Final Penetration Test Report

Madeline Farina

3 December 2019

Contents

Executive Summary

The following document is a complete report on the vulnerability research and penetration testing conducted on the network of M3g4c0rp Mega Crops company. It was performed by representatives of testing firm Pr0b3.com. Due permission was granted by Matt Mason, M3g4c0rp Director of Engineering, to conduct the following research in order to asses the risk of exploitation by malicious parties, all of which was outlined in a Scope of Work document. Risks along with verification of their presence, tools used to discover said risks, and remediation tactics to mitigate them has all been provided below.

Background

These tests were carried out on a Kali virtual machine bearing the IP address 172.24.0.10. While the exact number of external and internal IP addresses to be tested was unknown, it was stated in the Scope of Work all machines in range 172.30.0.0/24 (excluding 172.30.0.1) are in the scope of the M3g4c0rp domain and are thus available for testing. Furthermore, there is a private IP block accessible from innerouter.m3g4c0rp.com with IP address block 10.30.0.0/24. All hosts there are also in scope. 172.30.0.1 is not in scope due to being an upstream interface assigned to a network service provider other than M3g4c0rp.

The complete list of hosts to be tested with their corresponding Operating Systems and IP addresses are as follows, found using the **CeWL** wordlist creator and **fierce** domain scanner like so: **cewl http://www.m3g4c0rp.com** > **wordlist.txt** then **fierce -dns m3g4c0rp.com -wordlist wordlist.txt**:

1. **www, ns, pop** : Debian 10 : 172.30.0.128
2. **innerouter** : pfsense 2.4 : 172.30.0.3
3. **herd** : Windows 7 or 8 : 10.30.0.98
4. **patronum** : Windows 8 or 10 : 10.30.0.97
5. **pdc** : Windows Server 2012 or 2016 : 10.30.0.90
6. **devbox** : Debian : 10.30.0.32
7. **linuxserver** : Debian 10 : 10.30.0.128

At one point a social engineering scheme was set up against employee Phineas Philatelist at M3g4c0rp by providing a phishing URL for him to click. Phineas was told the web site at kali.pr0b3.com (172.24.0.10) was under construction, and so he would actively attempt to contact the supposed web site. No other information was provided. This allowed credentials to be captured from Phineas' browser using BeEF, discussed below.

General Findings

M3g4c0rp's system proved to be greatly vulnerable to attack, mostly due to the unwise actions of Brian Oppenheimer, the son of the company's system administrator. Generally, his configurations left much to be desired and caused many vulnerabilities to be exploited. Furthermore, it was noted a lot of exploits were possible because of weak passwords and default configurations in need of an immediate update.

Technical Report

Information Gathering

Passive Intelligence

Passive intelligence was used in the form of running Wireshark on the different domains to notice any interesting network activity. TCP packets were sent between 172.24.0.10 (the Kali vm) and the source, like 172.30.0.128.

The OpenVAS vulnerability scanner (a web-based GUI for scanning targets) was used to scan a number of hosts. The scan took 50 minutes and provided a list of 330 vulnerabilities identified, including Unencrypted Cleartext Login (running on port 512), OS End of Life Detection, VNC Brute Force Login (port 5900), and a possible backdoor titled Ingreslock (port 1524). The vsFTPd vulnerability was running on both ports 21 and 6200. The metasploit module associated with vsFTPd was **exploit/unix/ftp/vsftpd_234_backdoor** which could be used in the metasploit framework (discussed under Active Intelligence).

Passive intelligence was also used during an initial OpSec leak of information by the client which allowed the NTLM password hashes of users on the 10.30.0.98 machine to be gathered. These were later used to easily crack the passwords of said users.

Active Intelligence

After running an nmap TCP version scan on www.m3g4c0rp.com with the command **nmap -sV -O www.m3g4c0rp.com**, it was revealed this server had ports 22, 53, 80, and 8080 open, meaning the ssh, domain (ISC BIND 9.11), http (Apache httpd server), and tcpwrapped services were running on each port, respectively. The last port was something of interest, and upon further research it was discovered port 8080 was running nginx, and opening 172.30.0.128:8080 in a Firefox web browser led to a web page for a nginx web server. Some more research was conducted to find a number of vulnerabilities associated with the specific version of the nginx web server, the one with the highest score of potential risk being CVE-2013-0337. Other vulnerabilities were found for ISC BIND 9.11, mostly of low risk except for CVE-2017-3141 which had the highest risk score, and the apache httpd, the highest being CVE-2019-0217.

