GIAC Enterprises

Where your Fortune is our Business …

Malware Infection Tiger Team Final Report

Jim McMillan
Rob VandenBrink

14 Sept 2009

Executive Summary

The purpose of this paper is to explore the recent incident involving the DownAdUp (better known as Conficker) malware detection, and to provide recommendations to GIAC on further detection, remediation and prevention measures that may be required.

The conclusion we reached is that the recent event probably involved a Conficker infection, but not at GIAC. It was most likely a spam email sent from the companion virus Waledac, which "piggybacks" on Conficker.C and newer variants. The initial email probably originated from outside of GIAC, and was simply masquerading as a email from the GIAC CIO. We would need the email server logs from the target organization to comment any further on this.

However, the exploration into the incident has brought to light several related issues at GIAC that should be dealt with. The concern for the Conficker virus should be extended to the thousands of other malware packages that are created each day, and some basic changes to our IT infrastructure and team should be considered. These are separated into recommendations of low or no cost, and recommendations that should be considered for next years budget. First, let's consider the low/no cost recommendations:

- We should be devoting significant resources to a Security Awareness program within our user community. While it is not an easy task, the costs are relatively low, and this promises to deliver the most benefit of any of our recommendations in preventing future infections.
- We should create a Password Policy and communicate it to our user community. Enforcement of this policy can be done using Group Policy, and auditing for compliance to this policy can be done with any number of tools, including "john the ripper" and "cain". This recommendation is without cost, aside from time spent in implementation.
- A more stringent approach to applying vendor patches in a timely manner should be adopted. If possible, all patches should be applied within 24 hours of release. WSUS is a free tool from Microsoft that can help with this, and it can be extended (albeit in a clumsy way) to include patches from other vendors as well.
- We should scan our network for any workstations or servers that have not been patched for the Conficker vulnerability. This can be done at no cost using NMAP or WMIC based scripts (both included in this paper). The WMIC approach can in turn be expanded to include other specific patches if the need should arise in future.
- To detect future zero day infections, we should periodically scan our network for new, unknown processes running on workstations and servers (free scripts are included in this

paper). In addition, NMAP can be used to scan the network for new processes with listening TCP and UDP ports (free script-snips are included in this paper).

Some of the recommendations we arrived at, unfortunately, do bear a significant cost. These include:

- A Security Awareness program, specific to the IT group, should be implemented. Some facets of this, such as regularly visiting security websites or reading security blogs and papers, are free. The recommendations for training however do carry a cost, especially the recommendation for requiring most of our group to attend the SANS 401 course and attain their GSEC certification.
- An incident handling team should be created within GIAC. On the face of it, this does not seem to carry a cost, but in reality we desperately need training in a methodology for properly handling security incidents. For this reason we've recommended that at least the Incident Handling Team Leaders take the SANS SEC504 course and attain their GCIH certification.
- Network Admission Control (NAC) should be considered, as it will help us enforce our Security Policies before workstations actually attach to our network. However, the hefty price-tag means we should evaluate the costs against the benefits from a business perspective and make the decision then. The NAC section of this paper covers the benefits at a high level, and includes a budget cost. Proceeding in this direction is really a decision for the GIAC CIO.
- Implementing an Intrusion Detection or Intrusion Prevention (IDS / IPS) or Data Leakage Prevention (DLP) systems also involve significant costs. The initial costs of the solution may be high, but the personnel required to keep these systems running once in place would involve hiring at least one new tier 3 person into the IT group.

In summary, while we at GIAC did not suffer a security breach, the investigation into this incident has brought to light several measures we should consider to prevent future occurrences. As several of them are no-cost or low-cost options, we hope they can be considered for implementation.

Downadup (Conficker) Overview

In November 2008, there was a discovery of a new worm that took advantage of a vulnerability in the Windows Server service (see MS08-067). This worm was named Downadup by some Antivirus vendors, but is most commonly now known as Conficker. Conficker was defined by Microsoft as "a worm that infects other computers across a network by exploiting a vulnerability in the Windows Server service (Microsoft, 2009). In the beginning, upon successful exploitation of the vulnerability in the Server Service, the worm had the ability to spread and remotely execute any arbitrary code. According to a technical report from SRI International (Porras,Saidi, Vinod Yegneswaran, 2009), their honeynet was overwhelmed by Conficker in the early stages. From late Nov 2008 to Dec 2008 SRI International experienced 13,000 Conficker infections from over 1.5 million IP addresses in 206 countries. From SRI International's experience, they reported not having seen such a dominating outbreak since Sasser outbreak in 2004, and such poor AV detection of binary variants since Storm worm outbreak in 2007.

