PENETRATION TESTING AND VULNERABILITY ANALYSIS REPORT

NBN Corporation – December 19, 2024

Maya Humston
mlh9655@nyu.edu

Table of Contents

Executive Summary 2
  Overall Security Rating – Immediate Action Recommended! 2
Introduction 2
  Background 2
  Goals + purpose 2
  Rules of engagement 3
  Contact 3
  Important Immediate Security Information 3
Methodology 4
  How Findings Are Documented 4
  Assessment Process – Step by Step 4
Findings 10
  1 – Directory Traversal 10
  2 – Cross Site Scripting 10
  3 – Command Injection 11
  4 –Cross Site Request Forgery 11
  5 – Insecure Configurations – Anonymous FTP login 11
  6 – Weak Password Configurations 12
  7 – Insecure Configurations – sudo privileges 12
Conclusion 13
  Goals 13
  Results 13
  Targets 13
  Risk and Recommendation 13
Appendix A – System Reconnaissance: Topology, Ports, and Protocols 14
  Figure 1 – Open ports, protocols, services, and versions on NBNServer, discovered via NMAP scanning 14
  Figure 2 – Executables With SUID Enabled on NBNserver 14
  Figure 3 – Open ports, protocols, services, and versions on NBNClient, discovered via NCAT scanning 14
Appendix B – Discovered Exposed Data 1
  1 – Flags 1

2 – Customer Data – names + email addresses 3
3 – Employee Data – username + password 3
4 – System Data – users + passwords + groups + keys 3

Executive Summary

This document details an engagement report between the client NBN Corporation (hereby referred to as “NBN”) and Security Company (hereby referred to as “me” or “I”). The client approached Security Company for a penetration test on December 10, 2024. All testing was concluded and the report was finalized on December 19, 2024, and all sensitive data has since been purged from Security Company Systems.

Overall Security Rating – Immediate Action Recommended!

The rest of the document will describe in detail every finding of interest, as well as the testing methodology and vulnerability remediation recommendations. However, it is first necessary to state that the system that has been tested throughout this engagement is **critically vulnerable** and has almost **certainly** been compromised by threat actors. It is the recommendation of Security Company that the vulnerable systems be shut down immediately and protected from any external access. The risks to NBN of keeping the system active and externally accessible include **data exfiltration, employee identity compromise, malware injection, and serious destruction to NBN-owned systems**. An attacker may breach the system and elevate to root privilege with **extreme ease** and **little sophistication**, and thus such a breach is a matter of time if it has not already occurred.

Introduction

Background

Security Company was approached by NBN following a previous breach of their systems that led to the loss of sensitive employee and customer data. NBN stated that the vulnerability that allowed the breach has since been secured. However, there is reason to believe this external-facing server is still being targeted and another attack may be imminent. NBN requested a Red Team style engagement in an effort to mimic real threat actor attack vectors and identify other areas of weakness.

Goals + purpose

This engagement was a Red Team style assessment, intended to mimic a true threat actor as accurately as possible. Copies of the system was provided, but no login information was provided. This assessment started with as little privileged or identifying information as possible in an effort to reflect a real threat actor’s methodology. As the goal of this assessment is primarily to breach the system, the testing team had to gather as much information as possible through manual means.

The purpose of the test was to escalate access and privileges until damage could be done. Although damage can be inflicted on a system even without root access, gaining root access was the primary goal of the engagement. The process of the test can be summarized in three major steps: 1) scan the system and perform extensive reconnaissance 2) find and exploit vulnerabilities, including those leading to data access and exfiltration 3) elevate privileges to establish control over the system.

Rules of engagement

The client established terms to the assessment to ensure they can maintain uninterrupted operations while still testing the system as thoroughly as possible. In pursuit of that, NBN and Security Company established this engagement as a standard red-team assessment with the following stipulations.

- Denial-of-service attacks are not of interest and are out of scope.
- While data exfiltration is not the primary goal, it is still of interest. Any critical data that is exposed and can be exfiltrated (i.e. passwords, flags, customer/employee information) should be recorded and included in the assessment.
- Any sensitive data gathered via reconnaissance or exploitation should be handled with care and discretion, and copies of data located on Security Company systems should be properly disposed of post-engagement following appropriate protocols.
- Vulnerabilities should be found and exploited, and be detailed alongside a full recommendation for how to fix them.
- Any vulnerabilities enumerated should include their risk score in order to prioritize and rank findings.
- The timeline for this test is short due to the critically time-sensitive context. The engagement should be thorough but time sensitive vulnerabilities should be reported as quickly as possible.

Contact

The point of contact representing the client, NBN Corporation, is Bill Gibson, CISO. Email address is xxxxx@xxxx.xx.

The point of contact representing the service provider, Security Company, is Maya Humston. Email address is mlh9655@nyu.edu.

Important Immediate Security Information

Though previously stated in the Executive Summary, it cannot be overstated that this system is critically vulnerable and likely already compromised. Some of the vulnerabilities require complicated fixes, and a full patch would amount to a full overhaul of the system. It would be safest for NBN, its employees, and its customers to isolate this system completely from outside access. The vulnerabilities found are easy to exploit and require very little sophistication, but the effects of compromise would be potentially catastrophic. While it is entirely up to the client to weigh the risks for themselves, the estimated cost of a compromise is very high. It is the official recommendation of Security Company that such a breach be avoided at all costs.

Methodology

This section is intended to document the process I followed during this engagement. In detailing the chronological steps I took as I explored the system and exploited vulnerabilities, I attempt to explain the thought process an attacker may also follow to accomplish the same.

