BACHELOROPPGAVE:

Reccoon - automatisert rammeverk for external network footprinting

FORFATTERE:
Jan William Johnsen
Kasper Kristoffersen

DATO:
19.05.2014

Sammendrag av Bacheloroppgaven

Tittel:	Reccoon - automatisert rammeverk for external network footprinting	Nr: 10
		Dato: 19.05.2014
Deltakere:	Jan William Johnsen	
	Kasper Kristoffersen	
Veiledere:	Lasse Øverlier	
Oppdragsgiver:	Watchcom Security Group AS	
Kontaktperson:	Morten Gjendemsjø, Eivind Utnes, Bjørn Richard Watne	
Stikkord	Eksternt nettverksavtrykk	
Antall sider: 84	Antall vedlegg: 9	Tilgjengelighet: Åpen

Kort beskrivelse av bacheloroppgaven:
Under utføringen av penetrasjonstester er det viktig å begynne hele testen med å hente inn informasjon om kunden og deres nettverk. Dessverre er denne informasjonsinnhentingen en prosess som ofte blir nedprioritert. En dårlig utført informasjonsinnhenting kan føre til en mindre angrepsflate for sikkerhetskonsulentene som utfører penetrasjonstesten. I verste fall kan dette føre til at svakheter hos kunden ikke blir oppdaget.

Reccoon er en prototype av en rammeverk som er utviklet for å hjelpe sikkerhetskonsulentene med denne initielle fasen av penetrasjonstestene. Dette gjøres ved at rammeverket automatiserer prosessene i en external network footprinting, som er en modell for teknisk informasjonsinnhenting. Prosjektgruppen har (etter veiledning fra Penetration Testing Execution Standard) analysert hvilken informasjon som burde hentes fra kundens nettverk under en slik operasjon, og hvilke verktøy som er best egnet for å hente nettopp denne informasjonen.

Rammeverket er utviklet for Kali Linux. Kali Linux er et operativsystem som er spesialtilpasset for penetrasjonstesting og digital etterforskning, og alle modulene i rammeverket benytter seg av verktøy som er inkludert i standard-bibliotekene til dette operativsystemet.

I tillegg til å utføre operasjonene som henter inn selve informasjonen så fungerer Reccoon også som et rapporteringsverktøy. Sikkerhetskonsulentene har mulighet til å legge til og redigere objekter i rammeverket, og eksportere resultatene til en oversiktelig HTML-rapport. Dette legger en god grunnmur for de senere fasene av penetrasjonstesten.

Summary of Graduate Project

Title:	Reccoon - automated framework for external network footprinting	Nr: 10
		Date: 19.05.2014
Participants:	Jan William Johnsen	
	Kasper Kristoffersen	
Supervisor:	Lasse Øverlier	
Employer:	Watchcom Security Group AS	
Contact person:	Morten Gjendemsjø, Eivind Utnes, Bjørn Richard Watne	
Keywords	External Network Footprinting	
Pages: 84	Appendixes: 9	Availability: Open

Short description of the main project:
During the execution of penetration tests it's important to start the test itself by gathering information about the customer and their network. Unfortunately, the information gathering process tend to get discouraged. A poorly executed information gathering process may cause the security consultants to overlook points of interest in the targeted network. In a worst case scenario this may lead to vulnerabilities not beeing identified.

Reccoon is a prototype of a framework developed to assist the security consultants during this initial phase of the penetration test. This is done by automating the processes of an external network footprinting, which is a model for technical information gathering. The project group has (under the guideance of the Penetration Testing Execution Standard) analyzed what information should be extracted from the network during an operation such as this, and also what tools that are most eligible for this task.

The framework is developed for usage in Kali Linux. Kali Linux is an operating system tailored to fit penetration testing and digital forensics operations. The framework's modules utilizes tools included in the operating system's standard libraries.

In addition to manage the operations used to extract the information from the customer network, Reccoon serves as a reporting tool. The security consultants have the possibility to add and edit objects inn the framework, and export the results to a comprehensible HTML-report. This alltogether lays the foundations for the subsequent phases of the penetration test.

Forord

Reccoon har blitt utviklet av to studenter fra Høgskolen i Gjøvik gjennom vårsemesteret 2014. Vi studerer begge informasjonssikkerhet og ønsket oss derfor en oppgave som var rettet mot temaer innen dette fagfeltet. Utviklingen av et rammeverk som automatiserer ekstern network footprinting i starten av en penetrasjonstest virket derfor midt i blinken for oss. Det å jobbe med utviklingen av et slikt rammeverk er lærerikt på svært mange forskjellige områder. Vi må tilegne oss kunnskap om tekniker og verktøy for penetrasjonstesting, databehandling, programmering og utvikling. Dette er kunnskaper vi har stor tro på at kommer til nytte senere i skolegang eller arbeidsliv.

Det å få lov til å skrive en oppgave for et så respektabelt selskap som Watchcom Security Group AS er inspirerende i seg selv. Vi vil gjerne takke våre kontaktpersoner hos Watchcom: Morten Gjendemsjø, Eivind Utnes og Bjørn Richard Watne. Dere har vært hjelpsomme og hyggelige å arbeide sammen med. Vi vil også takke resten av selskapet for at vi fikk komme på besøk og se hvordan selskapet jobber, såvel som invitasjon til sikkerhetskonferansen Paranoia 2014.

Vi vil også gjerne takke vår veileder under bacheloroppgaven: Lasse Øverlier. Vi har begge to tidligere hatt Lasse som foreleser i faget *Ethical hacking and penetration testing*, og det er ingen tvil om at dette har vært en veileder som har passet oppgaven vår utmerket. Lasse har kunnet hjelpe oss med både tekniske aspekter under utvikling av rammeverket, såvel som de mer administrative områder i forbindelse med rapportskrivingen.

Til slutt vil vi takke alle medstudenter som har hjulpet oss med testing av rammeverket, og Jonas Kristoffersen som har hjulpet oss med design av logo.

Gjøvik, 17.05.2014

Jan William Johnsen

Kasper Kristoffersen

Innhold

Figurer

Tabeller

Definisjoner

Penetrasjonstest: Et avtalt angrep på et system med hensikt å avdekke svakheter.

Portskanning: Søking på et datanettverk for å avdekke hvilke tjenester enhetene kjører.

Nmap: Populært verktøy for portskanning.

Geolocation: Den geografiske plasseringen til et objekt

External network footprinting: En fase i en informasjonsinnhentingsprosess som krever interaksjon med målet for å avdekke hvordan nettverket ser ut for en ekstern aktør.

Host En datamaskin. Merk; denne kan også være virtuell.

C++: Et høynivå programmeringsspråk, utvidelse av programmeringsspråket C.

PHP: **P**HP: **H**ypertext **P**reprocessor. Et programmeringsspråk, ofte brukt i forbindelse med nettsider.

Metasploit Framework: Et rammeverk for utnyttelse av svakheter i et system (eng: exploits).

Kali Linux: En Linux-distibusjon for penetrasjonstesting og digital etterforskning.

PTES: **P**enetration **T**esting **E**xecution **S**tandard. En standard for utførelse av penetrasjonstester.

Python: Et høynivå programmeringsspråk.

Proof of Concept: Bevis på at et konsept er levedyktig.

XML: **E**xtensible **M**arkup **L**anguage. Et markeringsspråk. Ofte bruk til å dele data mellom systemer.

Request for Comments: En dokumentserie som ofte brukes for å publisere infromasjon om teknologiske standarder.

DNS: **D**omain **N**ame **S**ystem. System som kobler sammen IP-adresser og domenenavn.

OS: **O**perating **S**ystem. Et system for administrasjon av maskinvareressurser.

RAM: **R**andom **A**ccess **M**emory. Arbeidsminnet til en datamaskin.

MySQL: Et administrasjonssystem for relasjonsdatabaser.

Normalisering: Databasedesign for å minimere duplisering av data.

Domene: En navnestreng som brukes for å kunne adressere på Internett.

URL: **U**niform **R**esource **L**ocator. Ofte kalt nettadresse.

IP: **I**nternet **P**rotocol. Brukes i forbindelse med nettverksadressering.

ISP: **I**nternet **S**ervice **P**rovider. Leverandør av tilkobling til Internett.

Bruteforce: Teknikk for å knekke nøkkler ved å prøve og feile.

HTML: **H**yper**T**ext **M**arkup **L**anguage. Et markeringsspråk for formatering av nettsider.

Django: Et Python-basert web-rammeverk.

Apache: En type webserver.

Falsk positiv verifisering: Et testresultat som viser at en hendelse har skjedd, men som i realiteten ikke har skjedd.

Falsk negativ verifisering: Et testresultat som viser at en hendelse ikke har skjedd, men som i realiteten har skjedd.

Cracking: Ikke-tiltenkt metode for å skaffe tilgang til et beskyttet system.

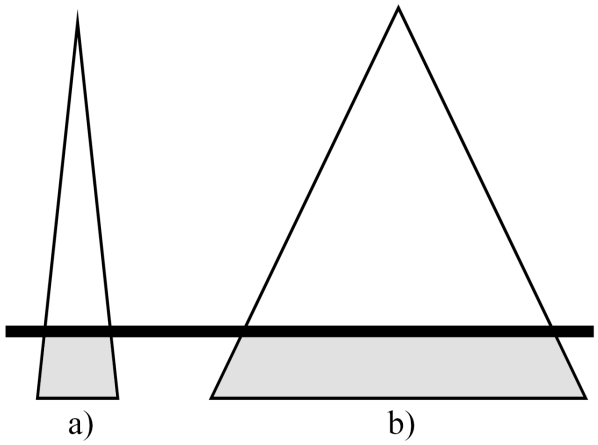
Tråd: Den minste sekvensen som kan administreres av et operativsystem.

1 Introduksjon

Hensikten med introduksjonen er å beskrive hvilket problemområde denne oppgaven adresserer, og hva produktet av denne oppgaven ønsker å gjøre for å forbedre situasjonen. I tillegg inneholder kapitlet en kort introduksjon av prosjektgruppen, oppdragsgiver og veileder.

1.1 Problemområde

Kunder gir ofte en kort tidsfrist for informasjonssikkerhetskonsulenter å gjennomføre penetrasjonstester mot virksomhetsnettverk. Siden informasjonsinnhentingsfasen ikke er like synlig i en rapport som et datainnbrudd, så blir denne fasen ofte oversett eller ikke utnyttet til det fulle [1]. Figur 1 illustrerer at tiden som blir brukt i denne fasen resulterer i størrelsen på angrepsflaten til målnettverket. Konsulentene vil med større sannsynlighet finne flere sårbare veier inn til nettverket hvis angrepsflaten er stor. Penetrasjonstester må planlegges for å få den mest omfattende kartleggingen av maskinvare, programvare og mennesker fra et nettverk. Det er essensielt at den innledende informasjonsinnhentingsfasen blir grundig gjennomført for å gi konsulentene et godt fundament for fortsettelsen av penetrasjonstesten.



Figur 1: Grått felt er innledende informasjonsinnhentingfase, og sort linje er overgangen til portskanningfasen. Trekantene representerer både tid og datamengde, og mengden blir tynnere ved arbeid mot et mål fordi mer spesifikk informasjon blir brukt. Grunnlinjene i trekantene viser at mer tid brukt i informasjonsinnhentingfasen, resulterer i en større datamengde å arbeide med i de neste fasene av penetrasjonstesten. a) kan finne noen av veiene inn i målnettverket, men b) vil finne både flere og mindre synlige veier inn.

