UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Blaž Berčič

PREIZKUS ODPORNOSTI INFORMACIJSKIH SISTEMOV NA VDORE

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Denis Trček

Ljubljana, 2013

Besedilo je oblikovano z urejevalnikom besedil Microsoft Word.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00371/2013
Datum: 01.03.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **BLAŽ BERČIČ**

Naslov: **PREIZKUS ODPORNOSTI INFORMACIJSKIH SISTEMOV NA VDORE**
PENETRATION TESTING OF INFORMATION SYSTEMS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

V diplomskem delu opredelite proces penetracijskega preizkusa, podajte vrste penetracijskih preizkusov in stopnje le-teh ter načine njihovega izvajanja. Dodatno predstavite glavna orodja na tem področju in njihovo praktično uporabo.

Mentor:

prof. dr. Denis Trček

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani Blaž Berčič,

z vpisno številko 63000006,

sem avtor diplomskega dela z naslovom:

Preizkus odpornosti informacijskih sistemov na vdore

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom

  prof. dr. Denisa Trčka

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne _____ Podpis avtorja: _____

ZAHVALA

Najprej bi se rad zahvalil svojim staršem za moralno in finančno podporo skozi vsa študijska leta.

Zahvaljujem se tudi mentorju, prof. dr. Denisu Trčku, za strokovno pomoč in nasvete pri izdelavi diplomske naloge.

Posebna zahvala gre moji ženi Petri in svaku Pavlu, ki sta mi pri končanju študija nudila pomoč in me spodbujala pri pisanju, da je bilo delo končano hitreje kot sicer.

KAZALO

POVZETEK

V zadnjih letih je s širitvijo interneta narastel tudi računalniški kriminal. Tarča napadalcev so tudi informacijski sistemi. Tega se vedno bolj zavedajo podjetja in posamezniki, katerih delo in zaslužek sta odvisna od nemotenega delovanja informacijske infrastrukture. Z namenom zaščite sistemov je potrebno poiskati in odpraviti njihove ranljivosti, preden jih odkrijejo zlonamerni napadalci in izkoristijo za vdor. Postopek, ki nam to omogoča, je penetracijski preizkus informacijskega sistema.

Cilj diplomske naloge je opredeliti penetracijski preizkus, opisati orodja, ki se tekom preizkusa uporabljajo in z njihovo pomočjo prikazati vdor v informacijski sistem. V teoretičnem delu je ocena ranljivosti predstavljena kot del penetracijskega preizkusa. Penetracijski preizkuševalec je umeščen med bele hekerje. Opisane so vrste in stopnje penetracijskega preizkusa. Predstavljena so orodja za izvajanje preizkusov: Nmap, Nessus, OpenVAS in Metasploit. Po opravljenem teoretičnem delu se ta orodja uporabijo za poizkus vdora v spletno trgovino. Z uporabo skenerjev ranljivosti se pridobijo informacije o možnih ranljivostih tega informacijskega sistema, z uporabo primerne izkoriščevalske kode pa se doseže njihovo izkoriščenje.

Ključne besede: penetracijski preizkus, ocena ranljivosti, bel heker, izkoriščevalska koda

ABSTRACT

In recent years the expansion of the internet also brought an increase in computer crime. The targets of the attackers are mostly information systems. Companies and individuals, whose work and earnings depend on the smooth functioning of the IT infrastructure, are becoming more and more aware of the problem. In order to protect these systems it is necessary to find and eliminate their vulnerabilities before malicious attackers find and exploit them in order to obtain unauthorized access. The process, which enables us to achieve this, is called penetration testing.

The aim of this thesis is to define the penetration test, describe the tools which are used during the test and finally use them to demonstrate intrusion into the information system. In the theoretical part a vulnerability assessment is presented as a part of the penetration test. In this case the penetration tester is considered to be a white-hat hacker. In the next part types and phases of a penetration test are described and some tools are presented that enable us to accomplish the task: Nmap, Nessus, OpenVAS and Metasploit. After the theoretical part these tools are used for demonstration of an attempt to break into an online store. By using the vulnerability scanners information about potential vulnerabilities is obtained and then vulnerability exploitation is achieved by using an appropriate exploit.

Key words: penetration test, vulnerability assessment, white hat, exploit

1 UVOD

Živimo v dobi globalizacije. Države in različne kulture se povezujejo med seboj intenzivneje kot kadarkoli doslej. Pomemben dejavnik, ki omogoča to povezovanje so elektronski mediji.

Internet kot medij se uporablja v tolikšni meri, da je postala družba od njegovega pravilnega delovanja do določene mere celo odvisna. Vedno več podjetij in organizacij vzpostavlja spletne aplikacije, ki so namenjene vrsti storitev, od elektronskega trgovanja, pridobivanja informacij, storitev javne uprave pa do spletnega bančništva in borznega trgovanja.

Za nemoteno delovanje teh storitev je ključno odkrivanje njihovih varnostnih pomanjkljivosti, še preden jih odkrijejo zlonamerni napadalci in storitve onemogočijo ali pa si z nepooblaščeno uporabo le teh pridobijo materialno korist. Učinkovita ocena varnosti posameznega informacijskega sistema ne temelji le na varnostnem pregledu in odkritju ranljivosti, ampak tudi na penetracijskem preizkusu informacijskega sistema.

V diplomskem delu najprej predstavimo proces penetracijskega preizkusa in prikažemo razliko med delom penetracijskega preizkuševalca in hekerji. V nadaljevanju pojasnimo, zakaj se ocena ranljivosti in penetracijski preizkus informacijskega sistema razlikujeta. V tretjem poglavju opišemo vrste penetracijskih preizkusov, ki se razlikujejo glede na količino podatkov o sistemu, ki jih imajo na voljo preizkuševalci. Četrto poglavje obsega opis stopenj penetracijskega preizkusa, ki so del metodologije za dokončanje preizkusa. V naslednjem poglavju predstavimo nekaj brezplačnih orodij, ki se uporabljajo v posameznih stopnjah penetracijskega preizkusa. To so orodja Nmap, Nessus, OpenVAS in Metasploit.

V šestem poglavju prikažemo, kako v praksi poteka penetracijski preizkus. Izvedemo ga na primeru spletne trgovine in pri tem predstavimo uporabo orodij opisanih v petem poglavju. Prikažemo poizkus vdora v sistem od zunaj in nato poizkus vdora iz notranjega omrežja. Pri tem predstavimo uporabo avtomatiziranih postopkov in na drugi strani ročno konfiguriranje ter izvedbo izkoriščevalske kode. V zadnjem poglavju podamo še končne ugotovitve.

2 PENETRACIJSKI PREIZKUS

2.1 Kaj je penetracijski preizkus

Penetracijski preizkus (angl. penetration test) je metoda s katero ovrednotimo varnost računalniškega sistema ali omrežja s simulacijo napada, kot bi ga izvedla zlonamerna nepooblaščena oseba. Proces vključuje aktiven pregled sistema glede morebitnih slabosti, tehničnimi pomanjkljivostmi ali ranljivostmi (angl. vulnerabilities). Pregled je izveden s stališča možnega napadalca in lahko vključuje izkoriščanje odkritih ranljivosti sistema.

Pomanjkljivosti, odkrite med penetracijskim preizkusom, predstavimo lastniku sistema skupaj z oceno možne škode za celotno organizacijo ter predlagamo protiukrepe za zmanjšanje tveganja [4].

Na tem mestu je potrebno pojasniti tudi razliko med penetracijskim preizkuševalcem (angl. penetration tester) in hekerjem (angl. hacker). Heker »računalniške zaščite« je oseba, ki zaobide zaščito in vstopi v računalniški sistem ne glede na namen. Poznamo tri vrste hekerjev, ki se razlikujejo glede na namen vdora:

- bele hekerje (angl. white hat hackers),
- črne hekerje (angl. black hat hackers),
- sive hekerje (angl. grey hat heckers).

Bel heker ali etičen heker z legalnim vdiranjem preizkuša varnost sistemov z namenom zaščite le teh. Med etične hekerje spadajo tudi posamezniki, ki izvajajo penetracijske preizkuse in ocene ranljivosti (angl. vulnerability assessment) informacijskih sistemov določene organizacije v okviru pogodbenega dogovora.

Črn heker, imenovan tudi kreker (angl. cracker), nelegalno vdira v sisteme z namenom pridobiti materialno korist s krajo podatkov ali škodovati sistemu z uničenjem podatkov in onemogočanjem dostopa do sistema pooblaščenim osebam.

Siv heker spada nekje vmes med belega in črnega hekerja. Nelegalno vdira v sisteme vendar ne z namenom škodovanja ampak z namenom ozaveščanja lastnika sistema in širše javnosti o pomanjkljivi varnosti določenega sistema [5].

2.2 Razlika med penetracijskim preizkusom in oceno ranljivosti

Mnogo ljudi, ki se ukvarja z računalniško varnostjo, nepravilno uporablja izraz ocena ranljivosti in izraz penetracijski preizkus kot sopomenki. Ocena ranljivosti je diagnostični proces pri katerem se osredotočimo na iskanje in ocenjevanje možnih ranljivosti v sistemu. Pri tem lahko uporabljamo avtomatizirana orodja posebej zasnovana za ugotavljanje, če so

odpravljene vse že znane ranljivosti v določenem računalniškem okolju. Pri iskanju ranljivosti se ustavimo, še preden ogrozimo delovanje informacijskega sistema.

Med penetracijskim preizkusom dejansko napadamo sistem. S simulacijo hekerjevega delovanja poizkušamo te ranljivosti izkoristiti za vdor v sistem in s tem dokazati, da le te v resnici obstajajo. Pri tem gremo tako daleč, kot nam dopušča pogodba [2,6].

Penetracijski preizkus je v primerjavi z oceno ranljivosti draga metoda.

3 VRSTE PENETRACIJSKIH PREIZKUSOV

Penetracijske preizkuse se lahko izvaja na več načinov. Najpogostejša razlika med njimi je količina podatkov o podrobnostih izvedbe sistema, ki je na voljo preizkuševalcem [4]. V osnovi ločimo tri načine izvedbe penetracijskih preizkusov.

3.1 Testiranje z metodo bele škatle

Testiranje z metodo bele škatle je način, ko preizkuševalci popolnoma poznajo infrastrukturo sistema, ki ga preizkušajo. Imajo dostop do omrežnih diagramov, izvorne kode, obsega naslovov IP in drugih uporabnih informacij. Ta vrsta testiranja se uporablja, ko je čas ključnega pomena, proračun in število dovoljenih ur pa omejeno. Takšno testiranje je najmanj realen prikaz, kaj lahko stori potencialni napadalec.

3.2 Testiranje z metodo črne škatle

Pri testiranju z metodo črne škatle nimajo preizkuševalci predhodno na voljo nobenih informacij o sistemu. Navadno poznajo samo domeno ali naslov IP spletnega strežnika. Pred začetkom analize morajo najprej določiti lokacijo in obseg sistema. Ta vrsta testiranja najbolj natančno prikaže delovanje potencialnega zunanjega napadalca, ki sistema ne pozna.

3.3 Testiranje z metodo sive škatle

Test z metodo sive škatle je križanec med testom z metodo bele in metodo črne škatle. To je najbolj uporabna vrsta penetracijskega preizkusa. Preizkuševalci dobijo le omejene informacije o sistemu in še te le na zahtevo. Tekom testa se jim postopno poda več informacij z namenom hitrejšega zaključka testiranja. Ta metoda poveča realizem v primerjavi z metodo bele škatle, hkrati pa je cenejša za naročnika glede na test z metodo črne škatle [3]. Posnema notranji vdor izveden s strani zlonamernega zaposlenega, ki že ima omejen dostop do sistema.

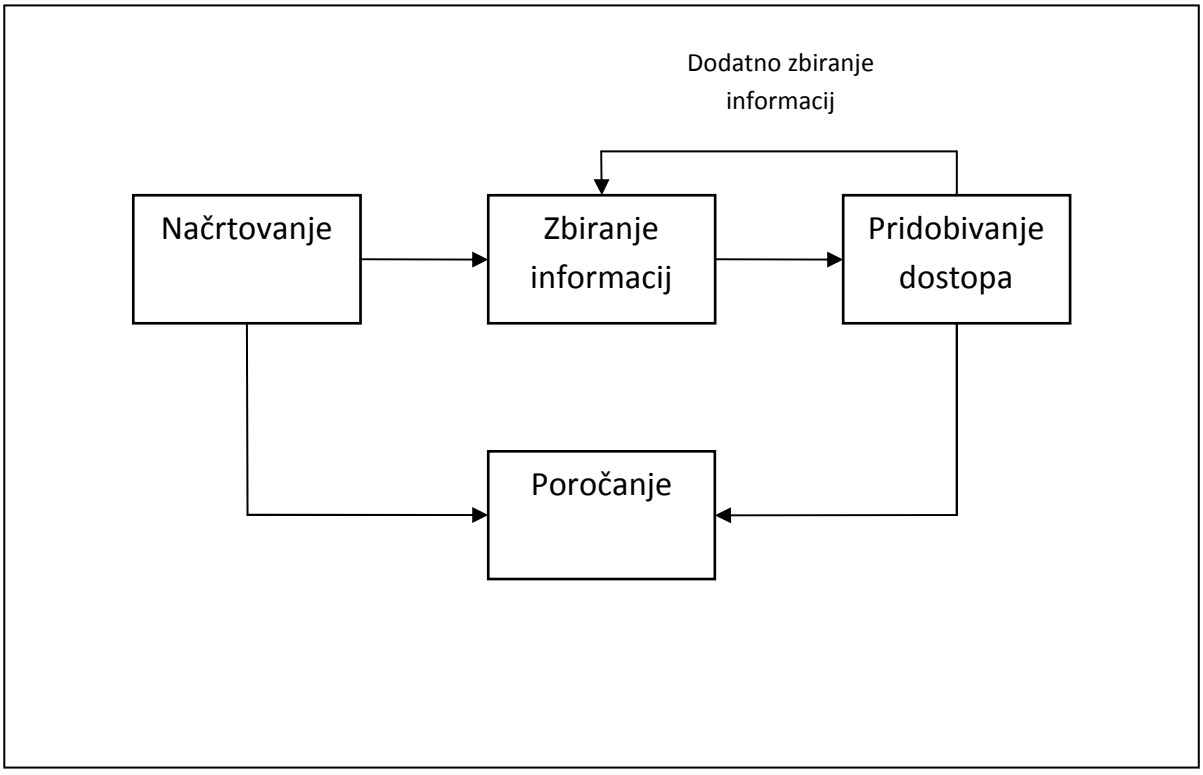
4 STOPNJE PENETRACIJSKEGA PREIZKUSA

Celoten proces penetracijskega preizkusa lahko razdelimo na več korakov ali stopenj, ki skupaj tvorijo celovito metodologijo za dokončanje penetracijskega preizkusa. Uporaba organiziranega pristopa (metodologije) je potrebna, ker omogoča preizkuševalcu, da se osredotoči na posamezen del in nato rezultate iz vsake stopnje uporabi v naslednjih stopnjah preizkusa. Z uporabo metodologije zapleten proces razčlenimo na več manjših, bolj obvladljivih nalog.

Obstaja več metodologij, ki proces razdelijo na štiri do sedem stopenj in za stopnje uporabljajo različna imena, vsem pa je skupno to, da omogočajo popoln pregled nad celotnim procesom [2].

Tipičen penetracijski preizkus informacijskega sistema je sestavljen iz naslednjih stopenj:

- načrtovanje (angl. planning),
- zbiranje informacij (angl. information gathering),
- pridobivanje dostopa (angl. gaining access),
- poročanje (angl. reporting).



Slika 1: Stopnje penetracijskega preizkusa (Vir:[8])

4.1 Načrtovanje

V stopnji načrtovanja se sestanejo naročnik in ekipa penetracijskih preizkuševalcev. Dogovorijo se o obsegu in ciljih preizkusa. V večini primerov je cilj preizkusa dokazati, da znotraj naročnikove omrežne infrastrukture obstajajo ranljivosti, ki se jih lahko izkoristi za vdor. Obseg preizkusa se določi z identifikacijo računalnikov, sistemov in omrežij v podjetju naročnika. Del dogovora je tudi oblika poročila v katerem so prikazani rezultati ali izid preizkusa [7].