As time progressed into early 2009, several variants of Conficker started emerging and new infection vectors were discovered. Conficker was slowly growing and maturing. As this evolution occurred, we saw it use various attack methods to propagate.

At first, it only exploited the Microsoft Server Service vulnerability to spread. This was very successful due to the number of machines that were left unpatched for a long time. As time went on and Conficker was updated, the variants accumulated more methods of propagation. These propagation methods include use of mapped drives, removable media, shares with weak passwords, and the task scheduler.

Not only did Conficker mature with propagation methods, its payload also matured. The payload went from the ability to generate many update URLs daily and resetting the restore point to additional "features" such as: modifying system settings, disabling security software, blocking security sites, peer-to-peer updating and authenticity and validity checking of downloads.

The following chart, from Microsoft, shows more detail of the propagation methods and payloads by variant.

Variant	Spreads Via...	Payload	Additional Information
Worm:Win32/Conficker.A **Discovered Date:** 21st Nov 2008 **Payload Trigger Date:** 25 Nov 2008 and later	-Exploiting the vulnerability outlined in Security Bulletin MS08-067.	-Generates 250 URLs daily that it checks for updates -Resets System Restore Point	The name of this family was derived by selecting fragments from 'trafficconverter.biz', a string found in this variant.
Worm:Win32/Conficker.B **Discovered Date:** 29th Dec 2008 **Payload Trigger Date:** 1 January 2009 and later	In addition to the method used by the .A variant (above): -Network shares with weak passwords -Mapped and Removable drives -Uses a scheduled task to execute copies of the worm on targeted machines	In addition to the .A variant's Payload (above - although .B uses a different method to generate URLs): - Blocks access to many security-related websites -Modifies system settings -Terminates system and security services	This variant built on the functionality of the .A variant by adding new spreading mechanisms and by making itself more difficult to remove.
Worm:Win32/Conficker.C **Discovered Date:** 20th Feb 2009 **Payload Trigger Date:** 1 January 2009 and later	Uses the same methods listed above for the .B variant.	In addition to the Payloads listed above for .A and .B: - Uses additional method for downloading files that utilizes Peer-to-Peer communications - Adds checks to verify the authenticity/validity of content targeted for download	Very similar to the .B variant in function (this variant has even been referred to as variant .B++).
Worm:Win32/Conficker.D **Discovered Date:** 4th Mar 2009 **Payload Trigger Date:** 1st April 2009 and later	No spreading functionality per se. Distributed as an update to machines previously infected with the .B and .C variants.	In addition to the Payloads listed above for .A and .B, with some variations: - Generates 50,000 URLs to download files from. This variant only visits 500 of the generated URLs within a 24-hour period. - Expands on efforts to hinder its removal from an affected machine.	Spreading functionality was removed from this variant. It continues to expand on its file downloading payload and targets a broader range of processes to terminate (appears to be targeting cleaning utilities designed specifically to remove Conficker). It also blocks access to additional security-related websites.
Worm:Win32/Conficker.E **Discovered Date:** 8th Apr 2009 **Payload Trigger Date:** 1st April 2009 and later	No spreading functionality per se. Utilized to update machines previously infected with the .B and .C and .D variants.	- Blocks access to many security-related websites -Modifies system settings -Terminates system and security services -Terminates itself on May 3	May have been distributed via the Conficker peer-to-peer network.

(Source: http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Win32%2fConficker)

Malware Function

Even today, no one really knows the intended purpose of Conficker. The payloads used during the outbreak were just there to allow it to spread, mature and survive. It appears that Conficker's purpose was to build a huge BotNet of hosts that could be controlled for any intended purpose. In later stages, adding a peer-to-peer architecture in addition to its initial client/server architecture, it became much more resilient in updating and propagating itself. This leaves us to question, was Conficker just a research project to gain strength and knowledge in these areas for something yet to come? Were the creators wanting to see how their design would hold up under widespread analysis and prevention so they could refine the worm to avoid detection and eradication? Or possibly, a tactic for diversion or resource expenditure? Even without a mass mailer or DDoS payload, Conficker caused quite a bit of disruption and effort in analysis. Or did Conficker's profile just become too large in the security community and popular media to be useful as a rental botnet? Even if the original intent was to sell generic botnet services, with an aggregate economic cost of over $9 billion (Cyber Secure Institute, 2008), Conficker had become so large in the media that trying to actually sell a Conficker based botnet would attract immediate attention from law enforcement.

