Universiteti i Prishtinës “Hasan Prishtina”

**Fakulteti Inxhinierisë Elektrike dhe Kompjuterike**



**Dokumentim teknik i projektit**

#### Lënda: Siguria e Informacionit

### Titulli i projektit: sqlmap

##### Emri profesorit/Asistentit Emri & mbiemri studentëve / email adresa

|  |  |  |
| --- | --- | --- |
| Prof. Dr. Blerim REXHA PhD.c Mërgim H. HOTI | 1. Egzona Grashtica | egzone.grashtica@student.uni-pr.edu |
| 2. Leand Thaçi | [leand.thaci@student.uni-pr.edu](mailto:leand.thaci@student.uni-pr.edu) |
| 3. Rrezarta Sallauka | rrezarta.sallauka@student.uni-pr.edu |

Prishtinë, 2021

## Përmbajtja

[Abstrakti 3](#_TOC_250005)

1. [Hyrje 4](#_TOC_250004)
2. [Qëllimi i punimit 5](#_TOC_250003)
3. [Pjesa kryesore 6](#_TOC_250002)
4. [Konkluzione (apo përfundim) 7](#_TOC_250001)

[Referencat 8](#_TOC_250000)

## Tabela e figurave

[Figura 1 Aplikacioni DVWA 9](#_Toc91427371)

[Figura 2 Krijimi i databazës testuese në mënyrë automatike 9](#_Toc91427372)

[Figura 3 Mjedisi i cili është i cenushëm ndaj sql injection 10](#_Toc91427373)

[Figura 4 Përcjellja e requestit përmes Developer Tools 11](#_Toc91427374)

[Figura 5 Printimi i tabelave të databazave pas injektimit 12](#_Toc91427375)

[Figura 6 Printimi i skemës së databazës 13](#_Toc91427376)

[Figura 7 Printimi i skemës së tabelës users 14](#_Toc91427377)

[Figura 8 Printimi i të dhënave të tabelës users 15](#_Toc91427378)

[Figura 9 Tentimi jo i sukseshëm për dëshifrimin e fjalëkalimeve 16](#_Toc91427379)

[Figura 10 Forma për blind sql injection 17](#_Toc91427380)

[Figura 11 Analiza e kërkesës POST BSI 18](#_Toc91427381)

[Figura 12 Printimi i emrave të databazave BSI 19](#_Toc91427382)

[Figura 13 Printimi i tabelave BSI 20](#_Toc91427383)

[Figura 14 Printimi i të dhënave të tabelës users BSI 21](#_Toc91427384)

[Figura 15 Rezulatet e ekzekutimit të custom query 22](#_Toc91427385)

[Figura 16 Krijimi i sql-shell 23](#_Toc91427386)

[Figura 17 Ekzekutimi i queries përmes sql-shell 23](#_Toc91427387)

[Figura 18 Printimi i databazës aktuale 24](#_Toc91427388)

[Figura 19 Printimi i userit aktual të databazës 25](#_Toc91427389)

# Abstrakti

Zhvillimi i internetit dhe rritja e përdorimit të tij në masë shumë të madhe në shoqërinë njerëzore ka rrjedh në një pikë ku pothuajse çdo e dhënë, duke i përfshirë edhe të dhënat personale është e ruajtur në të. Për këtë arsye ka lindur nevoja e ruatjes me çdo kusht të tyre, për shkak të mundësisë së keqpërdorimit të tyre.