1.2 Formål

Prosjektgruppens oppdragsgiver ønsker et rammeverk som automatiserer prosessene for å foreta external network footprinting av et målnettverk [2]. Dette er en del av den innle-

dende informasjonsinnhentingfasen. Rammeverket skal benytte og automatisere flere kjente verktøy for å få et klart bilde av hvordan målnettverket ser ut fra utsiden. Hvert program gir en bit av puslespillet, og kombinajonen av flere verktøy gir et mer helhetlig bilde av nettverket. Verdien av resultatet avhenger av dekning og kvalitet i informasjonen som blir hentet inn, og hvordan denne gjøres tilgjengelig.

1.3 Målgruppe

Rammeverket er først og fremst tiltenkt som et produkt for oppdragsgiver, men det er mulig at andre aktører også ville hatt den samme nytten av produktet. Disse aktørene er trolig også informasjonssikkerhetskonsulenter eller -eksperter som har en bakgrunnsforståelse for hvordan penetrasjonstester blir gjennomført. Prosjektgruppen har tatt utgangspunkt i oppdragsgiver sine arbeidsmetoder for utviklingen av dette rammeverket, og det vil derfor kanskje ikke passe for alle.

Denne rapporten utformes for å være en god introduksjon til prosjektgruppens arbeid. Dokumentet vil være beregnet til bruk i studiesammenheng på Høgskolen i Gjøvik, samt som dokumentasjon til oppdragsgiver for eventuell videreutvikling av rammeverket. Basert på tiltenk målgruppe for rapporten, så forventes det at leseren har en viss forståelse for informasjonsteknologi.

1.4 Prosjektgruppens bakgrunn og kompetanse

Begge utviklerene av dette rammeverket har utdanning innen det samme fagfeltet, bachelorstudie i informasjonssikkerhet ved Høgskolen i Gjøvik. I denne oppgaven vil prosjektgruppen benytte seg av forskjellig kunnskap som er tilegnet gjennom utdanningen, og kombinere disse for å lage et komplett og omfattende produkt av høy kvalitet. Gruppen har tidligere programmert hovedsakelig i objekt-orienterte språk som C++ og PHP, men har også erfaringer med flere programmerings- og skriptspråk. I tillegg til dette besitter gruppen kunnskap innen systemutvikling, penetrasjonstesting, database og operativsystemer. Noe som også vil komme godt med i prosjektperioden.

Prosjektgruppens kompetanse er i store deler teoretisk, og dette prosjektet vil gi mulighet for praktisk gjennomføring av teorien. I tillegg må prosjektgruppen tilegne seg nye kunnskaper i løpet av perioden, som for eksempel hvordan man utfører et systemutviklingsprosjekt av denne størrelsen, og hvordan external network footprinting gjennomføres.

1.5 Oppdragsgiver

Oppdragsgiver for prosjektet er selskapet Watchcom Security Group AS [3], en leverandør innen IT og informasjonssikkerhet. Selskapet har rundt 20 ansatte, med kontor i Bjørvika i Oslo.

Watchcom er en langsiktig partner innen informasjonssikkerhet som sikrer kundens verdier på en profesjonell og kompetent måte. Sentralt i selskapets forretningsmodell står kompetansedeling. Watchcom tror sterkt på at virksomheter ikke oppnår økt sikkerhet uten at de har god kunnskap om hvilke trusler en står overfor, hvilke konsekvenser dette kan medføre og hvordan en best kan beskytte seg.

Watchcom ønsker derfor å dele kompetanse gjennom alle sine virksomhetsområder, og gjennom dette kunne gjøre virksomheter i stand til å oppfylle de strengeste standarder innen informasjonssikkerhet.

Figur 2: Watchcom Security Group AS

1.6 Veileder

Veileder for prosjektet er Lasse Øverlier. Lasse foreleser ved Høgskolen i Gjøvik i faget *IMT 3491 Ethical Hacking and Penetration Testing*, og jobber i tillegg med informasjonssikkerhet ved Forsvarets Forskningsinstitutt (FFI). I 2007 disputerte Lasse for sin Ph.D. grad i Computer Science, med temaet ”Anonymity, privacy and hidden services: Improving censorship-resistant publishing".

1.7 Tidligere arbeid

Informasjonssikkerhetskonsulenter benytter ofte enkeltstående verktøy for å gjennomføre penetrasjonstester. Det er derfor veldig lite tidligere arbeid som er blitt gjort for å automatisere disse enkeltstående verktøyene. Det betyr at det ofte kun benyttes en brøkdel av verktøy som er tilgjengelige, og aggregeringen av data er et manuelt arbeid.

En av prosjektgruppens første idéer var å bygge videre på Recon-ng [4], som kom ut i slutten av 2012. Recon-ng er et modulbasert verktøy for informasjonsinnhenting, laget for å ligne Metasploit Framework [5], for en enklere overgang til å brukte dette verktøyet. Recon-ng løser noe av problematikken rundt automatisering, men brukeren må selv laste og starte hver modul manuelt. Det ville blitt en større jobb å forbedre dette eksisterende verktøyet på grunn av at gruppen måtte sette seg inn i andres kode, og i tillegg har ikke verktøyet tilstrekkelig stor database. Hovedargumentet for å ikke gjennomførte denne planen er at mye av informasjonen som blir funnet av Recon-ng aldri blir lagret.

Det var først 28. april 2014 (ca 3 uker før innlevering) at gruppen fant Spiderfoot [6], som er et open source footprinting verktøy. Hadde dette rammeverket blitt oppdaget før, så ville prosjektgruppen muligvis foretatt en vurderingen om å bygge videre på dette. Dessverre ble dette verktøyet oppdaget for sent.

Uansett så skiller oppgaveforslaget seg fra de verktøyene som er nevnt i avsnittene over. Hovedpunktet det skiller seg ut på er at det skal benytte eksisterende verktøy for å gjen-

nomføre external footprinting. Gruppen har benyttet Recon-ng som inspirasjon under utviklingen av rammeverket.

1.8 Prosjektmål

1.8.1 Effektmål

Hovedmålet for oppdragsgiver er at rammeverket skal effektivisere forarbeidet og kartleggingen av external network footprinting. Det skal automatisere bruken av flere verktøy, og må derfor aggregere data. Dette er for å fjerne uviktige eller duplikate data, sammenstille data som er relatert til hverandre, og så videre. I tillegg skal oppdragsgiver unngå unødvendig manuelt arbeid, og dermed spare tid i sine penetrasjonstester.

1.8.2 Resultatmål

Denne oppgaven leverer et rammeverk som er utviklet til operativsystem-distribusjonen Kali Linux. Rammeverket benytter eksisterende verktøy i operativsystemet for å finne external network footprint, se listen over verktøy som er implementert i 3.3. Rammeverket vil aggregere resultatene fra disse verktøyene for å gi et mer helhetlig bilde av det eksterne nettverket. Resultatene vil lagres i en database inntil det ikke lenger er behov for dem. Rammeverket vil kunne generere en lesbar teknisk rapport og en maskinvennlig rapport.

Det er viktig at rammeverket er designet slik at det er enkelt å legge til nye moduler eller nye verktøy som kommer på markedet etter innlevering av prosjektet.

1.8.3 Rammer

Det finnes flere fremgangsmåter og metoder for å utføre informasjonsinnhenting, men dette prosjektet vil kun fokusere på teknisk innhenting av data av external network footprinting. Dette vil si at rammeverket kun vil benytte seg av tekniske hjelpemidler for å hente inn informasjonen. Tekniske hjelpemidler er vanligvis programvare for å utføre forskjellige operasjoner mot en målenhet eller et nettverk. Eksempeler på ikke-teknisk informasjonsinnhenting kan være å søke etter informasjon på Google, eller å ta direkte kontakt med målet via telefon. Dette er lite hensiksmessig å automatisere da det ofte krever menneskelig intelligens og avgjøringsevne for å innhente informasjonen, eller for å identifisere om informasjonen er relevant.

Prosjektgruppen vil også utelukke noen punkter som faller inn under teknisk infromasjonsinnhenting. Disse er:

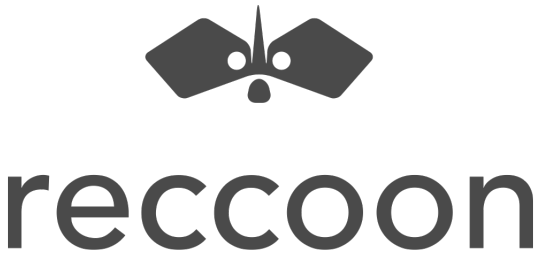
- *Lokale IP-adresser:* Siden en external network footprinting-prosess ser på målnettverket fra utsiden så er det ikke relevant å benytte seg av lokale IP-adresser, som kun benyttes på innsiden av nettverket.
- *Verktøy for sårbarhetsskanning:* Det er få punkter under external network footprinting som krever sårbarhetsscanning. Det å inkludere slike verktøy i rammeverket vil føre til en dårligere totaloversikt og ryddighet i rammeverket.
- *Begrensninger av WHOIS-protokollen:* Denne protokollen returnerer forskjellig informasjon avhengig av domenet som søkes på. Prosjektgruppen vil i første omgang kun få fullverdig WHOIS-informasjon fra .NO og .COM-domener.

Prosjektet vil foregå fra januar til juni 2014, med noen spesielle datoer:

- Innlevering av prosjektrapport 19. mai
- Presentasjon av prosjektet 5. juni.

1.9 Oppgavebeskrivelse

Sluttproduktet vil være et automatisert rammeverk som har fått navnet Reccoon. Dette navnet fikk prosjektgruppen etter å ha lekt med ordet *rekognosering* (eng: reconnaissance), og blandet det med *vaskebjørn* (eng: raccoon). Vi sammenligner rammeverket med en vaskebjørn som roter rundt på offentlige eiendommer for å finne informasjon, samt at ordlyden til de to ordene faller bra sammen på engelsk.



Figur 3: Reccoons logo, av Jonas Kristoffersen.

Oppgaven er levert av Watchcom Security Group AS, og omhandler utviklingen av et automatisert rammeverk for innhenting av external network footprinting. Oppdragsgiver har gitt få føringer for gjennomføringen av oppgaven. De eneste kravene er at prosjektgruppen kun skal benytte tekniske metoder for innhenting av informasjon; verktøyet skal utvikles for Linux-operativsystemer; rapportere tekniske funn i en menneskelig lesbar rapport og et maskinvennlig format. Hele oppgaveforslaget kan leses i vedlegg A.

Oppdragsgiver nevner at prosjektgruppen bør strebe etter å innhente elementer fra punktene under *Footprinting* i PTES [2]. Listen over elementer som vi henter inn er beskrevet i vedlegg H. Gruppen har ellers mye frihet til å gjennomføre prosjektet slik de selv ønsker.

Det er to hovedmål for oppgaven. Første hovedmål er å analysere flere verktøy som er inkludert i Kali Linux. For deretter å avgjøre hvilke av dem som er best egnet for å utføre en grundig informasjonsinnhenting. Under analysen må gruppen fokusere på å fjerne duplikater, data som er irrelevant, eller data som ikke kan bli brukt videre. Andre hovedmål er å implementere de utvalgte verktøyene som moduler i et automatisert rammeverk.

