Group 2: Phase 2 Report

Lockwood Topping, Eric Chapman, Tre Germany, Joseph Daher, & Manasa Mutpur
Department of Cybersecurity, Kennesaw State University
CYBR 7910: Capstone in Cybersecurity Practicum
Dr. Zhigang Li
November 18, 2024

Project Status Update

Since phase 1 of our server hardening project, our team has made significant strides in securing our environment. We changed all default passwords, created non-administrative users to provide more granular access controls, enforced HTTP-to-HTTPS redirection, configured HTTP Strict Transport Security (HSTS) to defend against man-in-the-middle attacks, bolstered WordPress security by installing Wordfence and implementing a web application firewall (WAF), incorporated intrusion detection systems (IDS) and intrusion prevention systems (IPS). We also set up Google Authenticator to enforce multi-factor authentication (MFA) for SSH, sudo, and su commands, restricted database permissions for the WordPress user, enabled Cockpit and dnf-automatic to manage system patches, and used rsync and cron jobs to handle automated backups. Additionally, we installed Wazuh-Manager for log management and configured Snort as our IDS/IPS to monitor and respond to potential intrusions.

To streamline server management and reduce troubleshooting complexities, fewer team members worked directly on the server. This was very important as the server had to be restored a couple of times due to Linux's built-in security structure – SELinux – shutting down SSH or restricting MFA applications from running properly. Those who were not working on the server directly instead worked to prepare the team's final presentation. This splitting of tasks ensured that all team members made maximum contributions towards the project goal. Additionally, meeting times were adjusted twice to accommodate everyone's schedules, allowing us to stay coordinated and on track with our milestones.

Implementation of Security Program

To effectively secure our server against cyber threats, we implemented a comprehensive security program focusing on administrative privilege restriction, system upgrades, robust access controls, network security, backup and disaster recovery, as well as logging and monitoring. This security program was carefully designed to address known vulnerabilities, prevent unauthorized access, and ensure system resilience.

1. Restriction of Administrative Privileges

To limit potential security risks from elevated access, we focused on restricting administrative privileges across SSH, sudo, MariaDB, and WordPress services. In SSH, we minimized access by establishing a non-root user with limited permissions. The creation of this non-root user is shown in Figure A1 in Appendix A, where the rest of the pictures in the implementation phase of this document will reside. We then configured the sudoers files to restrict the actions that user could perform, reducing exposure to high-risk commands. In MariaDB, we applied role-based access control (RBAC) to separate the administrative and standard user roles, enabling the database to only accept commands essential to each user's responsibilities. WordPress administrative access was limited by forcing WordPress to use one of the non-administrative database users (see Figure A2), ensuring that only certain tables were editable and other critical tables were protected. And lastly, we changed the passwords for the root user (see Figure A3) and deleted all other default user accounts for MariaDB (see Figure A4) and WordPress to ensure previously compromised accounts could not access the server or its services. This privilege restriction mitigates potential damage from unauthorized or compromised user accounts, supporting a principle of least privilege.

2. System and Software Upgrades

A regular update schedule is critical to maintaining security. Using dnf -y update and dnf -y upgrade, we began the initial upgrade of the system (see Figure A5). We also enabled Cockpit (see Figure A6) along with dnf-automatic (see Figure A7) to streamline the process of pushing and managing updates across the system. Cockpit provides a user-friendly interface that simplifies monitoring, making the process efficient and helping us avoid outdated, vulnerable packages. By automating updates, we mitigated risks associated with security patches and bug fixes. These upgrades helped close potential entry points for attackers and ensured that our server consistently benefited from the latest security enhancements.

3. Strengthened Access and Authentication

To further bolster access control, we introduced multi-factor authentication (MFA) using Google Authenticator (see Figures A8, A9). This additional layer of security requires users to input a time-sensitive code generated on a separate device, minimizing risks from password-based attacks. For SSH access, only verified users could authenticate, as we disabled root login and implemented MFA on all critical access points. Google Authenticator also secures sudo and su commands, requiring users to authenticate with MFA before executing any privileged commands (see Figures A10, A11). Integrating MFA within WordPress's wp-admin access and cockpits management portal again further enhanced the security of our website. This setup limits unauthorized access to sensitive areas and maintains high standards for user verification.

4. Network Security Enhancements

We reinforced network security by implementing intrusion detection and prevention measures. Snort 3, combined with PulledPork, was used to monitor and detect malicious activity (see Figure A12). PulledPork automates Snort rule updates, ensuring that our intrusion detection system (IDS) remains current. Additionally, we installed ModSecurity as a Web Application Firewall (WAF) to inspect HTTP traffic, while Fail2ban (see Figure A13) monitors login attempts and blocks suspicious IPs after multiple failed attempts. We enforced an HTTP-to-HTTPS redirection to secure data transmission over encrypted connections (see Figure A14); and lastly, Wordfence was deployed to secure the WordPress application against attacks, provide malware scanning and firewall protections, and enforce additional login security features (see Figure A15). This combination of tools helped to create a layered defense that detects and blocks potential threats while protecting sensitive data transmissions.

5. Backup and Disaster Recovery

To ensure backup recovery capabilities, we had KSU's technical support team set up regular restore points and assist with any rollbacks that were needed and that may be needed in the future. Additionally, we utilized rsync to backup important configuration and log files throughout the process so that any updates to the server could be manually reviewed and evaluated as needed (see Figure A16). Scheduled through cron, these backups run periodically to minimize data loss risks and support quick restoration if needed. By establishing a robust backup and disaster recovery protocol, we ensured that critical data and configurations could be restored in the event of an attack, system failure, or data corruption.

6. Logging and Monitoring

Effective logging and monitoring were critical to our security program's success. We used Wazuh Manager for centralized security information and event management (SIEM), which aggregates logs from various sources, including Apache, MariaDB, SSH, firewall, and WordPress logs. Wazuh's alerting capabilities allow us to identify and respond to potential threats in real time. Cockpit was also utilized to monitor system health, including CPU, memory, and disk usage, along with log management (see Figure A17). This comprehensive logging and monitoring setup provides a detailed audit trail, enabling incident detection, forensic analysis, and continuous security assessment.

Conclusion

In conclusion, our security program successfully established a multi-layered defense to protect our server from cyber threats and unauthorized access. By restricting administrative privileges, implementing regular updates, and strengthening authentication with multi-factor verification, we minimized potential vulnerabilities. Network security measures, such as Snort, ModSecurity, and Wordfence, enhanced our system's resilience against intrusion attempts. Additionally, our backup and disaster recovery strategy ensures data integrity and availability, while centralized logging and monitoring with Wazuh provides real-time insights and facilitates timely responses to security incidents. Altogether, these security measures create a robust, proactive approach to maintaining server security and operational reliability.

Strategies and Tools Used for Penetration Testing

To thoroughly assess our server’s security posture, we designed a comprehensive vulnerability analysis and penetration testing strategy. This approach first begins with target reconnaissance and then employs a diverse set of tools selected to exploit potential weaknesses across several critical attack vectors, including SSH, Cockpit, WordPress, SQL, and hashing and encryption attacks. By leveraging custom username lists, password lists, and recovery email lists tailored to the class environment, we aim to further enhance the strength of our attack and simulate realistic attack scenarios that may identify areas for security improvement.

1. Reconnaissance

Effective penetration testing begins with comprehensive reconnaissance, gathering as much information about the target system as possible. To begin, we created a custom phishing email that will be sent to the whole class requesting any updated usernames and passwords An example this email is shown in Figure B1 in Appendix B, where the rest of the pictures in the strategies and tools phase of this document will reside. We then used tools such as nmap, dirb, gobuster, ffuf, nikto, wpscan, sslscan, openssl, and curl for the remainder of this phase. nmap provides detailed insights into open ports and service versions, laying the groundwork for identifying vulnerable entry points (see Figure B2). dirb, gobuster, and ffuf are directory brute-force tools that help locate hidden files or directories, potentially revealing sensitive information or unprotected entry points (see Figure B3). Nikto scans for known vulnerabilities in web servers, while wpscan is specialized for WordPress vulnerabilities, identifying outdated plugins or weak admin credentials. sslscan and openssl allow for analysis of SSL/TLS configurations, uncovering encryption weaknesses that could be exploited in man-in-the-middle (MITM) attacks (see Figure B4). Finally, using curl to interact with the WordPress API can expose configuration details useful in other attack stages. These tools collectively create a detailed map of system vulnerabilities, forming a foundation for targeted attacks.