Metasploit is an open-source framework used for exploiting security vulnerabilities on host machines, opened in a command window with the command **msfconsole**. For the vsFTPd vulnerability, the exploit "exploit/unix/ftp/vsftpd 234 backdoor" was used, which was mentioned in the OpenVAS scan. After showing the payloads, it was revealed "cmd/unix/interact" as the only payload, so this payload was set as that, the RHOST as 172.24.0.42 (the target), and the LHOST as 172.24.0.10 (the Kali machine). A meterpreter session was started simply by using the **exploit** command, and this allowed shell access as root because of how the backdoor worked.

To exploit for the Tomcat vulnerability, a metasploit session was started in the command window and all the exploits related to tomcat were searched for with grep exploit search tomcat. This revealed a lengthy list of exploits to experiment with. The exploit exploit/multi/http/tomcat_mgr_deploy was chosen by both logical inference and suggestion by the client. After showing options, we set HTTPPassword and HTTPUsername as "tomcat", then set RHOSTS as 172.24.0.42 (the target) and RPORT as 8180. Then, we tried to set the payload as linux/x86/meterpreter/reverse tcp, but this led to "exploit completed but no session was created" in metasploit. Trying the linux/x86/meterpreter/bind tcp payload got the same result. Finally, by trial and error, the payload java/meterpreter/reverse http allowed the exploit to start a meterpreter session. When trying to download the /etc/shadow file after starting a meterpreter session, this operation failed because our user was not given the sudo privileges to access this file, so we were unable to become root. This is because the /etc/shadow file contains passwords in encrypted format (password hashes) for user accounts, hence why one would need to be a root user to download it.

Fuzz testing was used to exploit a buffer vulnerable program written by Brian Oppenheimer which was intended to provide information about M3g4c0rp servers when run. By using fuzz testing (a software testing technique involving invalid or unexpected inputs to the computer program in order to get it to do something unintended), shell access was eventually gained on 172.30.0.128. This was done by first running a nmap scan which showed port 1337 was open, running an unrecognized service described as "waste?". Netcat was used to connect to the specified service with **nc www.m3g4c0rp.com 1337** in a command window. This prompted for an admin username (after a few guesses, "brian" was the username that gained access). After login, three simple commands were provided, like "ip a", which brought to mind the question of whether it was possible to change these commands via buffer overflow. Since the admin username was specified to be below 15 characters, experimentation was conducted by inputting a large number of characters (between 16 and 80), and then typing "brian" as the username in the second attempt. This led to the discovery that the first 32 characters went to the buffer and the rest were overflow. So to have a command we wanted (one that would give us shell access), we inputted first 32 random characters, then followed with the command we wanted (/bin/sh) before using "brian" as the username. This led to /bin/sh being the first available command to use, which allowed shell access when used.

The source code was found in Brian's directory in /home, in his src directory. The source code file itself was named "toool.c". Analyzing it revealed the error which allowed exploitation via buffer overflow.

Password cracking was used to crack the root password on 172.24.0.42 with the password-cracking tools John the Ripper and Hashcat. After using a metasploit exploit to gain root access to the target machine, the /etc/shadow and /etc/password file were obtained and copied over to the Kali vm via sftp. After preparing the shadow file with the **unshadow password file shadow file >jpasswdfile**,

gunzip /usr/share/wordlists/rockyou.txt.gz,

john -wordlist=/usr/share/wordlists/rockyou.txt jpasswdfile, and **john –show jpasswdfile** commands, John the Ripper was useful for cracking 5 of the 7 passwords. Using hashcat was attempted to crack the other 2, but it did not produce satisfying results.