Conficker At GIAC: Our Incident

Now that we know more about Conficker, let's discuss the email incident at GIAC that prompted the formation of this tiger team.

Without more investigation, it's not possible to determine exactly the root cause of the recent Conficker incident that has raised the awareness about malware within GIAC. There are a few possibilities:
The CIO's workstation might in fact have been infected. This is not likely, as we have since checked and scanned that computer for malware using our AV application, and that computer also has all current operating system patches applied.

What is more likely is that what we are seeing is a result of some other host that is infected, not necessarily within GIAC. Conficker.C and E installs a companion virus, called Waledac, which then sends emails to infect other computers. It's likely that this is what we are seeing. An infected host may have sent such an email to our CIO's peer, with a reply-to address at GIAC. It was detected correctly as a Downadup/Conficker infection, but the source was not diagnosed correctly. A review of their email server logs will most likely show that the host that sent this email was not mail.giac.org

Detecting Conficker Vulnerable Machines

NMAP

NMAP is a free tool that is traditionally used as a port scanner, but has been expanded in recent years to include other functions via a generic scripting engine. NSE (NMAP Scripting Engine) scripts are based on LUA (a general purpose, network aware programming language).

Since version 4.85 Beta 5, Conficker detection has been a part of NMAP (Insecure.org, 2009). This was implemented as part of a larger script set, "smb-check-vulns". To scan a network for Conficker, or Conficker-vulnerable hosts, use the following syntax:

```
nmap -PN -T4 -p139,445 -n -v --script=smb-check-vulns --script-args safe=1
[targetnetworks]
```

At this time, this scan will check for several different vulnerabilities, Microsoft MS08-067, a Windows RPC vulnerability, and infection by the Conficker (or Downadup) virus. (Bowes, 2009)

The NMAP script was based on another set of scripts developed at universitätbonn, with a set of papers and scripts published by Felix Leder and Tillmann Werner. (Leder, Werner, 2009)

WMIC based Scripts

Another approach to scanning hosts for vulnerabilities is to use WMIC (Windows Management Instrumentation Console) to scan for the application of the Microsoft patch KB958644. This one-line script for instance will scan the pc running the script for that patch:

```
wmic qfe where hotfixid="KB958644" list full
```

(Skoudis, 2009)

To scan remote station, let's create a short script file called dauscanstation.cmd:

Dauscanstation.cmd:

```
ping -n 2 %1 | find "Reply" >nul
if not errorlevel==1 (
wmic qfe where hotfixid="KB958644" list full /node:%1
/user:<domain\administrator >/password:<domainadminpassword>
) else (
Echo station %1 is offline
)
```

Now we can use this dauscanstation.cmd file to scan an ip range on our network:

Dausubnet.cmd:

```
for /L %%i IN (1,1,254) do call scanstation 192.168.10.%%i >>dausubnet.out
```

We can expand on this to scan a complete Active Directory Windows domain:

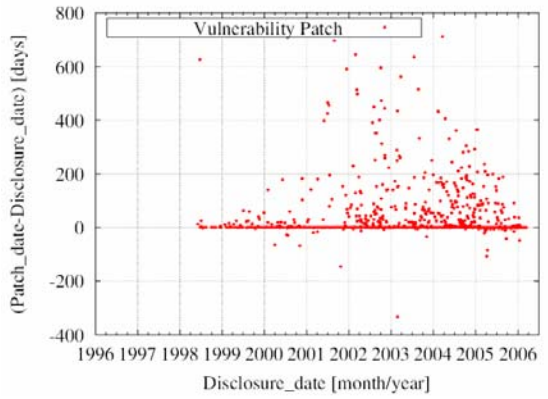
Daudomain.cmd

```
dsquery computer –limit 1000 | dsget computer -samid > dsquery.out
for /f "delims =  $" %%i in dsquery.out do dauscanstation.cmd %%i
>>daudomain.out
```

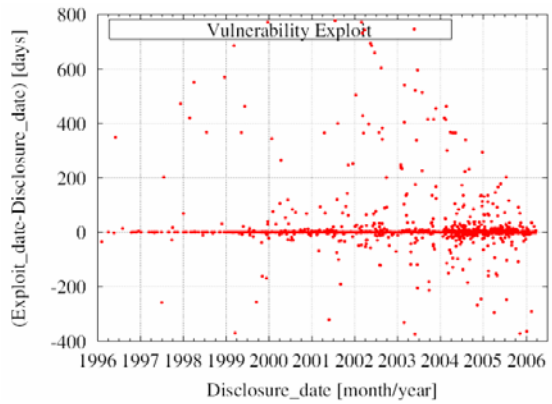
Future Zero Day attacks

Zero day attacks are listed among the SANS Institute's Top 20 Internet Security Problems, Threats and Risks. According to the SANS Institute, "a zero day vulnerability occurs when a flaw in software code has been discovered and exploits of the flaw appear before a fix or patch is available." (SANS Institute, 2009). Zero day attacks are very dangerous because they come with little to no warning, leaving many systems open for compromise while detection and awareness takes place, work-arounds are analyzed and implemented, and/or patches are developed and deployed.