2. SSH Access

SSH remains a critical access point, often targeted for unauthorized entry. We utilized ssh and hydra to test the robustness of SSH credentials. Hydra is a powerful tool for brute-forcing login credentials across various protocols, including SSH (see Figure B5). Using custom password lists, we simulated dictionary attacks to test password strength and identify accounts with weak or common passwords. These SSH tests highlight the importance of strong authentication, and testing with hydra allows us to ensure passwords meet security standards, reducing the risk of unauthorized access.

3. Cockpit Access

Given that Cockpit provides a web-based graphical interface for managing system resources, securing its login is critical. We used hydra again to perform brute-force attacks on Cockpit, leveraging our custom password lists. This ensures that Cockpit’s authentication mechanisms can withstand dictionary attacks and are not susceptible to weak passwords. By testing login resilience, we can ensure that only authorized users access this sensitive interface, protecting critical system controls.

4. WordPress

As a high-risk application, WordPress was tested extensively using curl, hydra, and Burp Suite (see Figure B6). Curl allows for command-line interactions with WordPress APIs, exposing sensitive endpoints and configurations. Hydra was employed to test the strength of WordPress login credentials, while Burp Suite was used to intercept and manipulate HTTP requests, allowing for vulnerability discovery such as cross-site scripting (XSS) and SQL injection points. Burp Suite's flexibility in altering HTTP requests is invaluable in testing web applications for common vulnerabilities. These tests are essential to assess the security of our WordPress installation, which is often a primary target due to its popularity and potential misconfigurations.

5. MariaDB

Our MariaDB attack strategy involved using MySQL and hydra through an SSH shell. By executing SQL commands directly in MySQL, we could test for potential SQL injection points, misconfigurations, or weak database accounts. Using hydra through SSH allowed us to perform brute-force attacks on database credentials, testing the strength of MySQL user accounts. This two-pronged approach identifies risks in database security, helping us mitigate unauthorized access or manipulation of sensitive data.

6. Hashes and Encryption

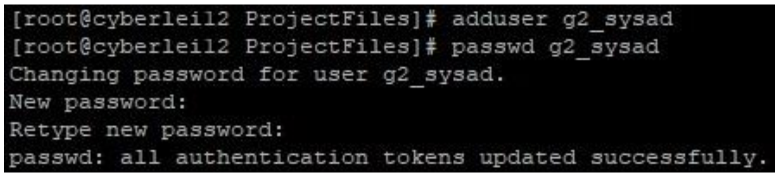
Finally, for testing password hashes and encryption strength, we selected John the Ripper. This tool is well-suited for cracking password hashes due to its extensive wordlist capabilities and customization options. Using John the Ripper allowed us to test the resilience of hashed passwords in the target server and in the MariaDB database, ensuring encryption standards are met to prevent unauthorized access even if the hashes are compromised (see Figure B7). This testing step validates the strength of our hashing algorithms and helps us enforce strong password policies.

Conclusion

In conclusion, our penetration testing process applied a robust suite of tools and strategies to thoroughly examine the security posture of our server across multiple attack surfaces. From initial reconnaissance with tools like nmap, dirb, and wpscan to specific, targeted testing on SSH, Cockpit, WordPress, and MariaDB, each phase was designed to uncover potential vulnerabilities that may compromise the system's integrity. The use of Hydra for brute-force testing and Burp Suite for web application vulnerabilities enabled us to simulate real-world attack scenarios, revealing areas where stronger authentication and configuration improvements are necessary. Our tests with John the Ripper underscored the need for secure hashing standards to protect sensitive data against unauthorized access. By proactively identifying and addressing these weaknesses, our strategy reinforces the server's defenses and enhances its resilience against potential cyber threats. This multi-faceted approach not only strengthens our current security posture but also establishes a foundation for ongoing vigilance and improvement in our cybersecurity practices. Through these assessments, we gain invaluable insights into maintaining a secure environment that effectively protects sensitive information and critical system resources.

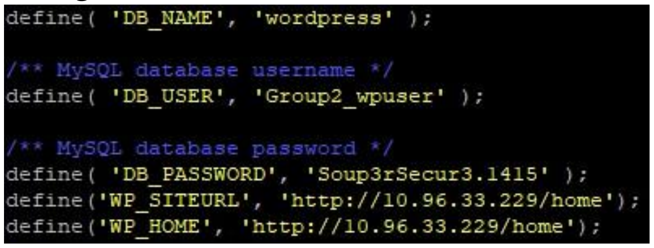
Appendix A: Implementation Pictures

Figure A1
Create non-root user



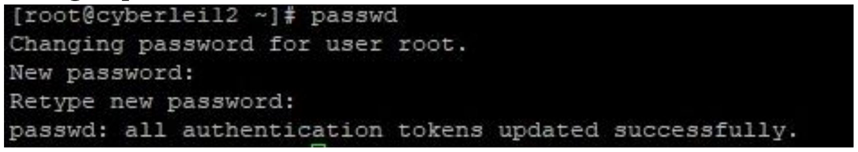
```
[root@cyberleil2 ProjectFiles]# adduser g2_sysad
[root@cyberleil2 ProjectFiles]# passwd g2_sysad
Changing password for user g2_sysad.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Figure A2
Setting WordPress to use the non-admin use



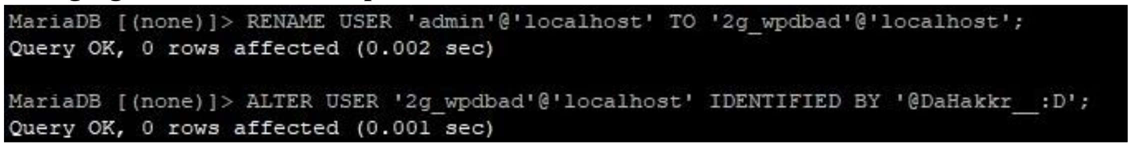
```
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'Group2_wpuser' );

/** MySQL database password */
define( 'DB_PASSWORD', 'Soup3rSecur3.1415' );
define('WP_SITEURL', 'http://10.96.33.229/home');
define('WP_HOME', 'http://10.96.33.229/home');
```

Figure A3
Changed password for root



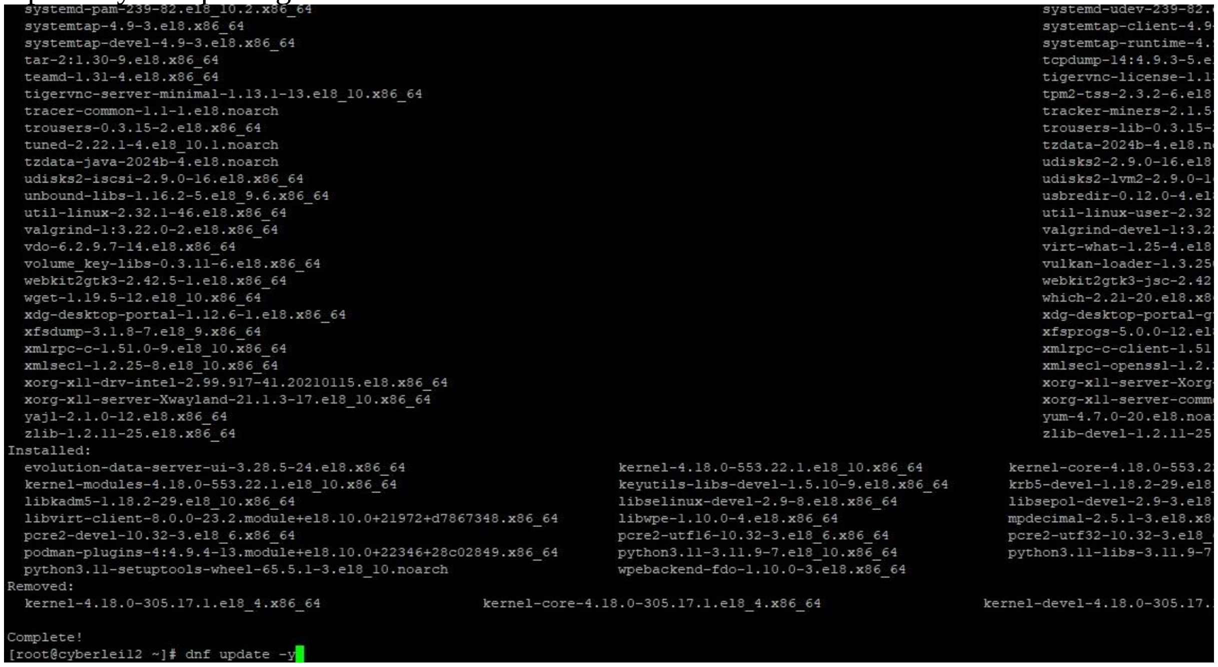
```
[root@cyberleil2 ~]# passwd
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Figure A4
Changing the username and password of the MariaDB admin user