How Findings Are Documented

Throughout the chronological description of the assessment, I will mention every vulnerability I found and/or exploited – these exploits often served as pivot points to expose another vulnerability or escalate privileges, and should be viewed as connected instead of distinct. However, I will keep finding descriptions here brief. Every vulnerability will be more thoroughly documented in the "Findings" section, alongside descriptions, risk scores, how the exploit can be recreated, and possible mitigations. Additionally, while exposure of sensitive data will be mentioned in this section, it will be similarly documented in great detail in Appendix A. The methodology section is intended to understand the system and the attack as a coherent whole.

Risk scores of findings are intended to give a general understanding of the level of risk each vulnerability presents. Many vulnerabilities have been given numerical risk scores according to CVSS indexing, and this report will provide those scores whenever possible – however, for the purposes of this assessment, the report will focus on assigning a category of "Low", "Medium", "High", and "Critical" for all vulnerabilities.

CVSS index scores are assigned by approximating such factors as the ease of exploitation, the attack vector, the attack complexity, and the privileges required, among others. While these scores are helpful, it is often better to use a more generalized version of this scoring with low/medium/high/critical. This can help communicate the level of risk, and help the client understand which vulnerabilities are of highest priority.

Assessment Process – Step by Step

Setup

Once a copy of the server and client has been setup, I am on the same subnet as the server. Therefore, the server and I can communicate directly, whereas the client machine is not on the same network as me, but can communicate directly with the server via subnet 2. My client machine will be referred to as 10.10.0.10, nbnserver as 10.10.0.66, and nbnclient as 172.16.1.2.

Begin with reconnaissance. To examine the server, I performed an NMAP scan of nbnserver. With TCP version scan enabled, I can determine which ports are open and running with TCP protocols, as well as the specific service and version running on each if applicable (Appendix A, Figure 1).

There are four major TCP services I can detect: two HTTP services, one SSH service, and one FTP service.

Web Application Examination

Begin with the HTTP service. The HTTP service running on both port 80 and port 8001 is Apache 2.4.29. These servers can be connected to by sending HTTP requests to 10.10.0.66:80 and 10.10.0.66:8001. It is immediately apparent that both services are nearly identical – the web application running on port 8001 appears to be a staging/test version of the application, while the application on port 80 appears to be the production environment. Most of the following tests can be performed on either application, but for the sake of simplicity, assume all further examination is performed on port 80.

Begin enumeration of the web application – all discoverable pages, directories, forms, etc. To do this, I employed a combination of manual exploration and automated scanning with Zap.

- Manual exploration showed me that there were two main pages directly accessible to a user: the homepage, and an employee login page. However, navigating to the robots.txt file showed that there were two directories that could be of interest: /data and /internal. Both directories can be navigated to in the searchbar, and both contain information and files that should not be accessible to users. (Finding 1)
- The Zap scan accomplished two things: Discovery of pages that could not be navigated to through the user interface (/data, /internal, /phpinfo.php), and automated attacks using common methodology. This scan helped with the discovery of findings 2-4 – cross site scripting, command injection, and cross site reference forgery. While there were other potential vulnerabilities discovered by Zap, I focused on these three highest priority ones.

Navigating through the /data directory exposed data that should not be accessible. Flag1 and Flag4 are stored in this directory (Appendix B-1.1); other images that may contain sensitive metadata are stored in this directory; and most importantly, there is a file called customer.list in this directory. The fact that this file is accessible to unauthenticated users is a severe vulnerability as addressed in Finding 1. However, it also directly exposes customer data (Appendix B-2), and can be used in tandem with other vulnerabilities described further.

The exposed phpinfo.php file is a vulnerability akin to information exposure. While none of the data exposed in this file is critically sensitive, it does include the versions of Apache, PHP, and other services that are running on the server, which can provide information to attackers regarding potential exploits. It also directly states the system user that is running this web application, www-data, which could be of use to attackers, and may contain exposed environment variables in plaintext. As this information exposure was not directly utilized in this assessment, I will not include it as a finding, but should still be considered a vulnerability.

Examining the customer.list file and the homepage reveals that submitting a name and email into the “Subscribe Now” form immediately populates the file /data/customer.list and appends this “new user” to the end. It does not provide appropriate input sanitization.

- The form can be submitted with only one field filled in. While not necessarily a vulnerability, this is likely not the intended behavior and gives attackers one less step to an attack.
- The “name” field accepts inputs containing dangerous characters like ‘”()?{}:; and does not sanitize these characters. The input accepts HTML tags like <script></script> or <img/> (Finding 2)

- The “name” field passes the input directly to an OS command. As such, inserting an OS command into this field will allow malicious commands to execute (Finding 3). **This is the vulnerability that I will now exploit to expose information and escalate privileges.** By inserting the string ‘; echo “$(cmd -arg)” ‘ into the name field, the system will execute “cmd -arg” and print the result to customer.list. I used this to view critical files like /etc/passwd with “cat”, examine directory contents with “ls” and “find”, and perform other critical commands referenced in future steps.

The next goal is to escalate to employee privilege to have read access to the contents of /internal. While the files in this directory can be accessed, the data within them is unhelpful without authentication. Attempting to login via login.php is unsuccessful without a proper username/password combination. However, failed logins do return some helpful information – namely, the exact SQL query that the system runs to verify the username and password.