In a recent survey of Microsoft and Apple patches, it was discovered that 31% of all patches were disclosed before a patch was available. In contrast, 81% of all vulnerabilities had exploits available at or before disclosure date. In the area of patching and exploits, the bad guys are clearly in the lead. However, that's no reason to give up or make it too easy for them.



Patch Availability:
1550 Patches
15% patched before disclosure
54% patched at disclosure (on Day 0)
31% patched after disclosure



Exploit Availability:
3428 exploits
23% available before disclosure (true zero days)
58% available at disclosure
19% available after disclosure

Patch Availability and Exploit Availability compared to Disclosure Date (Frei, Tellenbach, Plattner, 2009)

Zero Day attacks often use vulnerabilities in software that is used on a wide-scale basis. Applications such as Microsoft Windows, Microsoft Office and Adobe Reader or Flash are a few examples of the software packages that exploit developers are focusing on. With the widespread deployment of firewalls, attackers are looking at methods to deploy their exploits via social engineering due to lack of or poor user education in the majority of companies. Today's attackers are very good at convincing users to open attachments or click on links that will start the infection process unknowingly to the user. There are various methods employed, including but not limited to:

- Fake Antivirus sites.
- E-Mail links using popular news info as a front (hurricane relief and other disaster donations, swine flu education, the fake “Low Income Healthcare Enrollment” spam attack after Obama’s recent healthcare speech, etc).
- Man in the Middle attacks (invalid, fake, or forged certificates).
- Casual web surfing (browsing to a legitimate site that has been compromised or has malicious software behind the advertisements).
- Use of social media sites (Twitter, Facebook, MySpace, etc).
- Use of auction/sales sites with malicious content added to them (CraigsList, eBay, etc).

These zero day attacks will likely target data which includes proprietary information, personally identifiable information, credit card or other financial information, and any type credentials available. Attackers will tend to look for types of information that is valuable. Information that the can sell, use for fraud or identity theft, or just a simple transfer of funds. The fact that people often use the same credentials for different access classifications- ie. Same username and password for iGoogle and Online banking, is often a factor that is used in modern attacks.

Attack vectors for zero day attacks will greatly remain the same because of human habits. Spam, phishing, getting users to click on malicious links or press “OK” on invalid certificates are all attack methods that have been with us for 10 years and will probably still be with us 10 years from now. The high level approach of how attacks work have remained surprisingly constant over time. While the technical *details* of course have evolved with new technology, the presentation layer and underlying psychological methods of getting people to ”hack themselves” has improved exponentially.

A good illustration of “the more things change, the more they stay the same” is IP Version 6. The protocol is new in many environments, so new in fact that many firewalls and IPS solutions do not recognize it at all. In addition, IPv6 can be tunneled either through the teredo protocol via OSATAP (Intra-Site Automatic Addressing Protocol) or inside another IPv4. Recent tools that exploit IPv6 include relay6, 6tunnel, nt6tunnel and asybo, However, when viewed at a high level, as with a protocol such as HTTP, these tools simply take attacks that were common in the IPv4 world years ago, tools that tunnel information within other protocols such as HTTP, DNS or GRE, and port them directly to IPv6 using that tunnel encapsulation method. In addition, vendors are re-introducing old vulnerabilities into the new IPv6 stack. Microsoft for instance re-introduced the LAND attack into an early implementation of Windows Server 2003.

Detecting New Zero Days

Since antivirus applications cannot be counted on to detect new "zero day" malware, or in fact most new malware, other methods should be considered to detect new malware that may be operating inside the corporation. To this end, we'll discuss two methods of detecting malware – finding new processes in the domain, and finding new services with listening ports in the domain.