First, access was gained on 172.240.0.42 via metasploit in order to obtain the /etc/shadow and /etc/passwd files. Once attained, they were copied over to the Kali vm via sftp for use by John the Ripper. I did this by **unshadow password file shadow file >jpasswdfile**. I needed to unzip the rockyou.txt file before I could use it as a wordlist, so I did this with

gunzip /usr/share/wordlists/rockyou.txt.gz. Then I ran

john -wordlist=/usr/share/wordlists/rockyou.txt jpasswdfile, and **john –show jpasswdfile** to show the password entries. This showed 3 password hashes cracked: "batman" for sys, "123456789" for klog, and "service" for service. Running john -wordlist=/usr/share/wordlists/rock.txt –format=md5crypt- long jpasswdfile cracked an additional two passwords: "postgres" for postgres, and "user" for user.

An infrastructure misconfiguration found in the PFSense Web GUI at https://172.30.0.3 was exploited on multiple occasions to gain access to target machines on the 10.30.0.0/24 network. This misconfiguration was that the default credentials for the admin user were not changed, so login access was easily gained, which allowed a connection to be made via remote desktop to machines like herd.m3g4c0rp.com and patronum.m3g4c0rp.com. This could be done with port forwarding: going to Firewall > NAT > Port Forward in the top menu, adding a new port forwarding configuration, setting the destination as the WAN address option, the MS RDP option for Destination Port Range, 10.30.0.98 for the Redirect target IP, and MS RDP option for Redirect Target Port. RDP was chosen because otherwise, the rdesktop command couldn't be used later on. For this port forwarding to take effect, all that needed to be done is saving the configuration and clicking the "apply changes" button. Then, running the command **rdesktop 172.30.0.3 -g 90%** in a command window on the Kali vm brought up the windows remote desktop screen. Three user accounts were shown: Bogus User, Sharon Shephard (the CEO of M3g4c0rp), and Local Admin. Because of the aforementioned OpSec leak, we were able to get the password for Sharon Shephard by typing her hash into a NTLM hash-cracking website. This revealed her password, which allowed login on the herd.m3g4c0rp.com machine.

To exploit herd, one must first connect via the remote desktop service (with port-forwarding still configured) while also sharing a directory containing the PowerSploit PowerUp.ps1 module (done with the command

rdeskstop -r disk:/usr/share/windows-resources/powersploit/Privesc 172.30.0.3 -g 90%. Once logged in, one should mount the share when running PowerShell with **net use** then set the executionPolicy to bypass by running **powershell -exec bypass**. Afterwards, the PowerUp.ps1 module should be imported with **Import-Module** \\TSCLIENT \\win32 \\PowerUp.ps1. The \\TSCLIENT \\win32 is the remote directory found in the output of ipconfig. Vulnerabilities can be searched for by running **Invoke-AllChecks** in Powershell which looked for unquoted service paths, writeable service executables unattended install files, hijackable .dll locations, and various registry and file system misconfigurations. For service permissions, it outputted that the service BITS in C: \\Windows \\System32 \\svchost.exe -k netsvcs, the AbuseFunction being Invoke-ServiceAbuse -ServiceName 'BITS'. Running **Invoke-ServiceAbuse -serviceName 'BITS'** created a new user with administrative privileges named "john" with the password "Password123!".

Patronum was also exploited, though it was crucial to set up port-forwarding to achieve access to it. To set up port-forwarding, one can't simply set up a new configuration in the PFSense web GUI; instead, one must use a Meterpreter session connected to Patronum. Once a remote desktop login for Patronum is accessed, a command window was pulled up even without being logged in, by hitting the shift key repeatedly. The command **powershell -exec bypass** was run to upgrade to a Powershell session, then **net user user password /add** to create new user. To add this user to the Administrators group, thus gaining admin privileges, run **net LocalGroup Administrators user /ADD**. At the login page, one can then login as PATRONUM
user. Then one can peruse other users' directories, where a file named mykeys.txt containing users and passwords was found in Matt Mason's user directory.