Siguria kibernetike është njëra nga çështejt më të rëndësishme të kohërave të fundit, për asye se shumë të dhëna janë të vendosura në botën digjitale dhe ruatja e tyre përbënë sfidë në vete. Organizatat në gjithë botën marrin sulme kibernetike çdo ditë. Sidomos organizatat të cilat mbajnë të dhëna delikate dhe me interes për përfitime të tjera. Informacioni është “mjet” me rëndësi të lartë, që shumë njerëz dëshitojnë ta kenë, arsye kjo për të qenë caku i sulmeve kibernetike. Andaj është punuar mjaftë shumë dhe po vazhdon të punohet në mbotjejn ndaj këtyre sulmeve, pasi që numri i tyre vazhdimisht është duke u rritur. Por njëkohësisht me rritje të sulmeve në botën e tekonologjisë janë zhvilluar shumë vegla si dhe softuer të ndryshëm për parandalim i tyre. Secili prej tyre mund të përdoret për të gjetur dobësitë e aplikacioneve të ndryshme. Njëri prej llojeve të sulmeve është injektimi SQL, përmes së cilit arrihet në merren të dhëna duke inkorporuar SQL query në aplikacionin e caktuar. Njëra nga veglat për detektim, testim dhe paradalim është Sqlmap, e cila do të disktutohet në këtë punim.

# Hyrje

Me rritjen e sulmeve çdo ditë e më shumë, atëherë rritet nevoja e sigurisë më të lart, me ç’rast interesimi për të pasur më shumë njohuri ndaj kësaj teme është i shtuar. Gjithashtu aftësia për të ditur dobësit të një produkti që je duke e realizuar ndihmon në përmisimin e tij. Sulmi SQL injection bëhet me insertimin e SQL query përmes futjes së të dhënave prej klientit në aplikacion

Llojet e dëmeve që mund të shkaktohen janë:

* Keqpërdorimi i të dhënave
* Humbje financiare
* Manipulimi i të dhënave në bazë e të dhënave
* Dëmtim i markës së biznesit
* Shitja e informatave dhe të dhënave tjera të konsumatorëve etj.
* Vjedhja e informacioneve

Arsyeja kryesore për përdorim të këtij mjeti është realizimi i parandalimeve dhe kontrllave të cilat mund të organizohen duke analizuar respektivisht kontrolluar çdo parametër hyrës. Forma të ndryshme të teknikave të kontrollave me AI duke bërë ndarjen e parametrave hyrës normal dhe ato që mund të jenë të dëmshme. Parametra hyërs të mos lejojnë hyrje të simboleve që përdoren në query dhe të kenë gjatësi të caktuar. Gjithashtu validimi i të dhënave dhe filtrimi i tyre.

Sqlmap është njëra nga mjetet open source të njohura për testime automatike në procesin e zbulimit dhe shfrytëzimit të dobësive apo defekteve që mund të ketë apo shkaktohen nga të ashtuquajtuara injektime SQL tek ndonjë Bazë e të dhënave apo një sistemi relativ të menaxhimit të bazës së të dhënave, atëherë është përzgjedhur ky mjet. Sqlmap ka një numër të madh të çelësave për çasje në marrjen e të dhënave nga baza e të dhënave nga baza të ndryshme të të dhënave e deri te hyrja në sisteme dhe ekzekutim të komandave në një sistem operativ.Automatizimi është një nga cilësit më të mira të kësaj vegle.

Sqlmap është i zhvilluar në gjuhën programuese Python, që e bënë këtë vegël të pavarur nga sistemi operativ.

# Qëllimi i punimit

Veçori e rëndësishme e Sqlmap është që lejon apo i jep mbështetje përdoruesit në shkarkimin dhe ngarkimin e skedarëve të sistemit në llojet e bazave të të dhëna si: Microsoft SQL Server , MySQL ose PostgreSQL. [1] [4]

Sqlmap i jep mbështetje te plotë MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, Informix, MariaDB, MemSQL, TiDB, CockroachDB, HSQLDB, H2, MonetDB, Apache Derby, Amazon Redsh Vertica, Mckoi, Presto, Altibase, MimerSQL, CrateDB, Greenplum, Drizzle, Apache Ignite, Cubrid, InterSystems Cache, IRIS, eXtremeDB dhe FrontBase sistemeve të menaxhimit të bazës së të dhënave.

Sqlmap ka shumë veçori të cilat janë arsye të mjaftueshme për të përzgjedhur këtë mjet për testim dhe pasaj parandalim të sulmeve.