Finding New Processes In the Domain:

This script expands on our previous dauscanstation script, to scan a host for running processes. This output can easily reach over 100, so for a domain of 100 workstations, a list of 10,000 processes can be expected. In order to make this useful, we need a method of getting a "difference report" from one run of the script to another. The basic script uses WMI to list all running processes on a station, using the command:

```
wmic process list brief
```

The output is a bit "wordy" and difficult to read, but it has the information that we're looking for:

```
HandleCount  Name                        Priority  ProcessId  ThreadCount  Wor
kingSetSize
0            System Idle Process         0         0          2            286
72
2287         System                      8         4          86           262
144
26           smss.exe                    11        1564       4            450
560
968          csrss.exe                   13        1708       13           784
3840
696          winlogon.exe                13        1760       21           405
0944
398          services.exe                9         1804       16           383
3856
416          lsass.exe                   9         1816       21           152
7808
223          svchost.exe                 8         1996       19           559
1040
```

Our goal however, is to scan stations remotely. For this we'll modify the script to read:

```
 wmic process list brief /node:%1 /user:<domain\administrator>
/password:<domainadminpassword>
```

However, when combined with the script output from other stations, it can be difficult to parse out changes in the environment from one day to the next. For this reason, we'll need to strip out information in this output that will change from day to day (process number for instance). Also, we'll need to treat our text file more like a database, and include the station name next to the process name. With this approach, we can then use a standard tool such as diff (or windiff) to extract any new processes that have cropped up in our environment between one run of the scan and the next.

```
Wmic process list brief /node:%1 /user:<domain\administrator>
/password:<domainadminpassword> | /format:csv | cut -d, -f 1,3
```

Our output is now stripped down, and includes on the information that is required:

```
RVLTOP System Idle Process
RVLTOP System
RVLTOP smss.exe
RVLTOP csrss.exe
RVLTOP winlogon.exe
RVLTOP services.exe
RVLTOP lsass.exe
RVLTOP svchost.exe
RVLTOP svchost.exe
RVLTOP svchost.exe
```

Finally, we'll want to combine the output of all of our scans in order to make finding new processes and patterns that much easier. To that end, we'll use variants of our original "loop" scripts to call the final "scanprocesses.cmd" file, the final scripts look like this:

Processes_in_a_subnet.cmd

```
@echo off

for /L %%i IN (1,1,254) do call scanprocesses 192.168.10.%%i
>>procs_subnet.out
```

processes_in_the_domain.cmd:

```
@echo off

dsquery computer –limit 1000 | dsget computer -samid > dsquery.out

for /f "delims =  $" %%i in (dsquery.out) do scanprocesses.cmd %%i
>>procs_domain.out
```

Scanprocesses.cmd:

```
@echo off
```

```
ping -n 2 %1 | find "Reply" >nul

if not errorlevel==1 (

Wmic process list brief /node:%1 /user:<domain\administrator>
/password:<domainadminpassword> | /format:csv | cut -d, -f 1,3

) else (

Echo station %1 is offline

)
```

Now, to find new processes, we can use diff. for instance, the following diff command runs a report between the first and 14th of the same month:

Diff 09012009\procs_domain.out 09132009\procs_domain.out

```
96a96,98
> RVLTOP YA.exe
> RVLTOP wave.exe
> RVLTOP wmic.exe
```

We see that the station "RVLTOP" has 3 new processes running. Upon further investigation, it turns out that these processes represent the client starting a VOIP soft phone application, which is in general use within the corporation. Introducing an audit process such as this will result in a fair number of false positives. To reduce this, we might include a list of "known good applications" to exclude from our report. Similarly, a list of "known bad applications" can also be made to ensure that these are also dealt with in a timely fashion. This can be done either at collection time (in scanprocesses.cmd) or at report time (in the diff report). These might have the form:

Goodapps.txt	Badapps.txt
svchost.exe vpnagent.exe Rtvscan.exe TimeZone.exe YA.exe	Skype.exe Pinball.exe

For blacklisted processes, we'll want notification as soon as the scan completes, so should be integrated into the scan_domain_processes.cmd file – this might take the form:

```
@echo off

dsquery computer –limit 1000 | dsget computer -samid > dsquery.out

for /F "delims =  $" %%i in (dsquery.out) do scanprocesses.cmd %%i
>>procs_domain.out

for /F %%I in (badapps.txt) do type procs_domain.txt | find "%%I"
>>bad_apps_and_stations_to_investigate.txt
```

Filtering out whitelisted applications is essentially in issue in interpreting reports – this is best done during the diff report process. This might take the form:

Diffrpt.cmd:

```
Diff 09012009\procs_domain.out 09132009\procs_domain.out >diff1.out
For /F %%i in goodapps.txt do call diffloop
Copy diff1.out diff.txt
Del diff1.out
Del diff2.out
```

Diffloop.cmd

```
type diff1.out | find /v "%%1"  >diff2.out
copy /Y diff2.out diff1.out
```

Finding New Services with Listening Ports

In almost an identical manner, the network can be scanned periodically for new network services (ie open ports) on domain and non-domain workstations on the network. These cans could be done with free tools, such as NMAP, the reports could be saved in text format and reported on using diff scripts similar to those outlined above for windows processes running across the domain.

