of options for managing the whitelist. While there are some unique approaches specific to one or two solutions, most will include one or more of the following methods. While these options each can ease the pain of managing a whitelist, they can also increase the likelihood for compromise of the whitelist. Issues regarding the protection of the whitelist will be discussed in a later section.

#### 3.1. The Gold Image

Ideally, everyone will start creating their whitelist by first hashing a standard workstation image. This option is great for situations where systems are fairly static and unlikely to change much. However, this is only going to be a starting point for end user workstations which will likely change quite often in most organizations. After building using an image to build your first whitelist, keeping it up to date will be the biggest challenge facing most groups.

#### 3.2. Digital Certificates

One of the most effective techniques for easily managing a whitelist is to automatically trust certain publishers of software. Most software vendors digitally sign their applications. This digital signature can be used by many whitelisting vendors to automatically approve software from a specific vendor into the whitelist. For instance, in Microsoft Windows environments, automatically trusting Microsoft Corporations digital signature will automatically trust any digitally signed software from Microsoft. This can help greatly when it comes time for updates that patch Tuesday can bring. Also, this allows organizations to create more general policies where known software publishers can be trusted easily while still blocking malicious code. This can be highly effective in situations where providing some level of user freedom is important while still reducing the impact of malicious code.



### 3.3. Trusted Update Methods

Another method of keeping a whitelist up to date is the concept of trusted updaters. These are predefined accounts, processes or network locations which are automatically trusted. Automatic trust means that an application can be installed and automatically added to the enterprise whitelist.

A key process which ideally needs to be trusted is patch management. One could load patches and re-calculate the hashes against a known good image, but that would add significant time to the patching process. Instead most organizations will choose to automatically trust updates from their patch management systems.

Other options for automating trust include trusted network shares and trusted user accounts. A trusted network share is used to place installation files for approved programs. The associated file hashes are automatically added to the whitelist. Organizations may also choose to allow certain “administrator” accounts the right to install applications and have the associated hashes automatically added to the whitelist.

## 4. Aiding Incident Response

Application whitelisting gets a lot of attention as a prevention tool. Ideally, most organizations will be using this technology in this manner. However, organizations which cannot lockdown systems due to political or operational issues are still benefitting from the technology. Many organizations deploy whitelisting technology in a monitor only mode. This provides visibility into the executables which are running on end user systems and can be used to detect, confirm and respond to attacks.

### 4.1. Detecting Attacks

Whitelisting provides incident responders with a centralized log of the executable files existing on workstations throughout the organization. This can be extremely valuable in detecting and responding to attacks. Some whitelisting vendors also include the ability to detect known malicious files. Therefore, the whitelisting solution can be used in a similar manner as antivirus to detect known malicious files. Obviously, this approach has limited success, but can be complimentary to existing detection capabilities. In addition, some solutions can give a risk level to files

allowing for review of files which are hard to define as known good or known bad. An example is shown in Figure 2 in which the tool Dumpsec from Somarsoft has been discovered on a workstation. This tool is used to dump permissions and audit settings as well as user, group and replication data. Dumpsec is a great tool for system administrators, but could also be used by attackers to gain vital system information.



	Date Created ▼	Computer	File Name	Publisher or Company	User Name	Trust	Threat	Local State
	Oct 05 2010 08:32:14PM		dumpsec.exe	Somarsoft, Inc.				Approved

**Figure 2**

Beyond detecting known bad or questionable files, the data contained in whitelisting solutions can be used to detect files outside what is expected. For instance, assume an organization has deployed application whitelisting, but is unable to lockdown machines. Security staff could filter out certain files to create lists of questionable files which should be investigated. “With the Parity software, we can create baselines of our existing standard images... We can then filter out all of the files that are in my baselines, assuming those are known-good files. Next, I’ll toss out all of the files that are digitally signed. True, malware could be signed, but I can identify all of the signed files in a different report in Parity and that would be rather trivial to spot. I’ll also use the reasonable attribute of Threat to toss out all of the clean files. Threat is a verification on a hash level that Bit9 has an exact copy of that file in our ParityKnowledge repository and it has checked out to be clean by all of the leading Anti-Virus scanners. Finally, I’ll filter out all of the files larger than 1MB. Why would I do that? Well I took a look at over 10 million pieces of malware that we have collected for our knowledgebase, and statistically-speaking, 99% of malware over the past decade has been smaller than 1MB. I thought that was pretty amazing, but it actually makes sense: who wants to try and surreptitiously move a 10MB file around a network and onto hundreds of machines?” (Petrosky, 2010) Using this technique can provide a list of files which should be investigated for potentially malicious code.

Another useful detection method is to use other sources of alerts such as antivirus or intrusion detection logs and combine these with application whitelisting. For example, think of an intrusion detection alert, either network or host based, which detected a workstation being targeted with a known exploit. Was the attack successful? Even if the specific attack was blocked, many of today’s attacks employ multiple exploits so how do

you know another wasn't successful? Whitelisting logs are a great way to quickly determine the answers to these questions. Even if done manually, this can still save the analyst a significant amount of time and effort. In a perfect world, these logs would be correlated in a SIEM and presented to an analyst as a high priority event when an attack was detected and new code executed within a similar timeframe.

## **4.2. Incident Response**

When doing incident response, one could argue that the most important, yet often the most neglected phase in the incident response lifecycle is the preparation phase. "It makes sense to design networks and equip personnel in such a way that the organization can detect and handle security events as quickly and efficiently as possible to minimize the losses associated with downtime and data theft." (Casey, 2010). Whitelisting is a tool which can take your incident response process to a new level of effectiveness and efficiency. Whitelisting logs can provide a detailed look into the introduction of new executable files on a system over a period of time. This can help responders quickly determine how and when the compromise occurred. While there are other methods for determining similar information, such as a forensic review, they are not nearly as efficient. Beyond efficiency, whitelisting simply provides another layer of visibility into what is truly happening inside an organization. An added benefit is that this layer of visibility stays with the system regardless of location. This can help detect incidents while systems are away from the corporate network.

After identifying a successful attack, how does a responder determine the scope of the incident? Is the malicious code limited to a single workstation or has it spread to other systems inside the organization. This scenario provides another example of how whitelisting shines during IR. After finding a malicious executable on a single workstation, responders can quickly query the database to see if file hash exists on any other workstations in the environment. This is a quick and effective method for determining the scope of the incident.

## **4.3. Forensic Integration**

Whitelisting data can be a great ally to a forensic investigator. During an investigation, examiners can use a corporate whitelist to quickly limit the scope of their

investigation by removing known good files. Commercial tools are also starting to see the value of this integration. Guidance Software's EnCase Enterprise and Mandiant Intelligent Response both have options to include access to the Bit9 software registry as part of their tools. These tools help reduce investigation time by eliminating data which does not need to be reviewed. Michael Montecillo, Principal Analyst with Enterprise Management Associates' Security & Risk Management practice, said the following regarding the Mandiant and Bit9 integration. "The combination of the two serves to reduce the number of incidents an organization experiences and at the same time minimizes the impact of any successful attacks. For an enterprise, this means cost savings through better effectiveness of already existent processes." ("Mandiant, bit9 join," 2010)

## 5. Attack Scenarios

The following section is designed explain some of the more common malware attacks seen in today's environments and to test application whitelisting against these attacks. When specific attacks are analyzed, the client/victim system is a Windows-based virtual machine. The Bit9 Parity client will be used as the application whitelisting client. Each attack will be run twice, once without the client installed and once while running the Bit 9 client in lockdown mode.