-Karakteristikat gjenerike

Sqlmap mund të lidhet direkt me bazën e të dhënave pa kaluar përmes një SQL injektimi, duke siguruar kredencialet, adresën IP, portin dhe emrin e bazës së të dhënave, ku si përfundim mund të nxjerr numrin e përdoruesit, hash fjalëkalimin, privilegjet, rolet, tabelat dhe kolonat. Poashtu mund të bëjë njohjen automatike e formateve të hash-it të fjalëkalimit dhe mbështetje për thyerjen e tyre duke përdorur një sulm të bazuar në fjalor. Mbështet frijen e të gjitha tabelave të bazës së të dhënave, një sërë shënimesh ose kolona specifike sipas zgjedhjes së përdoruesit. Mbështet kërkimin për emra specifik të bazës së të dhënave, tabela specifike në të gjitha bazat e të dhënave ose kolona specifike në të gjitha tabelat e bazave të të dhënave. Mbështetje për të shkarkuar dhe ngarkuar ndonjë file nga sistemi i file-it themelor të serverit të bazës së të dhënave kur softueri i bazës së të dhënave është MySQL, PostgreSQL ose Microsoft SQL Server. Gjithashtu mund të ekzekutoj komanda arbitrare për të rimarrë prodhimin e tyre standard në serverin e bazës së të dhënave që qëndron në sistemin operativ kur softueri i bazës së të dhënave është MySQL, PostgreSQL ose Microsoft SQL Server. Mbështetje për të përsëritur strukturën e tabelave të bazës së të dhënave back-end dhe shënimet në një bazë të të dhënave lokale SQLite 3 dhe mbështetje për të vendosur një lidhje të paqëndrueshme TCP jashtë bandës midis makinës sulmuese dhe serverit të bazës së të dhënave që qëndron në sistemin operativ. [4]

-Karakteristikat e numërimit dhe fingerprint-it

Mbështet softuer të thjeshtë të ueb serverit dhe teknologji fingerprint të ueb aplikacioneve. Mbështetje për të numëruar përdoruesit e bazë e të dhënaves, hash të fjalekalimeve, rolet dhe privilegjet e përdoruesve, bazat e të dhënavet, tabelat dhe kolonat. Formatin për njohjen automatike të hash të fjalekalimeve dhe mbështetjen për thyerjen e tyre me një sulm të bazuar në ndonjë fjalor. Mbështetje për të brute-force emrin e tabelave dhe kolonave. Mbështetje për kërkimin e emrave të vecantë të bazë e të dhënaves, tabelave të vecanta në bazë e të dhënave apo kolonave të vecanta në tabela.Kjo ështeë e nevojshme për të identifikuar tabelat që përmbajnë të dhënat kredenciale të aplikacionit, ku emrat e kolonave përkatese përmbajnë string si emrin dhe fjalëkalimin. Mbështetje për ekzekutimin e custom SQL statements si në rastin kur një klient interaktiv SQL lidhet me bazë e të dhënaven back-end, sqlmap automatikisht zbërthen statement e dhënë dhe përcakton se cila metodë përshtatet më së miri për injektim.

-Takeover karakteristikat

Mbështet shkarkimin dhe ngarkimin të çfarëdo file-i nga serveri i bazës së të dhënave që qëndron në themel të sistemit kur programi i bazës së të dhënave është MySQL, PostgreSQL ose Microsoft SQL Server, përkrahe poashtu injektimin e funksioneve të personalizuara të përdoruesit: një përdorues mund të përpilojë një bibliotekë të përbashkët, pastaj të përdorë sqlmap për të realizuar ndryshime brenda funksioneve të përdoruesit të sistemit të menaxhimit të bazë e të dhënaves. Këto funksione mund të ekzekutohen, dhe të hiqen opsionalisht, përmes sqlmap. Kjo suportohet kur softueri i bazës së të dhënave është MySQL ose PostgreSQL.

**Disa nga komandat për injektim sql**

**Komanda -p**

Sqlmap automatikisht teston të gjitha GET dhe POST parametrat. Kur vlera e --level është >=2 teston gjithashtu edhe vlerat e HTTP Cookie header. Kur kjo vlere eshte >=3 teston HTTP User-Agent dhe vlerat e HTTP Referer header për SQL injeksionet.