Scanstationports.cmd

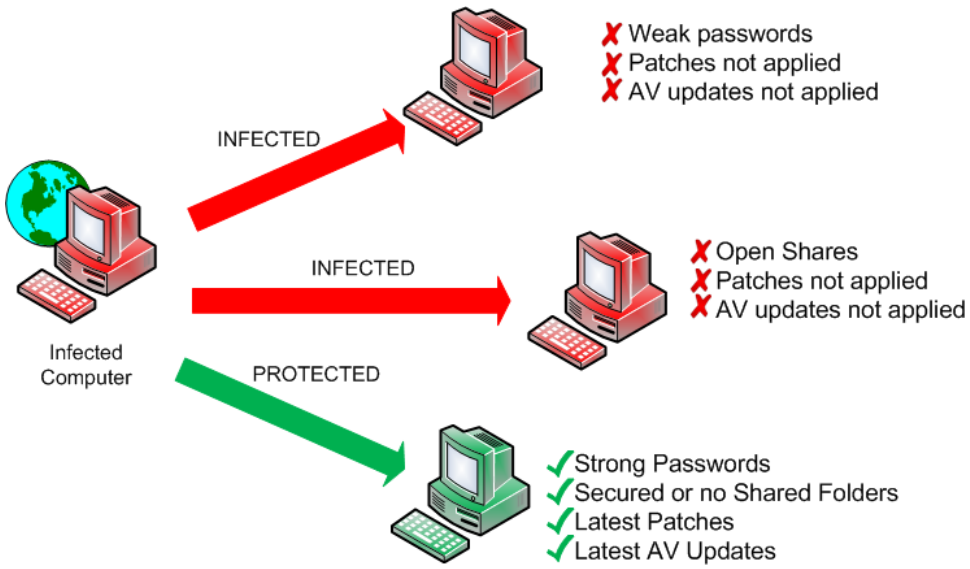
```
Nmap –sT  %1 | find "/" | find /v "Starting" | sed s/$/,%1/
```

This parses out the nmap output for a single station to include only the port information, and to append each line with the workstation name. This permits us to call scanstationports.cmd in exactly the same way as scan_domain_processes calls scanprocesses, complete with whitelisting / blacklisting of ports and diff reports.

Prevention of Zero Day Infections

Conficker was widespread, caused significant damage to businesses and took significant effort on clean up. However, the prevention for this virus was simple – Apply patches in a timely manner, ensure that complex passwords are in use, install Antivirus updates in a timely manner, and configure machines with secured shares or no shares. These are measures that every IT department should be implementing as part of regular business. Password complexity and open shares would generally be part of an ongoing "Security Awareness" training program.

In short, prevention of Zero Day infections at GIAC should be spearheaded by Security Awareness Training, communication and enforcement of a corporate Password Policy, timely patches, and timely AV updates.



Security Awareness

Security Awareness training is the most critical area we should focus on to reduce the malware infection rate at GIAC Enterprises. Training can be done in dedicated sessions, or in more informal settings. For instance, attending SANS SEC351 (which we are certified to present) would be a good option for our new-hire orientation program. Alternatively, we could create our own training course based on NIST SP 800-50, "Building an Information Technology Security Awareness and Training Program.". For a more immediate benefit, however, we need to target our existing employees, and a mandatory 3 hour class is probably not an effective way to do this quickly. For existing employees, we could:

- offer a series of “Lunch-and-learns”,
- produce a series of posters on security topics
- write a series of “splash pages” that users must read before they can login to the network.
- Have a monthly contest based on the information presented that month, with a sizable prize (something like an iPod might be appropriate).

In all of these things, the approach should be to keep it fun. Creating a comical character that always gets things wrong might be an approach, or presenting the monthly contest as a scavenger hunt with the prize at the end is always fun.

Strong authentication

Malware, and Conficker in particular, will take advantage of open shares and blank, default and simple passwords. Conficker in particular in some variants had a process to brute-force passwords. For this reason it’s important that as a corporation, GIAC writes a Password Policy outlining clear policies on password strength and frequency of password changes, with non-technical guidance for our users in plain English. We then need to communicate these policies to our user community, and follow this communication up with periodic audits and assessments of domain and local credentials. Our password strength policy should then be implemented as a Group Policy within Active Directory, to ensure that our users comply going forward. As we ramp up this policy, frequent might be advisable. Once the policy is in place, tapering down to monthly or bi-monthly assessments should be sufficient.