V tej fazi se podpiše tudi razne pravne dokumente, ki ščitijo ekipo preizkuševalcev pred posledicami, če bi med testom šlo kaj narobe. Preizkuševalci se tekom testa srečujejo z omejitvami, ki za zlonamernega napadalca ne veljajo. Nekatere izmed teh omejitev so:

- *Čas*: V realni situaciji ima zlonamerni heker dovolj časa, da temeljito pripravi svoj napad. Penetracijski preizkuševalec mora upoštevati zakonske roke, ki so določeni v pogodbi, kot tudi delovni čas organizacije v kateri izvaja preizkus. Nobena organizacija namreč ne želi, da zaradi preizkusa ne bi delovale določene storitve ali da bi trpelo njeno poslovanje.
- *Druge omejitve:* Preizkuševalec je vezan na pravno pogodbo. Pogodba navaja sprejemljive in nesprejemljive ukrepe, ki jih lahko uporablja tekom testa [8].

Faza načrtovanja še ni začetek dejanskega preizkušanja.

4.2 Zbiranje informacij

S stopnjo zbiranja informacij se dejansko začne penetracijski preizkus. Ta stopnja se nadalje deli na naslednje podstopnje:

- sledenje (angl. footprinting),
- skeniranje (angl. scanning) in popisovanje (angl. enumeration),
- odkrivanje ranljivosti (angl. vulnerability identification).

4.2.1 Sledenje

Sledenje je proces izdelovanja profila napadene informacijske infrastrukture. V tej stopnji napadalec preuči področja, kot so internet, intranet, ekstranet in možnosti oddaljenega povezovanja. Informacije, ki jih zbira, vključujejo registrarna imena domen, dodeljeni obseg naslovov IP, uporabljene omrežne protokole, podatke o administratorju sistema, naslove elektronske pošte, listo zaposlenih, telefonske številke itd.

Sledenje je pasivna dejavnost, pri kateri se izogibamo nepotrebnim stikom s ciljnim sistemom. Informacije zbiramo iz javno dostopnih baz na internetu.

4.2.2 Skeniranje in popisovanje

Skeniranje oz. mapiranje (angl. mapping) omrežja tarče sestavlja odkrivanje aktivnih sistemov (delujočih računalnikov), strežniških programov in vrst operacijskega sistema.

Najpogostejša oblika preverjanja, če je na izbranem naslovu IP delujoč računalnik, je uporaba ICMP (Internet Control Management Protocol) zahtev »echo« v danem obsegu naslovov IP [1]. Ko odkrijemo delujoče računalnike preverimo katera vrata imajo odprta, zaprta ali filtrirana in kateri operacijski sistem je nameščen na njih.

Pri popisovanju gre za dejaven stik z napadenim sistemom, kjer kot odjemalec pošiljamo poizvedbe strežniškim programom, ki tečejo na operacijskem sistemu tarče. Pri tem poizkušamo identificirati verzijo strežniških programov, uporabniške in administratorske račune ali druge možne točke za vstop.

4.2.3 Odkrivanje ranljivosti

Potem ko smo identificirali ciljne računalnike in zbrali dovolj informacij o njih, poizkušamo določiti ranljivosti, ki obstajajo na teh računalnikih. Ranljivost je pomanjkljivost v programski opremi ali konfiguraciji sistema, ki se lahko izkoristi za vdor. Večina odkritih ranljivosti je posledica manjkajočih programskih popravkov [2].

Pri iskanju ranljivosti lahko ročno preiščemo prosto dostopne baze ranljivosti na internetu: na primer narodno bazo ranljivosti (angl. national vulnerability database) ali pa informacije o zadnjih odkritih pomanjkljivostih programske opreme poiščemo na varnostnih forumih, blogih, novičarskih skupinah itd. V tej fazi pogosto testiramo spletne aplikacije z vnosom neveljavnih znakov ali naključnih nizov v vnosna polja. Pri tem preverimo nenadzorovan odziv aplikacij in sporočila o napakah. Na ta način mnogokrat odkrijemo nove, še neznane ranljivosti.

Druga možnost je uporaba avtomatskih orodij imenovanih skenerji ranljivosti (angl. vulnerability scanners), s katerimi preiščemo računalnike za znanimi ranljivostmi. Ta orodja imajo običajno lastne baze s podrobnimi informacijami o najnovejših ranljivostih [8].

Priporočljiva je kombinacija tako ročnega kot avtomatskega preverjanja sistemov za možnimi ranljivostmi. Z ročnim preverjanjem odkrijemo nove ranljivosti, ki jih avtomatski skenerji zgrešijo, vendar to zahteva več predhodnega znanja in potrebnega časa.

4.3 Pridobivanje dostopa

Pridobivanje dostopa ali napad na sistem je glavna stopnja penetracijskega preizkusa. Tudi to stopnjo preizkusa lahko delimo še na dve podstopnji:

- izkoriščanje ranljivosti (angl. vulnerability exploitation),

- povečevanje pooblastil (angl. privilege escalation).

4.3.1 Izkoriščanje ranljivosti

Izkoriščanje ranljivosti je stopnja, pri kateri poizkušamo predhodno ugotovljene možne ranljivosti izkoristiti za vdor v sistem. Če nam to uspe, s tem tudi potrdimo obstoj teh ranljivosti. Za izkoriščanje ranljivosti uporabimo izkoriščevalsko kodo (angl. exploit).

Izkoriščevalska koda izrablja varnostno pomanjkljivost v programih in operacijskih sistemih tako, da omogoči zagon škodljivega programa oziroma kode. Pri tem uporabi zmogljivosti napadene programske opreme tako, da ta izvrši arbitrarno kodo, prebere, kopira ali izdela datoteke in napadalcu omogoči neavtoriziran dostop v sistem [1].

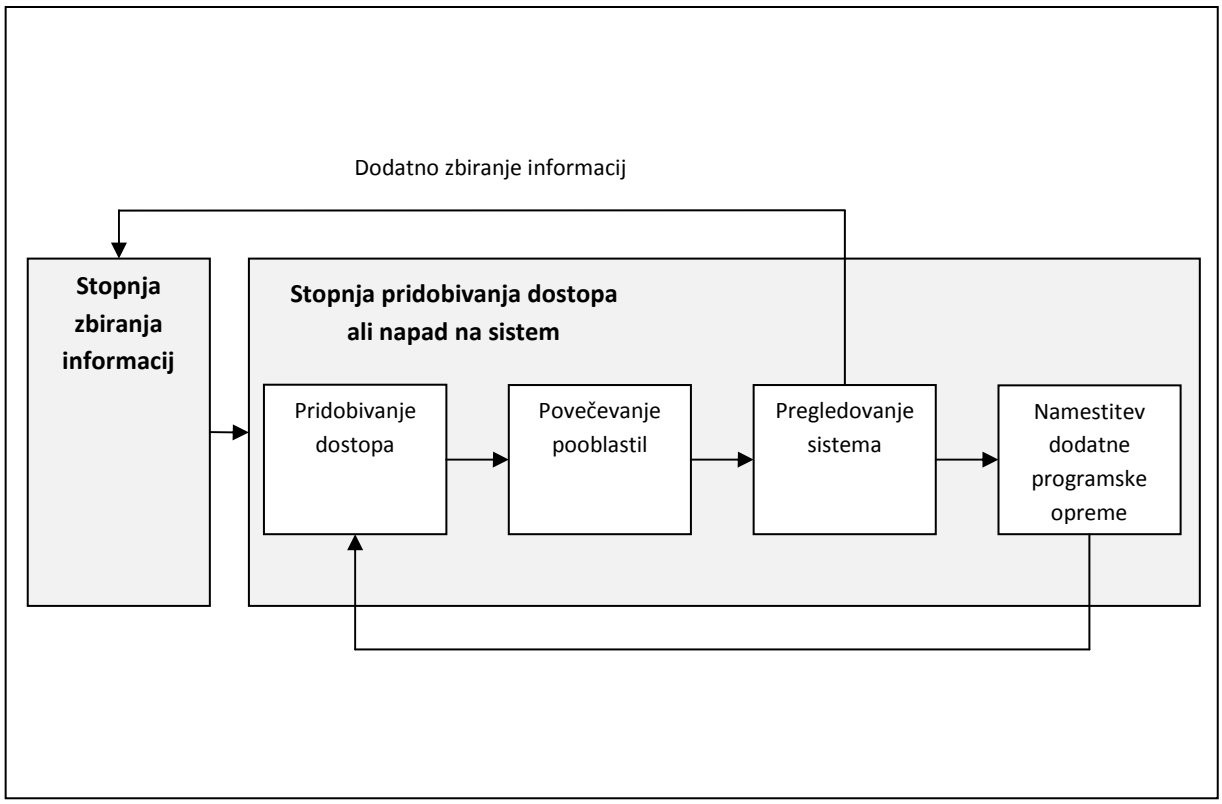
Pri iskanju primerne izkoriščevalske kode imamo na voljo spletne baze, ki nudijo izkoriščevalsko kodo za večino znanih ranljivosti. Primer take baze je Packet Storm Security. Obstajajo tudi orodja, kot je na primer Metasploit Framework, ki imajo implementirano lastno zbirko izkoriščevalskih kod. Izkušeni penetracijski preizkuševalci pa lahko napišejo lastno izkoriščevalsko kodo za izkoriščenje določene ranljivosti.

Pri uporabi izkoriščevalske kode je potrebno paziti, da ne motimo delovanja napadenega sistema, oziroma da bi bil sistem pri tem dalj časa nedelujoč. Nekatera podjetja celo zahtevajo, da se določenih odkritih ranljivosti na kritičnih sistemih ne sme izkoristiti. Tako mnogokrat ne testiramo občutljivosti sistemov na napade za zavrnitev storitve (angl. denial of service – kratica DoS). Pri DoS napadu gre za pošiljanje velikega števila zahtev za izvajanje omrežne storitve, kar lahko povzroči nedostopnost storitve za uporabnike [9].

V kolikor v prejšnji stopnji preizkusa na sistemu nismo odkrili nobene ranljivosti, ki bi jo bilo mogoče izkoristili za vdor v sistem, lahko z dovoljenjem naročnika preizkusa uporabimo tudi metodo družabnega inženiringa (angl. social engineering). V tej zvrsti napada igra glavno vlogo človek, ki zaradi pomanjkanja računalniškega znanja in zaupanja napadalcu omogoči vstop v omrežje [1]. Pri družabnem inženiringu tako preverimo varnostno ozaveščenost zaposlenih v podjetju, ki je predmet penetracijskega preizkusa.

4.3.2 Povečevanje pooblastil

Precej pogosto z uspešnim izkoriščenjem ranljivosti in posledično vdorom v sistem pridobimo dostop na ravni navadnega uporabnika. V takem primeru je potrebna nadaljnja analiza, s katero ocenimo tveganje, ki ga taka ranljivost lahko povzroči na napadenem sistemu. Tak scenarij je prikazan na povratni zanki med stopnjo pridobivanja dostopa in stopnjo zbiranja informacij.



Slika 2: Stopnja pridobivanja dostopa s povratno zanko do stopnje zbiranja informacij (Vir:[8])

S pridobitvijo dodatnih informacij o sistemu lahko z uporabo primernih orodij in z izrabo novo odkritih lokalnih ranljivosti pridobimo administratorske pravice.

Preizkuševalec v tej fazi poizkuša izrabiti napaden sistem za napad na druge sisteme v notranjem omrežju. Seveda pa mora za to dobiti dovoljenje naročnika.

Namen povečevanja pooblastil je pridobitev najvišje ravni dostopa do sistema.

4.4 Poročanje

Zadnja stopnja v celotnem procesu je poročanje. Lahko se izvaja vzporedno z ostalimi tremi stopnjami ali pa ob zaključku stopnje pridobivanja dostopa. Poročanje je najpomembnejša stopnja preizkusa, saj naročnik dobi in plača končno poročilo. Na podlagi poročila naročnik tudi izvede ukrepe za zaščito informacijskega sistema.

Pri izdelavi končnega poročila moramo upoštevati ciljno publiko; torej sistemske administratorje kot tudi upravni odbor organizacije. Tehnično plat poročila in odkrite ranljivosti moramo zato ustrezno predstaviti z grafi in slikami ter tako prikazati, kakšen učinek imajo lahko na poslovanje podjetja.

Pripraviti moramo povzetek poročila na eni ali dveh straneh, kjer na kratko opišemo aktivnosti, ki smo jih izvedli tekom preizkusa, navedemo najbolj kritične ranljivosti in podamo priporočila za odpravo teh ranljivosti. Na podlagi navedenih odkritij naročnik izdela oceno stroškov izvedbe priporočil.

Popolno poročilo mora vsebovati podroben tehničen opis vseh odkritih ranljivosti in priporočila za njihovo ublažitev. Vse varnostne luknje, ki smo jih odkrili in uspešno izkoristili moramo dokumentirati z dokazi: na primer s posnetkom zaslona (angl. screenshot) uspešno izvedene izkoriščevalske kode. Ničesar ne smemo prepustiti domišljiji naročnika. Natančna dokumentacija prikaže sposobnost uspešnega penetracijskega preizkuševalca [8].

Poročilo je tako sestavljeno iz naslednjih komponent:

- povzetka poročila,
- podrobno opisanih odkritij,
- stopnje tveganja odkritih ranljivosti,
- učinka ranljivosti na poslovanje,
- priporočil za ublažitev ranljivosti.

5 ORODJA ZA PENETRACIJSKE PREIZKUSE

Orodja so osnova za delo vsakega penetracijskega preizkuševalca. Pri penetracijskih preizkusih se uporablja množica različnih orodij, v grobem pa jih delimo v dve skupini:

- orodja za zbiranje informacij ali odkrivanje ranljivosti,
- orodja za izkoriščanje ranljivosti.

Meja med tema dvema skupinama ni jasno začrtana, saj lahko orodja za odkrivanje ranljivosti vsebujejo določene funkcije za izkoriščanje ranljivosti in obratno - orodja za izkoriščanje teh ranljivosti vsebujejo tudi poizvedovalni del.

V tem poglavju bomo na kratko predstavili brezplačna poizvedovalna orodja Nmap, OpenVAS in Nessus ter zbirko orodij za izkoriščanje ranljivosti Metasploit Framework.

5.1 Nmap

Nmap je verjetno eno izmed najbolj znanih in uporabljanih orodij v penetracijskih preizkusih. Gre za skener oziroma popisovalnik omrežja, ki ga je razvil Gordon »Fyodor« Lyon in je prosto dostopen na spletnem naslovu http://nmap.org/. Uporablja se za raziskovanje omrežja in varnostne revizije.

Orodje omogoča:

- odkrivanje delujočih računalnikov v omrežju,
- pregledovanje vrat (popis odprtih vrat na enem ali več računalnikih),
- detekcijo strežniških programov, ki poslušajo na odprtih vratih (ime in verzijo aplikacije),
- določanje vrste in verzije operacijskega sistema,
- določanje naslova MAC omrežne kartice,
- določanje vrste filtrov paketa ali požarnega zidu.