### 5.1. Binders

"Binders are utilities that allow the user to bind one application to another, in essence creating a Trojan application. The idea is that the carrier application will entice the user to launch it; examples include games and other executables. When the victim launches the carrier application, he or she sees the application run, and nothing seems amiss. All the while, however, the Trojan application runs, often behind the scene, unbeknownst to the victim." (Carvey, 2007)

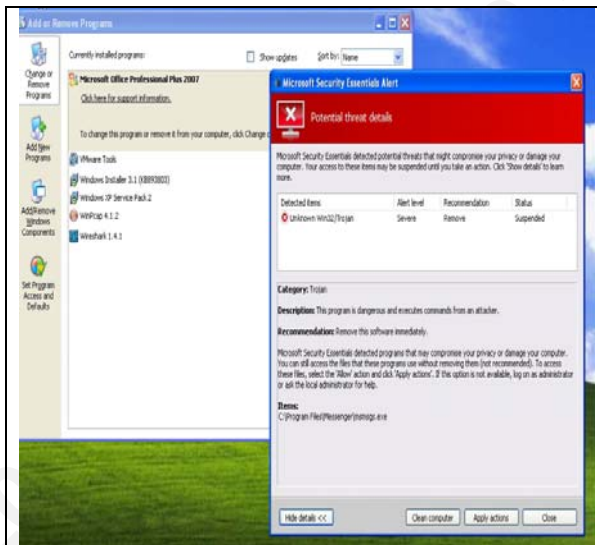
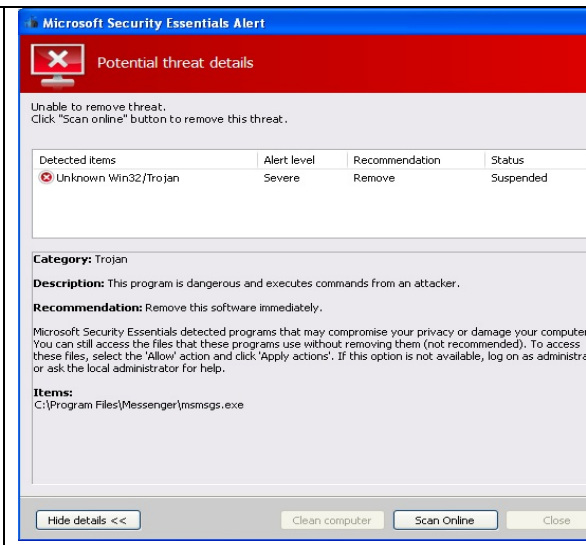
This kind of attack is a great illustration of how application whitelisting works. The whitelist may or may not list the original application, but for our purposes we will assume it does. The original application is permitted to run since its hash has been added to the whitelist. However, after "binding" the malicious application to the original, the

hash will change. This will ensure that the new application does not run. Needless to say, application whitelisting is highly effective against this kind of attack.

## 5.2. Fake/Rogue Antivirus

Fake/Rogue Antivirus has to be one of the more annoying malware trends of the past few years. These attacks are quite simple in the sense that they typically rely on convincing users to click on a link purporting to be security software, in order to install malware which pretends to find a security problem on the computer and won't let the user continue to use the computer without paying for their "solution" to the problem. From an enterprise standpoint, these attacks are costly in terms of the people resources lost to downtime and cleanup. However, these attacks also produce huge profits for attackers. "The two settling defendants were part of a massive deceptive advertising scheme that tricked more than a million consumers into buying "rogue" computer security products, including WinFixer, WinAntivirus, DriveCleaner, ErrorSafe, and XP Antivirus, according to the FTC's complaint." ("Ftc settles with," 2009)

The following screenshots show an example of a rogue antivirus program, ThinkPoint.

	
<p>Notice how the rogue antivirus program pretends to be Microsoft Security Essentials even though this program is not installed.</p>	<p>Looking at the details, it appears a Trojan has been detected. Note the executable is Microsoft Messenger.</p>

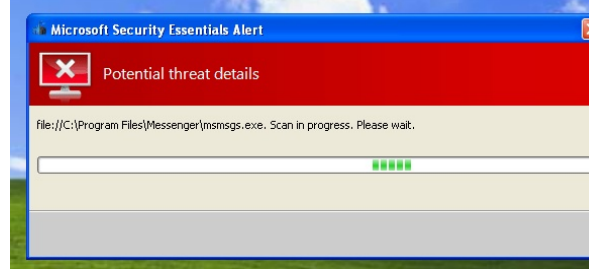
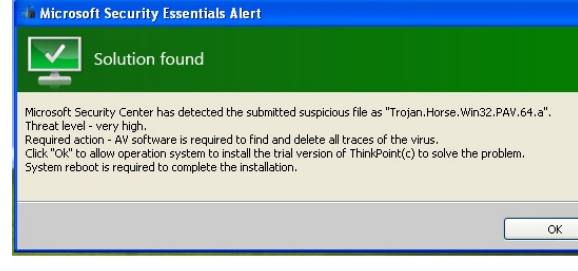

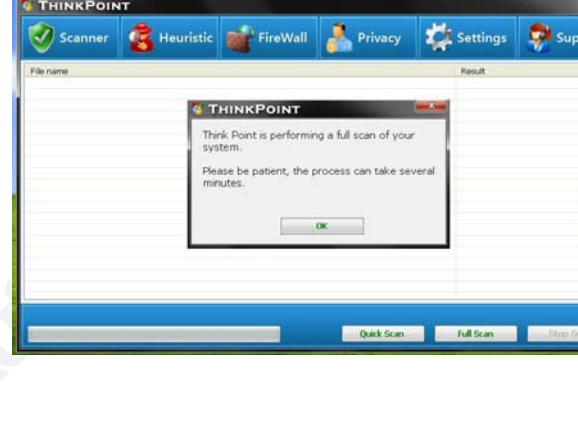

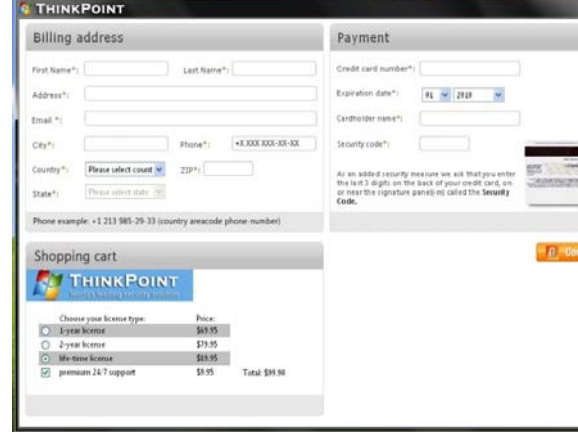
	
<p>After clicking “scan computer” a fake scan is delivered to the user.</p>	<p>A solution is found. After clicking “Ok” the machine reboots.</p>
	
<p>After rebooting this screen indicates that ThinkPoint – World’s leading security solution is loading.</p>	<p>The program then presents the user with an option to conduct a full scan of the computer.</p>
	
<p>Very quickly the following message appears showing 11 items that can't be restored as the “heuristic module missing”</p>	<p>Clicking “Install the full version” brings the user to a payment page where they can “purchase” a full version.</p>

Figure 3

Application whitelisting is very effective against this type of attack. The initial executable download is simply blocked and not allowed to execute, preventing all the of steps show above. Figure 4 shows the warning message provided by the Bit9 Application Whitelisting client after downloading the Fake/Rogue AV executable.