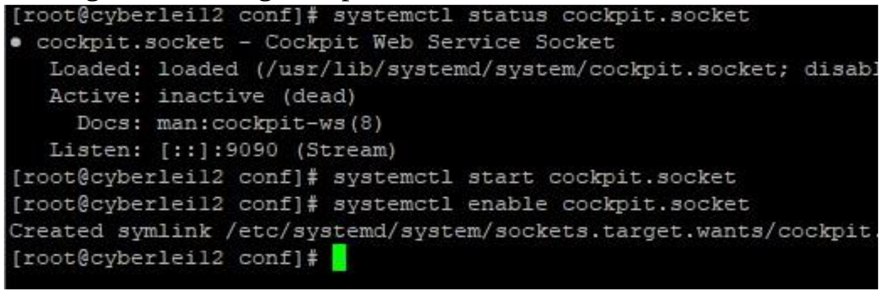
```
MariaDB [(none)]> RENAME USER 'admin'@'localhost' TO '2g_wpdbad'@'localhost';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> ALTER USER '2g_wpdbad'@'localhost' IDENTIFIED BY '@DaHakkr__:D';
Query OK, 0 rows affected (0.001 sec)
```

Figure A5
Update system packages



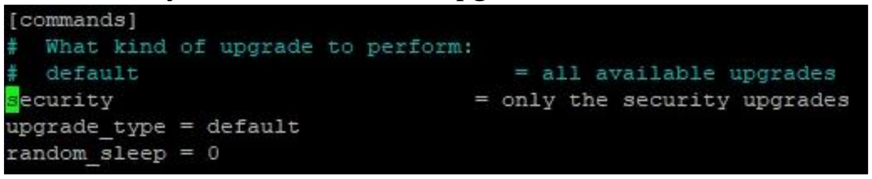
```
  systemd-pam-239-82.el8_10.2.x86_64                                                                                      systemd-udev-239-82.
  systemtap-4.9-3.el8.x86_64                                                                                              systemtap-client-4.9
  systemtap-devel-4.9-3.el8.x86_64                                                                                        systemtap-runtime-4.
  tar-2:1.30-9.el8.x86_64                                                                                                 tcpdump-14:4.9.3-5.e
  teamd-1.31-4.el8.x86_64                                                                                                 tigervnc-license-1.1
  tigervnc-server-minimal-1.13.1-13.el8_10.x86_64                                                                         tpm2-tss-2.3.2-6.el8
  tracer-common-1.1-1.el8.noarch                                                                                          tracker-miners-2.1.5
  trousers-0.3.15-2.el8.x86_64                                                                                            trousers-lib-0.3.15-
  tuned-2.22.1-4.el8_10.1.noarch                                                                                          tzdata-2024b-4.el8.n
  tzdata-java-2024b-4.el8.noarch                                                                                          udisks2-2.9.0-16.el8
  udisks2-iscsi-2.9.0-16.el8.x86_64                                                                                       udisks2-lvm2-2.9.0-1
  unbound-libs-1.16.2-5.el8_9.6.x86_64                                                                                    usbredir-0.12.0-4.el
  util-linux-2.32.1-46.el8.x86_64                                                                                         util-linux-user-2.32
  valgrind-1:3.22.0-2.el8.x86_64                                                                                          valgrind-devel-1:3.2
  vdo-6.2.9.7-14.el8.x86_64                                                                                               virt-what-1.25-4.el8
  volume_key-libs-0.3.11-6.el8.x86_64                                                                                     vulkan-loader-1.3.25
  webkit2gtk3-2.42.5-1.el8.x86_64                                                                                         webkit2gtk3-jsc-2.42
  wget-1.19.5-12.el8_10.x86_64                                                                                            which-2.21-20.el8.x8
  xdg-desktop-portal-1.12.6-1.el8.x86_64                                                                                  xdg-desktop-portal-g
  xfsdump-3.1.8-7.el8_9.x86_64                                                                                            xfsprogs-5.0.0-12.el
  xmlrpc-c-1.51.0-9.el8_10.x86_64                                                                                         xmlrpc-c-client-1.51
  xmlsec1-1.2.25-8.el8_10.x86_64                                                                                          xmlsec1-openssl-1.2.
  xorg-x11-drv-intel-2.99.917-41.20210115.el8.x86_64                                                                      xorg-x11-server-Xorg
  xorg-x11-server-Xwayland-21.1.3-17.el8_10.x86_64                                                                        xorg-x11-server-comm
  yajl-2.1.0-12.el8.x86_64                                                                                                yum-4.7.0-20.el8.noa
  zlib-1.2.11-25.el8.x86_64                                                                                               zlib-devel-1.2.11-25
Installed:
  evolution-data-server-ui-3.28.5-24.el8.x86_64                         kernel-4.18.0-553.22.1.el8_10.x86_64            kernel-core-4.18.0-553.2
  kernel-modules-4.18.0-553.22.1.el8_10.x86_64                          keyutils-libs-devel-1.5.10-9.el8.x86_64         krb5-devel-1.18.2-29.el8
  libkadm5-1.18.2-29.el8_10.x86_64                                      libselinux-devel-2.9-8.el8.x86_64               libsepol-devel-2.9-3.el8
  libvirt-client-8.0.0-23.2.module+el8.10.0+21972+d7867348.x86_64       libwpe-1.10.0-4.el8.x86_64                      mpdecimal-2.5.1-3.el8.x8
  pcre2-devel-10.32-3.el8_6.x86_64                                      pcre2-utf16-10.32-3.el8_6.x86_64                pcre2-utf32-10.32-3.el8_
  podman-plugins-4:4.9.4-13.module+el8.10.0+22346+28c02849.x86_64       python3.11-3.11.9-7.el8_10.x86_64               python3.11-libs-3.11.9-7
  python3.11-setuptools-wheel-65.5.1-3.el8_10.noarch                    wpebackend-fdo-1.10.0-3.el8.x86_64
Removed:
  kernel-4.18.0-305.17.1.el8_4.x86_64                kernel-core-4.18.0-305.17.1.el8_4.x86_64                kernel-devel-4.18.0-305.17.

Complete!
[root@cyberleil2 ~]# dnf update -y
```

Figure A6
Starting and enabling cockpit



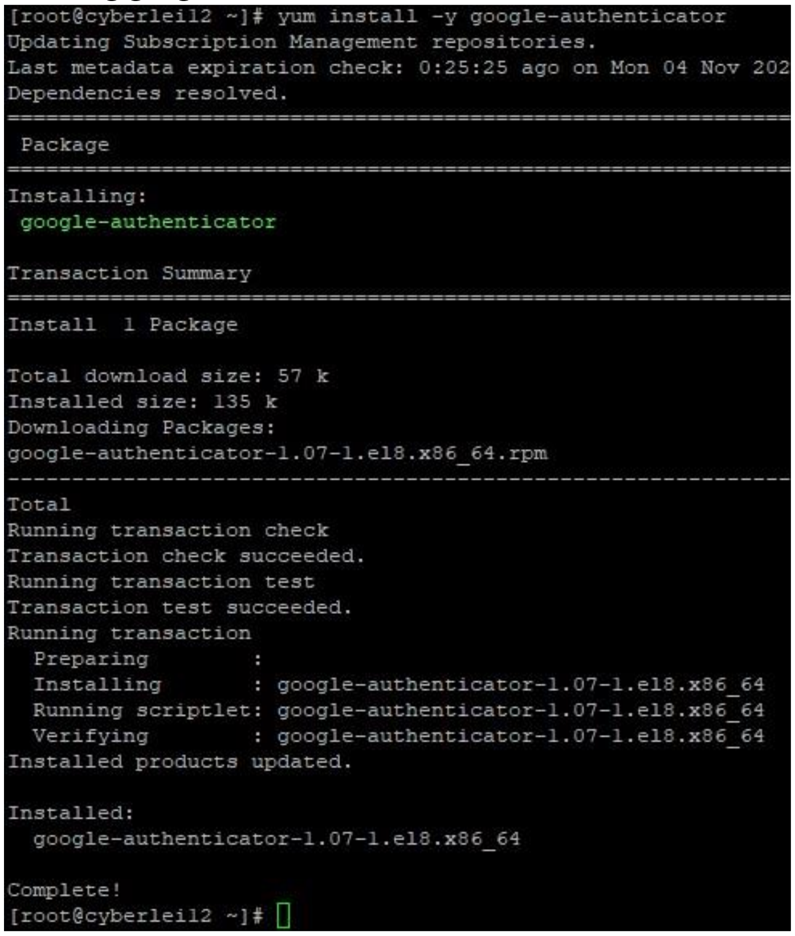
```
[root@cyberleil2 conf]# systemctl status cockpit.socket
● cockpit.socket - Cockpit Web Service Socket
   Loaded: loaded (/usr/lib/systemd/system/cockpit.socket; disabl
   Active: inactive (dead)
     Docs: man:cockpit-ws(8)
   Listen: [::]:9090 (Stream)
[root@cyberleil2 conf]# systemctl start cockpit.socket
[root@cyberleil2 conf]# systemctl enable cockpit.socket
Created symlink /etc/systemd/system/sockets.target.wants/cockpit.
[root@cyberleil2 conf]#
```