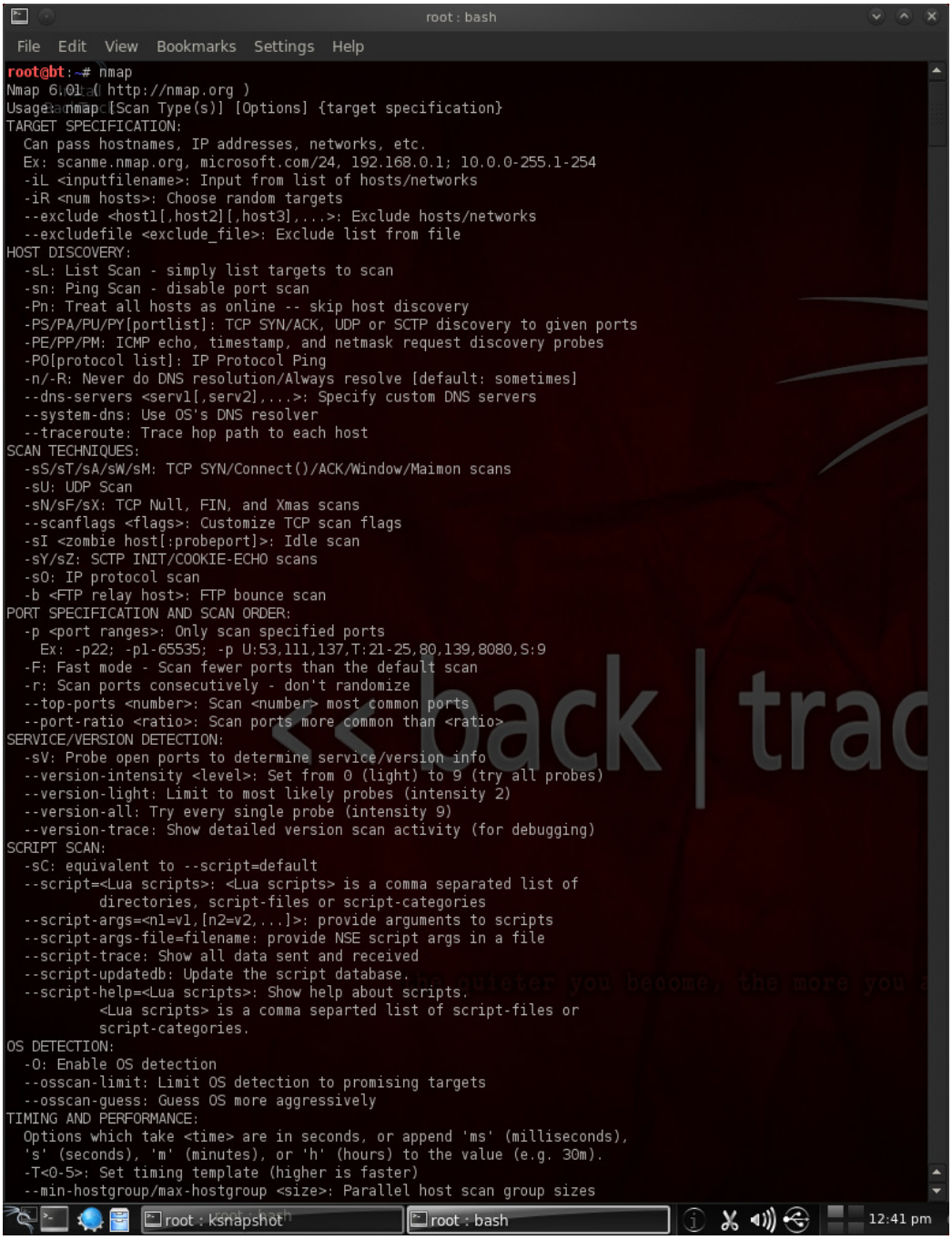
Nmap deluje na večini operacijskih sistemov kot je Linux, Microsoft Windows, Solaris, Mac OS X, različne verzije BSD. Standardni programski vmesnik je ukazna vrstica (angl. command prompt). Program je mogoče daljinsko krmiliti preko spletnega vmesnika, obstaja pa tudi grafični uporabniški vmesnik imenovan Zenmap.

Program Nmap se uporablja na sledeč način:

Nmap [tip skeniranja] [dodatne opcije] <naslov IP ali spisek naslovov IP >

Kot prvi argument navedemo tip skeniranja. Uporabimo lahko kombinacijo različnih tehnik skeniranja. Za drugi argument navedemo dodatne opcije skeniranja (obliko izpisa rezultatov,

izpis rezultatov v datoteko, vrata za skeniranje itd.). Kot zadnji argument navedemo naslov IP ali spisek naslovov, ki jih želimo skenirati.



Slika 3: Osnovni ukazi v programu Nmap

Podprtih je več tehnik za skeniranje vrat. Nekatere so bolj agresivne in očitne, druge pa so bolj prikrite, tako da skeniranja sistemi za zaznavanje vdorov ne opazijo. Glede na izbrano tehniko skeniranja so lahko tudi vrnjeni rezultati drugačni [10]. Podprte tehnike so:

- skeniranje z vzpostavitvijo povezave TCP (angl. TCP connect() scan),
- skeniranje tipa TCP SYN,

- skeniranje TCP ping,
- TCP ničelni tip skeniranja (angl. TCP Null scan),
- skeniranje s paketi TCP FIN,
- skeniranje TCP tipa božične jelke (angl. TCP Xmas Tree scan),
- skeniranje TCP ACK,
- skeniranje TCP z oknom paketa (angl. TCP window scan),
- skeniranje TCP RPC,
- skeniranje TCP tipa Maimon,
- skeniranje tipa UDP,
- skeniranje FTP bounce,
- skeniranje protokola IP,
- skeniranje SCTP tipa INIT,
- skeniranje SCTP tipa COOKIE ECHO.

5.2 Nessus

Nessus je popularen skener ranljivosti podjetja Tenable Network Security. Program z omejeno funkcionalnostjo je brezplačen za osebno uporabo v domačem okolju (licenca Nessus HomeFeed). Za pridobitev brezplačne licence se je potrebno registrirati.

Za komercialno uporabo je na voljo licenca Nessus ProfessionalFeed, za katero je potrebno plačati letno naročnino 1200 USD.

Namenjen je celovitemu pregledovanju lokalnih ali omrežnih računalnikov za možnimi ranljivostmi. Med skeniranjem pregleduje vrata in preverja morebitne pomanjkljivosti programske opreme, ki bi jih bilo mogoče izkoristili za vdor. Nekatere od teh pomanjkljivosti so zastarela in ranljiva programska oprema, neprimerna konfiguracija (računi s privzetimi gesli ali brez gesla), prisotnost tveganih storitev ali prikritih procesov (angl. daemons) itd. Omogoča preverjanje odpornosti sistema na napade DoS ter testiranje aplikacij, ki za varno komunikacijo uporabljajo sloj varnih vtičnic (angl. Secure Sockets Layer – kratica SSL) kot so https, smtps, imaps itd. Opremiti ga je mogoče s certifikatom, da se lahko vključi v varnostno okolje, ki temelji na javnih ključih.

Nessus vsebuje poseben skriptni jezik za pisanje napadov NASL (Nessus Attack Scripting Language) s katerim je mogoče hitro sestaviti varnostni test, uporabimo pa lahko tudi programski jezik C. Vsak varnostni test je napisan kot zunanji dodatek (angl. external plugin). Bazo varnostnih testov (zunanjih dodatkov) je mogoče dnevno posodabljati.

Temelji na arhitekturi odjemalec-strežnik. Odjemalec je uporabniški vmesnik, kjer se konfigurira način skeniranja, dodatke, tarče za skeniranje. Ukaze posreduje strežniku in na koncu skeniranja generira poročilo z rezultati. V njem so navedene odkrite varnostne

pomanjkljivosti in nasveti za odpravo le teh. Poročilo je mogoče izvoziti v besedilne datoteke ter datoteke XML, HTML in LaTeX. Strežnik izvaja vsa varnostna skeniranja oziroma napade. Vsa mrežna komunikacija med odjemalcem in strežnikom je zaščitena.

Odjemalec in strežnik sta lahko nameščena na ločenih računalnikih in različnih operacijskih sistemih. Strežnik je na primer nameščen na sistemu Linux in odjemalec na sistemu Windows ali pa sta oba nameščena na istem računalniku in operacijskemu sistemu. Strežnik podpira večino današnjih operacijskih sistemov, kot so Windows, Linux, Mac OS X, free BSD, Solaris. Odjemalec pa poleg že naštetih še mobilni operacijski sistem iOS in Android [1,11].

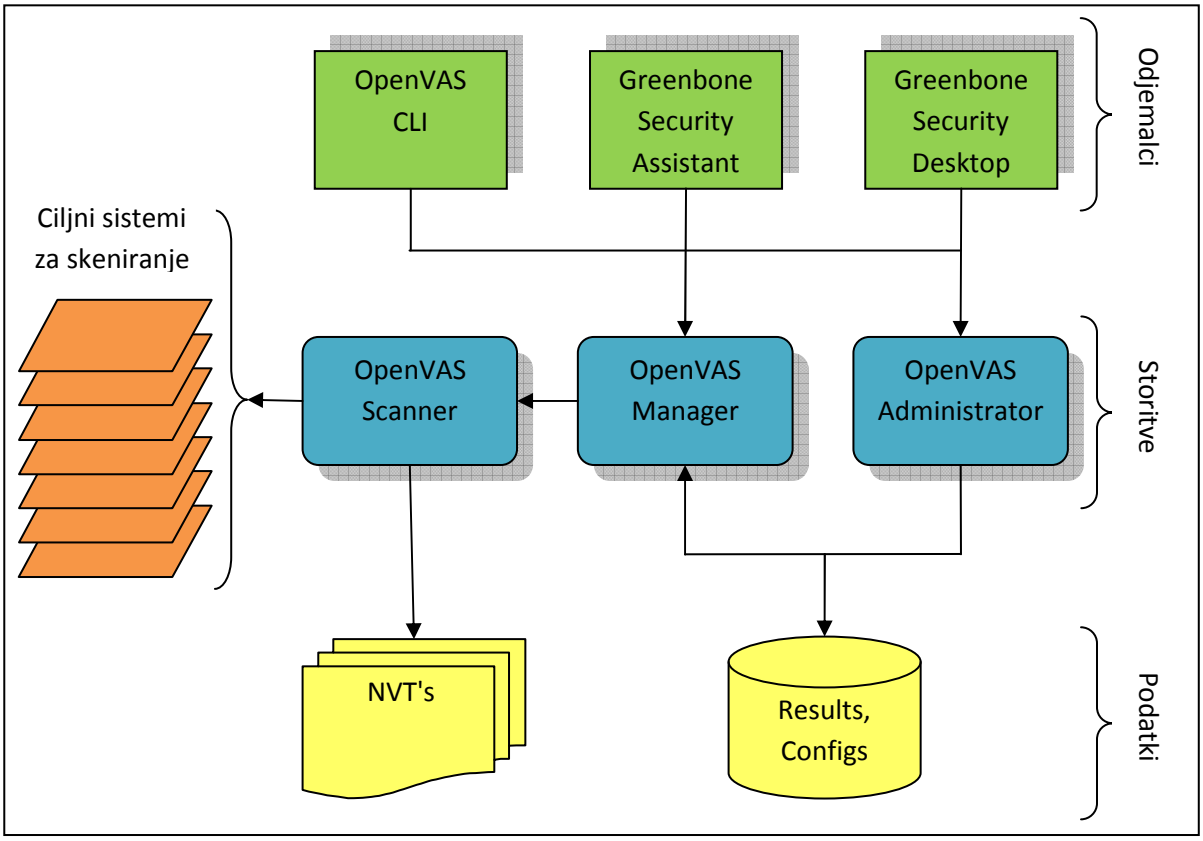
5.3 OpenVAS

OpenVAS (angl. Open Vulnerability Assessment System) je ogrodje (angl. framework) sestavljeno iz več strežniških programov in orodij, ki skupaj omogočajo celovito skeniranje sistema za možnimi ranljivostmi in ponujajo rešitve za odpravo le teh. Vsi produkti OpenVAS so brezplačni, večina pa jih je zaščitena z odprtokodno programsko licenco GPL (angl. General Public Licence) [12].

Ogrodje je nastalo kot podveja programa Nessus, ko je ta leta 2005 spremenil odprtokodno programsko licenco v zaprto in postal komercialen produkt. Iz tega razloga je OpenVAS močno podoben programu Nessus. Ravno tako temelji na arhitekturi odjemalec-strežnik, ki zajema več delov:

- *OpenVAS CLI*: omogoča upravljanje OpenVAS orodij preko ukazne vrstice.
- *Greenbone Security Assistant*: je spletni uporabniški vmesnik, s katerim lahko nastavimo in spremljamo profile za skeniranje, sprožimo skeniranje ciljnih sistemov in na koncu kreiramo poročilo.
- *Greenbone Security Desktop*: je orodje, ki tako kot OpenVAS CLI in Greenbone Security Assistant omogoča upravljanje OpenVAS orodij preko grafičnega vmesnika na namizju.
- *OpenVAS Scanner*: izvaja skeniranje ciljnih sistemov z uporabo testov omrežne ranljivosti (angl. network vulnerability tests – kratica NVT).
- *OpenVAS Manager*: je osrednja komponenta ogrodja OpenVAS. Sprejema naloge od komponente OpenVAS Administrator in komponent za upravljanje (OpenVAS CLI, Greenbone Security Assistant, Greenbone Desktop Security) in nato dodeljuje naloge za izvedbo skeniranja komponenti OpenVAS Scanner. Upravlja tudi zbirko podatkov SQL, kjer so shranjene vse nastavitve programa in rezultati skeniranj.
- *OpenVAS Administrator*: omogoča upravljanje z uporabniki in viri (na primer posodobitvami).

- *NVT's*: gre za zbirko testov, ki odkrivajo možne ranljivosti. Trenutno zbirka vsebuje več kot 25000 testov (maj 2012) in jo je možno dnevno posodabljati.
- *Results, Configs*: zbirka podatkov SQL z nastavitvami in rezultati.



Slika 4: Struktura OpenVAS ogrodja (Vir:[12])

Vse komunikacija med komponentami OpenVAS ogrodja je zavarovana z uporabo protokola SSL. Strežnik izvaja skeniranje, odjemalec pa se uporablja za konfiguriranje skeniranja in dostop do rezultatov. Strežnik deluje samo na operacijskem sistemu Linux in BSD, odjemalec pa tudi na operacijskem sistemu Windows in Mac OS X.

5.4 Ogrodje Metasploit

Ogrodje Metasploit (angl. Metasploit Framework – kratica MSF) je odprtokodno razvojno orodje za izdelavo varnostnih orodij ter izdelavo in uporabo zlonamerne kode. Uporablja se za penetracijske preizkuse in regresijsko testriranje programske opreme [13]. Z njegovo uporabo dejansko preverimo, če se možne ranljivosti, odkrite med varnostnim skeniranjem, da izkoristiti za vdor oziroma povzročitev škode informacijskemu sistemu.

Ogrodje je del Metasploit projekta, ki ga je leta 2003 ustanovil H.D. Moore. Leta 2009 je projekt prevzelo podjetje Rapid7 in poleg brezplačnega ogrodja Metasploit, ki je zaščiteno z licenco BSD (angl. Berkeley Software Distribution), na tržišče dodala še komercialni različici imenovani Metasploit Express in Metasploit Pro [14].

Ogrodje Metasploit je napisano v programskem jeziku Ruby, vključuje pa tudi komponente napisane v programskem jeziku C in zbirniku. Deluje na operacijskem sistemu Linux, Windows in Mac OS X. Vsebuje že preko 900 primerkov izkoriščevalske kode za različne operacijske sisteme in se stalno posodablja. Omogoča uvoz rezultatov, ki smo jih pridobili s skenerji ranljivosti. Nato primerja možne ranljivosti z izkoriščevalsko kodo, ki jo ima na voljo in izbere najbolj ustrezno za uspešno izvedbo vdora.