Gjithashtu mund të specifikohen manualisht një listë e parametrave që dëshrirojmë t’i testojmë në sqlmap.

**Komanda: --dbms**

Sqlmap në mënyrë automatike detekton sistemin e menaxhimit të bazës së të dhënave të ueb aplikacioneve. Versioni 9.0 i Sqlmap, mbështet plotësisht këto sisteme menaxhimi të bazë e të dhënaves:

· MySQL

· Oracle

· PostgreSQL

· Microsoft SQL Server

· Microsoft Access

· SQLite

· Firebird

· Sybase

· SAP MaxDB

Nëse për cfarëdo lloj arsye Sqlmap dështon në detektimin e back-end DBMS pas identifikimit të ndonjë SQL injektimit, mundësohet emërtimi i DBMS sipas vetë preferancave të përdoruesit

**Teknikat e Sqlmap**

**SQL Injektimi i bazuar në Boolean**

SQL Injektimi me bazë Boolean është një teknikë e cila mbështetet në dërgimin e një SQL query në bazën e të dhënave. Ky lloj i injektimit detyron aplikacionin të kthejë përgjigje (rezultat) tjetër në

varësi të pyetjes. Në varësi të rezultatit boolean (TRUE ose FALSE), përmbajtja brenda përgjigjes HTTP do të ndryshojë ose do të mbetet e njëjtë. Rezultati mund të lejoj një sulmues të gjykojë nëse ngarkesa e përdorur kthehet True ose False edhe pse nuk janë marrur të dhëna nga baza e të dhënave. Sapo të gjendet një dobësi apo backdoor, Sqlmap në mënyrë automatike zëvendëson ose shton SQL query dhe pret përgjigjiet. Këto përgjigjie i krahason dhe i ndan varësisht nga madhësia. Ky sulm është më i ngadalshëm dhe mund të sheh edhe më shumë informata në bazën e të dhënave, që mund të jenë të dëmshme për përdoruesit. [5]

**SQL Injektimi i bazuar në kohë**

Time-based SQL injektimi është injektim i llojit inferencial ose sulm me injektim të verbër. Është një lloj sulmi në të cilin nuk transferohen të dhëna midis sulmuesit dhe bazës së të dhënave dhe sulmuesi nuk do të jetë në gjendje të marrë rezultate aq lehtë dhe kjo është edhe njëra nga arsyejet pse quhet edhe një sulm me injeksion të verbër. Një sulmues mund të rindërtojë dhe të krijojë një strukturë të re të bazës së të dhënave brenda bazës së të dhënave. Në një sulm të bazuar në kohë, një sulmues dërgon një komandë SQL te serveri me kod për të detyruar një vonesë në ekzekutimin e query-it në mënyrë që nga vonesa e përgjigjes të dihet nëse rezultati është True apo False. Ky lloj i sulmit është nga lloji i sulmeve shumë të ngadalshëm që web aplikacionin mund ta lë në një rrezik të lartë. [3]

**SQL Injektimi i bazuar në gabime**

Error-Based Injektimi mbështetet në mesazhet e gabimit të hedhura nga serveri i bazës së të dhënave për të marrë informacion në lidhje me strukturën e bazës së të dhënave. Në shumë raste gabimi i gjeneruar nga baza e të dhënave është e mjaftueshme që sulmuesi të kuptojë tërësisht bazën e të dhënave. Prandaj çaktivizimi i lajmërimeve të drejtpërdrejta për përdorues ose vizitorë që nuk kanë çasje në të dhëna në web aplikacion.