A recent Linux vulnerability was exploited on devbox.m3g4c0rp.com. Similar steps from above were conducted to gain ssh access: setting up port-forwarding through herd, using veil to generate a payload in Metasploit, using rdesktop to connect remotely to herd with a shared directory, and starting a meterpreter session. In the meterpreter session, **portfwd add -l 12345 -p 22 -r 10.30.0.32** was used to set up port-forwarding. Then on the Kali vm, one could connect to devbox via ssh with the credentials found in the mykeys.txt file mentioned above. Since the linux kernel version on devbox was found to be 4.19.0-5-amd64, it was likely vulnerable to a vulnerability involved exploitation using a specific line from the /etc/sudoers file. This file had the line "ALL=(ALL:!root) /bin/ps" which meant one could run the ps command as sudo by running **sudo -u-1 ps**. Normally, one would get an error message if trying to run a command as root without being in the sudoers file, but when this command was run, there was no error message, which indicated the vulnerability could be used for exploitation. By running cp /bin/bash ps, the ps command was effectively changed into a command which would open up a shell. So by running **sudo -u-1 ps** again, one could get a shell as root user.

As mentioned above, a social engineering scheme was set up in order to perform web exploitation to capture credentials or other useful information. BeEF is the Browser Exploitation Framework, useful for generating and delivering payloads directly to a browser and thus involves cross-side scripting (XSS). In this case, a Reflected XSS attack would be performed because of how the target would be clicking a URL which would contain attack scripts. The first task was to identify what web page or pages Phineas was attempting to grab. Based on the background information provided, an HTTP server would need to be provided for Phineas to ping in order to figure out this request. Wireshark was used to track any interesting activity aside from the packet transfers between 172.24.0.10 (the Kali vm) and 172.24.0.1 (the gateway). An HTTP server was also started running on port 80 in the command window with **python -m SimpleHTTPServer 80**. When interesting activity was noted in Wireshark from the IP address 172.30.0.128, the command window showed the URL containing the particular path Phineas was trying to access. After identifying this piece of information, the next task was to use BeEF to hook Phineas' browser and capture any useful information or credentials. Using the BeEF program from the Kali menu, one can specify a new password and log in as the user "beef" with said password in the web GUI. In order to use BeEF to hook the browser, the browser must execute hook.js on the BeEF Server and deliver hook via persistent or reflected XSS (in this case, reflected XSS). One can go to 172.24.0.10:3000/hook.js in the web browser and save the page for use on the Kali vm. Eventually, the target was able to be hooked and his IP appeared in the BeEF web GUI in the side menu of hooked targets. There one can peruse information about his browser session in the GUI.

Vulnerability Assessment

The following vulnerabilities have been sorted from highest to lowest risk, and remediation tactics have been included.

Vulnerabilities

PFSense The PFSense web GUI found at https://172.30.0.3 could easily be logged into since the default password for the "admin" user was never changed. This is an extremely risky misconfiguration, as it means anyone could log in and change the firewall rules to allow malicious access from an outside source.

remediation: This can be done with the User Manager in the web GUI. Using default usernames and passwords is a common misconfiguration which can easily be exploited, so it is important to always change default credentials when possible.

Brian's buffer vulnerable program: The vulnerability in this activity was due to the poorly written source code for the service running on the 172.30.0.128 machine. It allowed exploitation via buffer overflow for several reasons. The buffer was of fixed length, so later on when the function which got the admin user credential used this array, the extra characters in the string carried over from the buffer to the next frame of the stack. Also, it wasn't checking the length of the input for the admin name, nor was it running a check to see if the commands to be run after login were the same as the ones originally specified. This allowed infinitely long input as well as the attacker being able to use their own, specified commands.

remediation: For future reference, arrays specified as the buffer should not be of fixed length, or if so, functions should be written which check the length of the input. Although in my opinion, the program is so vulnerable I would recommend completely removing it from the server.

Phishing: The social engineering scheme showed the dangers of phishing scams, or more specifically, how clinking dangerous URLs can lead to reflected XSS attacks and acquisitions of sensitive data from the attacker.

(remediation: To avoid problems like this in the future, employees at M3g4c0rp should not click any suspicious URLs sent in emails, especially ones that direct them to sites ”under construction”.