Two factor authentication, such as RSA keys or Phonefactor (a low cost 2-factor system that uses telephone numbers) might be an option at GIAC. These have the advantage of ensuring that However, given the mandate to contain the costs of security improvements, this may not be a viable option at this time.

Patching

The importance of keeping systems up to date on security patches cannot be stressed enough. Even though many exploits to vulnerabilities are available prior to vendor patches, it's still important to get updates in place as soon as possible in order to protect us from the malware that comes out after. And just because an exploit is available does not necessarily mean that the malware community is taking advantage of it. Remember, malware authors treat this as a business, so using kits to assemble new mal-code is much more prevalent than incorporating new exploits from scratch into code. Finally, not keeping systems up to date is analogous to leaving your car door unlocked. Locking your car door is not a real obstacle to a determined car thief, but if a door is locked, the thief is much more likely to move on to find an easier target.

WSUS is a free Microsoft approach to patching it's operating systems and applications, and could be used effectively at GIAC. If, after study, we decide that other business applications need to be patched in a similar way, we can expand our WSUS deployment with custom scripting, or we could migrate to a commercial product such as Altiris Total Management Suite or GFI Languard.

Antivirus and Malware Applications

The risk in Antivirus applications is that the signature based approach that all of the AV vendors use is by definition always behind the curve. When new malware comes out, the AV vendors first need a sample of the malware, then need to code a signature and detection method for it. This generally means and 8-24 hour lag on signatures as compared to release times of mal-code.

For this reason, a recent trend has been short-lived malware. Of the 37,000 new samples of viruses, worms and trojans that anti-virus firm Panda Security receives daily, 52 percent spread for just 24 hours. Nineteen percent last for two days, and nine percent persist for three days. This trend makes it extremely difficult for a signature based approach to "keep up" with current malware.

Again, however, this is no reason to give up. A reasonable approach for GIAC to take is to layer our malware defenses. We might take a 3 vendor approach, with different AV solutions for our mail filtering, web filtering and workstation AV engines. This will however mean that we'll also have 3 different management consoles and 3 different reporting solutions for malware.

IT Security Awareness

Something that has come to light over the recent incident at GIAC is the lack of Security Awareness within the IT group. We would like to recommend an ongoing discussion in this area. Some things we'd like to see might be:

- IT members doing regular research projects

- IT members regularly visiting security sites, such as isc.sans.org, pauldotcom.com, digg,.com/security and other security sites and blogs.
- Our IT group has an requirement for ongoing training in the security space. It's our recommendation that all members of our team attend at least one security course or major event per year. If a certification is associated with the event or class, in most cases it should be attempted. In many cases attendance of SANS SEC401 along with the associated GSEC certification should be considered a requirement.

All of these steps in combination should raise the overall awareness of our IT team on security matters. This means that we'll be more in tune with things that will affect GIACs ongoing security requirements, such as new attacks and defenses, new compliance requirements and concerns, and events and data breaches in other companies similar to GIAC.

NAC / NAP

NAC (Network Admission Control), or the Microsoft NAP (Network Access Protection) can both be used to enforce a defined security policy on our entire workstation population, including visitors. Both approaches act as GUI interfaces to the 802.1x protocol implemented on modern switches. They both work in this manner:

- A workstation powers on, and it's switch port sees them come online
- The switch puts them into an assessment vlan, where their antivirus and patching posture is assessed
- If they are a visitor, they might be confined to a visitor VLAN with only internet access. They may or may not be assessed.

If an assessed machine does not pass for any reason, the user is notified, then it is updated by the NAC server before it is allowed on the corporate network.

A few things should be identified before a NAC solution is identified as a requirement for GIAC:

- All switches must be 802.1x compliant. If possible, they should be all from one vendor to make things simpler to manage. This means that all of the unmanaged switches in our environment must be upgraded and then discarded. If a station on an unmanaged switch is put into the "jail" or remediation VLAN, all other stations on that switch will go with it and be isolated from their business applications.
- NAC goes through a similar function when a user VPNs in, or connects wirelessly.
- The base cost of a NAC solution for GIAC will be roughly $15,000 (based on our 200 seat desktop community), separate from any switch upgrades that may be required.
- When selecting our NAC solution, we should ensure that it will work with our existing desktop antivirus solution.