Figure A7
Automatically install all available upgrades



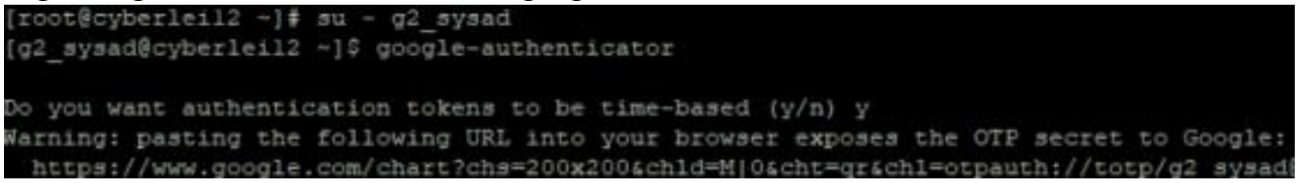
```
[commands]
#  What kind of upgrade to perform:
#  default                            = all available upgrades
security                           = only the security upgrades
upgrade_type = default
random_sleep = 0
```

Figure A8
Installing google-authenticator



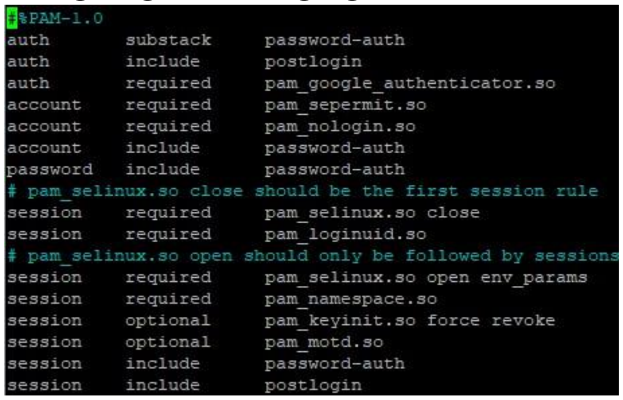
```
[root@cyberleil2 ~]# yum install -y google-authenticator
Updating Subscription Management repositories.
Last metadata expiration check: 0:25:25 ago on Mon 04 Nov 202
Dependencies resolved.
=============================================================
 Package
=============================================================
Installing:
 google-authenticator

Transaction Summary
=============================================================
Install  1 Package

Total download size: 57 k
Installed size: 135 k
Downloading Packages:
google-authenticator-1.07-1.el8.x86_64.rpm
-------------------------------------------------------------
Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :
  Installing       : google-authenticator-1.07-1.el8.x86_64
  Running scriptlet: google-authenticator-1.07-1.el8.x86_64
  Verifying        : google-authenticator-1.07-1.el8.x86_64
Installed products updated.

Installed:
  google-authenticator-1.07-1.el8.x86_64

Complete!
[root@cyberleil2 ~]#
```

Figure A9
Registering the non-admin account with google-authenticator



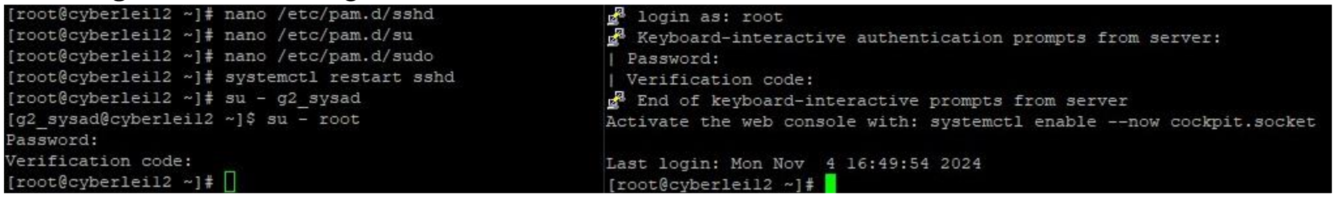
```
[root@cyberleil2 ~]# su - g2_sysad
[g2_sysad@cyberleil2 ~]$ google-authenticator

Do you want authentication tokens to be time-based (y/n) y
Warning: pasting the following URL into your browser exposes the OTP secret to Google:
  https://www.google.com/chart?chs=200x200&chld=M|0&cht=qr&chl=otpauth://totp/g2_sysad
```

Figure A10
Configuring ssh to use google-authenticator



```
#%PAM-1.0
auth       substack     password-auth
auth       include      postlogin
auth       required     pam_google_authenticator.so
account    required     pam_sepermit.so
account    required     pam_nologin.so
account    include      password-auth
password   include      password-auth
# pam_selinux.so close should be the first session rule
session    required     pam_selinux.so close
session    required     pam_loginuid.so
# pam_selinux.so open should only be followed by sessions
session    required     pam_selinux.so open env_params
session    required     pam_namespace.so
session    optional     pam_keyinit.so force revoke
session    optional     pam_motd.so
session    include      password-auth
session    include      postlogin
```

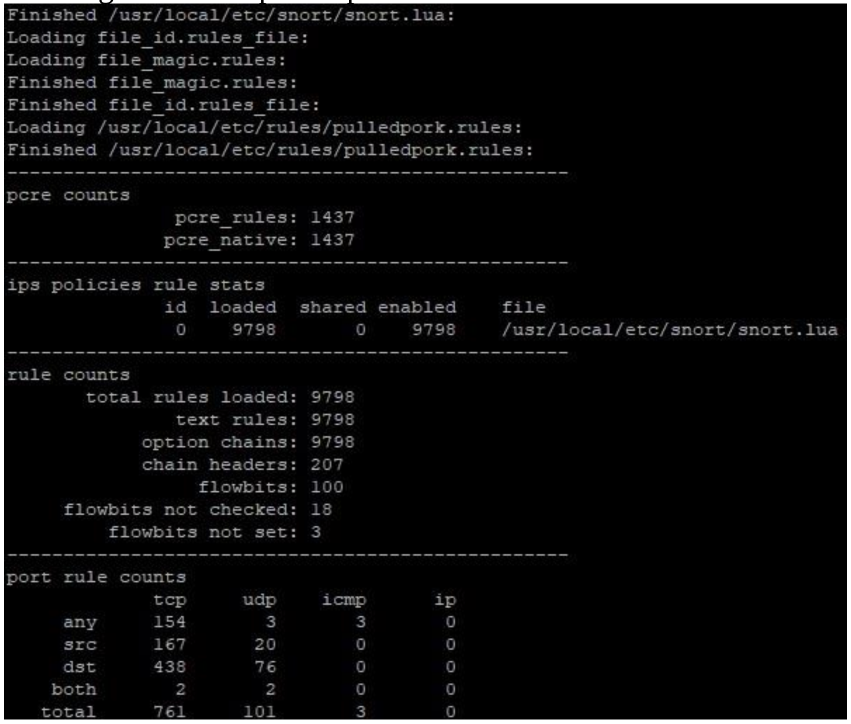
Figure A11
Enforcing MFA when using "su" commands



```
[root@cyberleil2 ~]# nano /etc/pam.d/sshd
[root@cyberleil2 ~]# nano /etc/pam.d/su
[root@cyberleil2 ~]# nano /etc/pam.d/sudo
[root@cyberleil2 ~]# systemctl restart sshd
[root@cyberleil2 ~]# su - g2_sysad
[g2_sysad@cyberleil2 ~]$ su - root
Password:
Verification code:
[root@cyberleil2 ~]#
```

```
login as: root
Keyboard-interactive authentication prompts from server:
| Password:
| Verification code:
End of keyboard-interactive prompts from server
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Mon Nov  4 16:49:54 2024
[root@cyberleil2 ~]#
```