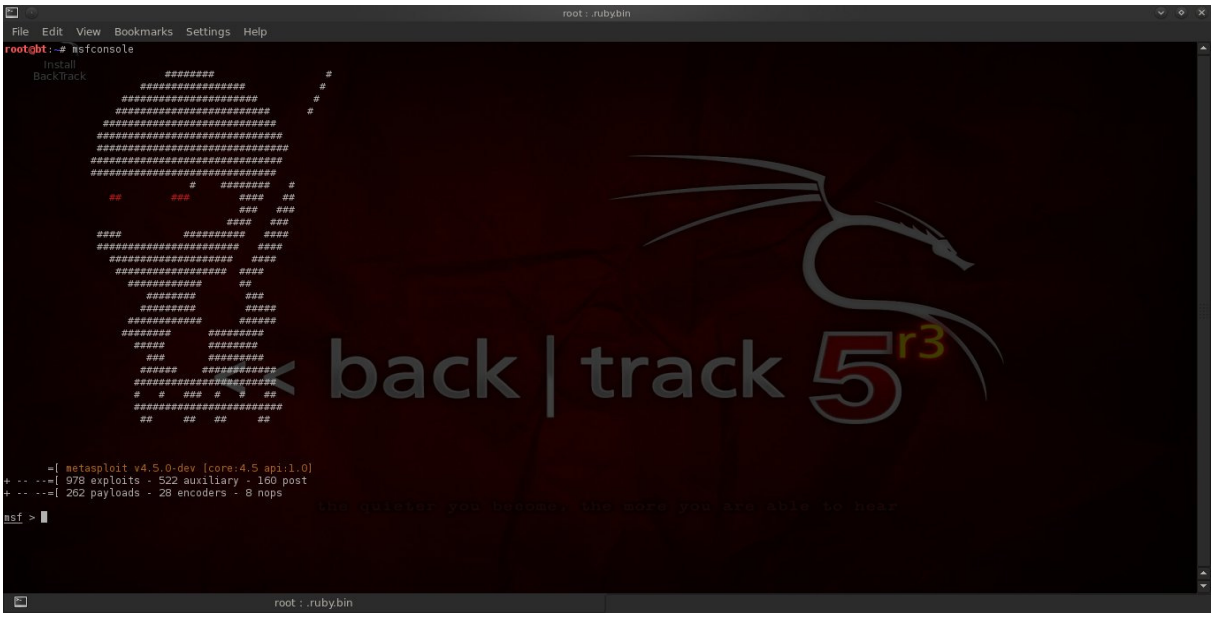
Uporaba ogrodja za vdor v sistem poteka preko naslednjih korakov:

1. Izbira in konfiguracija izkoriščevalske kode.
2. Preverjanje ali je ciljni sistem dovzeten za izbrano izkoriščevalsko kodo (opcijsko).
3. Izbira in konfiguracija bremena (angl. payload). Breme je koda, ki se bo na sistemu izvedla po uspešnem vdoru.
4. Izbira tehnike kodiranja bremena z namenom, da sistemi za zaznavanje in preprečevanje vdorov (angl. Intrusion Detection and Prevention Systems – kratica IDPS) bremena ne zaznajo.
5. Izvedba izkoriščevalske kode.

Z MSF je mogoče upravljati preko štirih uporabniških vmesnikov:

- *msfconsole*: interaktivni vmesnik v ukazni vrstici;
- *msfcli*: neinteraktivni vmesnik v ukazni vrstici;
- *msfweb*: spletni vmesnik;
- *msfgui*: grafični vmesnik.

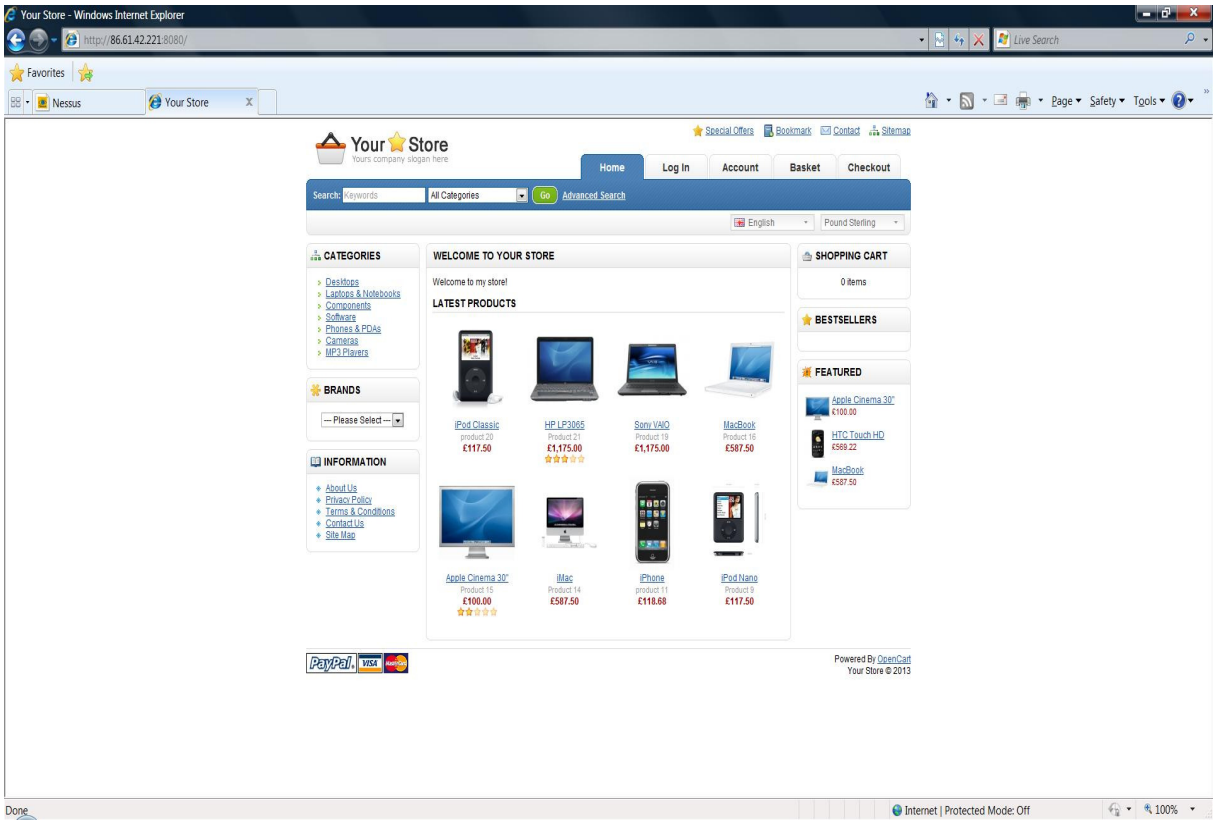
Glavni in najbolj priljubljen je vmesnik *msfconsole* z lastnim naborom ukazov.



Slika 5: Ukazna vrstica msfconsole – začetni zaslon

6 PRIMER PENETRACIJSKEGA PREIZKUSA

V tem poglavju bomo prikazali penetracijski preizkus na primeru spletne trgovine, ki teče na testnem strežniku. Z naročnikom smo se za začetek dogovorili za izvedbo preizkusa z metodo črne škatle, se pravi za poizkus zunanjega vdora. Informacij o ciljnem sistemu torej nimamo, posredovan nam je bil samo naslov IP spletnega strežnika. Po vnosu naslova IP v spletni brskalnik vidimo, da na strežniku teče spletna računalniška trgovina OpenCart.



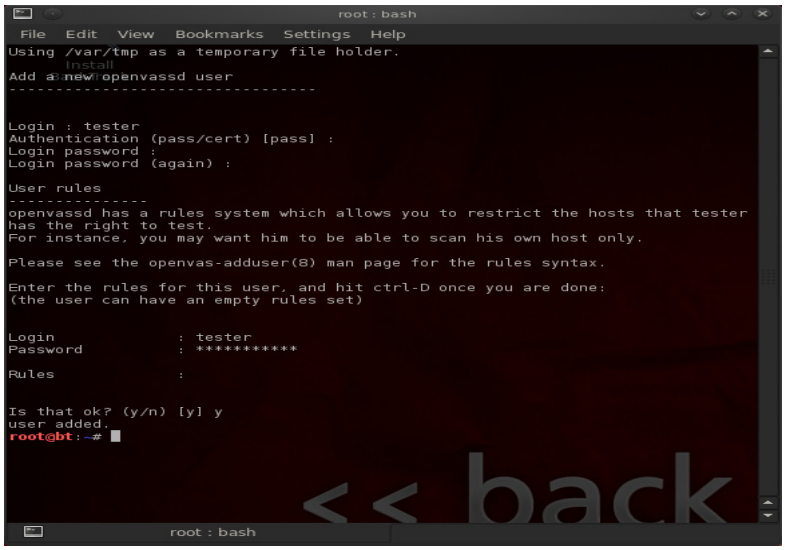
Slika 6: Spletna trgovina na podanem naslovu IP

Pri izvedbi penetracijskega preizkusa bomo uporabili operacijski sistem Linux, distribucijo BackTrack verzijo 5, ki že vsebuje mnogo varnostnih in penetracijskih orodij kot na primer Nmap, OpenVAS, Metasploit itd.

Z orodjem Nmap skeniramo ciljni računalnik z ukazom *nmap --top-ports 100 -sV -O 86.61.42.221*. Kot rezultat dobimo spisek odprtih in zaprtih vrat, tip in verzijo strežniških programov, ki poslušajo na teh vratih ter vrsto in verzijo operacijskega sistema. Na sistemu so odprta vrata 135, 139, 443, 1025, 3306 in 8080. Spletni strežnik je Apache verzija 2.2.6, operacijski sistem ciljnega računalnika pa je z veliko verjetnostjo Microsoft Windows Server 2003.

Slika 7: Rezultat skeniranja s programom Nmap iz zunanjega omrežja

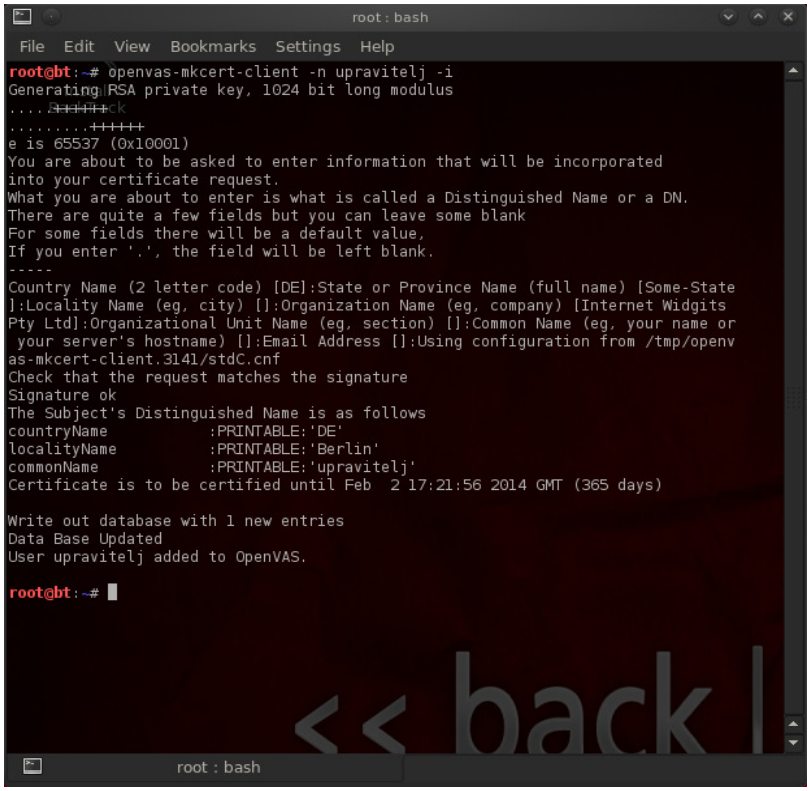
V nadaljevanju bomo ciljni računalnik pregledali s skenerjem ranljivosti OpenVAS, ki pa ga moramo pred prvo uporabo še konfigurirati. Najprej moramo dodati novega uporabnika tako, da v meniju OpenVAS izberemo ukaz *OpenVAS Adduser*. Odpre se terminalsko okno, kamor vnesemo uporabniško ime in geslo za uporabnika, ki ga želimo dodati.



Slika 8: Dodajanje novega uporabnika v ogrodje OpenVAS

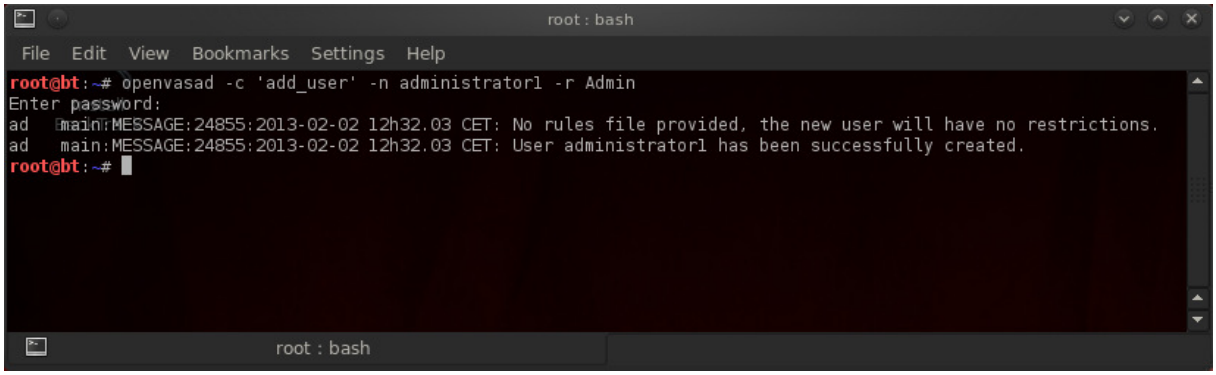
Potem je potrebno kreirati certifikat SSL, kar storimo z uporabo ukaza *OpenVAS Mkcert*. Naslednji korak je sinhronizacija zbirke varnostnih testov NVT z uporabo ukaza *OpenVAS NVT Sync*. S tem smo s spleta prenesli najnovejše teste za odkrivanje ranljivosti. V nadaljevanju poženemo *OpenVAS Scanner*, kar prvič traja kar nekaj časa, saj se v tem koraku preverijo in naložijo vsi novi varnostni testi NVT, ki smo jih prenesli v prejšnjem koraku. Nato ponovno zgradimo bazo z ukazom *openvasmd --rebuild*, kar storimo vedno po posodobitvi varnostnih testov NVT. Nadalje nastavimo komponento OpenVAS Manager

tako, da naredimo certifikat za uporabnika z ukazom *openvas-mkcert-client -n upravitelj -i*, s čimer smo dodali uporabnika z imenom upravitelj.



Slika 9: Dodajanje upravitelja v OpenVAS Manager

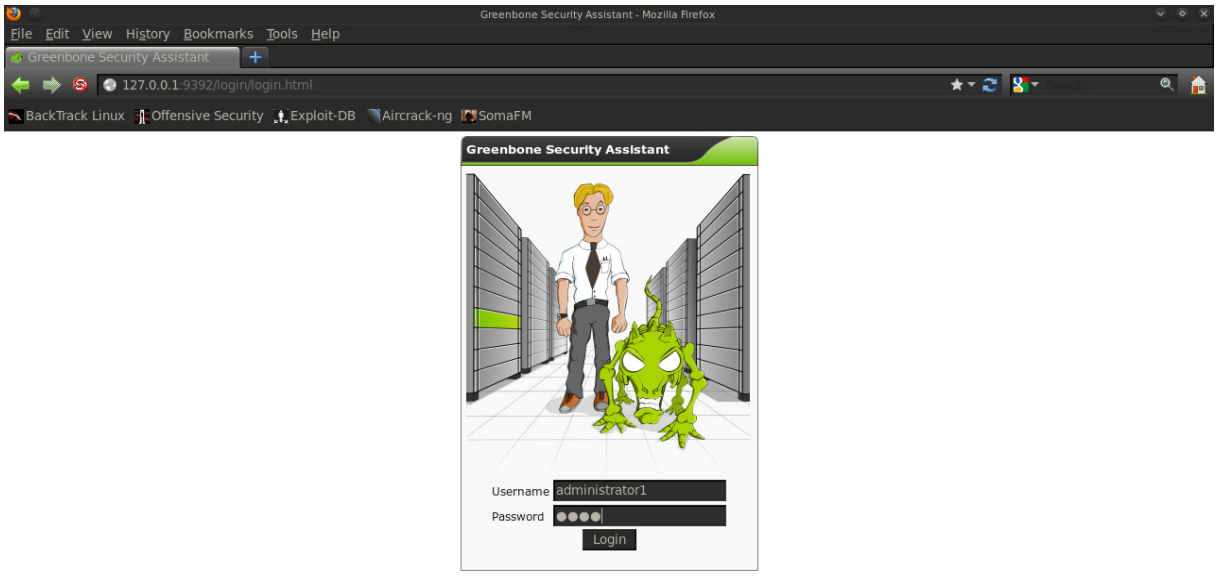
Dodamo še uporabnika s pravicami administratorja, ki ga bomo uporabljali za izvedbo varnostnih skeniranj. To storimo z ukazom *openvasad -c 'add_user' -n administrator1 -r Admin*, kjer smo uporabnika poimenovali administrator1.