For at oppdragsgiver kan benytte den innsamlede informasjonen enklere og raskere, ønsker prosjektgruppen at rammeverket også skal ha mulighet til å dele sine data med andre rammeverk, som f.eks. Metasploit Framework. Rammeverket skal samle inn så mye relevant informasjon som mulig, både fra målets eget nettverk og fra andre åpne kilder på Internett. All informasjonen skal enkelt kunne eksporteres til en oversiktelig rapport som kan brukes som et hjelpemiddel senere i penetrasjonsprosessen.

1.10 Problemstilling

Ut ifra planleggingen av prosjektarbeidet har gruppen kommet frem til en problemstilling for oppgaven: *Hvorvidt kan en external network footprinting automatiseres? Og hvordan vil dette eventuelt påvirke tidsbruken av gjennomføringen og kvaliteten på informasjonen som blir innhentet?*

1.11 Valg av utviklingsmodell og organisering av arbeid

Gruppen benytter en inkrementell utviklingsmodell for rammeverket. Etter planen vil dette gi oss et rammeverk med en solid bunnstruktur, og det vil også være enkelt å legge til flere moduler ved senere behov. Rammeverket vil bli utviklet i Python [7]. Implementasjonen vil dermed gå raskt og vi vil få et "Proof of Concept" etter relativt kort tid. Python passer bra til databaseprogrammer og er godt egnet for eksport til XML-format. For å kontrollere at rammeverket henter inn den riktige informasjonen, så vil resultatet fra rammeverket bli sammenlignet med resultatet fra de implementerte verktøyene.

Mer om prosjektgruppens organisering av arbeid, ansvarfordeling og initielle planlegging kan du lese mer om i forprosjektet i vedlegg D.

2 Kravspesifikasjon

Dette kapitlet inneholder en beskrivelse av de funksjonelle og operasjonelle kravene som er satt for rammeverket. Dette inkluderer rammeverkets omgivelser og begrensninger. Formålet med denne spesifiseringen er å skape klarhet i hva som skal utvikles og hvordan det vil bli utviklet.

2.1 Funksjonelle krav

Rammeverket vil knytte sammen eksisterende enkeltstående verktøy til et og samme brukergrensesnitt. Disse verktøyene er ikke utviklet av prosjektgruppen, men følger med i standardbibliotekene til Kali Linux. Målet med rammeverket er at oppdragsgiver skal spare tid og manuelt arbeid i prosessene rundt gjennomføringen av external network footprinting. Rammeverket blir utviklet på en slik måte at det vil følge metodene og prosessene for datainnhenting som oppdragsgiver benytter idag (vedlegg E, referat for 19.12.2013). Dermed sitter oppdragsgiver allerede inne med kunnskap om hvordan rammeverket vil fungere i praksis. Dette vil gjøre det lettere for oppdragsgiver å utvide metoden med flere verktøy ved en senere anledning.

Rammeverket blir utviklet for å støtte kun én type sluttbruker, som oftest vil være en informasjonssikkerhetskonsulent som utfører penetrasjonstester. Det vil med andre ord ikke bli implementert noen form for brukerkontoadministrasjon. Det er nødvendig at brukeren av rammeverket innesitter gode kunnskaper om hvordan man skal benytte dataene som blir innhentet, og også kriteriene for hva som skal hentes inn for å gjennomføre en fullverdig external network footprinting. Rammeverket vil ikke hjelpe brukeren med informasjon om hvordan operasjonene skal foregå, og det er derfor denne kunnskapen er nødvendig.

Oppgaveforslaget (vedlegg A) beskriver at rammeverket skal være automatisk, så sluttbrukeren skal kunne spesifisere ett domene som blir målet for automatiserte skanninger. På grunn av den potensielle store mengden med data som kan bli innhentet, så bør resultatene deles opp og gi sluttbrukeren muligheter til å spesifisere vidre skanninger på de valgte resultatene. Skanningene vil hente elementer som er beskrevet i vedlegg H. External network footprintingen vil begynne når sluttbrukeren starter første skanning av målet. Dette er for å finne alle hosts som er tilgjengelige fra Internett, og fylle opp databasen med data for videre behandling. Ordet "*alle*", i denne sammenhengen, er alle hosts som blir funnet av tredjeparts vertøy. Rammeverket vil aggregerer de innsamlede dataene for å fjerne duplikater, og ellers flette informasjonen sammen på en oversiktelig måte. Skanningene har en logisk struktur, fordi noen operasjoner avhenger av data som er innhentet fra tidligere skanninger.

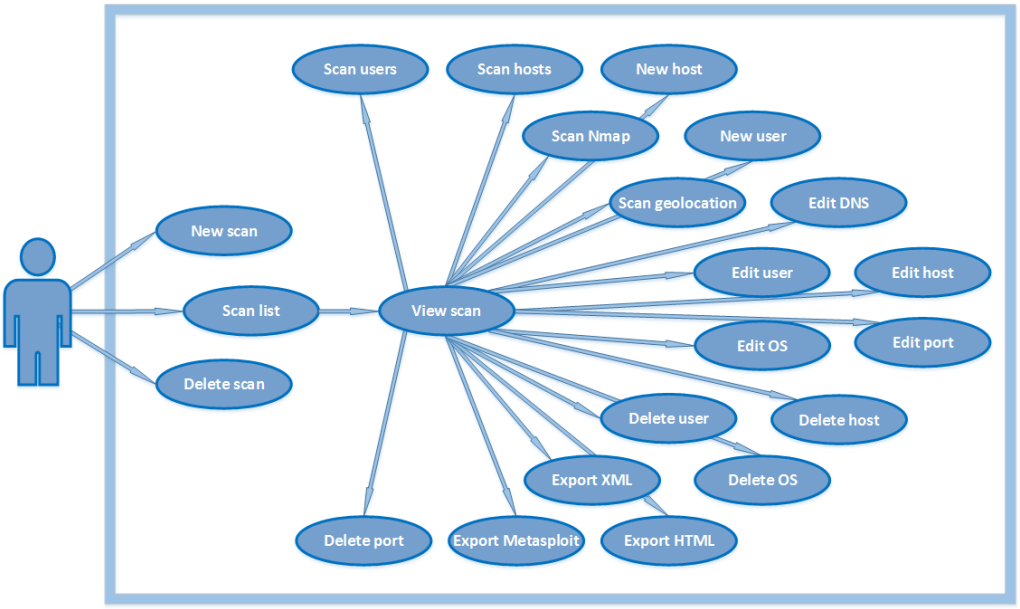
Etter den initielle skanningen har sluttbrukeren mulighet til å gjennomføre flere skanningtyper (for eksempel Nmap og geolokasjon) mot spesifikke resultater fra foregående skanning, og i tillegg gjøre søk etter users og eventuelt nye hosts. Nmap kan, til en

viss grad, identifisere operativsystem som blir benyttet av en host. Men Request for Comments (RFC), som beskriver hvordan protokoller skal implementeres, blir tolket på forskjellige måter av forskjellige utviklere. Derfor kan Nmap returnere flere sannsynlige identifiseringer, f.eks. så ligner Windows XP veldig mye på Windows 7 i nettverkstrafikk. Det betyr at det er muligheter for at det vil ligge noen resultater i databasen som ikke stemmer. Derfor må sluttbrukeren få muligheten til å redigere alle data som blir samlet inn, samt fjerne eller legge til elementer.

Når brukeren er ferdig med de skanningenene som de ønsker å gjennomføre, så skal de kunne få generert en detaljert og oversiktelig rapport, og også en maskinvennlig rapport. Disse rapportene skal inneholde alle aggregerte data som er blitt hentet inn fra skanningene. Ved dette tidspunktet så er sluttbrukeren normalt ferdig med skanningen, og har ikke noe videre behov for at den ligger i databasen. Sluttbrukeren skal kunne slette skanningen permanent, for å unngå at resultatene tar opp unødvendig plass. Det vil også bli lagt inn en funksjon for å eksportere resultatene av skanningene til Metasploit, for enkel overgang til de senere fasene i penetrasjonstestingen.

2.2 Bruksmønster

Et bruktsmønster (eng: use case) er en enkel representasjon av interaksjonen med systemet. Dette vil skildre sluttbrukeren av systemet, og de ulike måter som denne brukeren samhandler med systemet på [8].



Figur 4: Bruksmønsterdiagram

2.2.1 Høynivå bruksmønsterbeskrivelse

For at sluttbrukeren ikke må starte helt fra forsiden hver gang de starter skanninger eller gjennomfører endringer, så må rammeverket returnere brukeren til den aktuelle skanninginstansen. Dette blir gjennomført ved naturlige steder, og bør følge bruksmønsterets

trestruktur. F.eks. *New scan* returnerer til *View scan*.

Navn:	New scan
Aktør:	Sluttbruker
Mål:	Starte en ny instanse av en ny skanning, som blir brukt til videre skanninger.
Normal flyt:	1. Sluttbrukeren går inn på en visningen for å legge til ny skanning. 2. Skriver inn skanning-tittel og måladresse. 3. Skriver inn eventuelle parametere for hvert enkelt program. 4. Trykker på en knapp for å lagre ny skanning. Dermed legges den nye skanningen inn i databasen.
Resultat:	Den nye skanningen lagres i databasen. Brukeren returneres tilbake til startsiden og den nye instansen blir vist som ”skanning pågår”. Initiell external network footprinting skanning vil pågå i bakgrunnen.

Tabell 1: Høynivå beskrivelse av *New scan*.

Navn:	Scan Nmap
Aktør:	Sluttbruker
Mål:	Bruke tredjeparts program Nmap for å skanne etter åpne porter, tjenester, operativsystem, og versjon til disse.
Normal flyt:	1. Sluttbrukeren går inn på en eksisterende skanning. 2. Velger deretter å gjennomføre Nmap-skanning. 3. Skriver inn parametere for hvordan skanningen vil bli utført. 4. Velger hvilke hosts som skal skannes. 5. Trykker på en knapp for å starte skanningen.
Resultat:	Brukeren blir returnert tilbake til den valgte skanning instansen og skanningen blir vist som ”skanning pågår”. Nmap-skanningen vil pågå i bakgrunnen.

Tabell 2: Høynivå beskrivelse av *Scan Nmap*.

Navn:	New host/user
Aktør:	Sluttbruker
Mål:	Legge til en ny host/user inn i databasen.
Normal flyt:	1. Sluttbrukeren går inn på en eksisterende skanning. 2. Velger å legge til en ny host/user. 3. Får opp en visning med felter som skal fylles ut, disse feltene representerer en kolonne i tabellen i databasen. 4. Trykker på en knapp for å lagre den nye host/user.
Resultat:	Den nye host/user blir lagret i databasen, og brukeren blir returnert tilbake til den valgte skanning instansen.

Tabell 3: Høynivå beskrivelse av *New host/user*.