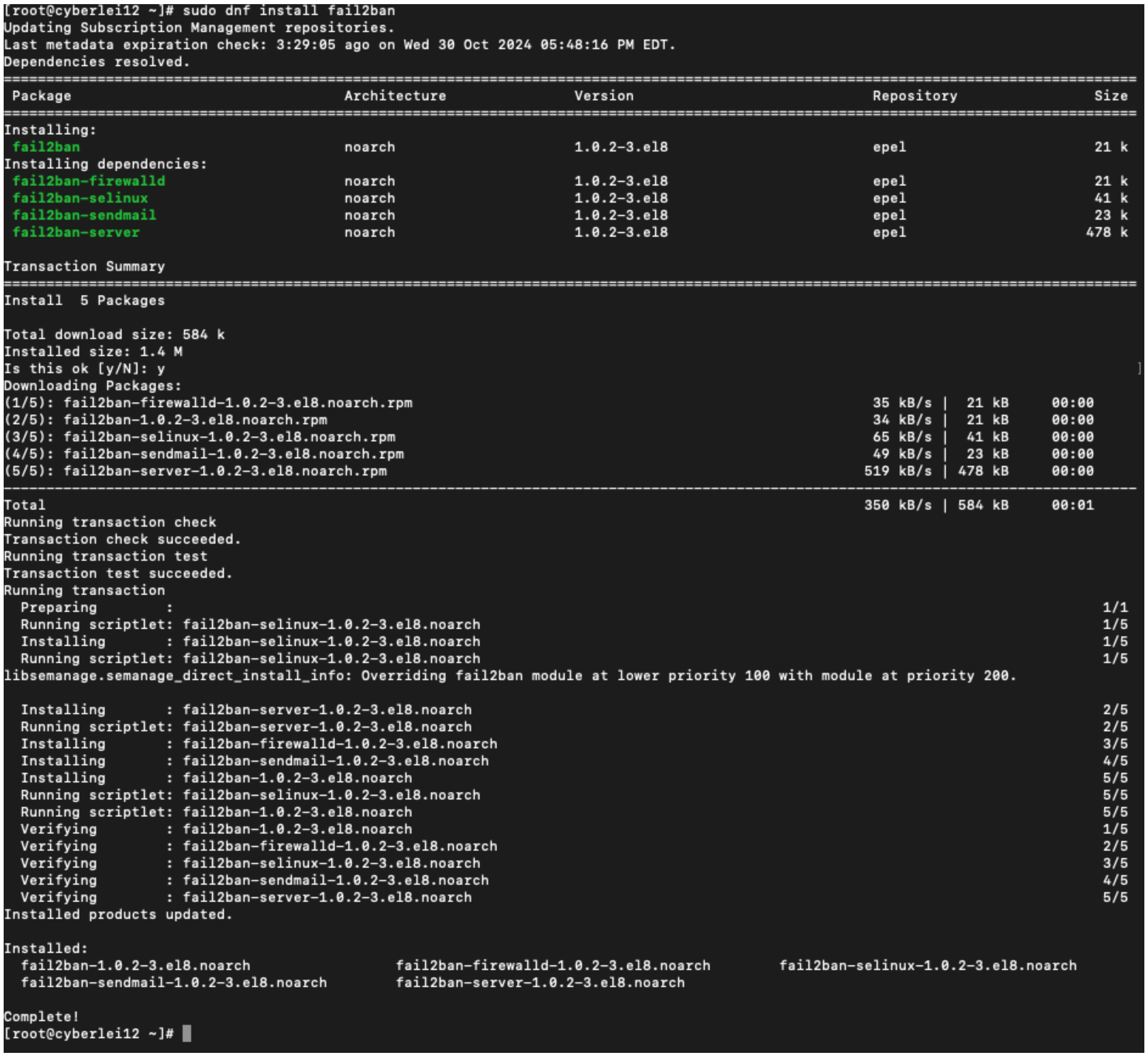
Figure A12
Running snort with pulled pork



```
Finished /usr/local/etc/snort/snort.lua:
Loading file_id.rules_file:
Loading file_magic.rules:
Finished file_magic.rules:
Finished file_id.rules_file:
Loading /usr/local/etc/rules/pulledpork.rules:
Finished /usr/local/etc/rules/pulledpork.rules:
--------------------------------------------------
pcre counts
               pcre_rules: 1437
              pcre_native: 1437
--------------------------------------------------
ips policies rule stats
              id  loaded  shared enabled    file
               0    9798       0    9798    /usr/local/etc/snort/snort.lua
--------------------------------------------------
rule counts
       total rules loaded: 9798
               text rules: 9798
            option chains: 9798
            chain headers: 207
                 flowbits: 100
     flowbits not checked: 18
         flowbits not set: 3
--------------------------------------------------
port rule counts
             tcp     udp    icmp      ip
     any     154       3       3       0
     src     167      20       0       0
     dst     438      76       0       0
    both       2       2       0       0
   total     761     101       3       0
```

Figure A13

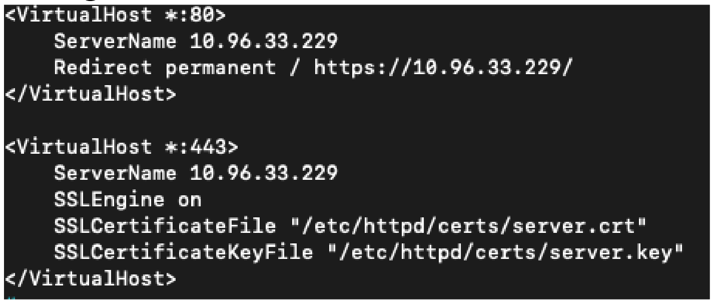
Installed fail2ban



```
[root@cyberlei12 ~]# sudo dnf install fail2ban
Updating Subscription Management repositories.
Last metadata expiration check: 3:29:05 ago on Wed 30 Oct 2024 05:48:16 PM EDT.
Dependencies resolved.
=====================================================================================================================================
 Package                                  Architecture                 Version                             Repository            Size
=====================================================================================================================================
Installing:
 fail2ban                                 noarch                       1.0.2-3.el8                         epel                  21 k
Installing dependencies:
 fail2ban-firewalld                       noarch                       1.0.2-3.el8                         epel                  21 k
 fail2ban-selinux                         noarch                       1.0.2-3.el8                         epel                  41 k
 fail2ban-sendmail                        noarch                       1.0.2-3.el8                         epel                  23 k
 fail2ban-server                          noarch                       1.0.2-3.el8                         epel                 478 k

Transaction Summary
=====================================================================================================================================
Install  5 Packages

Total download size: 584 k
Installed size: 1.4 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): fail2ban-firewalld-1.0.2-3.el8.noarch.rpm                                                     35 kB/s |  21 kB     00:00
(2/5): fail2ban-1.0.2-3.el8.noarch.rpm                                                               34 kB/s |  21 kB     00:00
(3/5): fail2ban-selinux-1.0.2-3.el8.noarch.rpm                                                       65 kB/s |  41 kB     00:00
(4/5): fail2ban-sendmail-1.0.2-3.el8.noarch.rpm                                                      49 kB/s |  23 kB     00:00
(5/5): fail2ban-server-1.0.2-3.el8.noarch.rpm                                                       519 kB/s | 478 kB     00:00
-------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                               350 kB/s | 584 kB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                            1/1
  Running scriptlet: fail2ban-selinux-1.0.2-3.el8.noarch                                                                        1/5
  Installing       : fail2ban-selinux-1.0.2-3.el8.noarch                                                                        1/5
  Running scriptlet: fail2ban-selinux-1.0.2-3.el8.noarch                                                                        1/5
libsemanage.semanage_direct_install_info: Overriding fail2ban module at lower priority 100 with module at priority 200.

  Installing       : fail2ban-server-1.0.2-3.el8.noarch                                                                         2/5
  Running scriptlet: fail2ban-server-1.0.2-3.el8.noarch                                                                         2/5
  Installing       : fail2ban-firewalld-1.0.2-3.el8.noarch                                                                      3/5
  Installing       : fail2ban-sendmail-1.0.2-3.el8.noarch                                                                       4/5
  Installing       : fail2ban-1.0.2-3.el8.noarch                                                                                5/5
  Running scriptlet: fail2ban-selinux-1.0.2-3.el8.noarch                                                                        5/5
  Running scriptlet: fail2ban-1.0.2-3.el8.noarch                                                                                5/5
  Verifying        : fail2ban-1.0.2-3.el8.noarch                                                                                1/5
  Verifying        : fail2ban-firewalld-1.0.2-3.el8.noarch                                                                      2/5
  Verifying        : fail2ban-selinux-1.0.2-3.el8.noarch                                                                        3/5
  Verifying        : fail2ban-sendmail-1.0.2-3.el8.noarch                                                                       4/5
  Verifying        : fail2ban-server-1.0.2-3.el8.noarch                                                                         5/5
Installed products updated.

Installed:
  fail2ban-1.0.2-3.el8.noarch                    fail2ban-firewalld-1.0.2-3.el8.noarch        fail2ban-selinux-1.0.2-3.el8.noarch
  fail2ban-sendmail-1.0.2-3.el8.noarch           fail2ban-server-1.0.2-3.el8.noarch

Complete!
[root@cyberlei12 ~]#
```