Slika 10: Dodajanje administratorja v OpenVAS Administrator

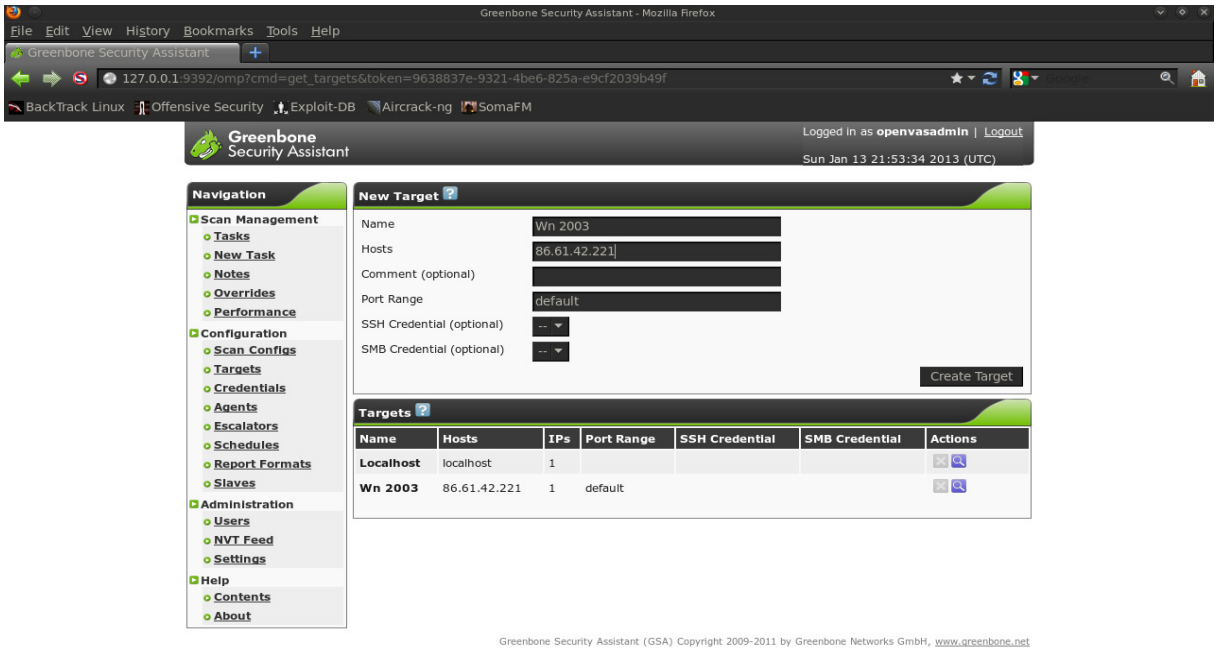
Po končani konfiguraciji komponent OpenVAS Manager in OpenVAS Administrator ju poženemo in potem ti dve storitvi tečeta v ozadju. OpenVAS Manager poženemo z ukazom *openvasmd -p 9390 -a 127.0.0.1*, OpenVAS Administrator pa z ukazom *openvasad -a 127.0.0.1 -p 9393*. Poženemo še Greenbone Security Assistant z ukazom *gsad --http-only --*

listen=127.0.0.1 -p 9392. Vse tri komponente uporabljamo na lokalnem računalniku, zato uporabljamo lokalni naslov IP 127.0.0.1 in privzeta vrata za poslušanje. S tem je konfiguracija ogrodja OpenVAS končana. Za izvedbo skeniranja bomo uporabili spletni uporabniški vmesnik Greenbone Security Assistant, ki ga poženemo z vnosom naslova *127.0.0.1:9392* v spletni brskalnik. Odpre se stran za prijavo, kamor vnesemo uporabniško ime in geslo administratorja, ki smo ga kreirali pred tem.



Slika 11: Prijava v spletni vmesnik Greenbone Security Assistant

Po prijavi v program najprej kreiramo tarčo za skeniranje, tako da jo poimenujemo in vnesemo njen naslov IP.



Slika 12: Kreiranje tarče za skeniranje v spletnem vmesniku Greenbone Security Assistant

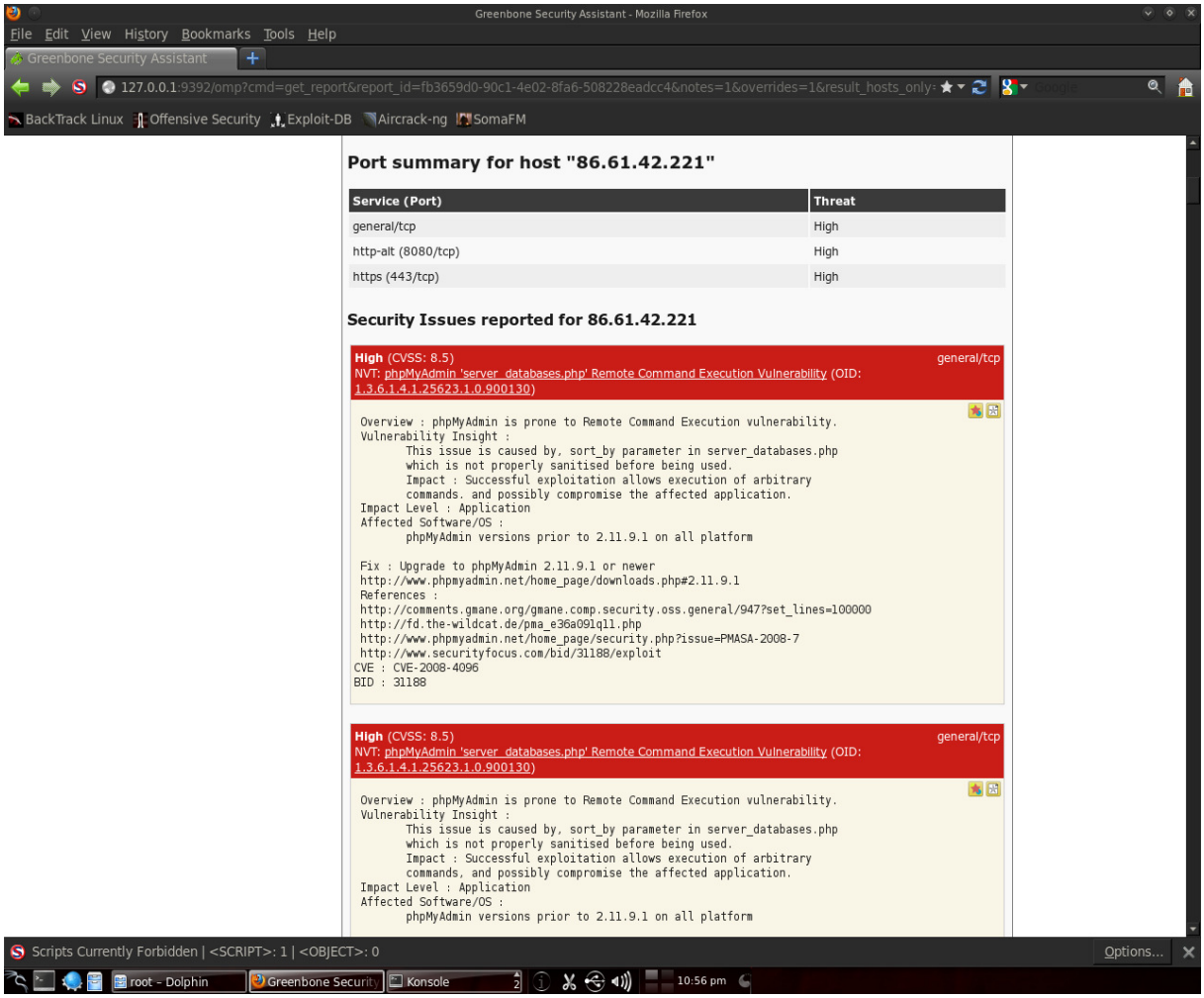
Potem ustvarimo novo opravilo, kjer izberemo našo tarčo in način skeniranja. Privzeto OpenVas ponuja štiri načine za skeniranje:

- *Full and fast*: uporabi večino varnostnih testov NVT in optimizira skeniranje na podlagi predhodno zbranih informacij.
- *Full and fast ultimate*: uporabi večino varnostnih testov NVT vključno s tistimi, ki lahko povzročijo nedelovanje storitev. Skeniranje optimizira na podlagi predhodno zbranih informacij.
- *Full and very deep*: uporabi večino varnostnih testov NVT. Ne zaupa predhodno zbranim informacijam, zato je počasen.
- *Full and very deep ultimate*: uporabi večino varnostnih testov NVT vključno s tistimi, ki lahko povzročijo nedelovanje storitev. Ne zaupa predhodno zbranim informacijam, zato je počasen.

Slika 13: Ustvarjanje novega opravila v spletnem vmesniku Greenbone Security Assistant

Nato sprožimo skeniranje naše tarče in na koncu testiranja pregledamo poročilo. V poročilu so odkrite možne ranljivosti razporejene v tri stopnje: ranljivosti z visoko, srednjo in nizko prioriteto. Za vsako možno ranljivost so navedene tehnične podrobnosti, kot so mesto in opis

napake v sistemu, vpliv na sistem v primeru uspešnega izkoriščenja ranljivosti, predlog za odpravo ranljivosti in reference na zunanje vire z opisom ranljivosti.

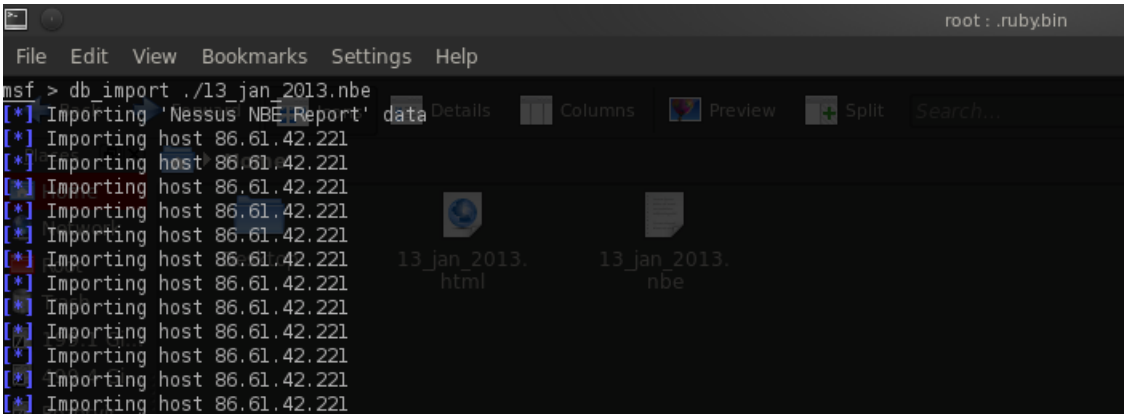


Slika 14: Izgled poročila skeniranja v spletnem vmesniku Greenbone Security Assistant

Poročilo lahko izvozimo v različnih formatih, v našem primeru pa smo se odločili za format NBE, ker bomo to poročilo uporabili v naslednjem koraku penetracijskega preizkusa.

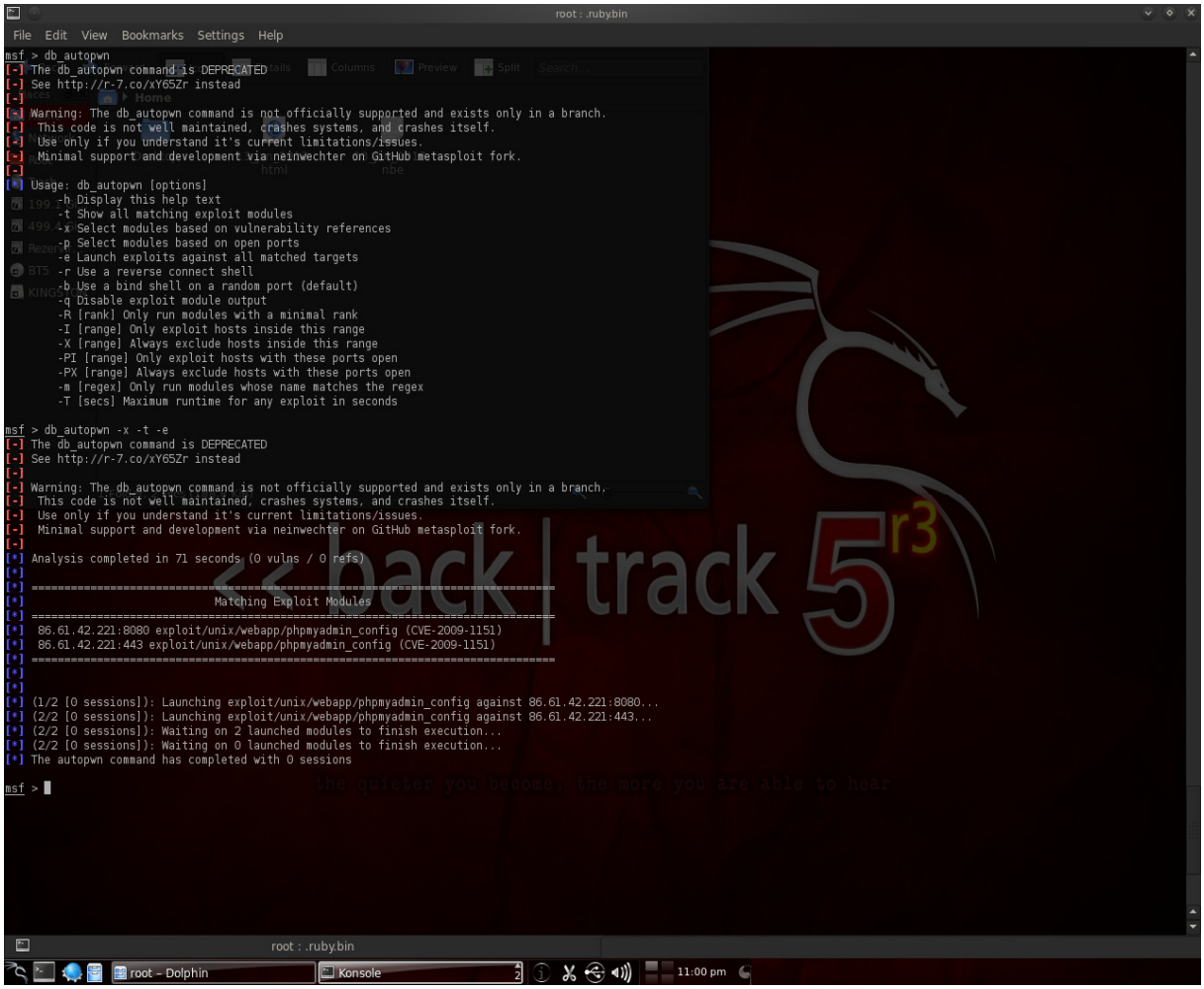
Za izkoriščanje odkritih potencialnih ranljivosti uporabimo ogrodje Metasploit, ki ga poženemo kot vmesnik v ukazni vrstici msfconsole. V MSF lahko ročno poiščemo in izberemo najbolj primerno izkoriščevalsko kodo ter breme za možno ranljivost, ki smo jo odkrili v prejšnjem koraku pri izvedbi skeniranja ranljivosti z ogrodjem OpenVAS. V našem primeru pa bomo uporabili funkcijo *autopwn*, ki bo avtomatsko preverila vse možne odkrite ranljivosti skupaj z zbirko izkoriščevalskih kod. V primeru ujemanja se bo izkoriščevalska koda izvedla in če bo uspešna, se bo odprla komunikacijska seja med našim računalnikom in napadenim sistemom. V tem primeru bomo torej uspešno vdrli v sistem.

Najprej v Metasploit uvozimo poročilo varnostnega skeniranja z ogrodjem OpenVAS. To storimo z ukazom *db_import ./ime_poročila.nbe*.