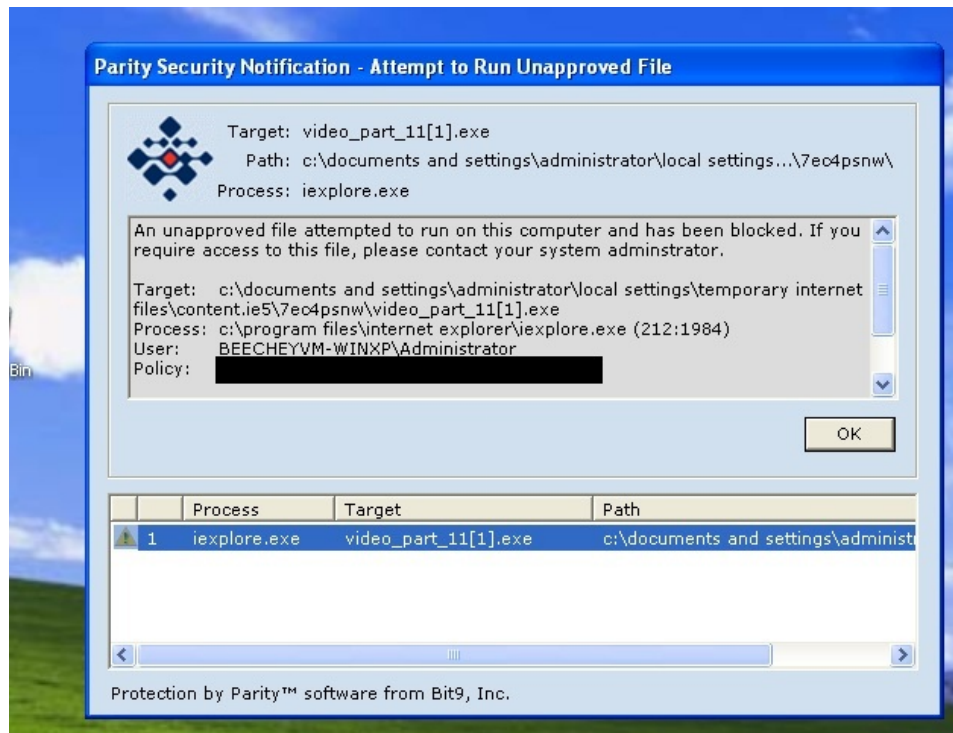


Figure 4

### 5.3. DLL Hijacking Vulnerabilities

DLL hijacking vulnerabilities received a lot of publicity in the fall of 2010 due to the widespread nature of the problem, inability for Microsoft to patch the problem and potential for remote execution. Nick Harbour of Mandiant wrote a great blog post covering the issue. “The problem currently making headlines comes down to the fact that when most programs recursively enumerate files, they do so by setting the current working directory to each directory they find before examining the files found in that directory. By setting the current working directory to a location, you expose DLLs found in that directory to be a part of the DLL search order and in some cases they will be loaded... The reason for all the attention this has received is that many applications will traverse files across a network share, allowing remote infection by applications that both



set current working directory for file traversals and have a mechanism where a DLL can be caused to load based on the file type or contents.” (Harbour, 2010)

The following example uses Metasploit to create and host a PowerPoint file which is used to exploit the DLL hijacking vulnerability and gain a meterpreter shell on the victim machine. First, setup Metasploit with the following configuration.

```

msf > use exploit/windows/browser/webdav_dll_hijacker
msf exploit(webdav_dll_hijacker) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(webdav_dll_hijacker) > set LPORT 9999
LPORT => 9999
msf exploit(webdav_dll_hijacker) > set EXTENSIONS "ppt pptx"
EXTENSIONS => ppt pptx
msf exploit(webdav_dll_hijacker) > set LHOST 10.
LHOST => 10
msf exploit(webdav_dll_hijacker) > exploit
Exploit running as background job.
msf exploit(webdav_dll_hijacker) >
Started reverse handler on 10.          :9999

Exploit links are now available at \\10.      \documents\

Using URL: http://0.0.0.0:80/
Local IP: http://10.          :80/
Server started.
msf exploit(webdav_dll_hijacker) >
  
```

Figure 5

After connecting to the malicious link and opening the PowerPoint document, process monitor shows two processes called rundll32.exe running underneath POWERPNT.EXE. These processes are the two meterpreter sessions which were created in this example.

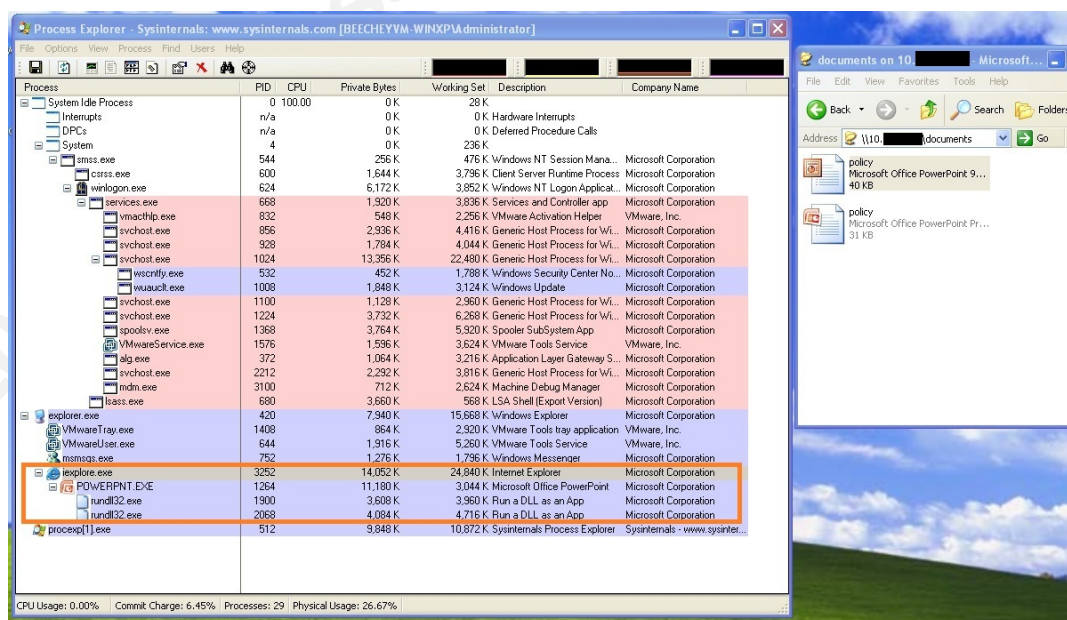


Figure 6



However, when we setup the same attack on a machine running the Bit9 parity client, we see very different results. First, note that there are no processes called rundll32.exe running under POWERPNT.exe. Second, a security alert has popped up notifying the user that an unapproved file attempted to run on the computer. In this scenario, these are the two malicious dll files used by the exploit.