Navn:	Edit DNS/host/OS/port/user
Aktør:	Sluttbruker
Mål:	Redigere en eksisterende DNS/host/OS/port/user i databasen.
Normal flyt:	1. Sluttbrukeren går inn på en eksisterende skanning. 2. Velger deretter å redigere en DNS/host/OS/port/user. 3. Får opp en visning med felter som skal fylles ut, disse feltene er allerede utfylt med data som ligger i databasen. 4. Brukeren gjør endringer i aktuelle felter. 5. Trykker på en knapp for å lagre endringene.
Resultat:	Verdiene blir overskrevet til databasen, og brukeren blir returnert tilbake til den valgte skanning instansen.

Tabell 4: Høynivå beskrivelse av *Edit DNS/host/OS/port/user*.

2.3 Operasjonelle krav

2.3.1 Plattform

Rammeverket blir utviklet i Kali Linux, og det skal kjøres på den samme distribusjonen i produksjonssammenheng. Standardbrukeren på Kali Linux har administratorrettigheter (*root* [9] i Linux miljøer), derfor beregnes det at rammeverket blir kjørt av brukere med administratorrettigheter. Det var naturlig å velge Kali Linux på grunn av at det er ofte den valgte distribusjonen for penetrasjonstester, og den inneholder de fleste tredjeparts programmer som blir brukt ved external network footprinting. Kali Linux krever minimum 10 GB diskplass, i386 og amd64 med 512MB RAM, og rammeverket skal kunne kjøre på dette minstekravet.

For å gjøre navigering, filtrering og sortering av data enklere, så har vi valgt å legge alt inn i en database. Vi velger MySQL som database, fordi den er den mest utbredte databaseplattformen. Dette er den plattformen gruppen har mest erfaringer med både

fra prosjekter på skolen og privat. MySQL følger med Kali Linux og er planlagt å kjøre lokalt på maskinen.

Selve rammeverket vil bli programmert med Python 2.7, og vil bli drevet av av web-rammeverket Django 1.6 for en enkelere databaseintegrasjon og utvikling av rammeverket. Selve designet av brukergrensesnittet vil bli satt opp ved bruk av Foundation 5.

2.3.2 Ytelse

Det er ikke spesifisert, verken i oppgaveforslaget eller i møter med oppdagsgiver, at rammeverket skal ha noen responstid eller ytelseskrav. Det er likevel nevnt at det ikke kan generere for mye trafikk slik at det blir utestengt fra målnettverket eller andre tredjeparts tjenester. Siden målet med oppgaven er at oppdragsgiver skal spare tid på external network foorprinting, så kom prosjektgruppen frem til at det er vesentlig at ytelsen av rammeverket sammenlignes med manuelt arbeid. Det er forventninger til at oppdragsgiver skal ha tidsbesparelser i penetrasjonstestingprosessen.

Det er ingen god måte å kvantifisere tiden som blir brukt for å skanne et målnettverk. Det er mange variabler som avgjør hvor lang tid en skanning bruker. Noen av disse variablene vil være antall maskiner på målnettverket, nettverkskapasitet, responstid, forsinkelser i respons, og avstand. Derfor vil ytelsen bli beregnet ut i fra tiden som blir brukt av rammeverket, sammenlignet med tiden av det manuelle arbeidet mot det samme målet. Gruppen forventer å se en betydelig redusering av tid som blir brukt til external network footprinting.

2.3.3 Brukervennlighet

Gruppen har som mål at rammeverket skal være intuitivt for sluttbrukere som innehar en forståelse av external network footprinting. Sluttbrukeren skal kunne redigere data for å passe sine behov. Det er viktig med funksjoner som lar brukeren gjøre disse endringene (legge til nye, redigere og slette eksisterende elementer).

Sluttbrukeren skal også kunne spesifisere egne parametere til de eksterne programmene som blir kjørt, og at disse feltene som lar brukeren spesifisere paramtere har eksempler på lovlig innhold. Når innholdet i feltet er feil, så skal brukeren bli gjort oppmerksom på feilen og få mulighet til å rette feilen. De eksterne programmene skal være mulig å starte uten at brukeren har gitt det noen egendefinerte parametere.

Rapporten som blir generert ved forespørsel fra sluttbrukeren skal ha et slik oppsett at den er velegnet for utskrift på papir.

2.3.4 Juss og etikk

På grunnlag av at rammeverket som prosjektgruppen produserer potensielt kan rettes mot mål som sluttbrukeren ikke har noen avtaler med, er det viktig å kartlegge om noen av rammeverkets operasjoner på noen måte kan bryte norsk lov.

Justis- og beredskapsdepartementet har ikke kommet til at det gode nok grunner for å anvende samfunnets strengeste reaksjoner på handlingen elektronisk kartlegging (skanning). Slik kartlegging har i rettspraksis blitt vurdert opp mot ulovelig bruk av løsøreg-

jenstand, straffeloven 1902 § 261 og § 393. Flertallet av høringsinstansene er negative til forslaget om å straffe skanning [10]:

"*Etter min oppfatning må den som har koblet sin datamaskin til Internett, og har valgt å la den svare på forespørsler, anses å ha gjort maskinen til en del av det informasjonssystem som Internett representerer. Ved å koble maskinen til Internett har datamaskineieren akseptert at det blir rettet forespørsler til maskinen om hvilken informasjon den har å tilby, og den aktivitet som skjer når maskinen svarer på slike forespørsler, kan da etter mitt syn ikke anses som uberettiget bruk av maskinen.*"

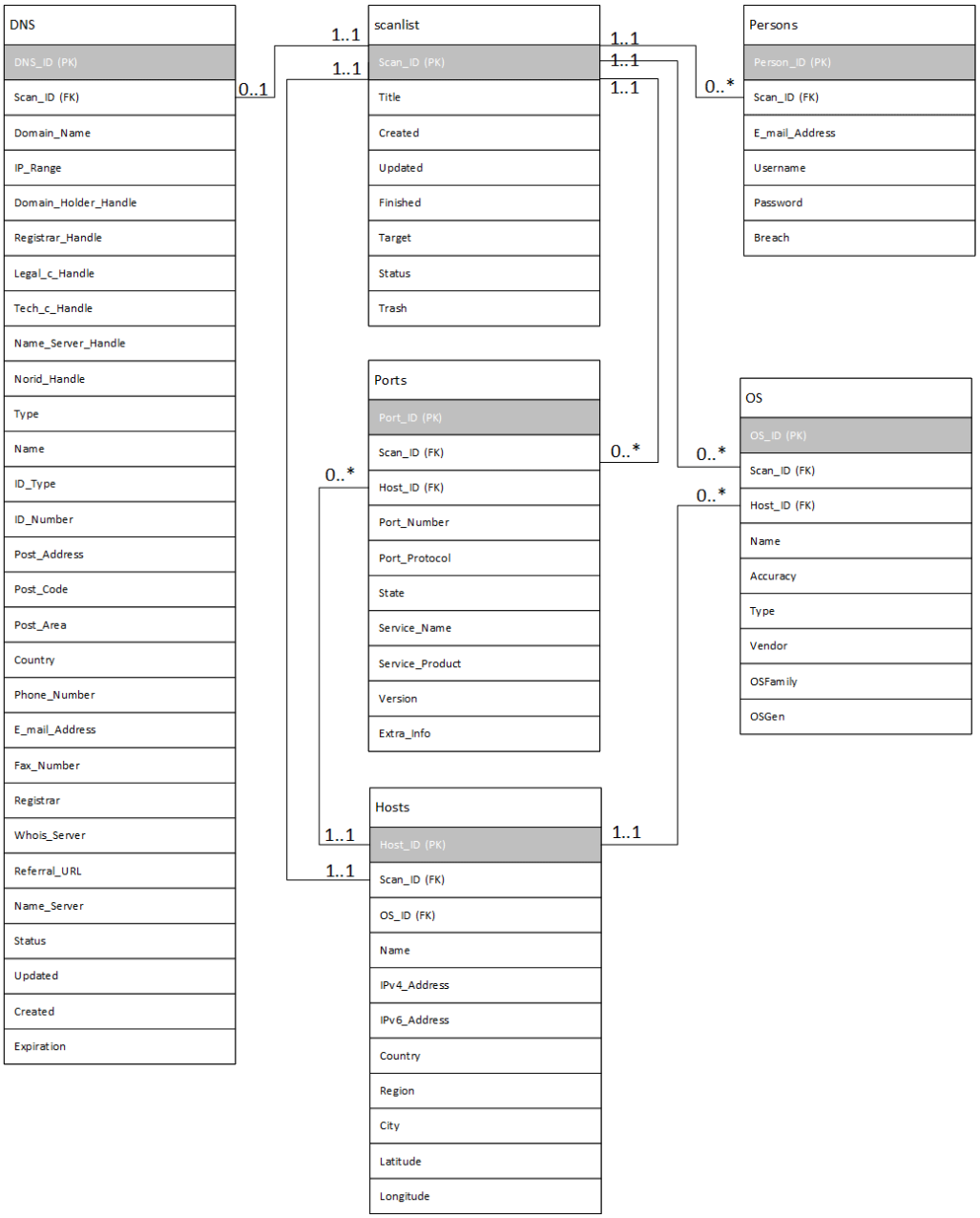
Undersøkelsene som gruppen har gjort har resultert til at det ikke er funnet noen lov eller rettspraksis mot informasjonen som blir innhentet av Reccoon. Det som er i gråsonen er e-post adresser. Det er fordi dette er å regne som personinformajson og ligger under personvernloven. Gruppen sin konklusjon er at e-post er offentlig informasjon på lik linje med telefonnummere. Reccoon ikke vil bryte noen eksisterende lover, men det vil lønne seg å innhente godkjennelser før innsamlingen av data.

Selv om det ikke finnes noen direkte lover mot informasjonsinnhentingen, kan ISP-er ha egne regler for hvilke operasjoner som er tillatt på sine nettverk. Om operasjoner som bryter med ISPs policier oppdages, kan dette i verste fall føre til utestengelse fra nettverket. Det er derfor viktig å sette seg inn i hvilke policier ISPs, og innehaver av nettverket, har satt.

2.4 Logisk databasemodell

Den logiske databasemodellen illustrerer klassene og relasjonene mellom dem. Klassene er en beskrivelse av en type objekter, og blir representert som lister med attributter, eller egenskaper. Relasjonene mellom klassene er representert med streker, og tallene viser hvilken type relasjon de har. For en beskrivelse av disse relasjonene, se: SQL for Beginners: Part 3 - Database Relationships [11].

Normalisering [12] av databasen er ikke blitt prioritert under dette prosjektet. Hovedmålet for prosjektet er å aggregere mest mulig data som skal inn i en lesbar rapport for informasjonssikkerhetkonsulentene. Databasen er ikke designet for å holde på data over en lang periode. Den er designet for å holde på alle data den finner, og skrive ut en rapport før alt slettes. Gruppen mener det er greit med noe duplisering av data.



Figur 5: Logisk databasemodell

Det konseptuelle klassediagrammet illustrerer dataene som rammeverket lagrer. Normalisering er ikke blitt prioritert, men gruppen har prøvd å redusere duplikater, og har dermed valgt å normalisert klassediagrammet til en viss grad. Designet kunne f.eks. normalisert geo-lokasjoner og IP-adresser i tillegg. Men siden databasen skal ikke holde på data over en lenger periode eller ha store tabeller, så det er greit at tabellene inneholder duplikater i kolonnene. Databasen er tiltenkt å lagre all informasjonen som kan finnes med modulene for å gjøre genereringen av rapporten så enkel som mulig.