Sticky Keys on Patronum: it was noted that if desktop access could be gained on the Patronum machine (either physically or remotely), it was possible to pull up a Powershell or command prompt window to create new users and add them to the Administrators group, even without logging in. This was because the shift sticky key was still set in place.

remediation: The shift sticky key configuration can be changed in the machine’s system preferences.

weak passwords: it was noted that practically every password cracked (like those 172.240.0.42) was typically either 10 characters or less, lowercase, cliche, containing dictionary words, or a combination of these traits, all of which make passwords much easier to crack by attackers. Also, it is crucial to not to reveal the hashes of one’s users’ passwords, as it is very easy to then crack these hashes and gain access to users’ accounts.

remediation: When user passwords are created, there should be a minimum length requirement (preferably, of 14 characters) and the password should include numbers, uppercase characters, and special characters; otherwise it is invalid. Also, it is crucial to not to reveal the hashes of one’s users’ passwords, as it is very easy to then crack these hashes and gain access to users’ accounts. Furthermore, passwords should never be stored as plaintext in any user’s directory on any machine.

Devbox’s sudo vulnerability: A root shell was able to be accessed on devbox even as a not-root user because of a vulnerability in the /etc/sudoers file. Non-sudo users should not be able to concatenate the /etc/sudoers file.

remediation: The permissions of the /etc/sudoers file should be updated so non-root users cannot read the file

BITS service misconfiguration on herd: For service permissions, the ’BITS’ service was exploited with the Invoke-ServiceAbuse command to create a new user with administrative privileges named ”john” with the password ”Password123!”.

remediation: the BITS service should be configured properly to not allow exploitation. There was also an unattended install file in C: \\Windows \\Panther \\Unattend.xml which should be addressed. In general, regular system checks should be run using PowerUp and Invoke-AllChecks to look for vulnerabilities.

nginx: One of the vulnerabilities with the highest score of potential risk, CVE-2013-0337, was a result of nginix's default configuration that uses world-readable permissions for the access.log and error.log files, which allows local users to obtain sensitive information by reading said files.

remediation: Default configurations in the access.log and error.log files should be updated to prevent local users from elevated access

ISC BIND 9.11: One of the vulnerabilities with the highest score, CVE-2017-3141, related to the BIND installer on Widows using an unquoted service path which could enable a local user to achieve privilege escalation if the host file permissions allowed it.

remediation: The permissions for the host file for the BIND installer should be changed to ensure only users with admin privileges can use it

Apache httpd server: One vulnerability of high risk, CVE-2019-0217 related to a certain function and running a threaded server, which could allow a user with valid credentials to authenticate using another username, bypassing configured access control restrictions.

remediation: Caution should be taken when running a threaded server

tomcat: Analyzing the OpenVAS scan produced a list of 35 vulnerabilities, although only 6 were shown and could be analyzed. Of these, the most severe were the Apache Tomcat Manager Remove Unauthorized Access Vulnerability and the Apache Tomcat Server Administration Unauthorized Access Vulnerability. The first vulnerability allowed login to the Tomcat Host Manager with user and password as "tomcat". An attacker could exploit this to upload and execute arbitrary code, which would facilitate a complete compromise of the affected computer. For the second mentioned vulnerability, it was possible to login into the Tomcat Server Administration at 172.24.0.42:8180/admin/index.jsp (once again, with "tomcat" as the username and password) which could be exploited by a remote attacker to gain access to sensitive information.

remediation: A password should never be identical to its associated username, or even similar. The tomcat password should be changed immediately to something more complex, or the tomcat user should be removed from tomcat-users.xml. Other vulnerabilities could be resolved by upgrades and enforcing the transmission of sensitive data via an encrypted SSL/TLS connection.

Conclusion

Overall, the M3g4c0rp company has shown itself to have a moderate amount of vulnerabilities which were almost entirely fixable by straightforward means. Testing was conducted using a variety of penetration tools and methods such as the Metasploit framework, network scanners, password cracking, and fuzz testing, all of which proved to have some level of success. The found vulnerabilities were listed in order of most to least severe, with remediation tactics included in order to remedy the problem.