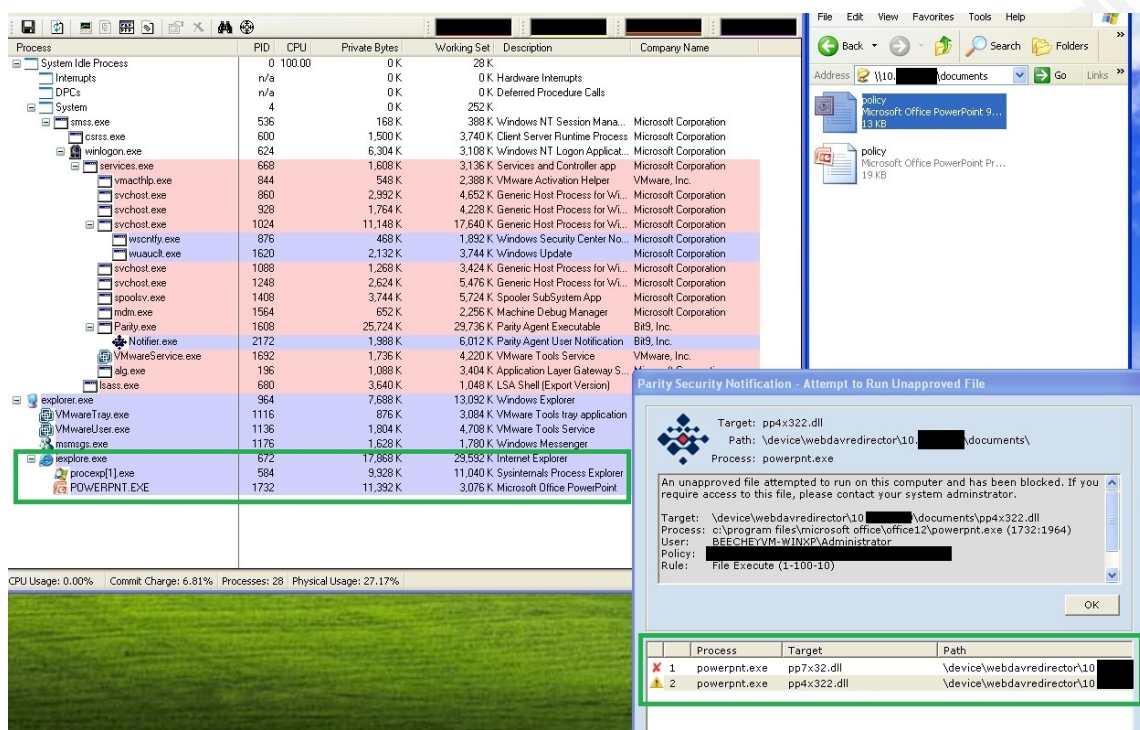


Figure 7

## 5.4. Drive-by Download Attacks

Drive-by download attacks have become a very common method for mass exploitation. “Drive-by download attacks are downloads that occur without the knowledge or consent of a user. After downloading, the application is invoked and is free to perform its nefarious purposes. The mere visit to a malicious web site can lead to the download and subsequent execution of malicious software on a visitor’s computer.” (Egele, Kirda, & Kruegel) Of course, these attacks rely on actually getting unsuspecting users to visit their malicious web sites. This can be done in a number of ways. “The three most common scenarios are: Search Engine poisoning, malicious forum posts, and malicious flash ads.” (Liston, 2010)

In order to make these attacks even more successful, attackers use sophisticated exploit packs to deliver the appropriate exploits based upon which browser and plug-ins

are being used. “Exploit packs — slick, prepackaged bundles of commercial software that attackers can use to booby-trap hacked Web sites with malicious software — are popular in part because they turn hacking for profit into a point-and-click exercise that even the dullest can master.” (Krebs, 2010) Figure 8 is a screen shot taken from a working Crimpack exploit kit and shows statistics for several exploits, operating systems, browsers and countries.

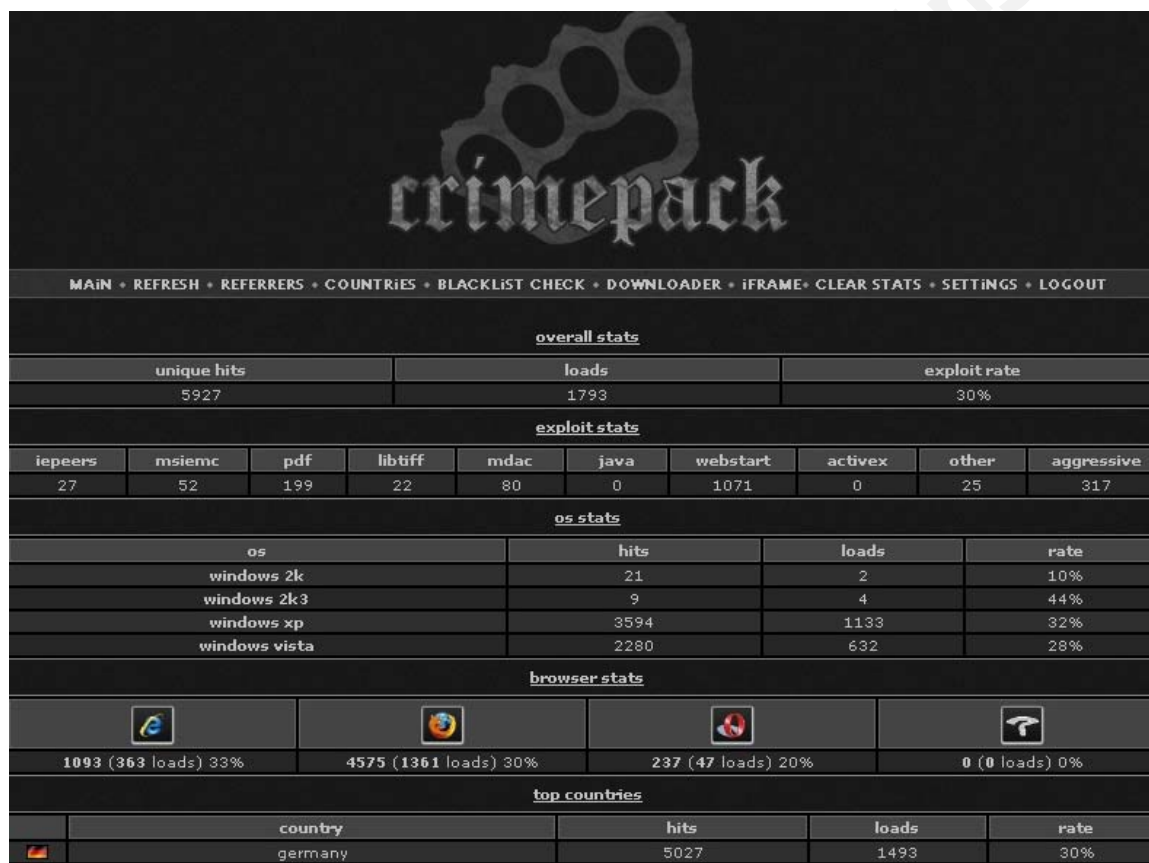


Figure 8 (Krebs, 2010)

The following example shows Bit9 Parity protecting a client against a drive-by download attack. In this case a wedding photographer's blog has been compromised and includes an extra treat for visitors. While browsing the blog, visitors are also unknowingly downloading an executable named file.exe from a Ukrainian IP address (78.26.187.48). Figure 9 shows flow data for the malicious download. Note the referring URL, HTTP Host and HTTP Get Request fields. Figure 10 shows the VirusTotal report for file.exe. Note the 25.6% detection rate by the various antivirus scanning engines.

Figure 11 shows the photographer website and Bit9 parity blocking file.exe from executing.

Flow Information					
Protocol:	tcp_ip	Application:	Web.Text.HTML		
Magnitude:	(3)	Relevance:	5	Severity:	1
First Packet Time:	2010-12-02 17:38:52	End Time:	2010-12-02 17:39:52		
Event Name:	Web.Text.HTML				
Low Level Category:	Web				
Event Description:	Application detected with state based decoding				
HTTP User-Agent (custom):	N/A				
HTTP Host (custom):	78.26.187.48				
HTTP GET Request (custom):	/god/jrbkmbpbjrbpalemi.php HTTP/1.1				
HTTP Content-Type (custom):	text/html; charset=Windows-1251				
HTTP Response Code (custom):	200 OK				
Google Search Terms (custom):	N/A				
HTTP Server (custom):	Apache/2.2.3 (CentOS)				
HTTP Version (custom):	1.1				
HTTP Referer (custom):	http://blog.████████.photo.com/2009/08/17/████████				


  

Source and Destination Information			
Source IP:	10.████████	Destination IP:	78.26.187.48
Source Asset Name:	N/A	Destination Asset Name:	N/A

Figure 9

VT Community [Sign in](#) Languages ▼

---




Virustotal is a **service that analyzes suspicious files and URLs** and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware detected by antivirus engines. [More information...](#)

0 VT Community user(s) with a total of 0 reputation credit(s) say(s) this sample is goodware. 0 VT Community user(s) with a total of 0 reputation credit(s) say(s) this sample is malware.