- While I will not include this finding in the enumeration section, it is still of concern.
- The fact that the webpage shows the query in plaintext can help an attacker understand the system commands that are being run, as well as the types of input sanitization (or lack thereof)
- The exposed query shows the password hash that is computed from the input. An attacker can see this and determine that the passwords are being stored as MD5 hashes with no salt. While this is not useful to attackers without more information, it is still information that should not be publicly exposed. Additionally, the use of an unsalted MD5 hash is an insecure storage method as the hash type is weak.
- The query itself is not formatted in a secure way. Using the previously described command injection vulnerability to examine the login.php page (cat /var/www/html/login.php), we can see that the command queries the table for users containing those two variables, and allows authentication success if the query results in *at least one user.* This is a severe vulnerability. If the attacker can manipulate the SQL query to output the whole table instead of just one row, the attacker can authenticate as an employee.
- **While I did not find a working SQL injection input and could not authenticate in this manner, there is very likely a SQL injection vulnerability here.**

Examining the login.php page with “cat” exposes more critical information. The PHP code contains four variables in plaintext that should NOT be exposed to attackers:

```
$servername = “localhost”;
$database = ‘nbn’
$username = ‘root’;
$password = ‘digital’;
```

- These variables should have been stored as hidden environment variables. These variables can be used as arguments to a mysql command that can execute any query directly.
- Using the command injection vulnerability, insert the command mysql as follows:
    - mysql -u root -pdigital -h localhost -D nbn -e “SHOW TABLES”

- This is a query specific to MySQL and MariaDB that shows the names of every table in a database. In this case, the only table was the table “users”.
    - mysql -u root -pdigital -h localhost -D nbn -e “SELECT * FROM `users`;”
- This outputs the entire contents of the table “users” to the customers.list file. There are only two employees in this table – Stephenson and Gibson. The passwords in this table were stored in MD5 hashes that were easily crackable with the help of Hashcat. (Further information in Appendix B-3)

We have now stolen the username and password combination of two employees. Logging in to login.php with these credentials successfully authenticates and directs me to the page /internal/employee.php, where there is a link to the page /internal/customers.php.

- This page provides the same list of customers found in customers.list, and also updates with form submissions to the “Subscribe Now” form. CSS attacks enumerated in Finding 2 are particularly important here – any script tag containing malicious code inserted into customer.list will be placed into the HTML of this page, and will execute when an employee loads this page.
- The page also loads a paragraph element containing Flag 2 (Appendix B-1.1).

FTP

Having accessed the passwords of two employees, we should pivot from the web application to the other services running on this server. Among these was FTP on port 65534. The initial NMAP scan determined that there was one username that was allowed anonymous access into FTP without a password (user ftp) (Finding 5).

- While this user has very low privileges and cannot leave the directory it was placed in (/home), it still may expose sensitive information. It was placed in the /home directory and can access /home/gibson, which is the directory in which user gibson is placed when they authenticate in. It is unlikely that ftp, an anonymous privileged user, should share a home directory with user gibson, the most privileged user on the server besides root.
- Additionally, /home/gibson contains potentially sensitive information in the form of flag3 (Appendix B-1.3).

Testing the two user/password combinations from the web application reveals that user gibson uses the same password for the web application as they do for system configs and for ftp/ssh. This password, “digital”, is a weak password, and its reuse is a major vulnerability (Finding 6).

Connecting to the server via FTP as user gibson / password digital is possible. However, the FTP shell is limited.

SSH

User gibson is able to connect to port 443 on which SSH is running. This gives a shell with pretty extensive capabilities.

Viewing /etc/passwd and /etc/group shows that user gibson is one of the only users with login capabilities, and is a member of several groups. Of most interest is the fact that gibson is a member of the group sudo, and can execute select commands with root privilege even without root password. Executing the command "sudo -l" lists all the commands that gibson is allowed to execute with sudo. There are exactly three:

- /bin/echo
- /usr/bin/whoami
- /usr/bin/tee

Additionally, I checked the executables that had the SUID bit set. I did so with the command "find / -perm /4000 2>/dev/null". The output is too long to list here (see Appendix A Figure 2). This was helpful for enumeration. There are often ways to manipulate these commands to escalate privileges. While I did not find or utilize an exploit in this manner, it is not confirmed that there is not still a vulnerability.

Instead, I took advantage of the three commands gibson is permitted to use with sudo. Although echo and whoami are less helpful, tee is critical. A complete description of this exploit can be found in Finding 7 – in sum, I used echo and sudo tee to append a line to /etc/passwd, even though gibson does not have write privileges for this file. I appended a line creating a new user with root privileges belonging to group root, and can now login as that user.

I have now gained root access to nbnserver via an SSH connection. A few more actions can be accomplished before pivoting to nbnclient.

- Copy /etc/shadow and /etc/shadow- files for personal use. These contained the hashed passwords for root and gibson. (Appendix B-4)
- Copy the private SSH keys found in /etc/ssh. Only root users have read access to these files, but they can be used to authenticate into root@nbnserver via ssh later. (Appendix B-4)
- Copy the system logs found in /var/logs/. This can assist in further data enumeration and exfiltration.
- Copy flag4 found in /var/www/html/data/flag4.jpg, which is only readable by root (Appendix B-1.4)

Pivot to nbnclient – 172.16.1.2

Having established a presence on nbnserver, we can begin enumeration of nbnclient.

Nbnserver does not have nmap installed. Instead, we can use netcat to perform a scan of all ports and protocols. There are many more ports open on the client than on the server (see Appendix A Figure 3), but only seemingly five of interest: SMTP, SSH, POP3, and IMAP2, and one custom application that appears to be used by employees to manage customer accounts. All other ports simply display random text when connected to.

SSH

Attempting to SSH into nbnclient as user gibson with password digital fails. Either gibson is not a user on this machine, or he uses a different password. However, connecting to nbnclient via SSH as user Stephenson with password pizzadeliver (as discovered in the SQL database) is successful (another example of password reuse; Finding 6)

Enumerate the data available to this user.