Klassediagrammet har én tabell som står i senter: *scanlist*. Når sluttbrukeren velger å starte en ny skanninginstanse, blir det lagt inn et nytt objekt i denne listen. Listen vil kun inneholde generell informasjon om en skanning, som for eksempel når skanningen ble startet og hva som er målet for skanningen. For å få lagt inn informasjon om målet (nettverksenheter, porter, operativystem, osv), må rammeverkets forskjellige moduler kjøres. Informasjonen de forskjellige modulene returnerer vil da bli lagt inn i de andre klassene som er knyttet opp mot *scanlist*.

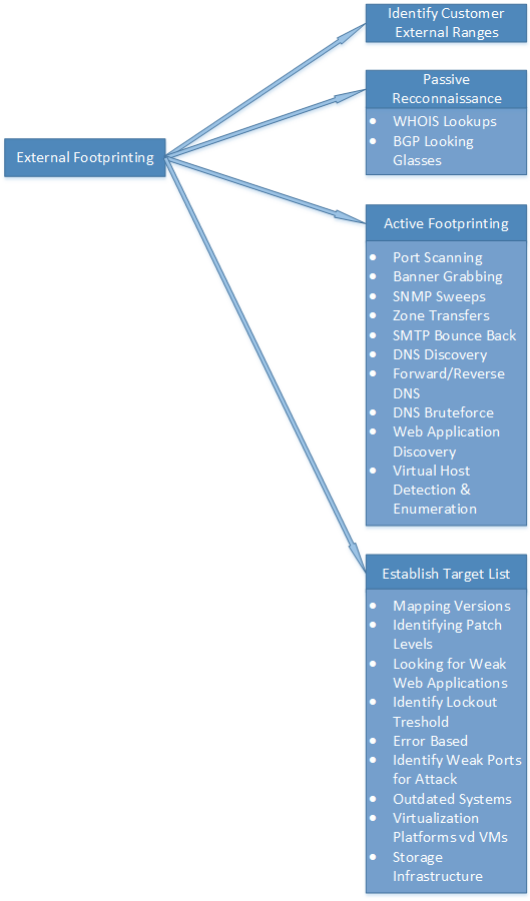
- *DNS*: Data fra moduler som finner informasjon om måladressens DNS-servere blir lagt inn i denne tabellen. Dette er helt klart den lengste tabellen, men det er ikke alle feltene som vil ha innhold etter en skanning. Dette er fordi verktøy som finner DNS-informasjon vil returnere forskjellig informasjon for forskjellige domener. For eksempel så vil en skanning av et .NO-domene returnere andre resultater enn et .COM-domene. Dermed kan noen av feltene være tomme etter en skanning.
- *Hosts*: Denne vil inneholde alle eksterne nettverksenheter som blir funnet på måladressen. Denne tabellen inneholder navnet til hver enhet (som er en URL), IP-adresser (IPv4 og IPv6), samt geografisk lokasjon for enheten. Det må nevnes at geografisk lokasjon er omtrentlig. Det avhenger av hvor detlajert hver ISP oppgir sine kunders lokasjon. Øvrig informasjon om disse enhetene hentes fra *ports*- og *OS*-tabellene.
- *Ports*: Data som blir hentet fra portskanning-modulen (Nmap) vil bli lagret her. For eksempel vil dette være hvilke porter som er åpne på en nettverksenhet, og hvilke tjenester som kjører på disse portene.
- *OS*: Her lagres data fra Nmap-skanningen som har operativsystem-identifisering aktivert. Det må nevnes at denne informasjonen er omtrentlig, og ikke hundre prosent korrekt. Dette er begrunnet i seksjon 2.1. Tabellen har en oversikt over hvilke operativsystemer og versjonene de forskjellige nettverksenhetene bruker.
- *Users*: Her vil data om målnettverkets brukere lagres. Dette vil hovedsakelig være ansattes e-postadresser, brukernavn, passord, og om adressen er registrert som lekket. Lekket vil si at e-postadressen har informasjon på avveie. Det er kun innhenting av e-postadresser og sjekken av disse adressene som kjøres automatisk. Brukernavn og passordfeltene må redigeres manuelt.

3 Analyse

Før gruppen kunne begynne med design og implementasjon av rammeverket, måtte gruppen tilegne seg en grundig forståelse av hvordan informasjonsinnhenting blir gjennomført. Mer spesifisert; den delen som omhandler å finne eksternt nettverksavtrykk. Utfordringen er å vurdere hva som er nødvendig informasjon og hvilke eksisterende programmer som burde benyttes for å hente inn denne informasjonen. Selve prosessen ved å analysere hvilke data som må hentes inn om målet, og hvilke verktøy som er de beste til å gjøre denne jobben er en stor del av oppgaven.

3.1 Penetration Testing Execution Standard

Oppdragsgiver nevner at gruppen burde prøve å samle inn data fra hvert punkt under *External footprinting* i Penetration Testing Execution Standard (PTES) [2].



Figur 6: Ekstern footprinting fra PTES

Figur 6 illustrerer PTES sin mening om hvilke informasjonspunkter som inngår i ekstern footprinting. Ekstern footprinting er listet opp som en del under intelligence gathering.

Hvert underpunkt er beskrevet i detalj på referansen [13]. Selv om det er ønskelig at rammeverket tar for seg alle underpunktene, så må hver enkelt punkt vurderes opp mot det ønskelige sluttresultatet til rammeverket.

Figuren har fire hovedpunkter. *Identify customer external ranges* er prosessen som ofte starter informasjonsinnhentingen. Denne må gjennomføres først. Det er for å avgjøre hvilke nettadresseområder som tilhører målet, og som i tillegg er legitime mål i følge kontrakten som ble inngått før penetrasjonstesten.

Det neste punktet *Passive reconnaissance* er informasjonsinnhenting fra kilder som ikke interagerer med målnettverket. Et eksempel er å benytte seg av WHOIS-protokollen for å finne informasjon om måldomenet. Dette er registrarinformasjon inneholder som oftest domenenavn og IP-blokker som blir benyttet. En registrar er en aktør som har autoritet til å registrere domener for kundene sine. For eksempel så er Norid ansvarlig for alle .NO-domener [14].

Active Footprinting er det motsatte av *passive reconnaissance*, med at informasjonsinnhenting skjer gjennom interaskjon med målnettverket. Denne rapporten vil ikke gå gjennom hvert eneste underpunkt i denne delen, men etter en diskusjon så har gruppen ønsket å inkludere disse i rammeverket: Port scanning, DNS discovery, DNS bruteforce, forward/reverse DNS.

Establish external target list er lister med informasjon som er samlet inn i de foregående punkter. Disse listene inneholder informasjon om brukere, e-post adresser, domener, nettverksenheter, -applikasjoner, og -tjenester. Listene må fylles ut med mer spesifikk informasjon for å ha noen videre nytte av dem. Fingerprinting er å identifisere detaljerte versjonsnummere mot nettverksenheter, -applikasjoner og -tjenester. For å identifisere disse så vil gruppen foreta aktive spørringer mot tjenestene.

3.2 Metodologi for informasjonsinnhenting

De fleste sikkerhetskonsulenter benytter seg av Internett for å finne initiell informasjon om målet, med både tekniske (DNS/Whois) og ikke-tekniske (søkemotorer, nyhetsgrupper, e-postlister, etc.) metoder [15]. De informasjonsinnhentingsmetodene som er nevnt krever ikke at aktøren etablerer kontakt med målsystemet. De benytter Internett fordi det finnes kilder der som alle kan benytte, og det vil ikke etterlate noen spor i målnettverkets logger. Denne initielle undersøkelsen blir fort veldig dynamisk, med at man finner data man ønsker å utforske nøyere. Det blir derfor vanskelig å automatisere de ikke-tekniske metodene.

Det neste skrittet er å finne fotavtrykket (eng.: footprint) av nettverket, og finne en sannsynlig nettverkstopologi, slik som det er nevnt under PTES veiledning. Denne prosessen bruker tekniske metoder for å finne levende hoster, port- og tjenesteskanning, nettverkskartlegging, identifisere kritiske tjenester, og operativsystem fingeravtrykk (eng.: fingerprint).

Denne kartleggingen vil hjelpe en aktør med å finjustere eventuell informasjonen som

er sikkerhetskonsulentene allerede innehar, og bekrefte eller avvise noen hypoteser om systemene. Det neste skrittet er at aktøren velger en eller flere mål for å gjennomføre sårbarhetsskanning: identifisere sårbare tjenester, søke etter kjente sårbarheter, utføre falske positive og falske negative verifisering, beregne sannsynlig innvirkning, og identifisere angrepsveier for utnyttelse.

I følge oppgaveteksten så skal prosjektgruppen fokusere på tekniske metoder for informasjonsinnhenting, og analysen har vært med på å bekrefte dette. Den har også gitt en metode for informasjonsinnhenting. Prosjektgruppen vil fokusere på opplisting, fot- og fingeravtrykk av nettverket til et spesifikt mål etter veiledningen gitt av PTES.

3.3 Analyse av verktøy

Kali Linux inneholder over 300 verktøy [16], og mange av disse verktøyene er ikke relevante å bruke i Reccoon. Eksempler på dette er verktøy for utnytting av sårbarheter (eng.: exploiting), verktøy for cracking av trådløse nettverk og verktøy for digital etterforskning. Det er også mange av verktøyene som ikke er godt egnet for automatisering, fordi de kun har grafiske grensesnitt eller krever kontinuerlig interaksjon fra brukeren.

Ved å benytte verktøy som er standard i Kali Linux vil gruppen unngå potensielle problemer ved installasjon og oppdateringer av Reccoon. Alternativet er at rammeverket også må håndtere installasjon av underliggende programmer som det er avhengig av for å kjøre modulene sine på en korrekt måte. Dette vil åpenbart være en mye mer komplisert operasjon enn å benytte forhåndsinstallerte verktøy. På grunnlag av dette vil prosjektgruppen først og fremst analysere de inkluderte verktøyene i Kali Linux.
Dette er punkter som gruppen brukte for å analysere hvert verktøy:

- Hvilke informasjon blir hentet inn?
- Er informasjonen som returneres korrekt?
- Finnes det andre verktøy som gjøre jobben raskere og bedre, eller returnerer mer informasjon?
- Blir det returnert mye duplikat informasjon?
- Dekker den returnerte informasjonen punkter fra PTES listen?
- Er det mulig å automatisere verktøyet i et rammeverk?

Når gruppen hadde analysert alle relevante verktøy, så lagde gruppen en liste av dem som ville gi en god dekning av de fleste puntkene fra PTES-listen. Verktøylisten er blitt delt inn i to deler: systemverktøy og personellverktøy. Hver av listene inneholder en kort beskrivelse av hvilken informasjon de innhenter, og i figur 7 vises hvilke verktøy som dekker de valgte punktene fra PTES. Listene blir presentert i den rekkefølgen gruppen ønsker at de skal implementeres.

3.3.1 Systemverktøy

dig: Domain Information Groper. Foretar DNS lookups og DNS Zone Transfer.

dmitry: Deepmagic Information Gathering Tool. Foretar Whois-oppslag på IP-adresse og domene.

dnsdict6: Bruteforcer subdomener, og finner IPv4 og IPv6 adresser til disse, inkludert NS og MX servere.

dnsenum: DNS og IP-block enumeration.

fierce: DNS enumeration.

host: DNS lookups. Oversetter navn til IP-adresse og motsatt.

nmap: Network Mapper. Identifiserer kjørende maskiner, port nummer, port protocol, port status, tjenestenavn og tjenesteversjon/-informasjon, operativsystem og -versjon.