File name:	<b>file.exe</b>
Submission date:	<b>2010-12-03 02:51:01 (UTC)</b>
Current status:	<b>finished</b>
Result:	<b>11 / 43 (25.6%)</b>

[Compact](#)
[Print results](#)

**VT Community**



not reviewed  
Safety score: -

Figure 10

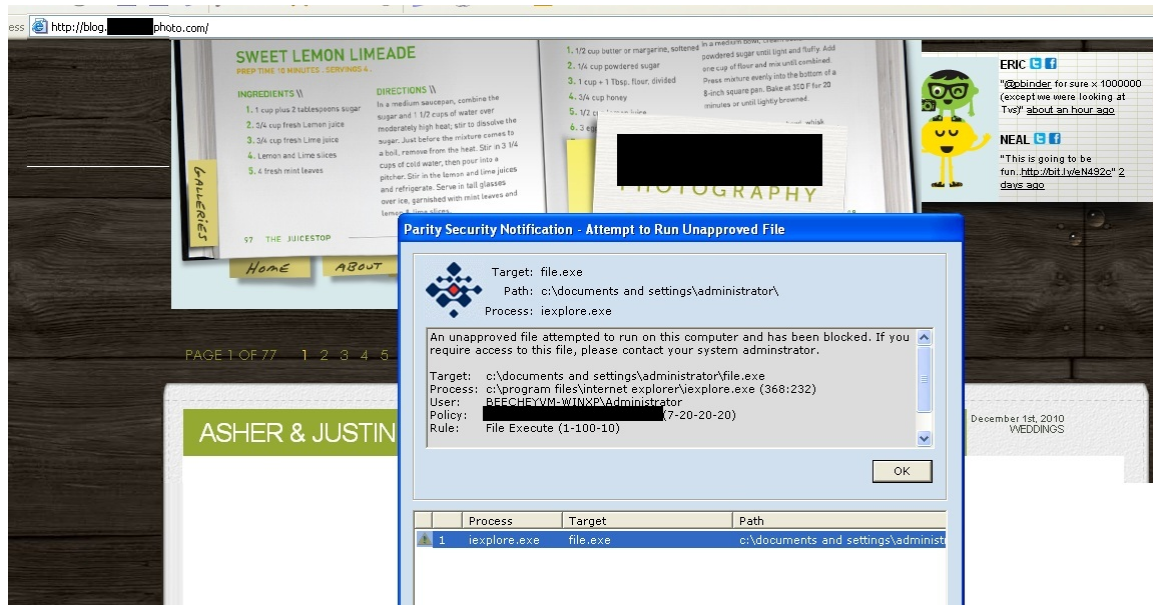


Figure 11

## 5.5. Web Application Attacks

Web application attacks have also been a major concern for security professionals over the past few years. While they differ from malicious code, their impact on an organization can be extremely significant. There are numerous kinds of web application attacks; but we will focus on the two most prevalent, SQL injection and Cross Site Scripting (XSS).

SQL injection is a serious concern for anyone running web applications which utilize back-end database servers. “The general idea is that an attacker can append his own SQL commands on to the end of a dynamically created query that is submitted to the SQL server backend. Without proper sanitation, a malicious SQL query can be easily created. To compound the issue, certain databases contain powerful functionality that can give an attacker direct access to the operating system” (Troost, 2009). SQL injection can be used to steal, delete or manipulate data as well as compromise backend servers in order to attack other internal systems.

Application whitelisting is not very effective against SQL injection simply because most of the attacks can be completed with applications likely already trusted in your environment. However, an organization which employs application whitelisting on both the external web server and internal database server does have the ability to limit the impact beyond the initial attack. For instance, after compromising the backend database

server, attackers will often download other, more specialized, attack tools in order to continue penetrating the network. Application whitelisting can be effective at blocking these attacks, and hopefully alerting responders to the initial compromise, by blocking the execution of the specialized attack tools.

Cross site scripting, or XSS, attacks are more focused toward client workstations. “According to NIST Special Publication 800-95, XSS attacks are possible when a valid Web service has their requests transparently rerouted to an attacker-controlled Web service, most often one that performs malicious operations.” (Wilhelm, 2009) XSS examples below will be conducted using the Browser Exploitation Framework (BeEF) which can be found at [www.bindshell.net/tools/beef](http://www.bindshell.net/tools/beef). The client machine is running Bit9’s parity software in lockdown mode to simulate a protected client.

First, the attacker must entice the unsuspecting user to visit a vulnerable web page. In this scenario we are using the page built into the BeEF console. After visiting the page, the browser is “hooked” and appears as a zombie inside the attacker’s console. At this point, the attacker has a variety of options at their disposal. In this example shown in Figure 11, we’ve chosen to send the hooked browser a message asking for them to enter their password. After the password is entered, the data is transferred to the BeEF console and is available to the attacker. See Figure 12 for details.

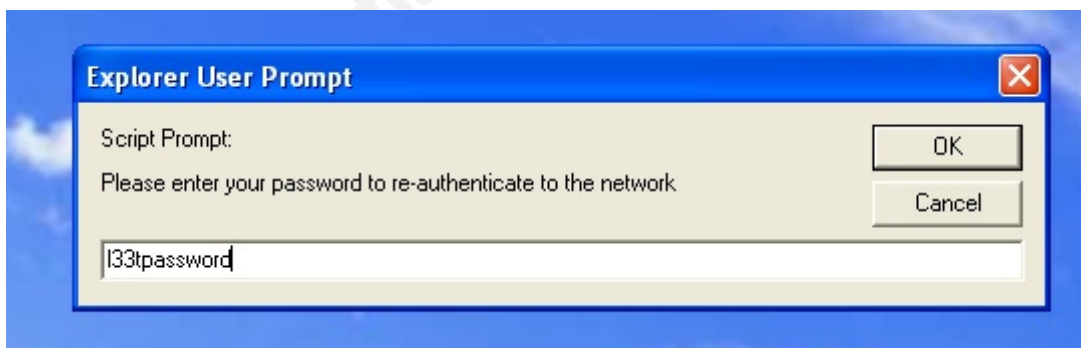


Figure 11



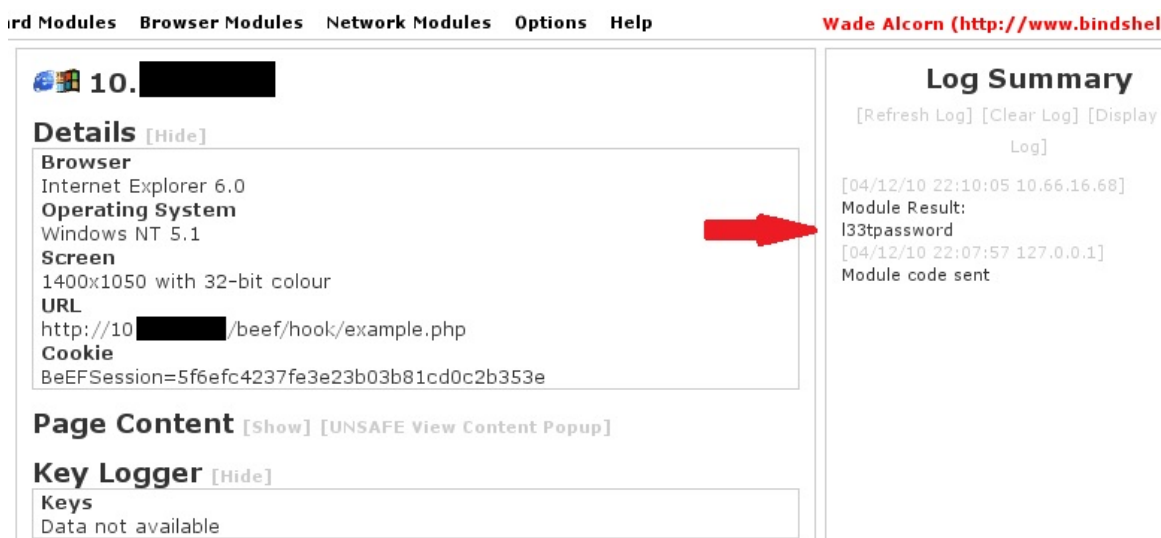


Figure 12

As you can see, application whitelisting has had no effect on this attack. Similar to SQL injection, each of the requirements for this attack to be successful are already included in the attacked system and trusted by the whitelisting solution. Any XSS attack which does not require additional, un-trusted software, cannot be stopped by application whitelisting.