- Ls reveals that there are three files in stephenson's home directory: flag7 (Appendix B-1.7), nbn, and nbn.backup.
- Cat /etc/passwd reveals that Stephenson and root appear to be the only two accessible users on the system
- Cat /etc/group reveals that Stephenson is not a member of any other group
- Sudo -l reveals that the only command Stephenson can execute with root privilege via sudo is /home/Stephenson/nbn. This appears to be the custom executable that is running at port 5355.

There does not appear to be an obvious way to escalate privileges via SSH as was accomplished on the server.

POP3, IMAP2

Both of these protocols utilize the Dovecot service for Pop3 and IMAP, as evidenced by the welcome message immediately upon establishing a TCP connection with netcat. It appears that a user may be able to login and investigate after appropriate authentication.

SMTP

Port 25 has an ESMTP service running. A simple connection to this service and initiation with "EHLO client" begins the interaction between the client and the mailserver. While not much information can be gathered, we can confirm there is a mail user named Stephenson ("VRFY Stephenson" returns "2.0.0 stephenson", which does not reveal much, but "VRFY gibson" returns "5.1.1 <gibson>: Recipient address rejected: User unknown in local recipient table").

Nbn – Custom Executable

The nbn executable is seemingly not very complicated. Upon execution, the user is presented with the opportunity to input a number 1-6. Inputting very large input seems to have no effect, as each char seems to be processed one at a time. Inputting 6 clears the screen as intended; 5 terminates the process; 4 does nothing without first creating an account; 2 and 3 present more options for user input, but instructions are unclear; and 1 allows the user to "create a new profile" by inputting a name and an address.

Of note is the fact that inputting very long values for user name or address will cause a segmentation fault and crash the program. It is very likely that this segmentation fault can be intentionally exploited by overflowing the buffer with user input, determining which bits are the ones that overwrite the EIP register, and crafting an exploit payload accordingly. This can be aided by the fact that gdp has been installed on nbnclient already, which will assist with examining and exploiting the nbn executable.

Wrapping Up

Although I was unable to escalate privileges on nbnclient beyond user Stephenson, accessing root on nbnserver is a major attack. Offloading the password hashes from /etc/shadow and the SSH private/public key pairs from /etc/ssh would allow me to continue signing in to root even if my custom-added user is deleted by the system. An attacker would have accessed customer data, which is a breach of privacy for customers, stolen employee login credentials, accessed privileged users on the server, and gotten root. There is undoubtedly more sensitive information on the client and more attack vectors to exploit given an engagement with a longer timeline and more resources.

Findings

This section contains every vulnerability exposed by the above process. Each finding is described in detail, alongside risk scores, mitigation recommendations, and specific exploit steps.

1 – Directory Traversal

Severity: High – the attacker has read access to sensitive files, and may traverse directories they should not have access to, leading to exposure of critical data
Description: The discovery of hidden files can lead to unintended actions performed by the user. The user is not supposed to access files in the /data folder, nor can they navigate there through the user interface. Instead, the user may enter that exact directory path into the search bar to access it directly.
Steps to exploit: Enter the link "10.10.0.66/data" into the search bar and observe the files in this folder that the user may now view.
Recommended fix: Relocate sensitive files to directories the user does not have access to, such as above the working directory instead of below.

2 – Cross Site Scripting

Severity: Critical – the attacker may insert malicious code into the website that will be executed by anyone who accesses it
Description: Cross-site scripting (XSS) is a web security vulnerability where an attacker injects malicious code into a trusted website, causing it to execute on a user's browser when they visit the site, potentially allowing the attacker to steal sensitive data or perform other malicious actions on the user's behalf. This often happens when a website doesn't properly sanitize user input, allowing malicious scripts to be inserted into the page.
Steps to exploit: Navigate to the homepage. In the "Subscribe" form, submit a script tag into the name field: <script>alert(1);</script>. Next, navigate to the employee login screen and login as an employee, then follow the link to future customers. Observe that the script tag loads into the

page and executes the alert – if this code had been something more malicious than a simple alert, the attacker could cause the employee to execute very dangerous commands.
Recommended fix: Ensure all forms have their input properly sanitized. Strip all unwanted special characters like tags <></>, and append quotes with slashes. Do not allow user input to directly populate the site for others unless the user input can be trusted.

3 – Command Injection

Severity: Critical – allows the user to execute any command on the system, limited by the permissions of the executing user
Description: Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell.
Steps to exploit: Navigate to the homepage. In the "Subscribe" form "name" input, enter this string: '; echo "$(cat /etc/passwd)"' Observe the file /data/customers.list – notice the bottom of the list has been appended with the contents of /etc/passwd. The "cat /etc/passwd" command can be replaced with many other commands, as long as they are not persistent, and command output will be written to customers.list.
Recommended fix: Ensure all user input is once again properly sanitized. Do not allow the insertion of quotation marks or other special chars unless necessary. Do not run a shell command with direct user input unless that input can be trusted.

4 – Cross Site Request Forgery

Severity: High – allows the attacker to construct a link that, when visited, will perform an action on behalf of that user
Description: Cross-site request forgery (CSRF) is a type of attack that tricks a user's web browser into performing an unwanted action on a trusted site. CSRF attacks exploit the trust that a site has in a user's browser.
Steps to exploit: In your search bar, directly visit the link /?name=user, then observe how customers.list becomes populated with this new user even though the form was not interacted with. The same can be done to login as an employee if the correct credentials are inserted into the link. Anyone who visits that link in their browser will submit these forms, regardless of intention.
Recommended fix: Ensure all forms are submitted using POST requests instead of GET requests so the links are not directly editable like this. Additionally, incorporate the use of session_id cookies that will provide a layer of authentication. The current method of sending a cookie with authenticated=0 or 1 is easily faked in very insecure.