3.3.2 Personellverktøy

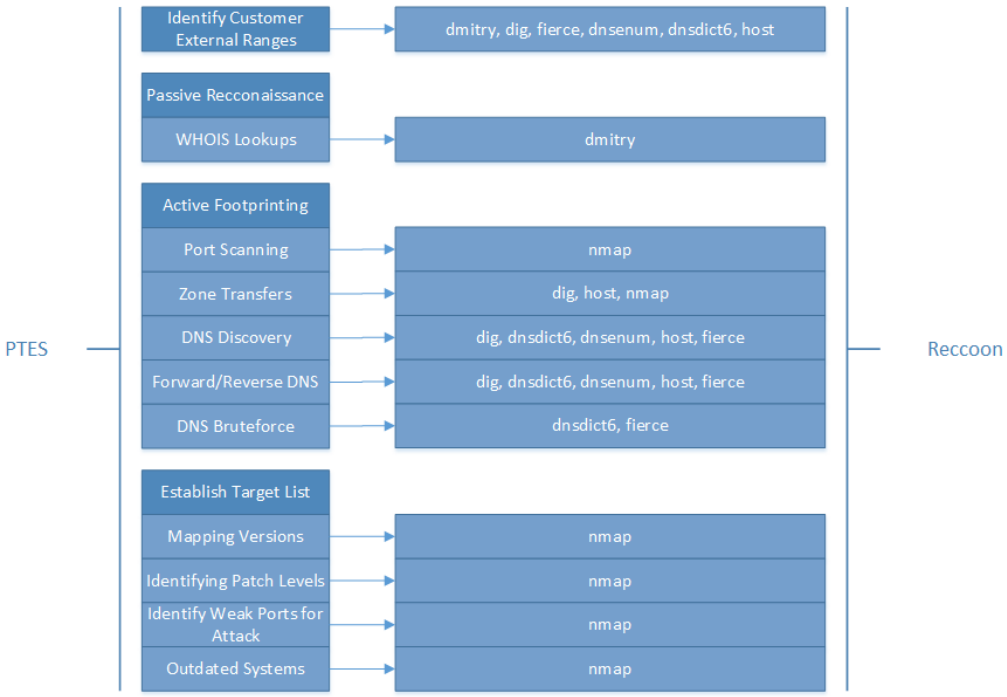
theharvester: Lister opp e-post adresser, IPv4 addresser, domenenavn og virtuelle hosts.

3.3.3 Tilleggsverktøy

geolocation: API fra ip-api.com. Returnerer geografisk lokasjon for en IP-adresse eller et domene.

haveibeenpwned: API fra haveibeenpwned.com. Returnerer om epost-adresse er kompromittert under dataangrep (breach).

msf: Eksport til Metasploit for de senere fasene i penetrasjonstesten (vulnerability scanning og exploitation).



Figur 7: Oversikt over hvilke verktøy i Reccoon som dekker de utvalgte punktene fra PTES.

3.3.4 Andre analyserte verktøy

Vi har også analysert disse verktøyene, som ikke vil bli inkludert i Reccoon av flere ulike grunner: telnet, traceroute, nslookup, recon-ng, xprobe2, nikto, metagoofil, dnsrecon, dnsmap, dnsrecenum6, dnstracer, dnswalk, Maltego, urlcrazy, zenmap, fragroute, fragrouter, ftest, lbd, wafw00f, alive6, arping, cdpsarf, detect-new-ip6, detect_sniffer6, fping, hping3, inverse_lookup6, miranda, ncat, netdiscover, passive_discovery6, thcping6, wol-e, jigsaw, twofi, 0trace, intrace, netmask, implementation6, implementation6d, sslscan, sslyze, tlssled, acccheck, nbtscan, smtp-user-enum, ike-scan, enumiax, ace, swaks, braa, tcpflow, wireshark, cisco-torch, cisco-auditing-tool, p0f, copy-router-config, merge-router-config, onesixtyone, snmpcheck, sslaudit, ssldump, sslh, sslsniff, sslsplit, sslstrip, stunnel4, irpas-ass, irpas-cdp, creepy.

4 Design

Dette kapitlet inneholder en beskrivelse av den grunnleggende arkitekturen til rammeverket. Først vil Model View Controller (MVC) og Model View Template (MVT) beskrives for å gi en innledning til denne arkitekturen. Deretter vil det graves dypere for å forklare hvordan rammeverket er blitt strukturert.

4.1 Innledning

Den originale planen var å ha en fult automatisk kommandolinjebasert eksekvering av flere moduler. Under både design- og prøveprogrammeringfasen ble det tydelig at vi ikke kunne bygge opp rammeverket på denne måten. Det ville tatt enormt mye tid hvis rammeverket skulle kjøre gjennom alle sine moduler mot alle objekter som blir funnet. Sluttbrukeren ville dermed ikke ha noen mulighet til å stoppe den automatiserte gjennomføringen. Det var heller ingen plan om å gi brukeren mulighet til å redigere resultatet.

Prosjektgruppen kom i stedet frem til å utvikle et grafisk administrasjonsgrensesnitt i HTML. I dette designet har de forskjellige mulighetene for skanning blitt skilt ut i mer selvstendige og enkeltstående operasjoner. For eksempel så kan brukeren velge å kun kjøre en skanning med Nmap, eller kun en geolokasjonskanning. På denne måten kan brukeren jobbe mer dynamisk og spesifisere relevante søk ettersom hva som er ønskelig. Prosjektgruppen har etterstrevet å ha høy styrke og lave koblinger i designet. Høy styrke vil si at en modul gjennomfører enkle oppgaver (gjerne én). Lave koblinger vil si at rammeverket forsøker å holde datautvekslingen minimal. For å få høy styrke og lave koblinger så letet prosjektgruppen etter et eksisterende rammeverk (eng. framework) som kunne være et skall til prosjektet. Gruppen fant et web-rammeverk i Python som heter Django.

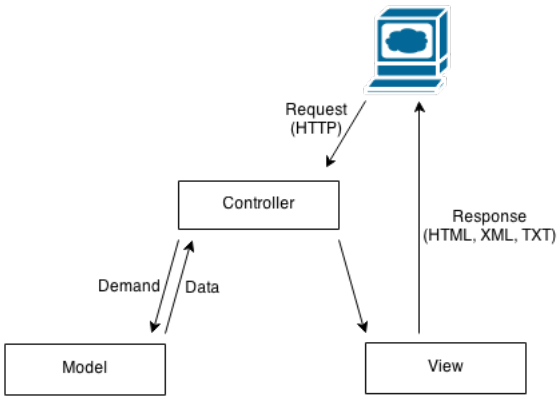
4.2 Django

Med Django kan prosjektgruppen utvikle og kjøre en developer-server for rammeverket på en lokal maskin. Deretter kunne rammeverket bli flyttet til en vanlig webserver, og med noen få endringer som tillater at Apache kan kjøre Python-scripts så har man en plattform som tillater samarbeid mellom flere sikkerhetskonsulenter. Dette er mer beskrevet i seksjon 7.2.

4.2.1 Model View Controller-teori

MVC-arkitekturen er den arkitekturen som oftest blir brukt for å gi et grafisk grensesnitt til en bruker. Arkitekturen har blitt veldig populær på grunn av at den ofte blir brukt for å lage applikasjoner til smartmobiler og nyere nettsider. Arkitekturen er bygget opp slik at den fordeler ansvarsområder og separere kode. Hver av enhetene har sin egen rolle, og medfølgende oppgaver å utføre. Modellen i figur 8 representerer data som er lagret et sted, normalt i en database. Den gir tilgang til informasjon, endringer, nye innlegg, og så videre. Det er et ekstra lag mellom kode og database. Visningen foretar informasjonvisu-

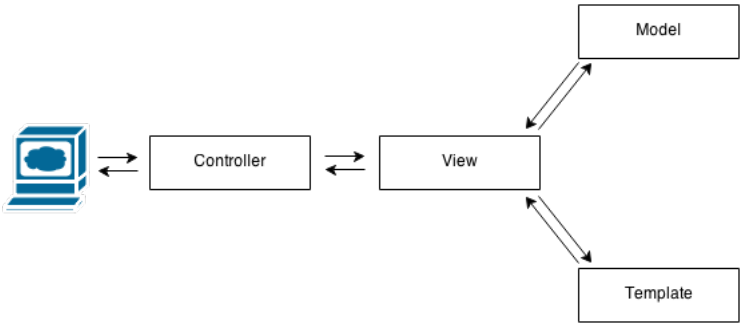
alisering. Dette er det eneste brukeren kan se. Til slutt så er det kontrolleren som utfører alle brukerhendelser. Den laster visninger og henter data fra modellen. Etter mulige formateringer av disse data så sendes det til visningen, som presenterer det for brukeren.



Figur 8: MVC-arktitektur

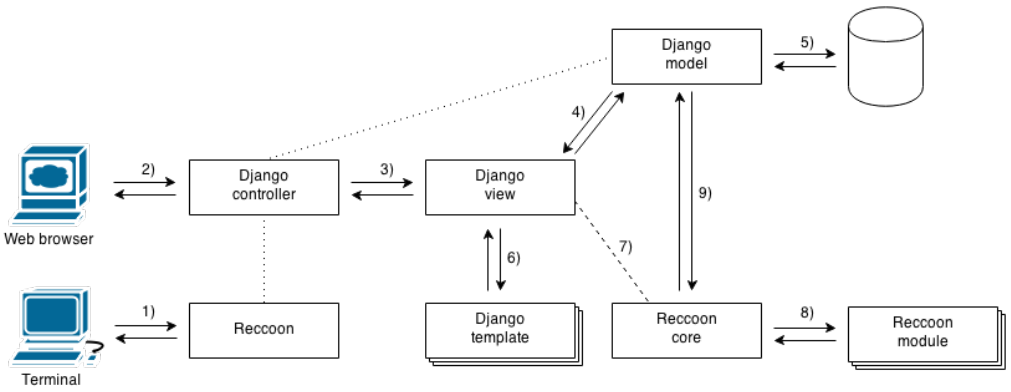
4.2.2 Model View Template-teori

Django er litt annerledes enn den klassiske MVC-arkitekturen. Django styrer selv kontrolleren, og dette gjør at arkitekturen kalles MVT. Denne arkitekturen benytter allerede modellen og visningen som tidligere er blitt beskrevet, og introduserer et nytt begrep: maler. Se figur 9. Før malen blir sendt så vil den gjennomgå en tolkning av Django-rammeverket, som om det var en kodefil. Django har en mal-motor som kan vise variabler, bruke betingede strukturer (if/else) og løkker (for), og så videre.