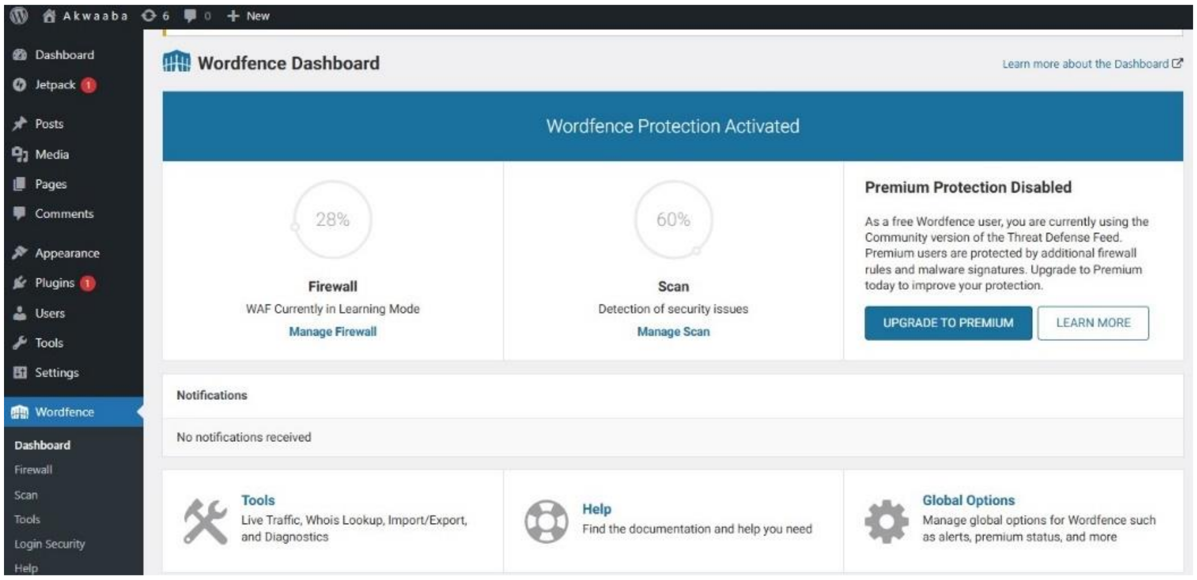
Figure A14

Forcing HTTP to HTTPS redirects



```
<VirtualHost *:80>
    ServerName 10.96.33.229
    Redirect permanent / https://10.96.33.229/
</VirtualHost>

<VirtualHost *:443>
    ServerName 10.96.33.229
    SSLEngine on
    SSLCertificateFile "/etc/httpd/certs/server.crt"
    SSLCertificateKeyFile "/etc/httpd/certs/server.key"
</VirtualHost>
```

Figure A15
Wordfence dashboard



Wordfence Dashboard

Wordfence Protection Activated

Firewall
WAF Currently in Learning Mode

Scan
Detection of security issues

Premium Protection Disabled

As a free Wordfence user, you are currently using the Community version of the Threat Defense Feed. Premium users are protected by additional firewall rules and malware signatures. Upgrade to Premium today to improve your protection.

Notifications

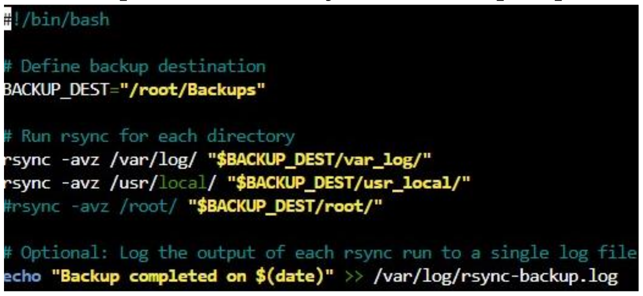
No notifications received

Tools
Live Traffic, Whois Lookup, Import/Export, and Diagnostics

Help
Find the documentation and help you need

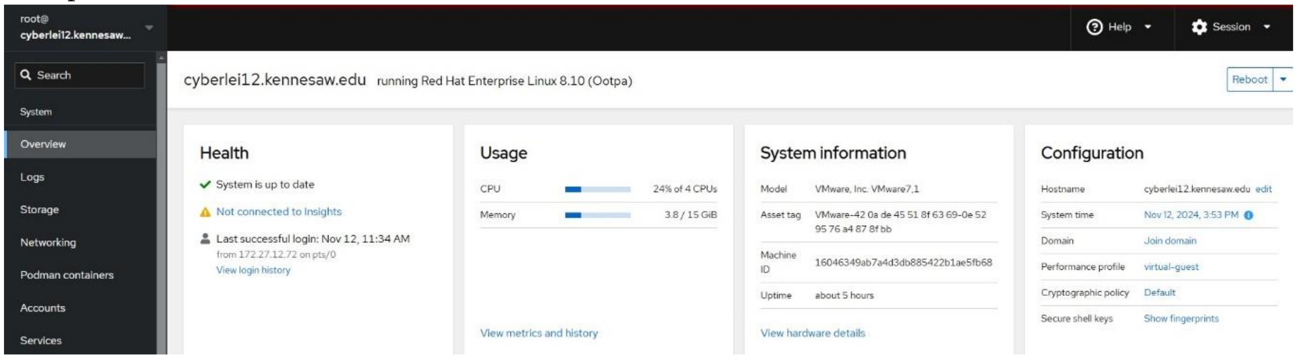
Global Options
Manage global options for Wordfence such as alerts, premium status, and more

Figure A16
Shell script which uses rsync to back up important logs and configuration files when run



```
#!/bin/bash

# Define backup destination
BACKUP_DEST="/root/Backups"

# Run rsync for each directory
rsync -avz /var/log/ "$BACKUP_DEST/var_log/"
rsync -avz /usr/local/ "$BACKUP_DEST/usr_local/"
#rsync -avz /root/ "$BACKUP_DEST/root/"

# Optional: Log the output of each rsync run to a single log file
echo "Backup completed on $(date)" >> /var/log/rsync-backup.log
```

Figure A17
Cockpit dashboard



cyberlei12.kennesaw.edu running Red Hat Enterprise Linux 8.10 (Ootpa)

Health
System is up to date
Not connected to Insights
Last successful login: Nov 12, 11:34 AM
from 172.27.12.72 on pts/0