**SQL Injektimi i bazuar në Union – Query**

SQL Injektimi i bazuar në union lejon një sulmues të nxjerrë informacion nga baza e të dhënave duke zgjeruar rezultatet e kthyera nga pyetja origjinale. Operatori i Unionit mund të përdoret vetëm nëse query-të origjinale / të reja kanë të njëjtën strukturë (numri dhe lloji i të dhënave të kolonave). Sulmuesi në këtë rast mund të nxjerrë informatat shumë shpejtë për arsye të mundsisë të kombinimit të shumë komandave në një. [10]

**SQL Injektimi - Stacked queries and out-of-band**

**Stacked queries -** Një pikëpresje (;) përdoret për të përfunduar një statement në SQL. Duke fshirë radhën origjinale dhe duke shtuar një të ri sulmuesi mund të ndryshoj të dhënat në bazën e të dhënave. Një pikëpresje lejon sulmuesin të ekzekutojë statements të shumta në bazën e të dhënave që ka ngjashmëri në sulmin e bazuar në Union. Injeksioni i bazuar në Union mund të bëhet vetëm duke përdorur statments SELECT, kurse te Stacked queries mund të bëhet me çfarëdo statements. Pra ky lloj i sulmit gjeneron një cenueshmëri të madhe të bazës së të dhënave.

**Out-of-band-** Nuk është shumë i përdorur e një ndër arsyet kryesore pse nuk përdoret është sepse varet nga tiparet që aktivizohen në serverin e bazës së të dhënave. Rastet kur përdoret ky lloj i sulmit është kur një sulmues nuk është në gjendje të përdorë të njëjtin kanal për të filluar sulmin dhe për të mbledhur rezultate. Kjo teknik i ofron një sulmuesi një alternativë ndaj teknikave të bazuara në kohë, veçanërisht nëse përgjigjet e serverit nuk janë shumë të qëndrueshme (duke e bërë një sulm përfundimtar të bazuar në kohë të pasigurt). [3]

**Përdorimi i Sqlmap për sulmimin e objektivave pa pëlqimin paraprak të ndërsjellë është i paligjshëm. Përgjegjësia është e përdoruesit dhe duhet t'i bindet të gjitha ligjeve të zbatueshme lokale, shtetërore dhe federale. Zhvilluesit nuk marrin asnjë përgjegjësi dhe nuk janë përgjegjës për çdo keqpërdorim ose dëmtim të shkaktuar nga ky program. Pasi që kjo vegël mund të përdoret edhe nga sulmuesit prandaj duhet pasur kujdes.**

# Sqlmap

## Përgatitja e mjedisit

Sqlmap është një vegël shumë e fuqishme, e cila automatizon shumë procese për sql injection dhe lejon që të përdoret edhe nga përdorues të cilët nuk kanë dijeni se si të implementojn shumicën e sulmeve për sql injection. Si e tillë duhet të përdoret në aplikacione të cilat jemi duke i zhvilluar ne, për qëllime testimi të vulnerabilities ose në aplikacione në të cilat kemi leje që të testojmë. Përdorimi i kësaj vegle në aplikacione ku nuk kemi leje për testime, mund të rezultoj në dëme të mëdha për kompaninë e cila zotëron aplikacionin në fjalë dhe njëkohësisht kalon në krim kibernetik.

Për qëllime të eksplorimit të mundësive që na ofron vegla sqlmap, ne do të përgatisim një ambient testues, me një aplikacion jo të mbrojtur mirë ose të cenueshëm të hostuar lokalisht. Për këtë qëllim kemi zgjedhur Damn Vulnerable Web App (DVWA). Damn Vulnerable Web App (DVWA) është një web aplikacion, i zhvilluar në PHP/MySQL që është jashtëzakonisht i cenueshëm. Qëllimet e tij kryesore janë të jetë një ndihmë për profesionistët e sigurisë për të testuar aftësitë dhe mjetet e tyre në një mjedis ligjor, të ndihmojë zhvilluesit e uebit të kuptojnë më mirë proceset e sigurimit të aplikacioneve në internet dhe të ndihmojë mësuesit/studentët për të mësuar sigurinë e aplikacioneve në internet në një mjedis klase. DVWA ofron web aplikacionin në 4 nivele të sigurisë : low, medium, hard dhe impossible. Përmes këtyre nivele mund të testohet se sa efikas në jetën reale është vegla.