Figur 9: MVT-arktitektur

4.3 Reccoon arkitektur



Figur 10: Reccoon-arktitektur

Her følger en gjennomgang av arkitekturen til Reccoon:

1. Sluttbrukeren starter Reccoon fra en terminal med kommandoen `python reccoon.py`. Brukeren kan også spesifisere IP-addresse og port som skal bli lyttet på. Reccoon vil sjekke om den får tilgang på databasen. Hvis alt er greit, så starter Reccoon Djangos developer server. Brukeren får tilgang til det grafiske grensesnittet via en nettleser, normalt på addressen `http://127.0.0.1:8000/`. Brukeren får beskjed om eventuelle feilmeldinger, og at serveren har startet eller sluttet.
2. Det er Django som selv håndterer forespørsler fra HTTP protokollen. Når Django får beskjed om å returnere en side, så leter den gjennom filen `urls.py` for å finne visningen som skal returneres. Hvis brukeren for eksempel spør etter startsiden (`/` eller `/index.html`), så leter Django i filen og starter funksjonen som tilhører addressen hvis den blir funnet. Resultatet fra denne funksjonen blir sendt som respons.
3. Funksjonen som ble forespurt blir startet av Django-kontrolleren. Denne funksjonen ligger i filen `views.py`, og inneholder flere funksjoner til forskjellige forespørsler. Det er denne visningen som henter, sorterer, filtrerer og ellers kontrollerer informasjonsflyten i grensesnittet. Denne funksjonen velger for eksempel å hente noen data fra databasen og sender det til en mal. Denne malen blir returnert til kontrolleren.
4. Modellen inneholder metainformasjon om data som finnes i databasen. Det er filen `models.py` som har en beskrivelse av klasser som blir liggende som tabeller i databasen. Klassene inneholder en beskrivelse av hver datatype. Disse beskrivelsene blir brukt av Django for datahåndtering til og fra databasen. Disse blir i tillegg brukt av Reccoon som blir beskrevet i punkt 9.
5. Django bruker modellen for å hente og legge inn data i databasen. Påloggingsinformasjonen Django benytter til databasen ligger i filen `settings.py`.
6. Når visningen har fått data som den har forespurt, så sendes disse til en mal. Malen vil genererer en HTML-side som blir returnert som respons.
7. For at sluttbrukeren ikke skal tro at nettsiden har hengt seg eller sluttet å respon-

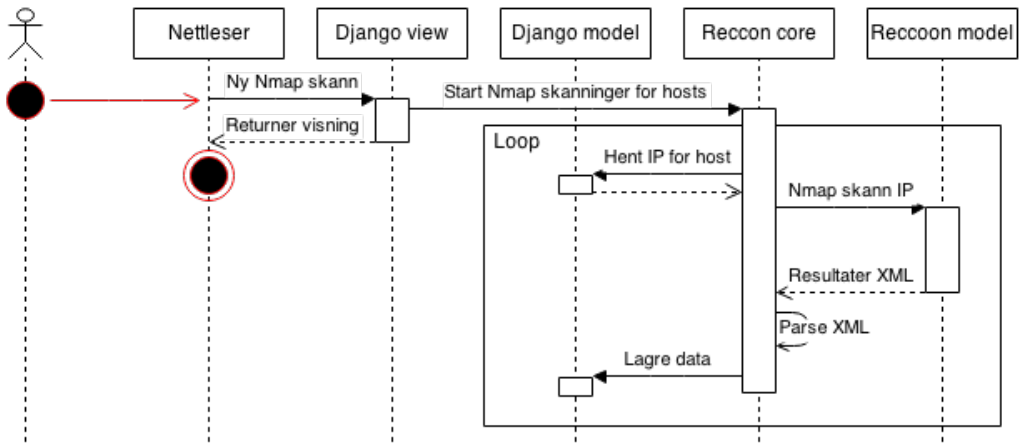
dere, så vil alle forespørsler til Reccoon være tråder. Det vil gjøre at Django kan fortsette å svare HTTP-henvendelser, og tillate at Reccoon arbeider bak kulissene. Det er derfor ingen direkte kommunikasjon mellom Django og Reccoon. Reccoon er uavhengig av Django, og Reccoons core kan kjøres alene med kommandoen `python core.py`. Hovedsakelig skal denne funksjonaliteten kun brukes i utviklingsprosesser. Denne måten er den beste for å rette feil i koden når den sammenkobles med Django, da denne metoden gir Python-feilmeldinger ut til kommandolinjen. Reccoons core benytter flere moduler for å samle inn data den trenger, dette blir deretter aggregert med data i databasen.

8. Som nevnt brukes modulene til å samle inn data. Data kan komme fra andre programmer som blir startet, eller fra nettsider. Hver modul vil sortere ut relevant data før resultatet blir returnert til Reccoons core.

9. Reccoons core får data fra en modul som må aggregeres. Reccoons core snakker med Django modellen for å sortere dataene yttligere, og til slutt lagre nye data inn i databasen.

4.3.1 Eksempel på informasjonsflyt

Dette er et kort eksempel for å vise hvordan en Nmap-skanning blir startet av systemet. I dette eksemplet så vises ikke Django-kontrolleren som håndterer seg selv. Sluttbrukeren har allerede visningen for å starte Nmap-skanninger mot nettverksenheter oppe i nettleseren sin, og velger parametere og mål som skal benyttes i skanningen. Når sluttbrukeren starter skanningen så blir det startet som en ny Reccoon-tråd, og rammeverket sender både parametere og mål til denne tråden. Reccoon henter IP-adressene til hvert mål, og sender disse IP-adressene og parametere til Nmap-modulen. Nmap-modulen starter Nmap-programmet, og sender fil-lokasjonen for XML-resultater. Reccoons core analyserer denne XML-filen og lagrer data i databasen. Denne filen blir slettet når det ikke lengre er behov for den.



Figur 11: Informasjonsflyt for ny Nmap-skanning

5 Implementasjon

Dette kapitlet tar for seg hvordan Reccoon implementeres. Dette er basert på kravspesifikasjonen og design-delen av rapporten. Her vil det presenteres hvordan rammeverkets ser ut, både visuelt og internt. Det vil også bli gitt eksempler på hvordan nye moduler legges til.

5.1 Lisens

Sannsynligheten for at prosjektgruppen er de siste som tilfører kode til Reccoon er stor. Oppdragsgiver og gruppen har ingen planer om å vedlikeholde eller videreutvikle rammeverket etter levering. Reccoon er utviklet ut ifra oppgaveforslaget til oppdragsgiver, og det har blitt levert en stabil prototype-versjon. Prosjektgruppen ønsker ikke å offentliggjøre rammeverket slik det er nå, før noen av forbedringene er blitt implementert. Disse er beskrevet i seksjon 7.2.

Prosjektgruppen er i tillegg av den oppfattning av at rammeverket har stort potensiale for forbedringer, men at det vil fungere godt i et produksjonsmiljø slik det fremstår i dag. Gruppen tenkte først at disse argumentene burde tilsi at rammeverket ble gjort tilgjengelig og med en open-source lisensiering. Men gruppen legger mest vekt på at rammeverket er en prototype og det er ingen plan for å vedlikeholde det. Derfor velger vi å ikke lisensiere eller utgi rammeverket på noen måte.

Unntaket er nye bacheloroppgaver eller akademiske henvendelser fra personer på Høgskolen i Gjøvik.

5.2 Utviklingsmiljø

Rammeverket er utviklet i Kali Linux, og er beregnet på å bli kjørt i det samme miljøet i produksjon. Det er derfor blitt fokusert på å benytte funksjonaliteter som følger med som standard i denne Linux-distribusjonen. Reccoon støtter Kali 1.0.6, utgitt 9. januar 2014.

Både versjon 2.6 og 2.7 av programmeringsspråket Python følger med i Kali, men prosjektgruppen har utelukkende benyttet den nyeste stabile versjonen, 2.7. Rammeverket er ikke testet med Python 2.6. Det følger med flere forskjellige databasesystemer, men gruppen har benyttet MySQL (versjon 5.5.35) fordi det er dette gruppen har mest erfaring med.

5.3 Filstuktur

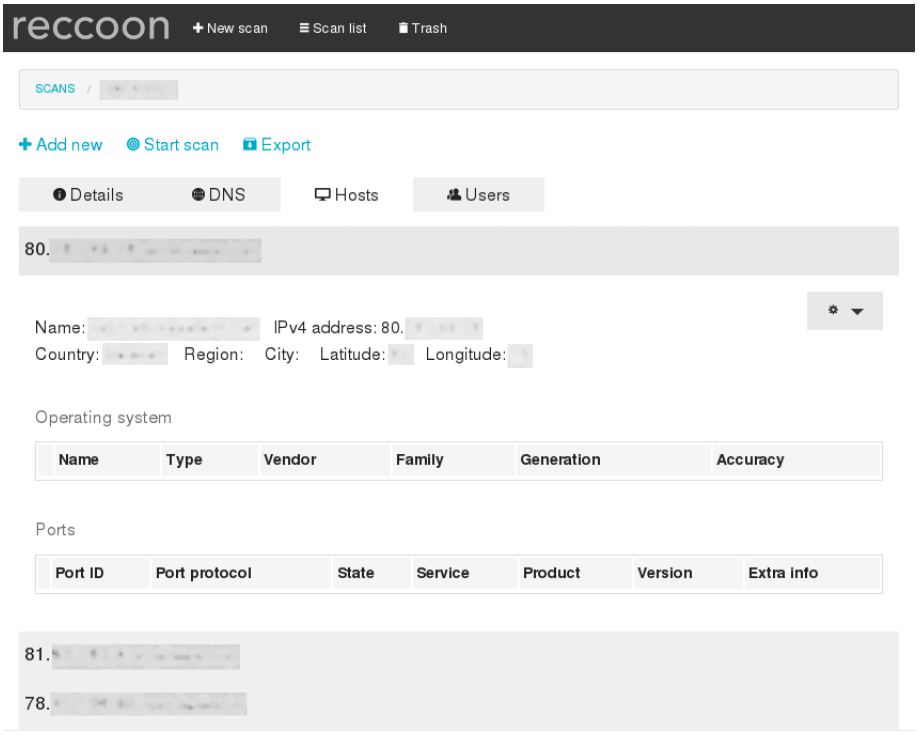
Webgrensesnittet følger filstrukturen til Django, som er en MVT-arkitektur (les mer i seksjon 4.2.2). Reccoon følger lagdelingsmodellen, og er inspirert av Recon-ng sin struktur. Det ble prioritert at Reccoon skulle være selvstendig, og tredjeparts moduler blir derfor inkludert i distribusjonen av rammeverket.

- `Reccoon/`
  - `data/` - *Midlertidige filer blir lagret her.*
  - `libs/` - *Tredjeparts moduler lagres her.*
  - `modules/` - *Reccoon moduler lagres her.*
  - `www/` - *Django og webgrensesnitt.*
    - `django/` - *Django filer.*
    - `ReccApp/` - *Reccoon webgrensesnitt.*
      - `templates/` - *Maler for webgrensesnitt.*
      - admin.py
      - models.py - *Databasemodell.*
      - tests.py
      - views.py - *Håndterer visningene, laster maler.*
    - `Reccoon/` - *Innstillinger.*
      - settings.py - *Generelle innstillinger for Django og Reccoon.*
      - urls.py - *HTTP-spørringer sjekkes opp mot denne.*
      - wsgi.py
    - `static/` - *Statiske bilder, Javascript og CSS filer.*
    - manage.py
  - core.py - *Logikken til Reccoon. Kan kjøres separat.*
  - reccoon.py - *Oppstartsprogram for Reccoon.*

5.4 Visuelt design

Det benyttes et front-end framework, Foundation 5, for det visuelle designet på nettsiden. Den er bygget opp med HTML, CSS, og Javascript. Dokumentasjon på Foundation finnes på deres nettsider [17]. Dette designet avviker noe fra den originale planen, fordi det oppsto et problem med at nettsidene ble for høye. Det var ikke satt noen krav for hvordan Reccoon skulle behandle dette, men prosjektgruppen ønsket å gjøre dataen oversiktlig og praktisk strukturert.