Slika 15: Uvoz poročila iz skenerja ranljivosti OpenVAS v ogrodje Metasploit

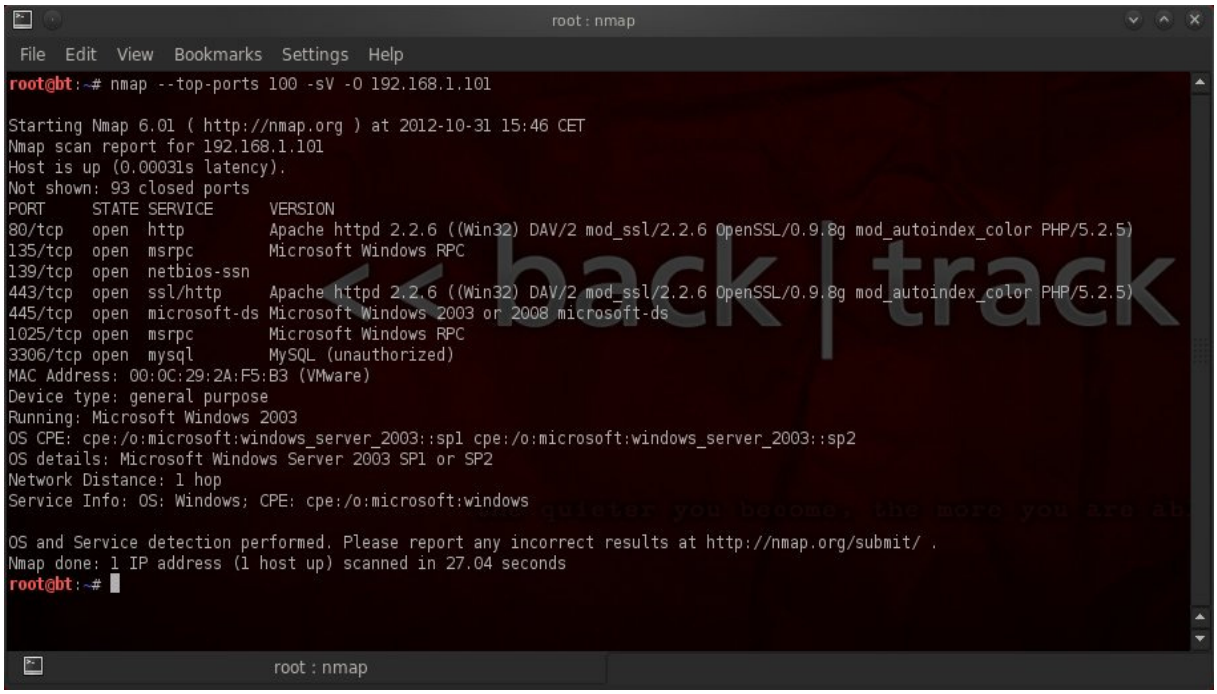
Nato poženemo funkcijo autopwn z ukazom *db_autopwn -x -t -e*. S parametrom -x povemo, naj funkcija izbere izkoriščevalsko kodo glede na ujemanje z možnimi ranljivostmi iz uvoženega poročila. Parameter -t prikaže primerno izkoriščevalsko kodo in parameter -e to kodo izvede.



Slika 16: Napad na ciljni sistem iz zunanjega omrežja z ogrodjem Metasploit in funkcijo autopwn

Po izvršitvi funkcije vidimo, da je bila primerna ena izkoriščevalska koda, ki se je izvedla na vratih 8080 in 443. Koda se ni izvedla uspešno, tako da zunanji napad ni uspel in nismo pridobili dostopa do napadenega sistema.

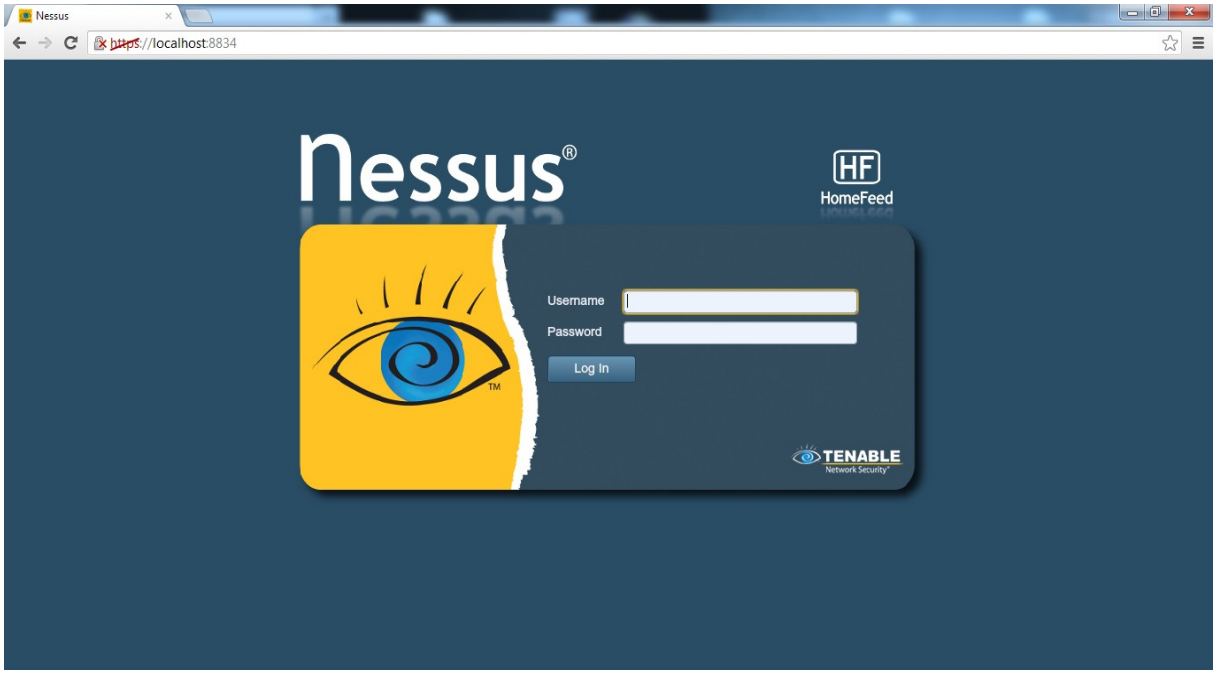
Po dogovoru z naročnikom bomo izvedli še preizkus z metodo bele škatle, torej poizkus vdora iz notranjega omrežja. Lokalni naslov IP spletnega strežnika je v tem primeru *192.168.1.101*. Po skeniranju s programom Nmap vidimo, da so na ciljnem računalniku odprta dodatna vrata 445.



Slika 17: Rezultat skeniranja s programom Nmap iz notranjega omrežja

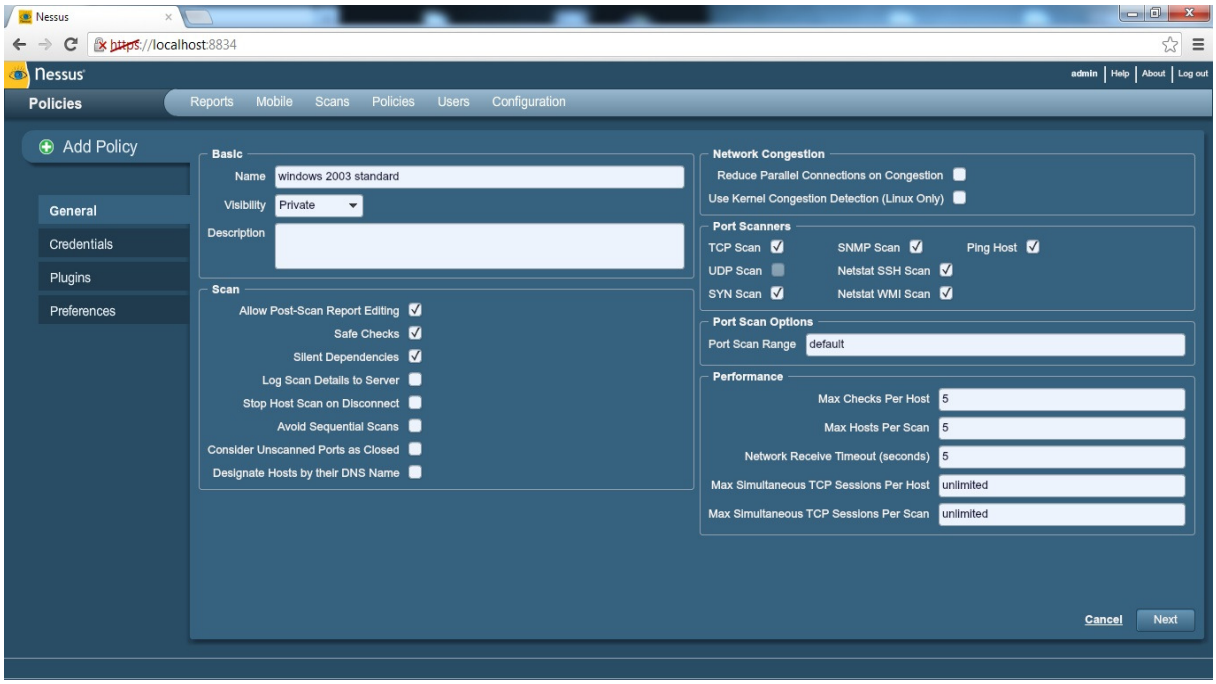
V primeru penetracijskega preizkusa notranjega omrežja bomo uporabili skener ranljivosti Nessus. Namestili in pognali ga bomo iz operacijskega sistema Windows, ki je nameščen na računalniku, ki se nahaja v notranjem omrežju.

Po namestitvi je potrebno program registrirati in kreirati novega uporabnika. Po končani registraciji se s spleta naložijo še najnovejši varnostni testi. Prepričamo se, da je zagnan Nessus strežnik kot proces z imenom *nessusd.exe*, nato pa poženemo Nessus odjemalec z vnosom lokalnega naslova IP in privzetih vrat *8834* v spletni brskalnik. Odpre se prijavno okno, kamor vnesemo podatke uporabnika, ki smo ga kreirali ob registraciji programa.



Slika 18: Prijava v spletni uporabniški vmesnik Nessus

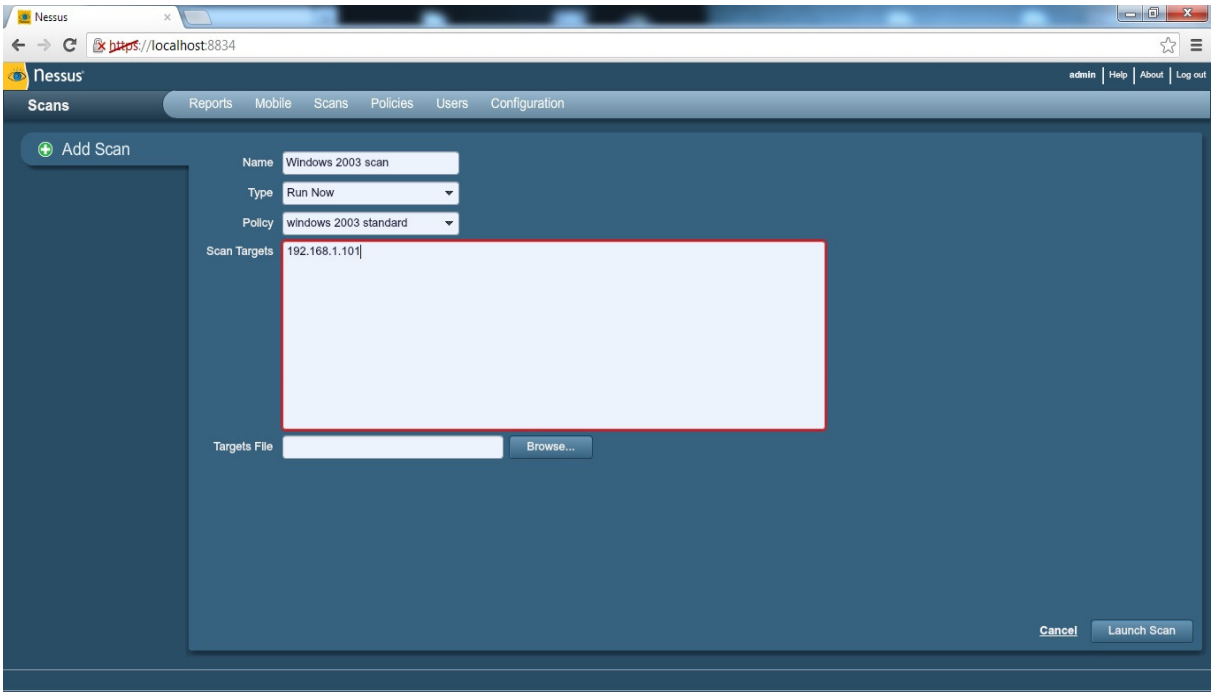
Po prijavi v program nastavimo politiko skeniranja. V zavihku z imenom *General* politiko najprej poimenujemo. Izberemo lahko izvajanje varnih testov, ki ne bodo zaustavili delovanja storitev na naši tarči. Definirati je možno točno določena vrata in tehniko skeniranja ter parametre, ki vplivajo na hitrost skeniranja. Če imamo na voljo uporabniške podatke za prijavo na našo tarčo, jih lahko vnesemo pod zavihkom z imenom *Credentials*.



Slika 19: Nastavitev politike skeniranja v programu Nessus

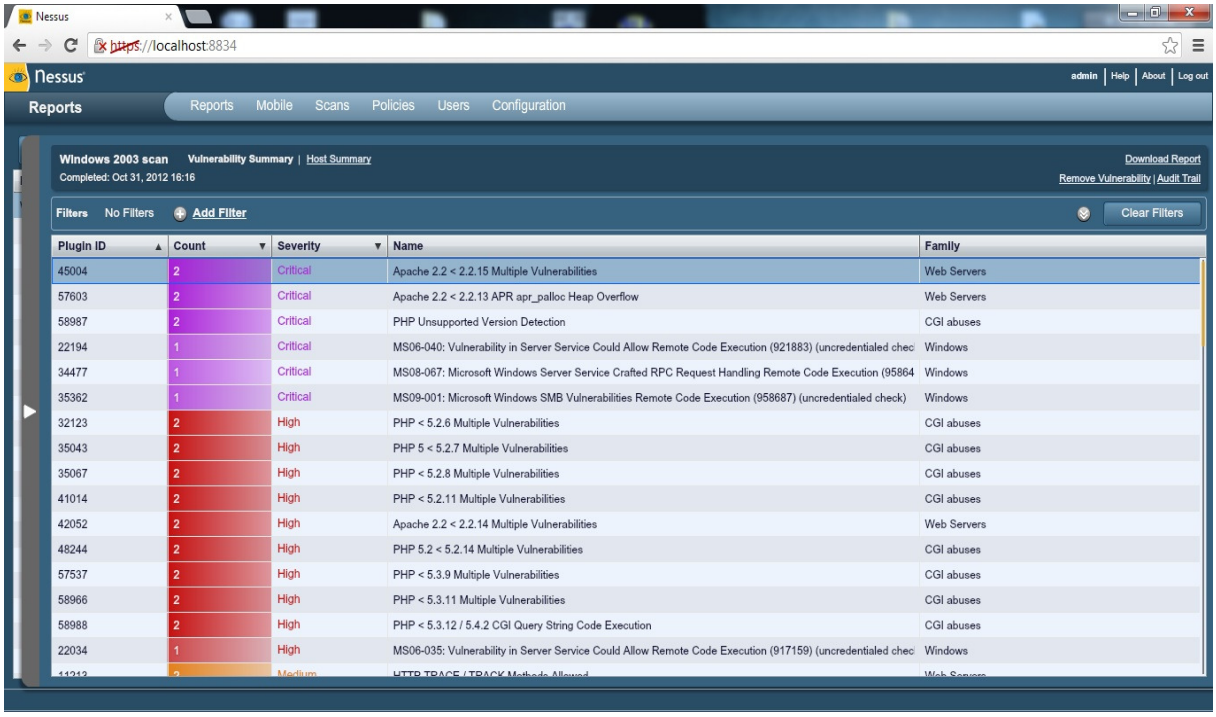
Pod zavihkom z imenom *Plugins* izberemo dodatke oziroma varnostne teste, ki jih bomo uporabili za skeniranje S tem politiko priredimo glede na operacijski sistem in instalirano programsko opremo tarče; na primer za skeniranje operacijskega sistema Windows in strežnika Windows ali za skeniranje operacijskega sistema Linux na katerem teče strežnik Apache.

Ko končamo s kreiranjem politike, izberemo tarčo za skeniranje. Tarčo poimenujemo, dodamo njen naslov IP in izberemo politiko skeniranja, ki smo jo kreirali pred tem.