Përveq aplikacionit viktim të simuluar normalisht na duhet dhde vegla sqlmap. Në sistemin operativ Kali Linux vjen e instaluar paraprakisht si pjesë e paketës së veglave të zgjedhura nga Kali Linux. Mirëpo në versione tjera të Linux mund të instalohet përmes komandës sudo apt-get install sqlmap.

## Përdorimi i veglës

Fillimisht përmes veglës docker hostohet lokalisht web aplikacioni DVWA. Komanda:

docker run --rm -it -p 80:80 vulnerables/web-dvwa

siguron që të hostohet lokalisht web aplikacioni DVWA.

Pasi të hostojmë DVWA, mund të qasemi në localhost dhe shfaqet pamja në Fig.1

Graphical user interface, application

Description automatically generated

Figura 1 Aplikacioni DVWA

Në aplikacionin DVWA mund të kyçemi përmes kredecialeve të paracaktuara admin/password. Pas kyçjes na shfaqet njç përshkrim i shkurt i aplikacionit dhe mundësia për të krijuar databazën e aplikacionit ose për të e resetuar atë , kjo shihet në figurën 2.

Graphical user interface, text, application

Description automatically generated

Figura 2 Krijimi i databazës testuese në mënyrë automatike

Në figurën 3 në pjesën e majtë mund të shihen kategori të ndryshëm për të cilat aplikacioni DVWA përgatit mjedisin për testimin e tyre. Për rastin tonë duke marrë parasysh veglën të cilën e kemi në diskutim, na shërbejnë SQL Injection dhe Blind SQL Injection.

### SQL Injection

Graphical user interface, text, application

Description automatically generated

Figura 3 Mjedisi i cili është i cenushëm ndaj sql injection

Për të e lehtësuar punën tonë do të përdorim Developer Tools. Në figurën 4 shihet se në sektorin Network në kuadër të Developer Tools shohim kërkesën tonë (request) në momentin që bëjmë submit formën. Në Network mund të shohim se kërkesa e bërë submit është e llojit GET, dhe gjithashtu shohim se Cookies që janë bërë submit bashk me kërkesën tonë është edhe PHPSESSID e cila përdoret për authentikim.

Graphical user interface, text, application, email

Description automatically generated

Figura 4 Përcjellja e requestit përmes Developer Tools

Duke marrë parasysh të gjitha informatat e mledhura nga analiza e pjesës së aplikacionit të cilin dëshirojmë të e sulmojmë, formojmë komandën e sqlmap.

sqlmap -u "127.0.0.1/vulnerabilities/sqli/?id=&Submit=Submit" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=low" –tables

Në parametrin **-u** caktojmë targetin tonë të cilin dëshirojmë të e sulmojmë , meqenëse kërkesa është e llojit **GET** caktojmë në URL edhe parametrat e kërkesës. Parametri **–cookies** , përmban cookiet të cilat duam të i bëjmë submit bashk me kërkesë, në rastin tonë dërgojmë PHPSESSID për të u authentikuar. Parametri **-tables** shërben për të printuar të gjitha tabelat nga databaza pas injektimit. Në figurën 5 mund të shihen rezulatatet e kësaj komande.

Graphical user interface

Description automatically generated

Figura 5 Printimi i tabelave të databazave pas injektimit

Në vazhdimësi do të eksplorojmë komanda të ndryshme të sqlmap duket i sqaruar gjithmonë vetëm ata parametra që shfaqen për herë të parë.

Në rast se pas injektimit dëshirojmë të printojmë skemën e databazës atëherë përdorim komandën në vijim.

sqlmap -u "127.0.0.1/vulnerabilities/sqli/?id=&Submit=Submit" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=low" --schema –batch

Parametri **–schema** shërben për të printuar të gjithë skemën e databazës, kurse parametri **–batch** shërben për të përdorur të gjitha opsionet e sygjeruara dhe të mos pyetet useri për vendime gjatë ekzekutimit të komandat. Rezulatet e komadës mund të shihen në figurën 6.