5 – Insecure Configurations – Anonymous FTP login

Severity: Medium – although little direct harm can be done and this anonymous user has very few privileges, it still led to the exposure of data.
Description: There is a user on nbnserver named "ftp", who can enter the ftp server with no password authentication. Any outside user may utilize this. The true vulnerability is that the ftp user enters the directory /home, and can access the subdirectory .home/gibson, which is home

directory of gibson, the most privileged user after root, and which contains potentially sesnsitive information.
Steps to exploit: start an FTP connection to the server with the following command: ftp ftp@10.10.0.66 65534. When prompted for a password, simply press return. From there, one can explore the directory and files.
Recommended fix: Remove this user entirely unless it is a necessary feature. If that's the case, create a subdirectory within /home that we can confine it to instead, so it cannot access the data of other users.

6 – Weak Password Configurations

Severity: Medium – increases the likelihood that an attacker may crack employee passwords and gain access to their accounts on the server.
Description: The passwords used by the users of this server are weak, not complex, and reused across multiple services. Additionally, the hash type used by the MySQL db is a weak, easily crackable hash.
Steps to exploit: copy the hashes in the 'users' MySQL table to a text file. Download Hashcat from the official site, as well as rockyou.txt. run the hashcat command for md5 that cracks the passwords. Observe that both passwords get cracked in seconds and will not deter an attacker much.
Recommended fix: Utilize a different hash type, like sha256, and include a salt in the hash. Additionally, have every user change their system passwords so they use different passwords for everything. Provide rules for the complexity of passwords, i.e. you must include a combination of numbers, uppercase characters, and special characters.

7 – Insecure Configurations – sudo privileges

Severity: High – allows unprivileged user write access to files they should not have write access to
Description: the configuration of the system places user gibson into the sudo group, which can execute select commands as root. However, there are many files that only the root user may have the ability to write to, such as /etc/passwd. However, the command "tee" is intended to assist in writing to files. If gibson has the ability to run "tee" with sudo, then this command will get executed as root without need for the root passwd, and therefore any file may get overwritten.
Steps to exploit: in the shell as gibson, execute the following commands:

Openssl passwd new (apply a hash function to the string "new") (result: pKP4/UnXgF60s)

Sudo echo "evil:pKP4/UnXgF60s:0:0:root:/root:/bin/bash" | sudo tee -a /etc/passwd

This appends a new user to the /etc/passwd file, who logs in with the password "new" and belongs to the root group. Login as this new user and the shell will have root privilege.
Recommended fix: remove tee as a valid command used by the sudo group.

Conclusion

Goals

The primary goal of this engagement was to simulate a real-world threat actor's approach to identify vulnerabilities and assess the security posture of NBN's system. By conducting thorough reconnaissance, exploiting weaknesses, and escalating privileges, the assessment aimed to uncover critical flaws and recommend actionable measures to mitigate risks. This red-team exercise emphasized the importance of proactive security measures to prevent potential compromise and safeguard sensitive data.

Results

This assessment exposed many severe vulnerabilities. Although I only enumerated findings that I specifically exploited (not just ones I suspect), there are certainly more than I stated. The methodology section exposes this as well. I also focused on medium, high, and critical severity vulnerabilities, but low severity vulnerabilities should still be of concern.

In total, I have enumerated 7 specific vulnerabilities in the findings section, along with a description and a potential mitigation. Findings that are less severe or less specific have been described in the methodology section.

Targets

The system is composed of two targets – a server, which is external facing, and a client, which is internal facing. The main concern was for the external facing servers, which have been breached previously. Indeed, this server can be breached again, as evideneced from my privilege escalation from nothing to root. The client is also insecure however, and if the sevrver is compromised by an attacker, the client is directly exposed and will be in danger.

Risk and Recommendation

This system is highly vulnerable. Even one critical vulnerability is enough to justify shutting down a system, and this system has at least two, potentially more. It is the recommendation of Security Company that the system be removed from public-facing interfaces until these vulnerabilities can be patched. Since the vulnerabilities are present in nearly every service and part of the system, it is likely the system will need to be completely rewritten from the ground up. While a daunting endeavor, it is not worth the risk to leave the company and the system exposed. It is likely the system has already been compromised, and NBN should operate under the assumption that attackers have privileged information already.

Appendix A – System Reconnaissance: Topology, Ports, and Protocols

Figure 1 – Open ports, protocols, services, and versions on NBNServer, discovered via NMAP scanning

Port	Protocol	Service	Notes
80/tcp	http	Apache/2.4.29 (Ubuntu)	Apache version 2.4.29 seems to be secure and up-to-date.
443/tcp	ssh	OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)	OpenSSH 7.6p1 seems to be secure and up-to-date.
8001/tcp	http	Apache httpd 2.4.29 ((Ubuntu))	
65534/tcp	ftp	vsftpd 3.0.3	There is an Anonymous FTP login allowed – user ftp, no password