Usage

CPU	24% of 4 CPUs
Memory	3.8 / 15 GiB

System information

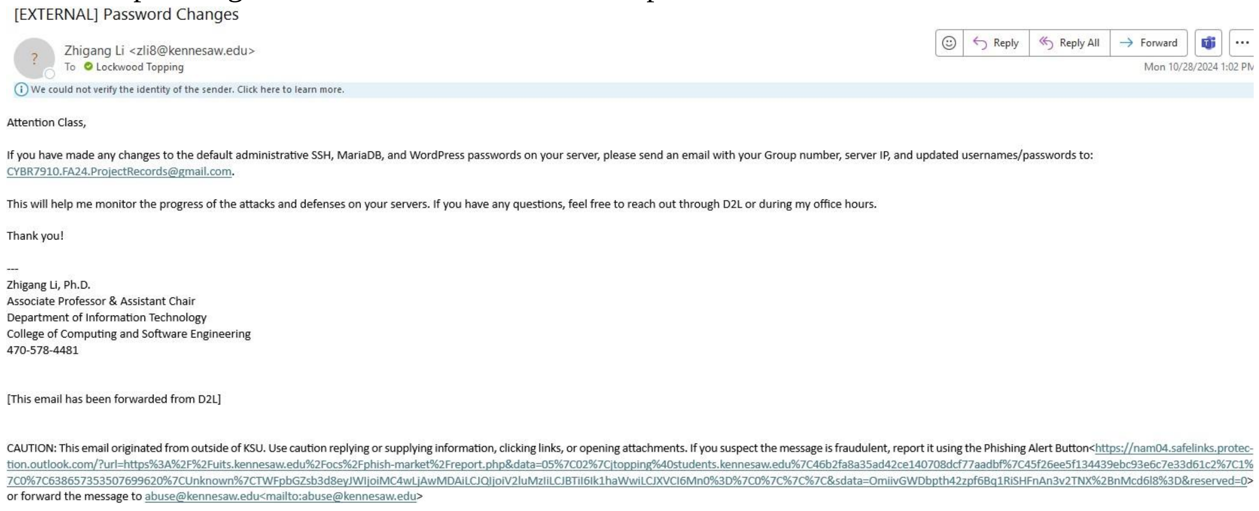
Model	VMware, Inc. VMware7,1
Asset tag	VMware-42 0a de 45 51 8f 63 69-0e 52 95 76 a4 87 8f bb
Machine ID	16046349ab7a4d3db885422b1ae5fb68
Uptime	about 5 hours

Configuration

Hostname	cyberlei12.kennesaw.edu edit
System time	Nov 12, 2024, 3:53 PM
Domain	Join domain
Performance profile	virtual-guest
Cryptographic policy	Default
Secure shell keys	Show fingerprints

Appendix B: Strategies and Tools Pictures

Figure B1
A custom phishing email to solicit usernames and passwords



[EXTERNAL] Password Changes

Zhigang Li <zli8@kennesaw.edu>
To Lockwood Topping
Mon 10/28/2024 1:02 PM

We could not verify the identity of the sender. Click here to learn more.

Attention Class,

If you have made any changes to the default administrative SSH, MariaDB, and WordPress passwords on your server, please send an email with your Group number, server IP, and updated usernames/passwords to: CYBR7910.FA24.ProjectRecords@gmail.com.

This will help me monitor the progress of the attacks and defenses on your servers. If you have any questions, feel free to reach out through D2L or during my office hours.

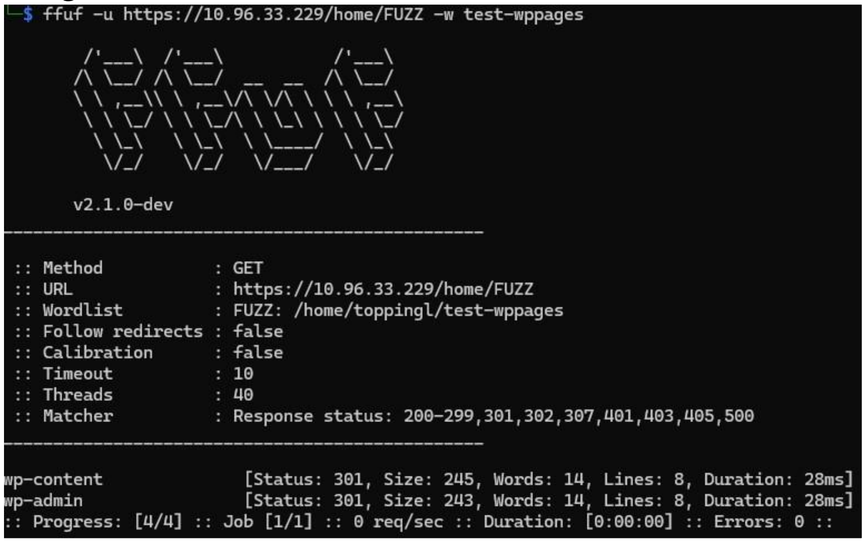
Thank you!

Zhigang Li, Ph.D.
Associate Professor & Assistant Chair
Department of Information Technology
College of Computing and Software Engineering
470-578-4481

[This email has been forwarded from D2L]

CAUTION: This email originated from outside of KSU. Use caution replying or supplying information, clicking links, or opening attachments. If you suspect the message is fraudulent, report it using the Phishing Alert Button<https://nam04.safelinks.protection.outlook.com/?url=https%3A%2F%2Fuits.kennesaw.edu%2Focs%2Fphish-market%2Freport.php&data=05%7C02%7Cjtopping%40students.kennesaw.edu%7C46b2fa8a35ad42ce140708dcf77aadbf%7C45f26ee5f134439ebc93e6c7e33d61c2%7C1%7C0%7C638657353507699620%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C0%7C%7C%7C&sdata=OmiivGWDbpth42zpf6Bq1RiSHFnAn3v2TNX%2BnMcd6l8%3D&reserved=0> or forward the message to abuse@kennesaw.edu<mailto:abuse@kennesaw.edu>

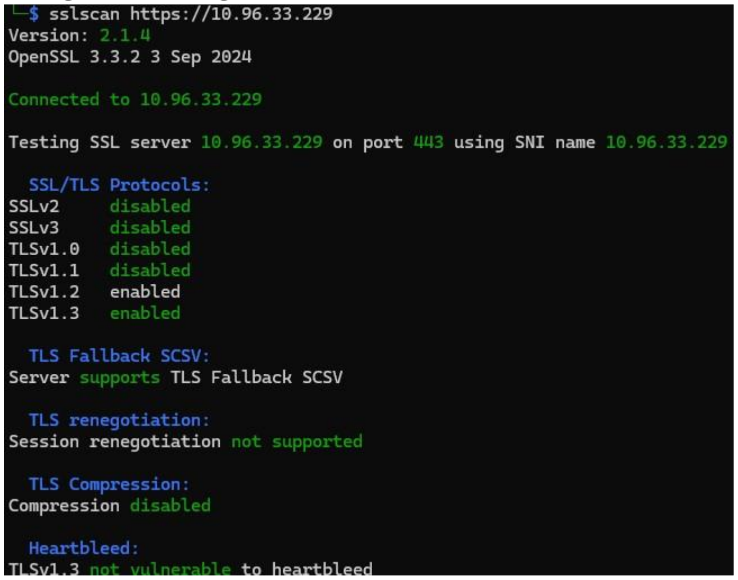
Figure B2
Using ffuf to find hidden directories on the web server



```
└─$ ffuf -u https://10.96.33.229/home/FUZZ -w test-wppages

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : https://10.96.33.229/home/FUZZ
 :: Wordlist         : FUZZ: /home/toppingl/test-wppages
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

wp-content              [Status: 301, Size: 245, Words: 14, Lines: 8, Duration: 28ms]
wp-admin                [Status: 301, Size: 243, Words: 14, Lines: 8, Duration: 28ms]
:: Progress: [4/4] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors: 0 ::
```

Figure B3

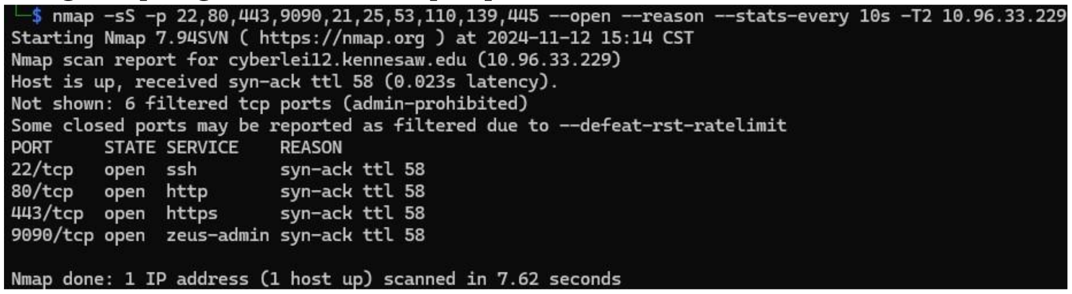
Using sslscan to get details about an ssl certificate for an HTTPS site



```
└─$ sslscan https://10.96.33.229
Version: 2.1.4
OpenSSL 3.3.2 3 Sep 2024

Connected to 10.96.33.229

Testing SSL server 10.96.33.229 on port 443 using SNI name 10.96.33.229

  SSL/TLS Protocols:
SSLv2     disabled
SSLv3     disabled
TLSv1.0   disabled
TLSv1.1   disabled
TLSv1.2   enabled
TLSv1.3   enabled

  TLS Fallback SCSV:
Server supports TLS Fallback SCSV

  TLS renegotiation:
Session renegotiation not supported

  TLS Compression:
Compression disabled

  Heartbleed:
TLSv1.3 not vulnerable to heartbleed
```