A screenshot of a computer

Description automatically generated with medium confidence

Figura 6 Printimi i skemës së databazës

Si shihet në figurën 6, serveri i cili jemi duke sulmuar përmban dy databaza, **dvwa** dhe **information\_schema,** ne na intereson vetëm dvwa. Për këtë arsye e targetojmë një databazë dhe gjegjsisht një tabelë specifike. Në rastin tonë tabelën **users**.

sqlmap -u "127.0.0.1/vulnerabilities/sqli/?id=&Submit=Submit" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=low" --columns -T users –batch

Parametri **–columns** na shërben për të printuar të gjitha kolonës e tabelës të cilën e përcaktojmë përmes parametrit **-T**. Rezulatet mund të shihen në figurën 7.

Graphical user interface, text

Description automatically generated

Figura 7 Printimi i skemës së tabelës users

Pasi kemi printuar skemën e tabelës users, shohim kolonën password e cila është më e rëndësishmja e kësaj tabele. Kështu që hapi i radhës është të i nxerrim të dhënat e ruajtura në këtë tabelë.

sqlmap -u "127.0.0.1/vulnerabilities/sqli/?id=&Submit=Submit" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=low" --dump -T users –batch

Parametri **–dump** shërben për të printuar të gjitha të dhënat e ruajtura në tabelë. Rezultatet mund të shihen në figurën 8.

Text

Description automatically generated

Figura 8 Printimi i të dhënave të tabelës users

Në figurën 8 të theksuara me ngjyrë të bardh mund të shihen se gjatë ekzekutimit të kësaj komande vegla ka detektuar se ka kolonë me emrin **password** dhe kërkon të di nëse ne dëshirojmë që vegla të tentoj me nje dictionary based attack të dëshifroj hash që gjenden në kolonën password, ose mundësia tjeter është të ruajmë hash në një file të veqant dhe më pas të mundohemi të i dëshifrojmë me ndonjë tool tjetër si p.sh aircrack ng. Meqenëse databaza ne fjalë nuk është e siguruar si duhet, fjalëkalimet janë të ruajtur në hash me MD5 të cilin vegla ka mundur të i dëshifroj dhe në figurën 8 mund të shihen fjalëkalimet edhe si plaintext.

Në vazhdim do të ekzekutojmë komandën për të i dëshifruar fjalëkalimet e përdoruesve të databazës , pra jo të aplikacionit.

Text

Description automatically generated

Figura 9 Tentimi jo i sukseshëm për dëshifrimin e fjalëkalimeve

Në figurën 9, shihet se vegla ka qenë jo e sukseshme në dëshifrimin e fjalëkalimeve të përdorueve të databazës. Pra jo çdo sulm është i suksesshëm ndaj çdo databaze, prandaj duhet të dijmë se qfarë sulme të përdorim.

### Blind SQL Injection

Graphical user interface, text

Description automatically generated

Figura 10 Forma për blind sql injection

Në figurën 10 shihet forma e cenushme ndaj sql injection. Mirëpo për dallim nga shembulli i kaluar kjo është një POST request, nga e cila nuk presim të dhëna. Prandaj duhet përdorur sulm i llojit blind sql injection.

Graphical user interface, text, application, email

Description automatically generated

Figura 11 Analiza e kërkesës POST BSI

Meqenëse kërkesa është e llojit POST , përveq Cookies duhet të dërgojmë edhe form data në body të kërkesës. Për këtë na ndihmojnë prap developer tools.

Pas analizës së kërkesës formojmë komandën.

sqlmap -u "127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id --dbs

Vërejmë se security ka kaluar në medium. Parametri **–data**  përmban form data të cilat në POST request nuk dërgohen në url. Parametri **-p**  përcakton se cilat parametra të databazës të testohen për vulnerabilities. Dhe parametri **–dbs** printon emrat e databazave të marra nga injektimi. Rezultatet mund të shihen në figurën 12.

A screenshot of a computer

Description automatically generated

Figura 12 Printimi i emrave të databazave BSI

Pas nxjerrjes së informacionit për emrat e databazave, targetojmë databazën dvwa.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id -D dvwa --tables --batch

Rezulatet mund te shihen në figurën 13.