Figure 2 – Executables With SUID Enabled on NBNserver

/usr/games/screensaver
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/snapd/snap-confine
/usr/lib/eject/dmcrypt-get-device
/usr/bin/chfn
/usr/bin/traceroute6.iputils
/usr/bin/pkexec
/usr/bin/at
/usr/bin/chsh
/usr/bin/newgidmap
/usr/bin/newuidmap
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/sudo
/usr/bin/gpasswd
/bin/fusermount
/bin/umount
/bin/ping
/bin/su
/bin/mount
/snap/core/6350/bin/mount
/snap/core/6350/bin/ping
/snap/core/6350/bin/ping6
/snap/core/6350/bin/su
/snap/core/6350/bin/umount
/snap/core/6350/usr/bin/chfn
/snap/core/6350/usr/bin/chsh
/snap/core/6350/usr/bin/gpasswd
/snap/core/6350/usr/bin/newgrp
/snap/core/6350/usr/bin/passwd
/snap/core/6350/usr/bin/sudo
/snap/core/6350/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core/6350/usr/lib/openssh/ssh-keysign
/snap/core/6350/usr/lib/snapd/snap-confine
/snap/core/6350/usr/sbin/pppd
/snap/core/6673/bin/mount
/snap/core/6673/bin/ping
/snap/core/6673/bin/ping6
/snap/core/6673/bin/su
/snap/core/6673/bin/umount
/snap/core/6673/usr/bin/chfn
/snap/core/6673/usr/bin/chsh
/snap/core/6673/usr/bin/gpasswd
/snap/core/6673/usr/bin/newgrp
/snap/core/6673/usr/bin/passwd
/snap/core/6673/usr/bin/sudo
/snap/core/6673/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core/6673/usr/lib/openssh/ssh-keysign
/snap/core/6673/usr/lib/snapd/snap-confine
/snap/core/6673/usr/sbin/pppd

Figure 3 – Open ports, protocols, services, and versions on NBNClient, discovered via NCAT scanning

Port	Protocol	Service
22/tcp	ssh	OpenSSH 7.5p1
25/tcp	esmtp	Postfix

110/tcp	Pop3	Dovecot
143/tcp	Imap2	Dovecot
5268/tcp	Custom/unknown	
5355/tcp	Custom/hostmon	
5782/tcp	Custom/unknown	
5843/tcp	Custom/unknown	
5854/tcp	Custom/unknown	
5854/tcp	Custom/unknown	
6174/tcp	Custom/unknown	
6573/tcp	Custom/unknown	
6868/tcp	Custom/unknown	
7437/tcp	Custom/unknown	
9562/tcp	Custom/unknown	
12824/tcp	Custom/unknown	
15035/tcp	Custom/unknown	
24204/tcp	Custom/unknown	
28478/tcp	Custom/unknown	
34246/tcp	Custom/unknown	
40998/tcp	Custom/unknown	
42780/tcp	Custom/unknown	
49881/tcp	Custom/unknown	
49953/tcp	Custom/unknown	
52396/tcp	Custom/unknown	
53852/tcp	Custom/unknown	
54597/tcp	Custom/unknown	
56585/tcp	Custom/unknown	
62049/tcp	Custom/unknown	
62992/tcp	Custom/unknown	
63034/tcp	Custom/unknown	
64128/tcp	Custom/unknown	

Appendix B – Discovered Exposed Data

1 – Flags

1. Flag1{CYBERFELLOWS_GOODLUCK}
   a. Insecure configurations led to directory traversal, exposing hidden files
   b. The /data folder – exposed by robots.txt and automated Zap scanning – contains important sensitive data, including flag1.txt
   c. Flag1.txt contains this flag in "ASCII art" format
2. Flag2{down_a_rabbithole}
   a. After successfully "logging in" as an employee (session hijacking, SQL injection, or password cracking), navigate to /internal/customers.php.

   b. This page has been populated with every item on the customers.list page, but also by flag2 at the very beginning.

3. Flag3{brilliantly_lit_boulevard}
   a. Upon connecting to the server via FTP (user ftp or gibson) or SSH (user gibson), the user has access to the directory /home/gibson.
   b. Within this directory is a file called flag3
   c. If using FTP, the file can be transferred to your local device with a GET command; if using SSH, this file can be viewed with a cat command
   a. The file contains a long story, an excerpt from a real book: https://www.goodreads.com/quotes/253186-the-deliverator-belongs-to-an-elite-order-a-hallow-subcategory
   d. However, of true interest is a line within the file containing the flag value. Find manually or find with grep.
4. Flag4{youre_going_places}
   a. The file flag4.jpg was stored in /var/www/html/data alongside flag1.txt. however, flag4.jpg does not have read privileges for anyone except root.
   b. After gaining root, this file can be offloaded to a local machine for further examination, or examined from the SSH shell
   c. To examine the metadata, there are a few options. Exiftool is usually reliable, but there was no useful information output by this tool.
   d. Instead, you can cat the file into xxd to hexdump the data, then find the metadata within that. Using grep works well. The flag was stored in the description tag.
5. –
   a. Although I could not find the file or flag itself, I found reference to the flag in the system logs.
   b. /var/log/apache2/access.log references a GET request from 192.168.1.17 for /hackable/flag5.txt. However, there are no folders called hackable or files called flag5.txt in the entire directory.
   c. /var/log/vsftp.log.1 also references the file – it seems the file was uploaded, downloaded, and finally deleted from the server.
   d. If there is a way to recover deleted files, or if there exists a backup, the flag would be in this.
6. –
7. Flag7{worlds_within_worlds}
   a. Upon connecting via SSH to the nbnclient as user Stephenson, we can see the file flag7 immediately. This file contains a long string of seemingly random alphanumeric characters, with == at the very end.
   b. The equal signs at the end immediately signal that this is base64 encoded data. Decoding this string with base64 -d < flag7 outputs another file of seemingly random digits, this time in binary. However, there is a readable string right at the beginning that reads “PNG”. Outputting this data into a new file called flag7.png and opening it with image viewer allows us to see the flag.
8. -