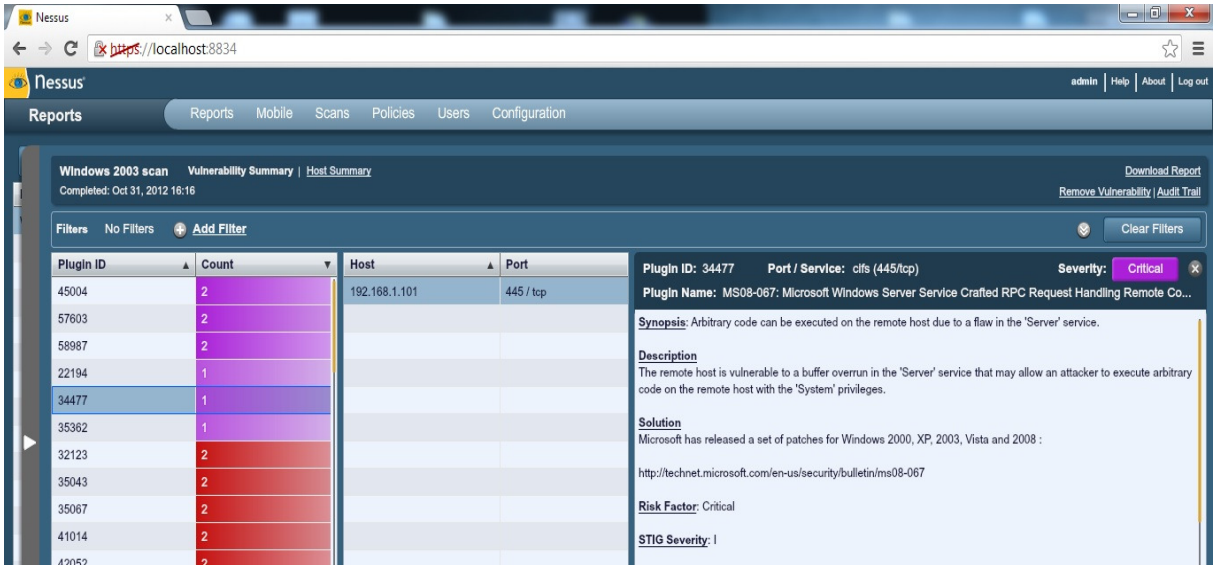
Slika 20: Izbira tarče za skeniranje v programu Nessus

Sprožimo skeniranje tarče. Po končanem skeniranju pregledamo poročilo. Možne ranljivosti so razporejene v več stopenj: kritične ranljivosti, ranljivosti z visoko, srednjo in nizko prioriteto. Vsaka ranljivost je podrobno opisana skupaj z načinom možnega izkoriščenja. Naveden je predlog za odpravo ranljivosti in povezava na zunanje vire z opisom ranljivosti.



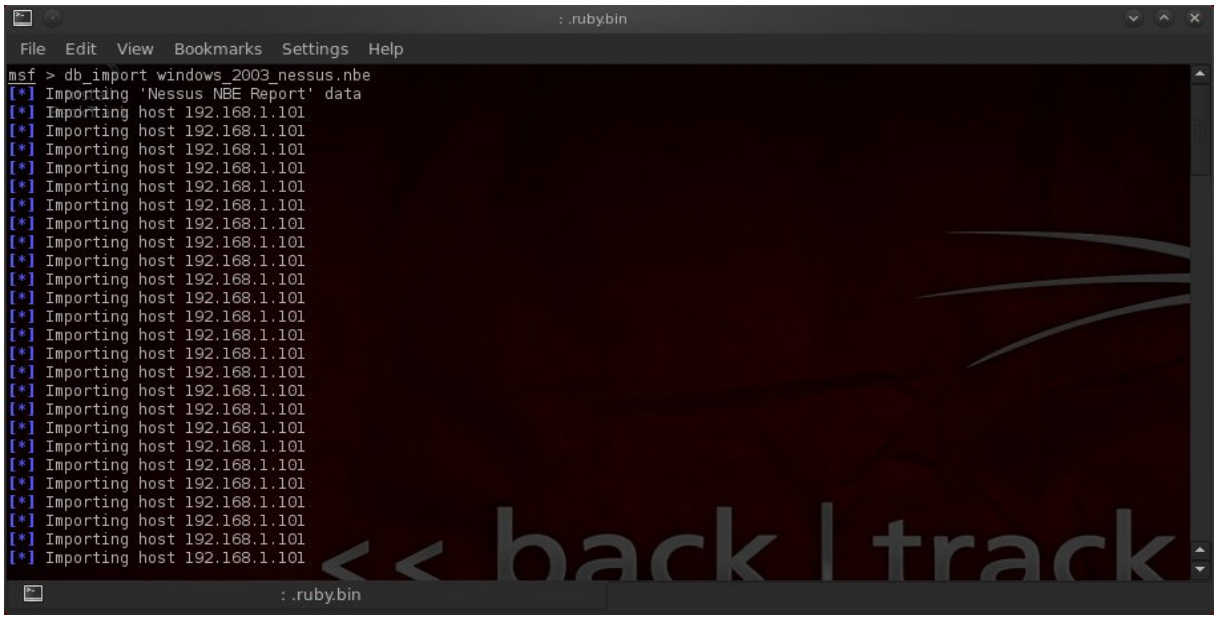
Slika 21: Pregled poročila skeniranja v programu Nessus

Za nas so zanimive predvsem možne ranljivosti z oznako kritično, ki se nanašajo na manjkajoče varnostne popravke podjetja Microsoft. Te ranljivosti so mnogokrat povezane direktno z izkoriščevalsko kodo v ogrodju Metasploit. V poročilu najdemo več takih ranljivosti, za katere je navedeno, da omogočajo oddaljeno izvedbo izkoriščevalske kode. Ena izmed njih je tudi manjkajoč varnostni popravek *MS08-067*.



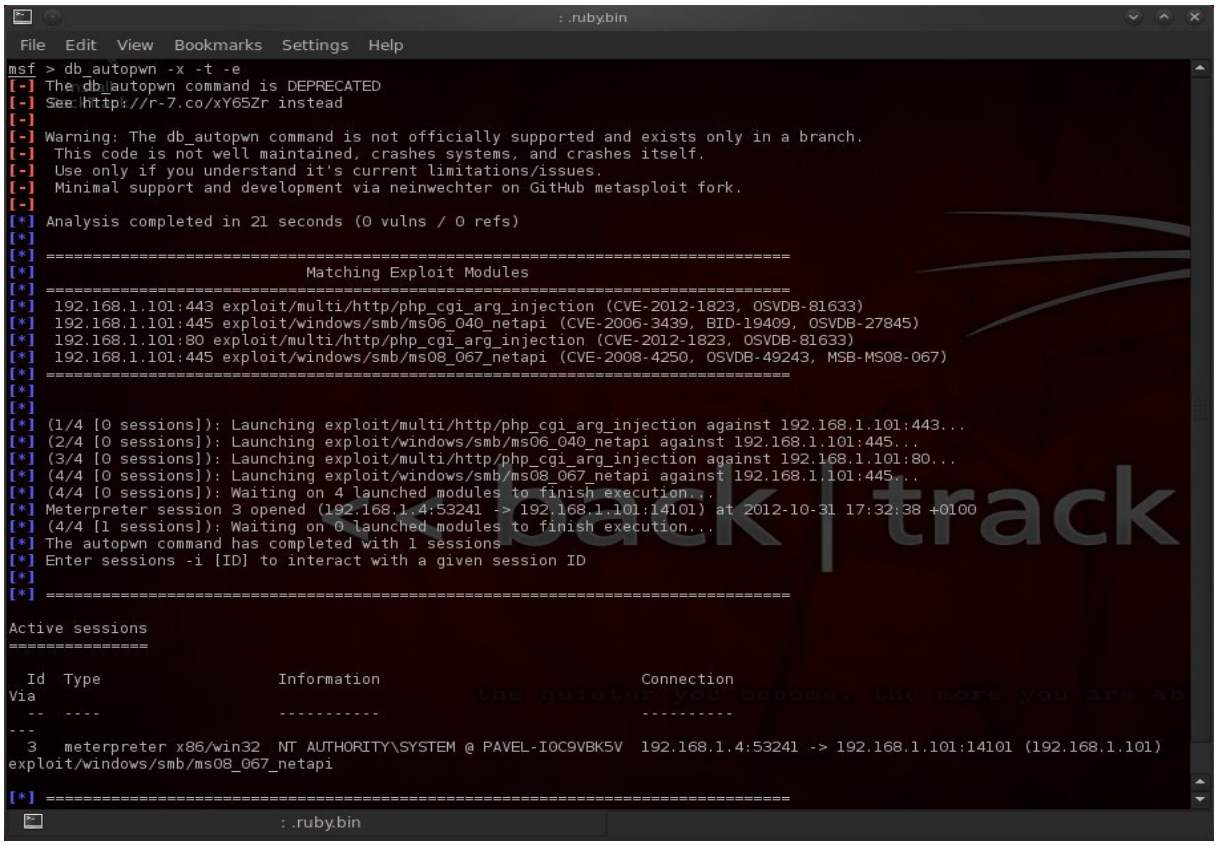
Slika 22: Manjkajoč varnostni popravek MS08-067

Poročilo lahko izvozimo v različnih formatih, odločimo pa se zopet za format NBE, ker bomo poročilo uvozili v ogrodje Metasploit.



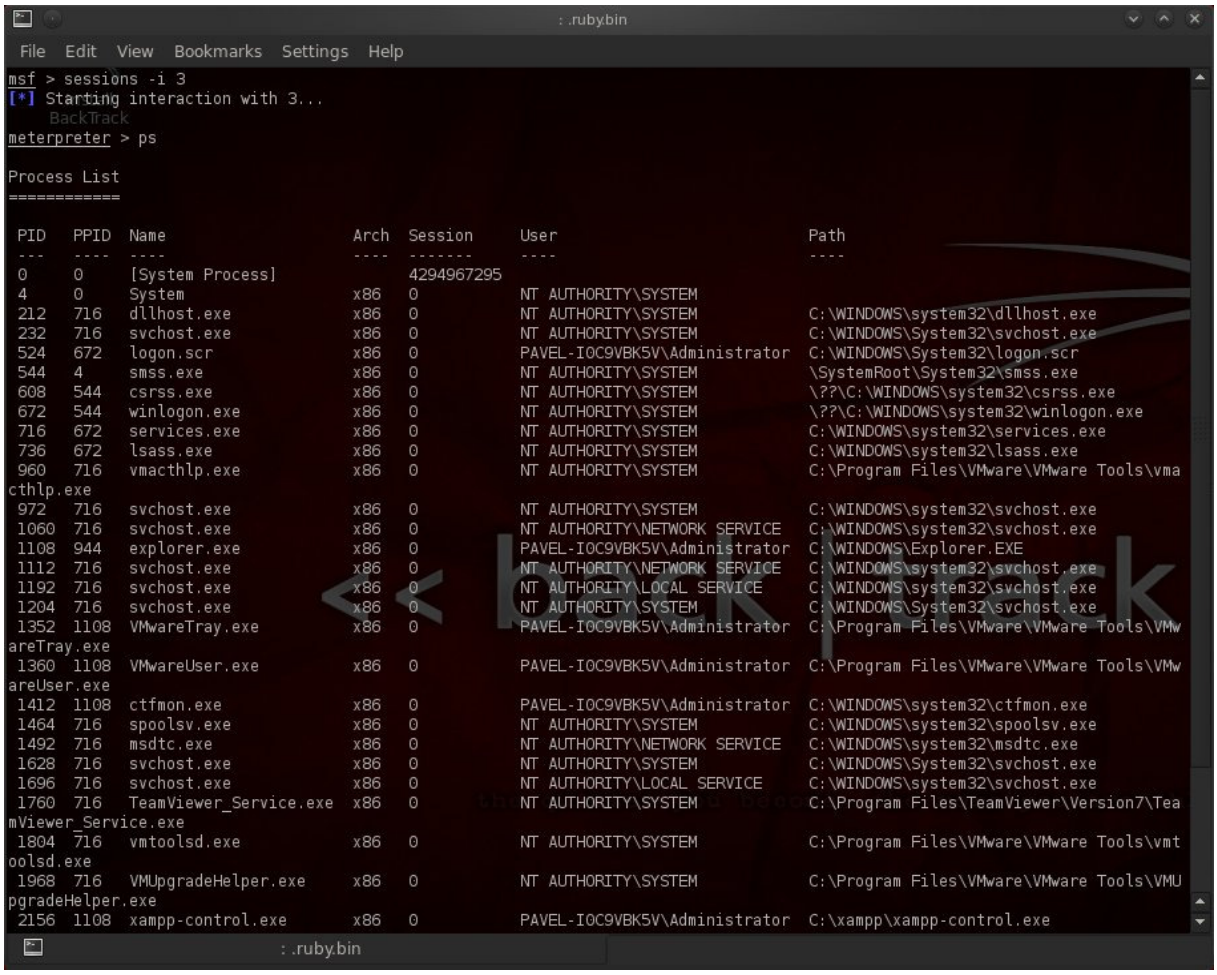
Slika 23: Uvoz poročila iz programa Nessus v ogrodje Metasploit

Ponovno uporabimo avtomatsko funkcijo *autopwn*. Po izvršitvi funkcije vidimo, da uvoženim ranljivostim ustrezajo trije sklopi zlonamerne kode, ki se izvedejo na vratih 80, 443 in 445. Koda se je v enem primeru uspešno izvedla.



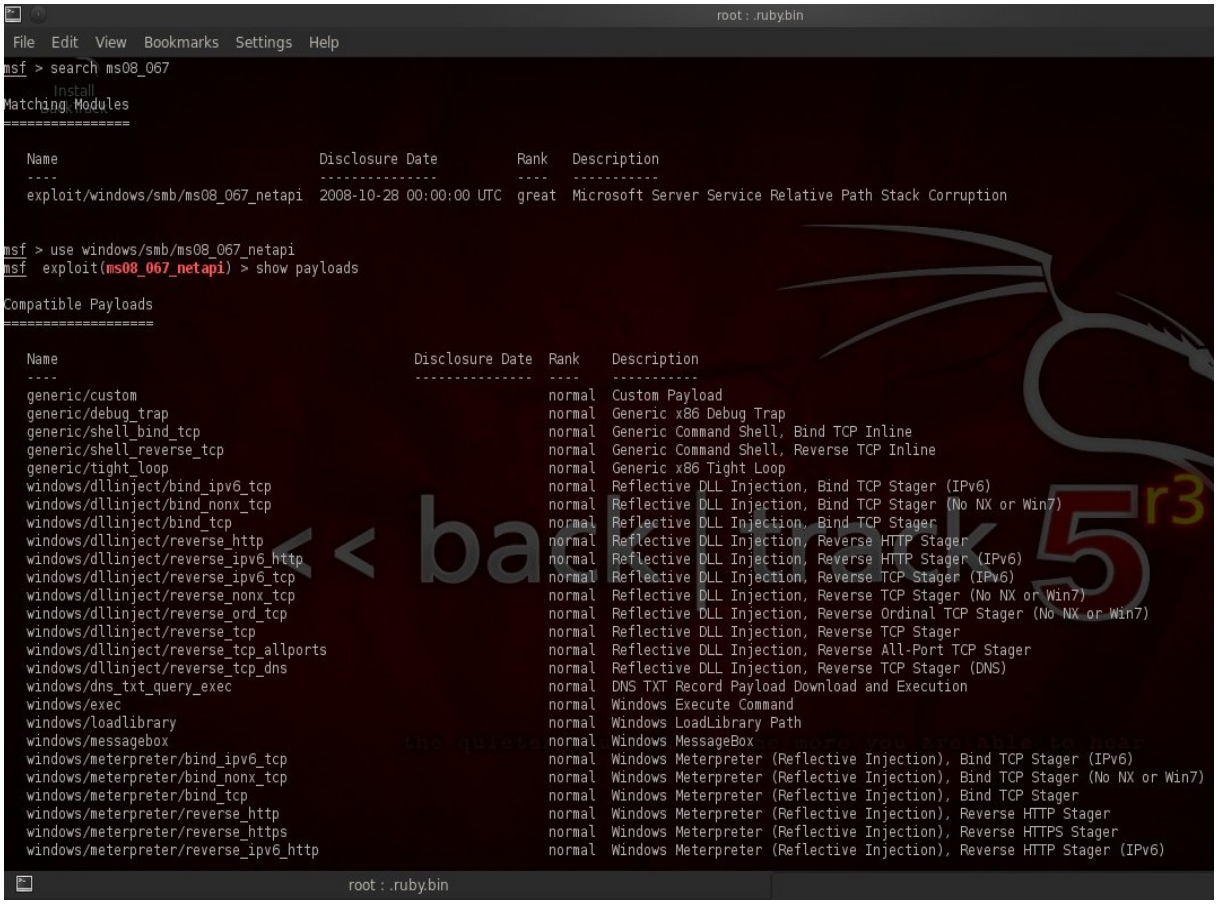
Slika 24: Napad na ciljni sistem iz notranjega omrežja z ogrodjem Metasploit in funkcijo autopwn

Po izvršitvi izkoriščevalske kode se je na ciljnem računalniku izvedlo še breme *Meterpreter* in odprla se je komunikacijska seja s številko 3. V nadaljevanju vstopimo v odprto sejo z ukazom *sessions -i 3* in z vnosom ukaza *ps* prikažemo vsa opravila, ki trenutno tečejo na napadenem računalniku. Imamo poln dostop do naše tarče preko napredne ukazne vrstice Meterpreter.