## 6. Attacking Whitelisting

Thus far we've covered several scenarios where whitelisting can be a highly effective solution for protecting against various kinds of attacks. However, there are no perfect security solutions and whitelisting is no exception. When implementing whitelisting, organizations need to consider what additional protections should be put into place in order to protect the whitelist itself.

### 6.1. Attacking Management Functions

Earlier we discussed various methods for making the management of whitelisting easier for systems administrators. However, these features can also create opportunities for attackers to bypass whitelisting. Understanding these risks and adding security controls where possible is important to maximize the effectiveness of your deployment.

### 6.1.1. Attack Software Distribution Points

Easily dealing with software updates and patches is key to any successful whitelisting implementation. Therefore, most organizations will trust specific directories or process in order to easily maintain their systems. This trust will allow the installation and automatic whitelisting of certain software. Attackers can use this knowledge to target software installation directories and systems to attempt to get their malicious software automatically trusted, possibly installed.

Organizations can limit their exposure to these attacks by ensuring that a limited number of people have access to update software installation directories and processes. In addition, file level auditing should be turned on and collected centrally for any trusted directory. This will allow someone to review changes, ensure proper procedures are being followed and hopefully identify any successful attacks.

### 6.1.2. Stealing Digital Signatures

One of the most effective management tools for a whitelist is approving software via digital signature. For example, trusting all software signed by Microsoft allows the automatic whitelisting of Microsoft operating systems and applications. This is especially effective when system patches and updates need to be deployed.

However, if an attacker can steal an approved digital signature and incorporate it into their malicious code, your whitelist would be useless. Stuxnet, arguably the most advanced piece of malicious code to date, utilized two stolen digital certificates. While this is an advanced attack, likely left for the more advanced adversary, it has become more common. “In the 15 years or so of serious malware production before 2010, there had been perhaps a handful of examples of malicious programs using digitally signed binaries to bypass antimalware systems. The emergence of Stuxnet earlier this year brought this tactic into the center of the spotlight, and now researchers say that the new mobile Zeus variant that is targeting Symbian and BlackBerry devices is following suit, using a stolen digital certificate to help cloak itself from security systems.” (Fisher, 2010)

The benefits of utilizing digital signatures for application approval still likely far outweigh the risks. However, high security organizations should take these threats seriously in their system design and consider other alternatives in managing their

whitelists. All organizations need to implement processes to respond when new malicious code samples are found to be using legitimate digital signatures.

### **6.1.3. Attack the Database**

Most application whitelisting vendors include some kind of centralized database for the creation and management of the whitelist. Often these applications include some kind of front end web application for administration purposes. These are prime targets for standard operating system and web application attacks. Certainly, these applications should not be public facing, but may become targets for compromise after an attacker gains an initial foothold in an organization. The ultimate goal of the attacker is to add entries for their malicious code.

In order to protect these assets, organizations should consider several system layers of additional defense. First, harden operating systems and required applications according to industry best practices and vendor whitepapers. The Center for Internet Security ([www.cisecurity.org](http://www.cisecurity.org)) is a great starting point for system hardening guides. Second, organizations should consider walling off application whitelisting servers from the rest of the network as much as possible. Router access control lists, network and host-based firewalls are good options for this layer of protection. Only ports absolutely required should be opened. Most solutions will require web access, via SSL, and at least one additional port for communicating with clients. Since many of these systems run on Windows servers, you will also need to consider windows domain based communications. Try to limit this traffic to as few servers and services as possible.

### **6.1.4. Attack the Client**

Whitelisting solutions require the installation and execution of agent software on each system. As with any software, there is the potential for vulnerabilities in these agents. Finding such vulnerabilities is not trivial; however the reward for the attacker would be substantial. Attacks could be used to disable, remove or trick the agent.

### **6.1.5. Malicious IT Insider**

Probably the most effective attack against whitelisting would be to have a malicious insider, with appropriate access, add hashes of targeted malware to the whitelist. This, of course, would take considerable time, effort and likely dollars to pull

Jim Beechey, [beechey@northwood.edu](mailto:beechey@northwood.edu)



off, but it not out of the realm of possibility considering today's organized and advanced adversaries. Organizations can limit their exposure to this kind of attack by implementing tight change control and approval procedures for whitelist changes including separation of duties between those updating and those reviewing whitelisting changes.

#### **6.1.6. Attack the Admin**

Successfully attacking the whitelist can be as simple as compromising the right credentials. This is no different than many of the attacks seen today. Once privileged account credentials are compromised such as root, domain admin or high level user access; attackers use these legitimate credentials to complete their attacks. In this case, the likely goal would be to compromise the account of a whitelist administrator in order to add a specific attack tool to the whitelist. This would allow the adversary to continue to compromise additional targets in the organization.

This type of attack could be used to add a specific entry to the centralized list, but likely would be much more effective against solutions and organizations which have allowed specific accounts to automatically install and whitelist software. These accounts would likely be used by Help Desk and PC Support Technicians to quickly address new software requests from end users. Unfortunately, this setup leaves one of the most gaping holes in the defense of a whitelist and should be used with caution.

Another reason to attack the admin is that they are probably the most likely systems to not have whitelisting software installed. My experience has been that systems which are most un-patched, deviate furthest from a standard image and most likely to not have AV software installed often exist in IT. Application whitelisting does create an additional step in testing new software and using non-standard tools, so it is fair to assume some system administrators may be reluctant to lockdown their own systems. These issues really boil down to the strength and enforcement of your internal policies.

The bottom line in minimizing each of these attacks is that a little bit of extra work can go a long way in providing additional protection to the key assets in your whitelisting deployment. Make these protections part of your initial deployment. Include

systems administrators as part of the project to increase communication of key issues and ensure they can be protected without adversely impacting their productivity.

### 6.1.7. Case Study Example

During a recent Mandiant “State of the Hack” Webinar, Christopher Glyer and Ryan Kazanciyan presented a mini-case study covering the use of application whitelisting in a recent APT investigation. After discovering the compromise and prior to remediation, the company in question decided to deploy application whitelisting on their domain controllers in order to best protect their enterprise password hashes. After the whitelisting tool was implemented the attacker copied pwdump to a domain controller successfully, unsuccessfully attempted to execute pwdump through a scheduled task and psexec and, finally, unsuccessfully attempted to disable whitelisting. Next, the attacker began to target the company’s software deployment infrastructure in order to run the code on the domain controllers and, ultimately, the application whitelisting control server itself. The company responded by deploying the whitelisting solution to their software deployment infrastructure, removing the control server from the domain and implementing two factor authentication. This is a fantastic study of how attackers will attempt to exploit flaws in your application whitelisting architecture and a great example of a limited deployment of application whitelisting having a significant impact. The full presentation can be found here: <http://www.mandiant.com/presentations/state1/>.