Because of the costs involved, it's recommended that this solution be evaluated for inclusion as a project in next year's budget.

IPS and Data Leakage Prevention:

On the face of things, Intrusion Prevention Systems (IPS) and Data Leakage Prevention (DLP) solutions look like they might be very similar. Both need to "sniff" every packet as it leaves, the network, and both need a significant investment in installation, tuning and ongoing maintenance. However, these the two solutions do differ significantly.

Data leakage prevention, at least in the context of the network data leakage we'll be concerned about from malware, is all about categorizing and recognizing data on the wire. In our case, we would be categorizing our customer information, and of course our Fortune Cookie Sayings, our main intellectual property. In both cases, our DLP system would need to know where this data was *not* allowed to travel. If any of this data is detected in an unauthorized path (outbound through the firewall in an email or tunnel for instance), the data stream would be stopped with a spoofed FIN or RST packet, and then an alarm would be raised and sent as an alert to the Incident Handling team.

Intrusion Prevention collects all data on the network, generally at significant points (DMZs, ingress and egress points for example), then reassembles the collected data into complete streams. This is then compared to the behavior or signatures of known attacks. If an attack is detected, it can either be treated as an alarm and/or alert, or the IPS can block the datastream. Similar to Data Leakage, it is important to tune IPS systems to reduce false positives, so that legitimate traffic will not generate false alarms or be blocked. For instance, accessing normal Microsoft shares, logging on to Outlook Web Access, and receiving an update from a Symantec Antivirus server will all generate alerts and may be blocked on an unturned IPS system. Alerts from an IPS system would be handled by an Incident Handling team, similar to the DLP solution.

When considering IPS or DLP solutions at GIAC, cost will become a significant factor. In both cases, we'll need a significant investment in hardware and probably software as well (although an open-source version of SNORT may be considered for IPS). However, the true cost will be in the ongoing system administration, reporting and tuning required by both systems. In both cases, it is estimated that a half person-year should be budgeted to maintain either system. If both are considered, this is an additive metric, a full person-year should be budgeted.

Incident Handling Team

Something that we allude to in both solution descriptions is an Incident Handling (IH) team, something which does not currently exist at GIAC. Creation of an Incident Handling team will also carry some costs, not so much in dedicated resources (though during an incident all or part of the IH team will be seconded from their regular

duties), but in initial and ongoing training. If we elect to create an Incident Handling team, it's important that we not only train our team members and define a methodology for handling incidents (SANS Security 504 would be an ideal choice for both), but will need to participate in regular Incident Handling exercises and ongoing training.

Because of the costs associated with Intrusion Prevention or Data Leakage Protection, if GIAC elects to go forward with either, it is recommended that the initial and total ongoing costs be more accurately defined, and these costs should be factored into the IT budget going forward.

References:

Microsoft Corporation, (2009). Malware Protection Center: Win32\Conficker. Retrieved from http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Win32%2FConficker on September 14, 2009

Phillip Porras, Hassen Saidi, and Vinod Yegneswaran, (2009). An Analysis of Conficker's Logic and Rendezvous Points. *SRI International Technical Report*, Retrieved from http://mtc.sri.com/Conficker on September 14, 2009

Cyber Secure Institute, (2008). Cyber Secure Institute on the Conficker Controversy. Retrieved from http://cybersecureinstitute.org/blog/?p=15 on September 14,2009

Insecure.org, (2009). Nmap Development: Nmap 4.85BETA5: Now with Conficker detection!. Retrieved from http://seclists.org/nmap-dev/2009/q1/0870.html on September 14,2009

Ron Bowes, (n.d.). Script smb-check-vulns.nse. *NMap.org*, Retrieved from http://nmap.org/nsedoc/scripts/smb-check-vulns.html on September 14, 2009

Felix Leder, Tillmann Werner, (2009). Containing Conficker. Retrieved from http://iv.cs.uni-bonn.de/wg/cs/applications/containing-conficker on September 14, 2009

Skoudis, EdEp (2009). Episode #16: Got That Patch?. Retrieved from http://blog.commandlinekungfu.com/2009/03/episode-16-got-that-patch.html on September 14,2009

SANS Institute, (2009). Top 20 Internet Security Problems, Threats and Risks. Retrieved from http://www.sans.org/top20/ on September 14, 2009

Stefan Frei, Bernhard Tellenbach, and Bernhard Plattner, (2009). 0-Day Patch Exposing Vendors (In)security Performance. Retrieved from http://www.blackhat.com/presentations/bh-europe-08/Frei/Whitepaper/bh-eu-08-frei-WP.pdf on September 14, 2009