2 – Customer Data – names + email addresses

NqF5Rz@yahoo.com : connie ////
long@gmail.com : capone ////
hjk12345@hotmail.com : ned ////
snoogy@yahoo.com : frank ////
polobear@yahoo.com : jess ////
mkgiy13@gmail.com : max ////
tempbeauties@live.com : peterpiper ////
amohalko@gmail.com : desiree ////
ramy43@gmail.com : greatone ////
dowjones@hotmail.com : stockman ////
yahotmail@hotmail.com : eugene ////
hydro1@gmail.com : maurice ////
boneman22@gmail.com : dennis ////
hamlin@hotmail.com : willie ////
nevirts@gmail.com : jackie ////
redtop@live.com : camille ////
langp@hotmail.com : pontoosh ////
jnardi@live.com : peter ////
4degrees@hotmail.com : ralph ////
fretteaser@hotmail.com : derek ////
bsquard@live.com : wilbur ////
zd0ns23@live.com : wrinkle ////
scheefca@live.com : gerry ////
enobrac@gmail.com : marcy ////
saazuhl1273@gmail.com : cauhuln ////
fwe315@live.com : evan ////
wilson@gmail.com : triad ////
navresbo@yahoo.com : heather ////
XO6Pn75pjjK@yahoo.com : sandy ////
darkness024@yahoo.com : randy ////
jjstrokes@live.com : beansko ////
zimago@yahoo.com : george ////
katrina@gmail.com : harald ////
awesome@gmail.com : larry ////
jess@yahoo.com : jesse ////

3 – Employee Data – username + password

Information from the MySQL database nbn, table 'users':

User_id	firtname	lastname	user	password	avatar	Last_ login	Failed_ login
1	gibson	gibson	gibson	e0e1d64fdac4188f087c4d44060de65e	data/ ourCEO.jpg	2019-04-21 14:08:55	123
3	stephenson	Stephenson	stephenson	942cbb4499d6a60b156f39fcbaacf0ae	data/ stephenson.jpg	2029-12-12 01:23:45	123

Password hashes cracked:

user	Password_hash	Password_cracked
gibson	e0e1d64fdac4188f087c4d44060de65e	digital
stephenson	942cbb4499d6a60b156f39fcbaacf0ae	pizzadeliver

4 – System Data – users + passwords + groups + keys

Password hashes found in /etc/shadow and /etc/shadow-:

user	File	Password_hash
Root	/etc/shadow	$6$x8yQ8PlY$/4jhqQfPE6vyFU7bU1UmjY.nXWEpxzz1c82x6MphQBlofiKN9/DzsXCSvv4RB/pYdmzOehx9cRbm3WlAtdedz1

Gibson	/etc/shadow	$6$evQQsME4$CRS5h3FhqUMQp4afUe/EsXqF5AGUgMnH0byX4kUaYOhroXI4CKJj36bjJV6gh3cJcHOwi3YpYsWCmboIygQv40
Root	/etc/shadow-	$6$V7ITyzbO$H3PY7DTLO6YB8MIIvKVmaeh7nweUSV4w8TQ2hCalCHJDlX976vLRw9eAJfYPE4JY60JG1mwmhEDs/Br7zjgpJ.
gibson	/etc/shadow-	$6$ADhVWD4o/ewiLkH5$lF8QFIoAJPoqEXcwSILJfNc33ZNulmJY9ROmY46eARULmBF91Rl.Zf60.yRB4qHJONvGwF4w9lhHJGoRtCqpP0

Group information found in /etc/group

```
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,gibson
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:gibson
floppy:x:25:
tape:x:26:
sudo:x:27:gibson
audio:x:29:
dip:x:30:gibson
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:
sasl:x:45:
plugdev:x:46:gibson
staff:x:50:
games:x:60:
users:x:100:
nogroup:x:65534:
systemd-journal:x:101:
systemd-network:x:102:
systemd-resolve:x:103:
input:x:104:
crontab:x:105:
syslog:x:106:
messagebus:x:107:
lxd:x:108:gibson
mlocate:x:109:
uuidd:x:110:
ssh:x:111:
landscape:x:112:
gibson:x:1000:
ftp:x:113:gibson
ssl-cert:x:114:
mysql:x:115:
```

