Cybersecurity: Buffer Overflow

Lezione 5: Case Studies e Laboratorio Pratico

Case Study: JPEG Overflow

- Librerie `libjpeg`, `gdiplus.dll`
- Overflow in commenti EXIF
- Basta aprire immagine malevola
- Citato da DilDog

Case Study: Windows DLL

- XP/2000: kernel32.dll no ASLR
- Uso di `WinExec`, `CreateProcess`
- IP offuscato con XOR (esempio DilDog)
- 112 byte exploit

Laboratorio Pratico

Setup Binari

```
# Senza protezioni
gcc -m32 -fno-stack-protector -z execstack -no-pie -o vuln vuln.c

# Con Stack Protector
gcc -m32 -fstack-protector -no-pie -o vuln_ssp vuln.c
```

Lab 1: Exploit Baseline

- Determinare offset con pattern
- Creare payload: NOP sled + shellcode
- Testare se shell ottenuta

Lab 2: Con DEP

```
gcc -m32 -fno-stack-protector -z noexecstack -no-pie -o vuln_dep vuln.c
```

- Provare return-to-libc
- Trovare `system` e stringa `"/bin/sh"`

Lab 3: Con ASLR

```
echo 2 > /proc/sys/kernel/randomize_va_space
```

- Usare `ROPgadget` per trovare gadget
- Costruire ROP chain

Lab 4: Offuscamento

Implementare XOR encoder per shellcode.

Testare senza/sotto NOP sled.

Strumenti

Strumento	Uso
GDB + PEDA/gef	Debugging
pwndbg	Estensioni GDB
ROPgadget	Cercare gadget
msfvenom	Generare shellcode
checksec	Vedere protezioni

Validazione Exploit

- Affidabilità >90%
- Testare con diverse lunghezze
- Usare NOP sled lunghi
- Verificare shell stabile

Difese Attuali

- Stack canaries forti
- ASLR pieno (PIE + librerie)
- CET (Control-flow Enforcement)
- Memory safe languages (Rust)

Conclusioni

- Buffer overflow classico ma ancora presente
- Fondamentale per pentester e defender
- Protezioni rendono exploit più complessi
- Abilità combinare ROP + info leak = expert

Domande?

Contatti: @pinolallo

Cybersecurity: Buffer Overflow – Pinolallo.com