Figure B4

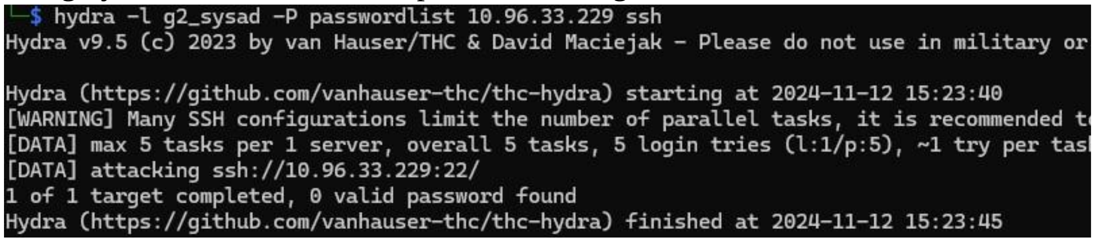
Using nmap to get details about open ports and services



```
└─$ nmap -sS -p 22,80,443,9090,21,25,53,110,139,445 --open --reason --stats-every 10s -T2 10.96.33.229
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-12 15:14 CST
Nmap scan report for cyberlei12.kennesaw.edu (10.96.33.229)
Host is up, received syn-ack ttl 58 (0.023s latency).
Not shown: 6 filtered tcp ports (admin-prohibited)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT     STATE SERVICE    REASON
22/tcp   open  ssh        syn-ack ttl 58
80/tcp   open  http       syn-ack ttl 58
443/tcp  open  https      syn-ack ttl 58
9090/tcp open  zeus-admin syn-ack ttl 58

Nmap done: 1 IP address (1 host up) scanned in 7.62 seconds
```

Figure B5

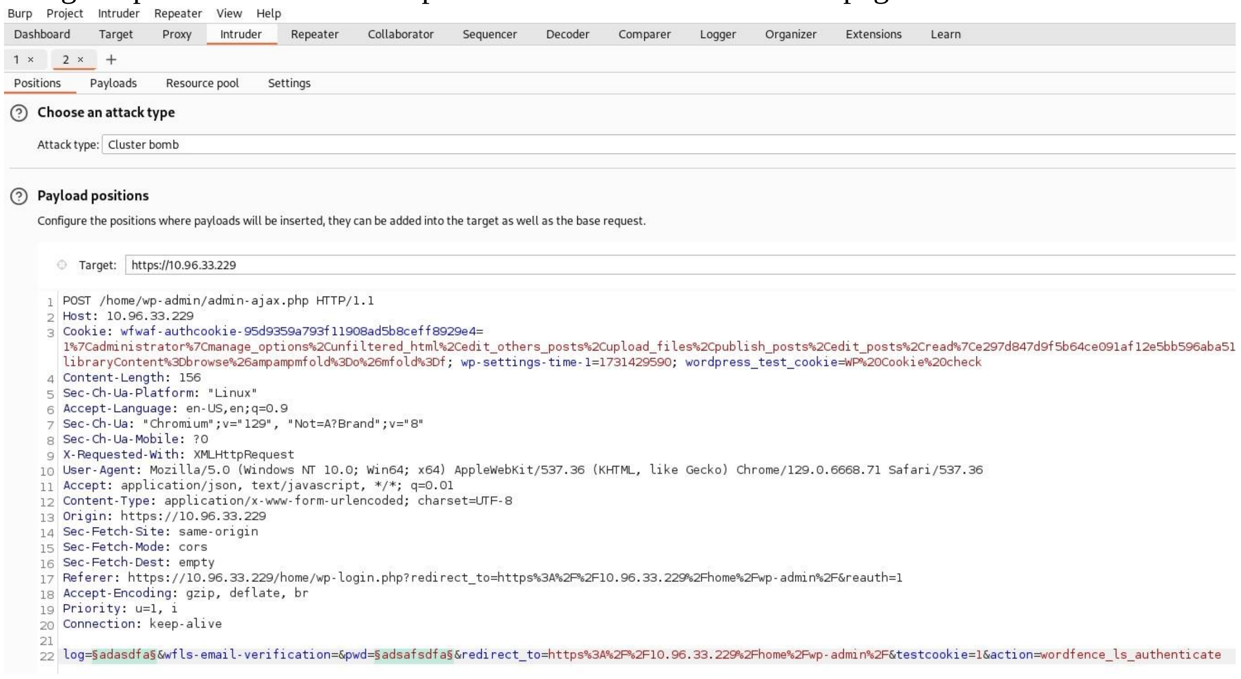
Using hydra to brute force a ssh passwords using a custom wordlist



```
└─$ hydra -l g2_sysad -P passwordlist 10.96.33.229 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-12 15:23:40
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended t
[DATA] max 5 tasks per 1 server, overall 5 tasks, 5 login tries (l:1/p:5), ~1 try per tas
[DATA] attacking ssh://10.96.33.229:22/
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-11-12 15:23:45
```

Figure B6

Using BurpSuite to send custom packets to the WordPress admin page


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
1 × 2 × +
Positions Payloads Resource pool Settings

Choose an attack type

Attack type: Cluster bomb

Payload positions

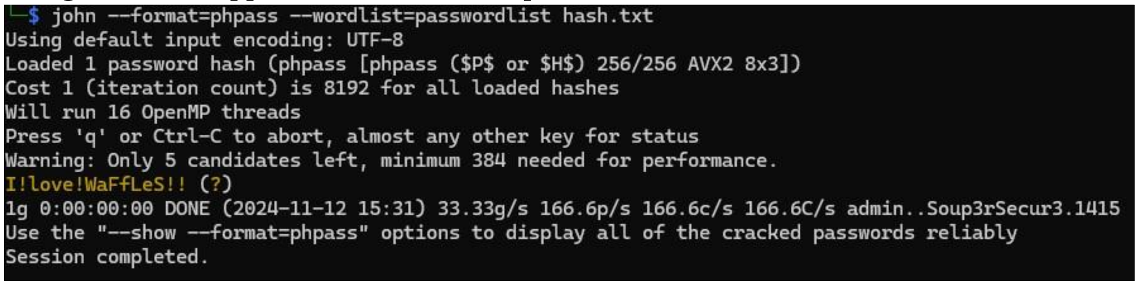
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: https://10.96.33.229

```
POST /home/wp-admin/admin-ajax.php HTTP/1.1
Host: 10.96.33.229
Cookie: wfwaf-authcookie-95d9359a793f11908ad5b8ceff8929e4=
1%7Cadministrator%7Cmanage_options%2Cunfiltered_html%2Cedit_others_posts%2Cupload_files%2Cpublish_posts%2Cedit_posts%2Cread%7Ce297d847d9f5b64ce091af12e5bb596aba51
libraryContent%3Dbrowse%26ampampmfold%3Do%26mfold%3Df; wp-settings-time-1=1731429590; wordpress_test_cookie=WP%20Cookie%20check
Content-Length: 156
Sec-Ch-Ua-Platform: "Linux"
Accept-Language: en-US,en;q=0.9
Sec-Ch-Ua: "Chromium";v="129", "Not=A?Brand";v="8"
Sec-Ch-Ua-Mobile: ?0
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6668.71 Safari/537.36
Accept: application/json, text/javascript, */*; q=0.01
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Origin: https://10.96.33.229
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://10.96.33.229/home/wp-login.php?redirect_to=https%3A%2F%2F10.96.33.229%2Fhome%2Fwp-admin%2F&reauth=1
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
Connection: keep-alive

log=§adasdfa§&wfls-email-verification=&pwd=§adsafsdfa§&redirect_to=https%3A%2F%2F10.96.33.229%2Fhome%2Fwp-admin%2F&testcookie=1&action=wordfence_ls_authenticate
```

Figure B7

Using John the Ripper to crack a hashed password



```
└─$ john --format=phpass --wordlist=passwordlist hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 256/256 AVX2 8x3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 16 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 5 candidates left, minimum 384 needed for performance.
I!love!WaFfLeS!! (?)
1g 0:00:00:00 DONE (2024-11-12 15:31) 33.33g/s 166.6p/s 166.6c/s 166.6C/s admin..Soup3rSecur3.1415
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed.
```