Text

Description automatically generated

Figura 13 Printimi i tabelave BSI

Pas kësaj targetojmë tabelën users.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id -T users --batch --threads 5 --dump

Parametri **–threads**  përcakton se sa threads të përdoren për ekzekutimin e komandës, në mënyrë që të ekzekutohet më shpejt. Rezultatet mund të shihen në figurën 14.

A screenshot of a computer

Description automatically generated with medium confidence

Figura 14 Printimi i të dhënave të tabelës users BSI

Përveq nxjerrjes së të dhënave në mënyrë automatike, me veglën sqlmap mund të ekzekutojmë edhe query të shkruar nga ne. Komanda në vazhdim e demonstron këtë veti.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id --sql-query="Select \* from guestbook" --batch --threads 5

Parametri **–sql-query**  përcakton pyetësorin (query) që dëshirojmë të ekzekutojmë pas injektimit.

Rezultatet mund te shihen në figurën 15.

Text

Description automatically generated

Figura 15 Rezulatet e ekzekutimit të custom query

Një ndër vetit më të avancuara të veglës sqlmap është sql-shell. Kjo na mundëson që pas injektimit me sql injection të krijojmë një shell në databazën e sulmuar . Pas krijimit të shellit mund të ekzekutojmë komanda në vazhdimësi.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id --sql-shell --batch --threads 5

Parametri **--sql-shell**  e udhëzon veglën që pas injektimi të krijoj një shellin për të cilin u diskutua më lart. Rezulatet mund të shihen në figurat 16 dhe 17.

Graphical user interface, text

Description automatically generated

Figura 16 Krijimi i sql-shell

lGraphical user interface

Description automatically generated

Figura 17 Ekzekutimi i queries përmes sql-shell

Meqenëse sqlmap është një vegël e cila ka shumë komanda dhe mundësi, ne do të i shqyrtojmë edhe dy komdanda të dobishme ndihmëse.

Në rastin kur në serverin të cilin po e sulmojmë ka shumë databaza te hostuara, atëherë është e dobishme të dijmë se cila i takon aplikacionit të cilin po e sulmojmë aktualisht. Për këtë na shërben komanda në vijim.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id --current-db --batch --threads 5

Parametri **–current-db**  udhëzon veglën të printoj databazën aktuale që përdoret nga aplikacioni.

Graphical user interface

Description automatically generated

Figura 18 Printimi i databazës aktuale

Për të ditur userin aktual përmes të cilit janë duke u ekzekutuar queryt, përdorim komandën në viijim.

sqlmap -u "http://127.0.0.1/vulnerabilities/sqli\_blind/" --cookie="PHPSESSID=fm26ua9o028nr1sa4a18qm8pf6; security=medium" --data="id=1&Submit=Submit" -p id --current-user --batch --threads 5

Graphical user interface

Description automatically generated

Figura 19 Printimi i userit aktual të databazës

# Konkluzione

Sqlmap është një vegël e mirë kur bëhet fjalë pë zbulimin e dobësive të injektimit sql. Me kaq shumë opsione të mbështetura dhe aftësi për të krijuar skriptë të personalizuar, ai dallohet nga shumë vegla tjera open source për testimin e cenueshmërisë së injektimit sql.

Thënë këtë , edhe pse vegla ofron automatizim të shumë procese dhe lehtëson mjaft punën, përdoruesi i saj duhet të ketë njohuri për sql injections në mënyrë që të dijë se si të e përdor në maksimum veglën dhe të dijë qfarë sulmesh të përdorë. Si punë shtesë për të ardhmen ky punim mund të plotësohet edhe më tutje duke eksploruar më shumë veti dhe opsione të kësaj vegle të cilat nuk kanë mundur të përfshihen në këtë punim.

# Referencat

1. <https://tools.kali.org/vulnerability-analysis/sqlmap>
2. <http://sqlmap.org/>
3. Bernardo Damel A.G., Miroslav Stampar “Sqlmap user’s manual“ April 10, 2011
4. <https://portswigger.net/web-security/sql-injection>
5. <https://beaglesecurity.com/blog/vulnerability/boolean-based-blind-sql-injection.html>