## 6.2. Attacking Vulnerable, Trusted Applications

One of the most important concepts to understand relating to application whitelisting is how solutions handle attacks against applications which are trusted, but have existing vulnerabilities. For instance, what happens when an attacker tries to run an exploit, against a workstation protected with whitelisting, which is missing a Microsoft Office security patch or running a vulnerable version of Adobe Flash? “What about *Office* exploits? They arrive in a *Word*, *Excel* or *PowerPoint* document. Some obscure field in the document is corrupted, causing a buffer overflow somewhere in the *Office* application that opens it. This (the corruption) is the exploit. The exploit causes control to be transferred to a small piece of code that resides in the document too (usually, but not always, close to the corrupted field). This small piece of code is called ‘shellcode’. Then

Jim Beechey, beechey@northwood.edu

it usually extracts the real malicious program appended (often in encrypted form) after the end of the document – or downloads it from somewhere and runs it. Now, a whitelist-based approach can prevent the dropped (or downloaded) executable from running. But it cannot stop the execution of the shellcode – not unless it stops the *Office* applications from running or disallows the opening of foreign documents – both of which would make the machine essentially unusable. And the shellcode doesn't really have to drop an executable – it's just easier to implement it this way. The shellcode runs directly in memory, in the context of the user who has opened the malicious document, and can do everything that the user is allowed to do.” (Bontchev, 2007)

The following example illustrates this issue. The client workstation running Windows XP SP2, is protected with whitelisting, but vulnerable to the issue described in Microsoft bulletin MS 08-067 (<http://www.microsoft.com/technet/security/bulletin/ms08-067.msp>). This is the infamous Conficker vulnerability which garnered much attention in 2008. The machine is attacked using the newly released Metasploit Express tool. Figure 13 shows the Metasploit Express successfully compromising the machine via the MS 08-067 vulnerability. Figure 14 (red box) shows various post exploitation options available under Metasploit Express. Each of these options successfully worked on our compromised host. The “virtual desktop” option worked intermittently. Each of the other options worked consistently, although we were not able to execute an uploaded file that was not part of the whitelist.

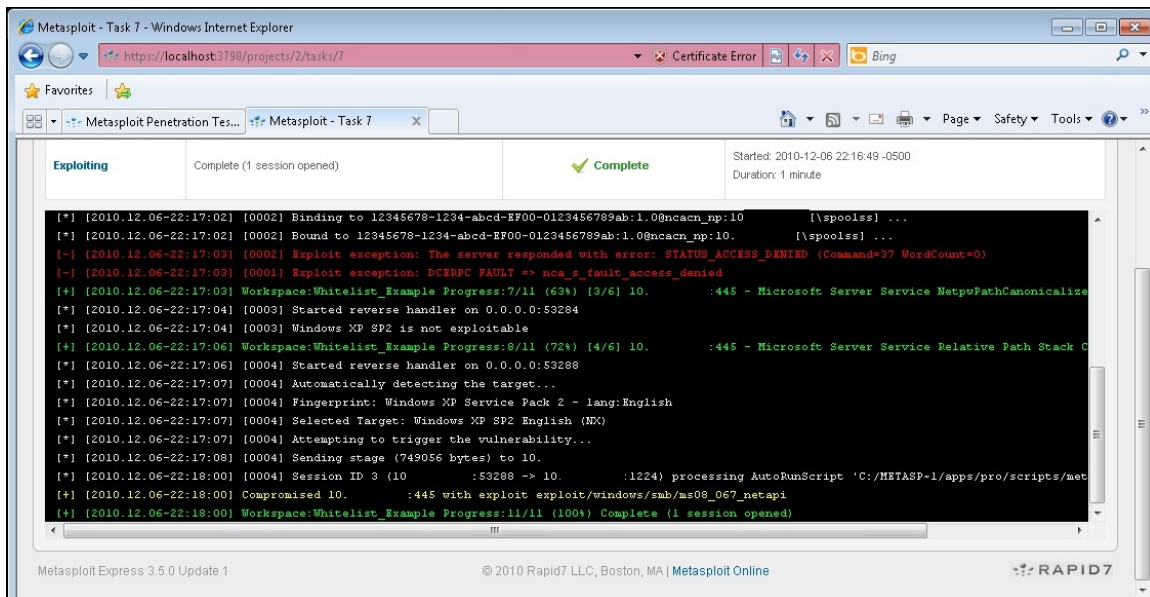


Figure 13

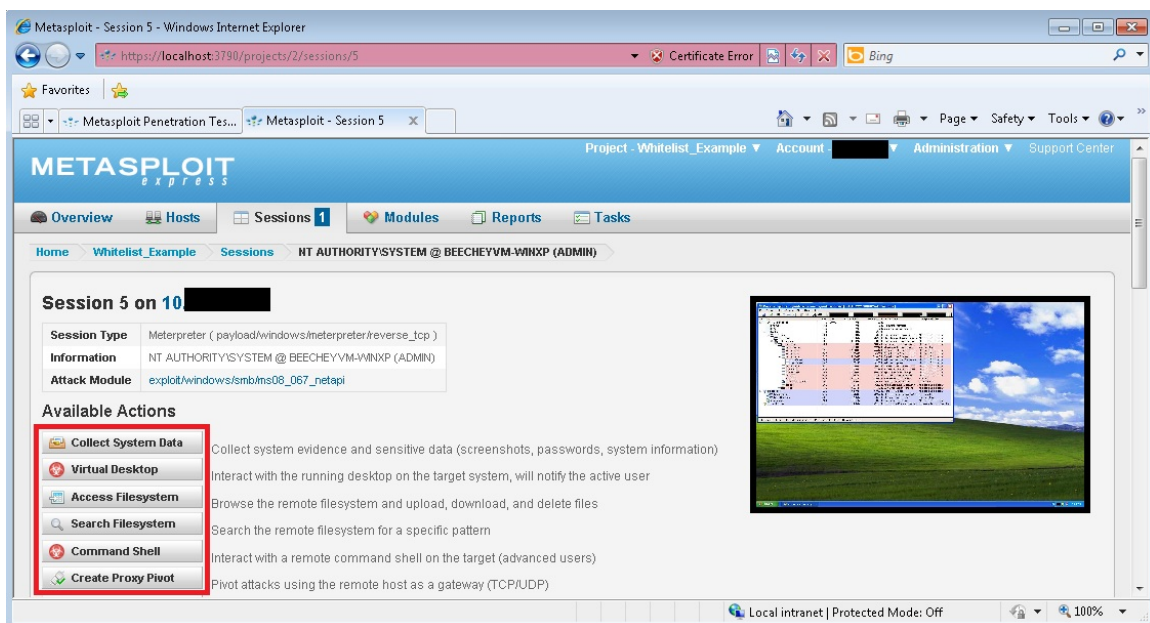


Figure 13

This example illustrates whitelisting's greatest challenge, memory. Greg Hoglund, founder of rootkit.com and HBGary, describes this issue in detail on his blog. "Please understand that files on disk are not the same as files in memory. And all that matters is memory. When a file is LOADED into memory, it CHANGES. This means on-disk MD5 sums do not map to memory. There are several reasons memory is different:

Jim Beechey, beechey@northwood.edu

- 1) Memory contains much more data than the on disk file
- 2) Memory contains thread stacks
- 3) Memory contains allocated heaps
- 4) Memory contains data downloaded from the Internet
- 5) Memory contains secondary or tertiary files that were opened and read
- 6) Memory contains data that is calculated at runtime
- 7) Memory contains data that is entered by a user

All of the above are not represented by the file on disk. So, none of the above are represented by the whitelist MD5 sum. Yet, when the file hash on disk passes for whitelisted, the running in-memory file is considered whitelisted by proxy. This is where the whole model breaks down.” (Hoglund, 2008) Therefore, application whitelisting does not remove the need for other complimentary technologies. Most importantly, patch management must be part of any organizations plans regarding whitelisting. In addition, while some suggest removing antivirus and HIPS technologies when deploying whitelisting, there still are circumstances where these technologies can assist in providing protection whitelisting is not able to provide.