Private SSH keys found in /etc/ssh:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAruqcIs51NulkwEGZemRpI1ZSvoEar4J89gVMhL14wTXnkTQe
T2YofCyt5Rp9rC8FLbrr25mUkTUXgF4eEWptDCoDT4l+HzJM6TbmA22yMyX/KRAE
02F2e7bWhTb1aOx7EuwC2arinJ0b1e35QcjPV1oi//YH0YutFHuuKcie2Ne8bhZp
VTUAGtvWXyM0TZpnQRMioVIJ654KZRzh6+CK9qwz+CG8V813jx+4YPXuyIzRcJck
Rtr3rds7pJHip/cklR+qaT8vPEgM4p49pLBfqgwMoKlJ9nQK3KBqO1du6EFpNQxf
/Y2HIgJ4ZD9I03rCKWMCjawDp6m3tuufaiAruwIDAQABAoIBAQChJa/3OTIwBtI1
dbpKUhuy3KKatCK7s4Igvj0CxfRsPJc85UFPcRS3YgpLWh6R9TfWK7GDvXhsVvdx
4kPdU6lnXYVzT36MizDkYxmB6xwTl1v+nq1zeQVJcOcuD7YElmnEwv7VPmRMWUn9
y6KqArFTDs3T9vy5Bu/MQS3xfM0rg//2k22DOCJI6yHppVAIb+tmJFbV4MyeAbbu
L0+Z5kDWysVI2l1Tqg0UfFIFbgzneVA67hhY8VUu1Jag6bwHEk2maIFIywn4z/Ob
ujS1TOptnrvUlzZo5yWlZb31CxI69fr9ThMHTDN0jWXwfCbiSOeRsEjXiQvL3yGs
gYJqQzOhAoGBANwIUE6lzcjz2B8OwBI4FBR3AWxL6BFE9ttVBgfg+x8QggeINGtF
jFk3IKeeWY6gu7GnK2rdVhHVNcCJ81uJBUZYgVdXEEr9pSwfcwiYCY+hMIr+HBnd
auV38k5gdff7EPi/rUa5iM8nA14kWz86h9bOV2hz691LksV8NX+TZw+jAoGBAMuC
```

VApkfBP/q5KGXXp4o1nPeNUQXRRpKfCo3L+QKsRdlJrHhqZPrU9/3xXol5DqwrfS
c1V0gIE0vTxC8I74SJJ3PUpJ7+zNMTym7EjRdHfzlbxrqkUNGQ9+7778+V0wmoXu
wcLS/Ag0qADYIVU8hM7v3CERThkPMJu8XT4WatUJAoGAYyBntZbi57ZbVlqZ01tq
SHH4tSZZTXZpVBSw29nSqIMSYXxOSUmS2hqI3prrM+Nby8RJPiKrBOuuBKjhdwp6
0t9KneN+VpqA0Cj29Rcxr4Bp0yR52nkzEGACTKcOXoLAeaA42xT4+oJT9RSh+xFO
Y0qgfFFFIHRMkiXMIEZy9ssCgYADOfK5I3SwIHqI3SZ4PZdhYh+pSRQgsbYfgZ3W
T6PN7yne9lDVLCg0PXO89i4I9x/VKDn53dn3sm1ZmjoUGi2UXN2U4u0x1OL18jOG
yANPr4XtMcvGQnnmO/3zCjGt34pjmzBpNU0XmDYdpU+J+WalnTnhMFQLo6kCnw7f
YB2beQKBgQDI6bB0leBZSLgOIwoYQg0A2D2599lObVi+KSWsPEX+QTG0v/2GjHYw
qdbd5GP6li1phF82SIbMZNoKNCm/c26gOtoKqK0zlbHvaRHdybHDvr2rDcGrvABy
+WSuZ/nsBkGtxGgPFw+BimkBS1Dqm+j28KE6bHDAzp/CUtH4OLEd9g==
-----END RSA PRIVATE KEY-----

-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQDEDjoO5igAw+Wm3MdtQEABiVdNkWCrK8cQ/NmYzj7irZq0csYC
7E+HzJXX2oVH2I43Ft/bNishMWMsN7rHKL+ZpLjzBXoQ+wcgdS9QxI/1YqP3siSy
xGS/x5DvDdYHJfdlCpUwj1TDq6DBC8jVN2oU2LJMt1xGxnWhW4aCDyoHQwIVANHX
hkBHbiTzVlsylKZnlzz1l5gDAoGBAMHkj1VUbRmZ8x8k7+xbdNi4nWHuCez3lcuq
FKNyDPCKoY84te6EeG4cM2b4dvJogZXQW5nkSScQMUEyno5QpevFeNgzcU6yHWfC
HLmPVBXVtHFHZXjyx3Y8lYH7o2jtihA/MVCd0OXGSMdBJ97xhidaE8FckW/M6JAz
ntzzV2vxAoGAHf++l+ww6Py8ulsGHDES5qsA/9/I4pRR/meaurYl7X57KUrObeRf
VAL8kfn9AJSdGRYnZEZW5mnFSOkCunE95jEpnYVAUCc3xv9pNuVGk8SdixR4S2Vh
qWKRFr9B7OdR9IIhILi8JYBOlYstK9YJ3+vUtRBwjprfMlQlmJSrB7oCFC0Hm4ur
rQqn3tYF1qEqpZFqF7H1
-----END DSA PRIVATE KEY-----
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIHpHDFcTN5L5sjYOjnAK4MXCPdklphxliXbedPjvLSB1oAoGCCqGSM49
AwEHoUQDQgAEm97ju1lkv1plTTB6JBvV1uIqPdTb11x0o3c0VProUXPdlIfq7LvY
zR2eNL4ZZnMXaXOgJoi/OtuxNpc77jOR6A==
-----END EC PRIVATE KEY-----

-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZ
W
QyNTUxOQAAACCTtI7qhX2KlEyZDK2VlFGfwThTE3vH7HDrBzlBel5tsgAAAJiZzMzBmczM
wQAAAAtzc2gtZWQyNTUxOQAAACCTtI7qhX2KlEyZDK2VlFGfwThTE3vH7HDrBzlBel5tsg
AAAEB6Zh9HB1rPlV1VwFdfL3B1kfx3TsU6+Cf/qZyhBe9YvJO0juqFfYqUTJkMrZWUUZ/B
OFMTe8fscOsHOUF6Xm2yAAAADnJvb3RAbmJuc2VydmVyAQIDBAUGBw==
-----END OPENSSH PRIVATE KEY-----

-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIHpHDFcTN5L5sjYOjnAK4MXCPdklphxliXbedPjvLSB1oAoGCCqGSM49
AwEHoUQDQgAEm97ju1lkv1plTTB6JBvV1uIqPdTb11x0o3c0VProUXPdlIfq7LvY
zR2eNL4ZZnMXaXOgJoi/OtuxNpc77jOR6A==
-----END EC PRIVATE KEY-----