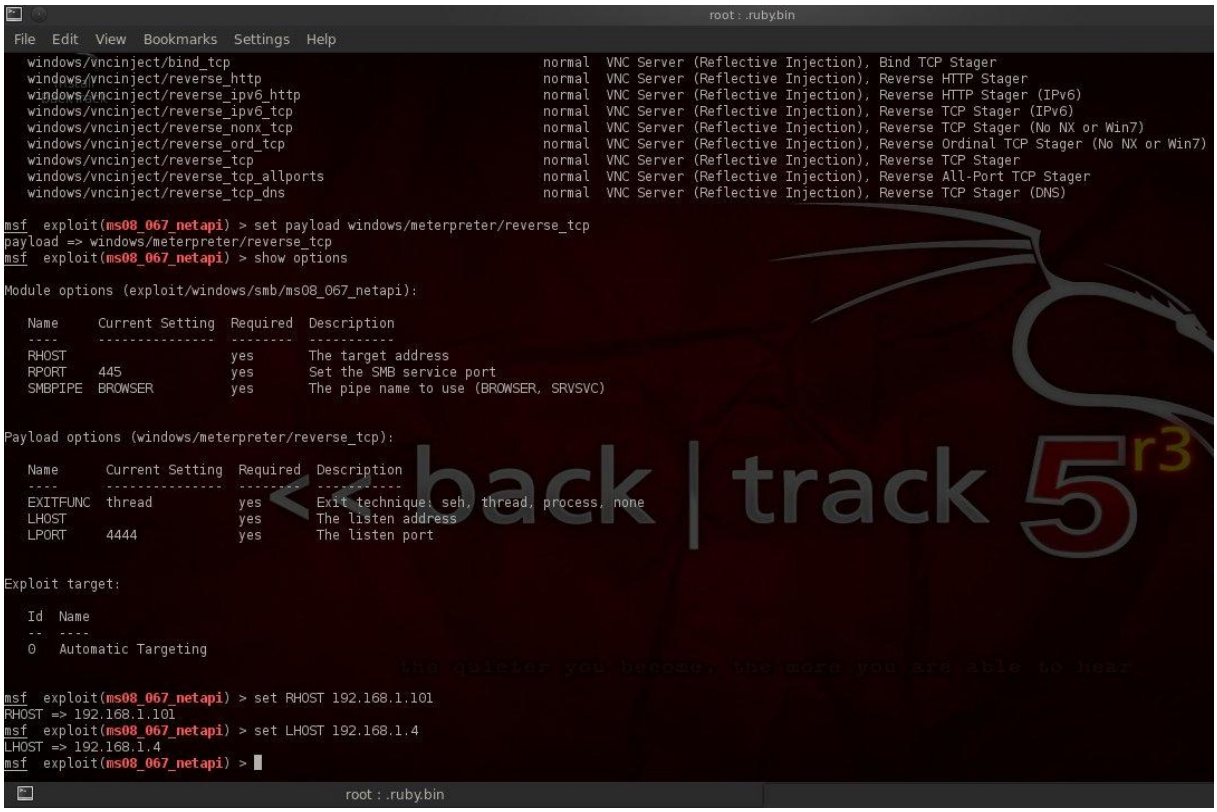
Slika 25: Prikaz procesov, ki tečejo na napadenem računalniku

Namesto uvoza poročila z odkritimi ranljivostmi in uporabe avtomatske funkcije *autopwn* v ogrodju Metasploit, lahko postopek izkoriščanja ranljivosti izvedemo tudi ročno. V ogrodju Metasploit poiščemo primerno izkoriščevalsko kodo z vnosom ukaza *search ms08-067*. Najdemo kodo z imenom *windows/smb/ms08_067_netapi*. Kodo uporabimo z ukazom *use ime_kode*. Primerna bremena, ki so na voljo za to izkoriščevalsko kodo, poiščemo z vnosom ukaza *show payloads*.

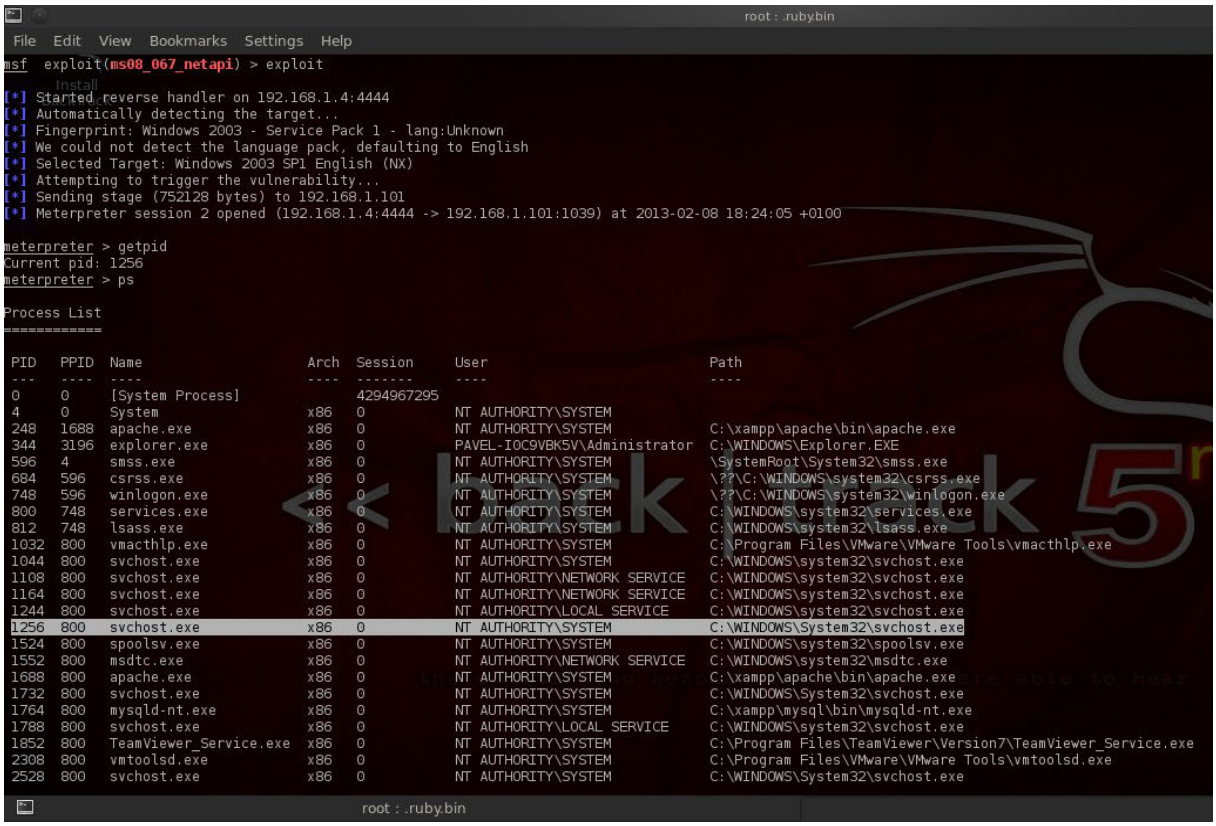


Slika 26: Iskanje primerne izkoriščevalske kode in bremena v ogrodju Metasploit

V našem primeru želimo zagnati breme, ki nam bo omogočilo ukazno vrstico na ciljnem sistemu. Na voljo imamo standardno ukazno vrstico *shell* in napredno ukazno vrstico *Meterpreter*. Standardna ukazna vrstica na napadenem operacijskem sistemu vzpostavi nov proces z imenom *cmd.exe*. To pomeni, da naše početje lahko hitreje zazna protivirusni program ali sistem za zaznavanje vdorov. Napredna ukazna vrstica Meterpreter se »naseli« v spomin poljubnega procesa na napadenem računalniku in tako ostane prikrita. Ker ima lahko naša tarča vklopljen požarni zid, izberemo tako breme, da se bo tarča sama povezala nazaj na naš računalnik. Tako breme je na primer *windows/meterpreter/reverse_tcp*. Breme zaženemo z ukazom *set payload ime_bremena*. Z vnosom ukaza *show options* preverimo, katere parametre moramo nastaviti, da bosta koda in breme delovala. Nastaviti moramo še naslov IP tarče, kar storimo z ukazom *set RHOST 192.168.1.101* ter naslov našega računalnika, kar storimo z ukazom *set LHOST 192.168.1.4*. Vrata tarče *445* so že nastavljena, prav tako vrata na našem računalniku *4444*, kamor se tarča poveže nazaj.



Slika 27: Nastavitev izkoriščevalske kode in bremena v ogrodju Metasploit



Slika 28: Napad na ciljni sistem iz notranjega omrežja z ogrodjem Metasploit in prikaz procesa, na katerega se je prilepilo breme Meterpreter

Nazadnje poženemo še izvedbo izkoriščevalske kode in bremena z ukazom *exploit*. Kot rezultat se odpre komunikacijska seja med našim računalnikom in tarčo, tako kot v primeru uporabe funkcije *autopwn*. Z ukazom *getpid* preverimo, na kateri proces na napadenem računalniku se je prilepilo breme Meterpreter. Kot rezultat poizvedbe dobimo proces *svchost.exe* z identifikacijsko številko 1256. Postopek je prikazan na sliki 28.

Napad na ciljni sistem izveden iz notranjega omrežja nam je torej uspel in s tem, ko smo odkrito ranljivost uspešno izkoristili, smo tudi potrdili njen obstoj. Na koncu penetracijskega preizkusa naročniku podamo poročilo o odkritih ranljivostih sistema, izpostavimo tiste, ki smo jih uspešno izkoristili ter predlagamo ukrepe za odpravo teh ranljivosti. V našem primeru bi bilo potrebno sistem posodobiti z najnovejšimi varnostnimi popravki.

7 SKLEPNE UGOTOVITVE

V diplomskem delu smo predstavili penetracijski preizkus in vlogo penetracijskega preizkuševalca pri zagotavljanju varnosti informacijskih sistemov. Pri tem smo jasno začrtali mejo med zakonitim preverjanjem varnosti informacijskega sistema in med nedovoljenim vdorom vanj. Pojasnili smo zmedo pri enačenju penetracijskega preizkusa z oceno ranljivosti.

Ugotovili smo, da se je za uspešen zaključek penetracijskega testa potrebno držati ene izmed uveljavljenih metodologij na tem področju. Testiranje bi bilo potrebno izvajati večkrat letno, saj zlonamerni hekerji stalno izboljšujejo izkoriščevalsko kodo in iščejo nove možnosti za vdor. Za preizkus določenega sistema je dobro uporabiti več penetracijskih preizkuševalcev, oziroma naj naslednji preizkus sistema izvedejo drugi preizkuševalci. Vsak preizkuševalec ima namreč drugačen pristop in znanje ter lahko odkrije ranljivosti, ki jih je njegov »sodelavec« spregledal.

Pri uporabi orodij za penetracijske teste smo se odločili za brezplačna orodja, ki so prosto dostopna. Za skener ranljivosti smo uporabljali brezplačno odprtokodno orodje OpenVAS in še komercialni skener Nessus, ki pa je za domačo uporabo brezplačen. V obeh primerih gre za kvalitetna produkta, menimo pa, da je za začetno konfiguracijo orodja OpenVAS potrebno več znanja. Nessus je bolje dokumentiran in vsebuje večjo količino varnostnih testov, kar omogoča uporabniku večjo fleksibilnost pri testiranju. Pri uporabi ogrodja Metasploit je izbira primerne zlonamerne kode in njena ročna konfiguracija za začetnike lahko zapletena. Marsikatera koda se ne izvede uspešno zaradi nepopolne ali napačne konfiguracije, kar je posledica pomanjkljivega znanja uporabnika. Smiselno bi bilo izboljšati funkcijo autopwn in odpraviti njene trenutne omejitve.

Pri poizkusu vdora v sistem smo pri ročnem testiranju izkoriščanja možnih ranljivosti poizkusili dokazati eno izmed ranljivosti. Za prikaz postopka penetracijskega preizkusa to zadošča. Dokazovanje vseh možnih ranljivosti bi zahtevalo veliko več časa in znanja s področja penetracijskega testiranja. Za izkoriščanje možnih ranljivosti smo uporabili že obstoječo izkoriščevalsko kodo, ki jo ponuja samo orodje. Poklicni penetracijski preizkuševalci pa bi izkoriščevalsko kodo lahko napisali tudi sami.

Neuspešen poizkus vdora v sistem še ne pomeni, da je sistem dobro zaščiten. V vsak informacijski sistem je možno vdreti, ovira sta le čas in znanje, ki sta potrebna za izvedbo. Zavedati se je potrebno, da bo v prihodnosti vdorov v informacijske sisteme vedno več. Slabe finančne razmere in življenjske stiske silijo ljudi v zlonamerna dejanja, kot so pridobitev materialnih koristi ali denarja na lahek, čeprav nezakonit način.

SEZNAM SLIK

LITERATURA IN SPLETNI VIRI

[1] Tomaž Bratuša, *Hekerski vdori in zaščita*, 2. razširjena izdaja, Ljubljana: Pasadena, 2006

[2] Patrick Engebretson, *The Basics of Hacking and Penetration Testing,* Rockland: Syngress Media, 2011

[3] Allen Harper, Shon Harris, Jonathan Ness, Chris Eagle, Gideon Lenkey, Terron Williams, *Gray Hat Hacking: The Ethical Hacker's Handbook,* 3rd Edition, New York: McGraw-Hill, 2011

[4] Spletna enciklopedija Wikipedija. Penetration test. Dostopno na: http://en.wikipedia.org/wiki/Penetration_test (uporabljeno: september 2012)

[5] Spletna enciklopedija Wikipedija. Hacker (computer security). Dostopno na: http://en.wikipedia.org/wiki/Hacker_(computer_security) (uporabljeno: september 2012)

[6] (2006) Stephen Northcutt, Jerry Shenk, Dave Shackleford, Tim Rosenberg, Raul Siles, Steve Mancini. Penetration Testing: Assessing Your Overall Security Before Attackers Do. Dostopno na: http://www.sans.org/reading_room/analysts_program/PenetrationTesting_June06.pdf (uporabljeno: september 2012)

[7] (2002) SANS Institute. Conducting a Penetration Test on an Organization. Dostopno na: http://www.sans.org/reading_room/whitepapers/auditing/conducting-penetration-test-organization_67 (uporabljeno: oktober 2012)

[8] Manish S. Saindane. Penetration Testing - A Systematic Approach. Dostopno na: http://www.infosecwriters.com/text_resources/pdf/PenTest_MSaindane.pdf (uporabljeno: oktober 2012)

[9] Spletni slovar. Terminološki slovar informatike. Dostopno na: http://www.islovar.org (uporabljeno: november 2012)

[10] Skener omrežja. Nmap. Dostopno na: http://nmap.org/ (uporabljeno: november 2012)

[11] Skener ranljivosti. Nessus. Dostopno na: http://www.tenable.com/products/nessus (uporabljeno: december 2012)

[12] Skener ranljivosti. OpenVAS. Dostopno na: http://www.openvas.org (uporabljeno: december 2012)

[13] Spletna enciklopedija Wikipedija. Metasploit Project. Dostopno na: http://en.wikipedia.org/wiki/Metasploit_Project (uporabljeno: december 2012)

[14] Orodje za izdelavo in uporabo zlonamerne kode. Metasploit. Dostopno na: http://www.metasploit.org/ (uporabljeno: december 2012)