Whitelisting vendors acknowledge these issues in memory and have been working to address the issue. Coretrace, McAfee and Bit9 all provide some level of memory protection in their whitelisting solutions. The goal is typically to stop buffer overflows from occurring in whitelisted processes which leads to many of the issues discussed above. Regardless of the vendor solution chosen, organizations that deploy whitelisting must consider these issues and plan their deployments accordingly.

## 7. Advanced Persistent Threat (APT)

Any discussion of attack scenarios in today’s environment is almost certain to include the advanced persistent threat or APT. Talk regarding this type of attack has dominated information security community in 2010. Commercial vendors have also latched on to the threat as a marketing opportunity. Application whitelisting vendors are

certainly no exception. In fact, many have been particularly aggressive in discussing their products ability to combat the advanced persistent threat.

When discussing APT, there are a couple of key issues to understand. First, the term APT and the threat it represents are not new. “The United States Air Force coined the phrase advanced persistent threat in 2006 because teams working within the service needed a way to communicate with counterparts in the unclassified public world.” (Bejtlich, 2010) Second, the problem with discussing APT in relational to a specific technology, specifically application whitelisting, is that APT is not a specific type of attack, but a classification of attacker. “**Advanced** means the adversary can operate in the full spectrum of computer intrusion. They can use the most pedestrian publicly available exploit against a well-known vulnerability, or they can elevate their game to research new vulnerabilities and develop custom exploits, depending on the target's posture. **Persistent** means the adversary is formally tasked to accomplish a mission. They are not opportunistic intruders. Like an intelligence unit they receive directives and work to satisfy their masters. Persistent does not necessarily mean they need to constantly execute malicious code on victim computers. Rather, they maintain the level of interaction needed to execute their objectives. **Threat** means the adversary is not a piece of mindless code. The opposition is a threat because it is organized and funded and motivated. Some people speak of multiple "groups" consisting of dedicated "crews" with various missions.” (Bejtlich, 2010)

Given this understanding of APT, one cannot say that this adversary can be defeated by any single technology, including application whitelisting. APT by nature, will use whatever means necessary to achieve their objective. This doesn't mean application whitelisting can't help make it more difficult for this adversary to penetrate your network or provide responders more visibility to detect their attacks. However, saying that application whitelisting can *defeat* this adversary is marketing hype.

## 8. Conclusion

The malware problem is significant and shows no signs of lessening. While new signatures and heuristic techniques are developed each day, clearly current protection technologies are struggling to keep up. Application whitelisting is not perfect. Managing

Jim Beechey, beechey@northwood.edu

the whitelist can prove difficult in large, open environments. The challenges in dealing with memory based attacks such as buffer overflows are clear. However, regardless of these challenges, application whitelisting can provide significant benefit to any organization. First, whitelisting provides a dramatic improvement in the level of visibility into files being introduced into an environment. This can be extremely helpful for incident responders. Second, organizations can use the technology to significantly reduce the risk of today's malware. This helps reduce the likelihood of system compromise and reduces cost of staff to deal with malware related issues. While no security solution can be described as a panacea, application whitelisting certainly is not propaganda. In fact, application whitelisting is the most effective way to significantly reduce the impact of malware in today's environments.

## 9. References

- Bejtlich, R. (2010, July). *Understanding the advanced persistent threat*. Retrieved from [http://searchsecurity.techtarget.com/magazinePrintFriendly/0,296905,sid14\\_gci1516312,00.html](http://searchsecurity.techtarget.com/magazinePrintFriendly/0,296905,sid14_gci1516312,00.html)
- Bontchev, V. (2007, August 1). *The dark side of whitelisting*. Retrieved from <http://www.virusbtn.com/virusbulletin/archive/2007/08/vb200708-whitelisting>
- Carvey, H. (2007). *Windows forensic analysis*. Burlington, MA: Elsevier.
- Casey, E. (2010). *Handbook of digital forensics and investigation*. Burlington, MA: Elsevier.
- Egele, M, Kirda, E, & Kruegel, C. (n.d.). *Mitigating drive-by download attacks: challenges and open problems*. Retrieved from <http://www.iseclab.org/papers/inetsec09.pdf>
- Firth, N. (2010, September 11). *'here you have' virus that promises free sex films causes havoc as it spreads across the world*. Retrieved from <http://www.dailymail.co.uk/sciencetech/article-1310890/Here-virus-causes-havoc-spreads-world.html>
- Fisher, D. (2010, September 30). *Stolen digital certificates becoming standard components*. Retrieved from [http://threatpost.com/en\\_us/blogs/stolen-digital-certificates-becoming-standard-malware-components-093010](http://threatpost.com/en_us/blogs/stolen-digital-certificates-becoming-standard-malware-components-093010)
- Ftc settles with two defendants in bogus computer scan case*. (2009, June 25). Retrieved from <http://www.ftc.gov/opa/2009/06/winsoftware.shtm>
- Harbour, N. (2010, August 31). Dll search order hijacking revisited [Web log message]. Retrieved from <http://blog.mandiant.com/archives/1448>

- Hoglund, G. (2008, June 30). Whitelisting is the next snake oil [Web log message]. Retrieved from <http://fasthorizon.blogspot.com/2008/06/whitelisting-is-next-snake-oil.html>
- Identify, authenticate and trust software*. (n.d.). Retrieved from <http://www.bit9.com/products/bit9-global-software-registry.php>
- Krebs, B. (2010, August 5). *Crimepack: packed with hard lessons*. Retrieved from <http://krebsonsecurity.com/2010/08/crimepack-packed-with-hard-lessons/>
- Krebs, B. (2010, June, 25). *Anti-virus is a poor substitute for common sense*. Retrieved from <http://krebsonsecurity.com/2010/06/anti-virus-is-a-poor-substitute-for-common-sense/>
- Liston, K. (2010, November 3). *Defeating drive-by downloads in windows*. Retrieved from <http://isc.sans.edu/diary.html?storyid=9880>
- MacDonald, N, & Silver, M. (2008, August 4). *Application control market update*. Retrieved from <http://my.gartner.com/portal/server.pt?open=512&objID=260&mode=2&PageID=3460702&docCode=159032&ref=docDisplay>
- Mandiant, bit9 join forces*. (2010, March 2). Retrieved from [http://www.mandiant.com/news\\_events/article/mandiant\\_bit9\\_join\\_forces/](http://www.mandiant.com/news_events/article/mandiant_bit9_join_forces/)
- Petrosky, M. (2010, September 7). Finding the needle in the haystack [Web log message]. Retrieved from <http://blog.bit9.com/bid/14039/Finding-the-Needle-in-the-Haystack>
- Savant protection - products - how it works*. (n.d.). Retrieved from [http://www.savantprotection.com/en/products/how\\_it\\_works.php](http://www.savantprotection.com/en/products/how_it_works.php)
- Sophos list of recently infected websites' photostream*. (2010, May 24). Retrieved from <http://www.flickr.com/photos/50473116@N05/>
- Trost, R. (2009). *Practical intrusion analysis: prevention and detection for the twenty-first century*. Boston, MA: Addison-Wesley Professional.
- Wilhelm, T. (2009). *Professional penetration testing: creating and operating a formal hacking lab*. Burlington, MA: Syngress.