Designet ble til slutt relativt minimalistisk med en hovedmeny øverst på siden, som gir brukeren oversikt over alle skanningene sine. For å korte ned høyden på nettsiden som ble generert, så er informasjonen fordelt mellom faner (Details, DNS, Hosts, og Users). Med kollapserbare tittellinjer så ble høyden ytterligere redusert. Dette gir bedre oversikt for brukeren. Vi ser i figur 12 at tittellinjen merket ”80” er åpen og gir brukeren yttligere informasjon om denne enheten, mens linjen merket ”81” er lukket. Under hovedmenyen er det en linje med breadcrumbs som gjør at brukeren lettere kan gå tilbake i sine visninger.



Figur 12: Visuell design. Bildet er sensurert.

5.5 Implementering av ny modul

På grunn av at rammeverket er skrevet i Python, så trengs det en gradvis oppbygging av nye moduler. Derfor anbefales det å følge metoden som blir beskrevet her. Denne metoden beskriver hvordan en ny modul blir koblet til både kjernen og nettlesergrensesnittet. I denne beskrivelsen skal det gjennomgåes hvordan Nmap-modulen ble lagt til. Det vil foregå i to separate operasjoner. Først må en mal bli opprettet under `Reccoon/www/ReccApp/templates/`. Kopier gjerne en eksisterende mal og bruk det som et utgangspunkt. Deretter må Django vite at det eksisterer en ny visning. Legg derfor til en kobling i Django-kontrolleren sin oversikt slik:

```
# Reccoon/www/Reccoon/urls.py
...
urlpatterns = patterns('',
    ...
    """ Add the new URL to the Django controller. """
    url(r'^nmapscan.html', 'ReccApp.views.nmapscan', name='nmapscan'),
    ...
)
```

Django vil kunne lete etter funksjonen `nmapscan` som blir implementert på måten som blir beskrevet under. I første omgang vil det holde å bare returnere den nye malen som er laget. På denne måten kan man laste malen for å gjøre endringer på den. Siden Nmap-visningen må ha et skjema for at brukeren skal kunne spesifisere parametere til Nmap, så skrives disse feltene ut til fil i følgende kode for å sjekke at alt fungerte slik det skulle.

```
# Reccoon/www/ReccApp/views.py
from core import nmap
...
```

```
def nmapscan(request):
  """ Save each form variable and write it to a file. """
  scan_id = request.POST.get('Scan', None)
  ...
  f = open('testfile', 'w')
  f.write(scan_id)
  ...
  f.close()
  """ Check for any forgotten variables in the file. """
  return render_to_response('nmapscan.html')
...
```

Etter det forrige steget så kan man skille mellom første HTTP-henvendelse (GET) og andre skjema-henvendelse (POST) i koden under. Nå skal feltene bli lagret i en datastruktur kalt *dictionary* i Python. Dette er for å sende de videre til Nmap-skanningen i Reccoon-kjernen.

```
# Reccoon/www/ReccApp/views.py
from core import nmap
...
def nmapscan(request):
  if request.method == 'GET':
    """ First: Generate the form-view for a Nmap scan. """
    return render_to_response('nmapscan.html',locals(),RequestContext(
    request))
  elif request.method == 'POST':
    """ Second: Save the formvariables into a dictionary (nmapparam).  """
    ...
    """ Create the hostList for selected hosts to scan. """
    if scanAll == 'Yes':
      """ Append all host IDs to hostList. """
    else:
      """ Append only selected host IDs to hostList. """
    """ Start the nmap thread. It will run in the background. """
    nmap(scan_id, hostList, nmapparam).start()
    """ Return the user back to the scanning page. """
...
```

Den andre delen i denne prosessen består av å utvikle modulen som henter inn data, og koble den sammen med logikken til Reccoon (kjernen). Nmap-modulen vil starte Nmap-programmet og gi det variable som om det skulle gått via terminalen. Det vil si at Reccoon sender teksten `nmap -Pn -sS -sV -O -T3 -oX /root/bacheloroppgave/Reccoon/www/../data/nmapscan_12.34.56.78.xml 12.34.56.78` til kommandolinjen i terminalen.

```
# Reccoon/modules/nmap.py
import anything

def nmap(scanID, hostList, params):
  """ Create a dynamic XML filepath and make parameter variables NULL. """
  ...
  """ Make the parameters to a string, and submit it to Nmap. """
  """ If a parameter is empty ('') then %s will not display anything. """
  self.command = 'nmap %s%s%s%s%s%s%s%s-oX %s %s' % (
    """ All parameters. """
  )
  """ Execute the command generated. """
  self.process = subprocess.Popen(self.command)
  """ Get terminal output & errors. """
  self.out, self.err = self.process.communicate()

  """ In this example, we only return a XML filepath, but on other """
  """ modules we return a dictionary with values. Return the suitable """
  """ datastructure for your needs. """
```

```
  return results

if __name__ == "__main__":
  """ This if-check is needed to test the new module results. """
  parameters = {'param1':value1, 'param2':value2}
  results = nmap('1', parameters)
  print(results)
```

Et tips er å benytte den datastrukturen som ble skrevet ut til fil fra `views.py` som `parameters` i koden over. Dette vil spare arbeid med å bygge opp variablene som blir brukt for å sende til Nmap. Benytt `if __name__ == "__main__":` for å teste at modulen skriver ut korrekt informasjon. `print self.out` som blir returnert fra `self.process.communicate()`. Deretter skal modulen kobles til og startes fra Reccoon kjernen. Det betyr at modulen må bli importert og ellers programmert i filen `core.py`.
Det aller siste skrittet i denne fremgangsmåten er å koble funksjonen fra `core.py` til webgrensesnittet i `views.py`. La funksjonen i kjernen arve fra `threading.Thread`, se eksempel i koden under. Dette vil la webgrensesnittet kalle på funksjonen `nmap.start()` som blir arvet fra `threading.Thread`. Dette vil tillate at Nmap startes som en tråd i bakgrunnen, og webgrensesnittet kan fortsette å svare på HTTP henvendelser fra brukeren.

```
# Reccoon/core.py
from nmap import *

class nmap(threading.Thread):
  def __init__(self, scanID, targets = [], params = {}):
    threading.Thread.__init__(self)
    self.scanObject = scanlist.objects.get(id = scanID)
    self.targets = targets # from views.py: hostList
    self.params = params # from views.py: nmapparams
    self.results = None

  def run(self):
    self.scanObject.Status = 1
    self.scanObject.save()
    """ For each target in targets. """
    for self.target in self.targets:
      self.host = Hosts.objects.get(id = int(self.target))
      """ Use either the IPv4/6 address to start the Nmap module. """
      ...
      self.results = nmapModule(self.host.IPv4/6_Address, self.params).run
    ()
      """ Parse the XML file returned from the module. """
      ...
      """ Create a new port/OS from the results. """
      ...
      """ Remove the XML from the filepath. """
      os.remove(self.results)

    self.results = None
    self.scanObject.Status = 0
    self.scanObject.save()
    sys.exit()
```

Bruk dokumentasjonen fra Django for å programmere i webgrensesnittet [18].

6 Testing og kvalitetssikring

For å kartlegge kvaliteten av rammeverket er det flere punkter som må testets og måles. Dette kapitlet vil gi innsikt i hvordan gruppen har gått frem for å undersøke om rammeverket innfrir de kravene som er satt i kravspesifikasjonen.

6.1 Testmetode

For å måle kvaliteten, så skal gruppen legge vekt på intuitet, brukervennlighet og kvaliteten på informasjonen som blir returnert. I tillegg skal tidsbruken som går med på å hente denne informasjonen måles. Det er fordi effektivitetsmålet til oppgaven er at rammeverket skal gi inforamsjonssikkerhetskonsulenter tidsbesparelser. For å dekke alle disse punktene vil prosjektgruppen foreta en grundig test av rammeverket. Denne testen vil bli delt inn i to deltester som sammenlagt vil gi et klart bilde av rammeverkets brukervennlighet, kvalitet og effektivitet:

- *Deltest 1: Brukervennlighet*: I den første deltesten vil rammeverket bli testet av utenforstående aktører, som i dette tilfellet vil være medstudenter av gruppemedlemmene. Testpersonene vil hver for seg benytte rammeverket til å utføre en ekstern network footprinting av et egendefinert mål. Gruppemedlemmene vil observere hvordan testpersonene benytter seg av rammeverket. I tillegg vil testpersonene svare på en spørreundersøkelse angående brukeropplevelsen av rammeverket, etter at du har fullført skanningen (vedlegg F inneholder spørreundersøkelsen som ble besvart).
- *Deltest 2: Kvalitet og tidsbruk*: Den andre delen av testen vil bli utført av gruppemedlemmene selv. I denne deltesten vil det legges fokus på kvaliteten av den returnerte informasjonen, samt tidsbesbarelsen ved å bruke rammeverket i stedet for å bruke verktøyene individuelt. Gruppen vil begynne med å skanne et mål ved hjelp av de enkeltstående verktøyene, og sette resultatene opp i en rapport. Deretter vil gruppen gjøre den eksakt sammen jobben ved bruk av rammeverket og sammenligne resultatene og tidsbruken til de forskjellige metodene.

6.2 Testmiljø og gjennomførelse

Vurdering av testmiljø

Under forbredelse av testmiljøer kom prosjektgruppen frem til av det er to mulige miljøer testene kan utføres i. Den første er å lage et eget domene som inneholder virtuelle maskiner. Denne løsningen krever mye arbeid og forbredelse i form av å sette opp maskiner med forskjellige operativsystemer og tjenester kjørende, nettverkskonfigurasjon, og ligendende. Gruppen var også usikker på om alle modulene ville virke ordentlig i et slikt virtualisert miljø. Det virtualiserte miljøet ville gjort det vanskeligere å teste for eksempel innhenting av brukerinformasjon. Det er fordi det blir benyttet verktøy som igjen bruker søkemotorer som Google for å finne dokumenter for å hente denne brukerinformasjonen. Dette testmiljøet er veldig forskjellig fra de virkelige systemene som Reccoon vil bli brukt mot, og blir ikke en god nok test for rammeverket.

Den andre muligheten er å foreta testene opp mot et reelt domene. Her vil det ikke være noen jobb med å sette opp et testnettverk, og testene kan utføres i et naturlig og realistisk miljø. Dette vil gi gode testresutater fordi det er slike miljøer rammeverket er beregnet å utforske. Men som vi nevnte i avsnittet om juss og etikk 2.3.4 kan dette havne i en juridisk gråsone. Siden det ikke er lovverk eller reglementer, verken for Høgskolen i Gjøvik eller Uninett, som direkte motstrider med operasjonene rammeverket utfører, velger gruppen å benytte seg av den sistenevnte testmetoden.

Deltest 1

Prosjektgruppen stiller med to datamaskiner til disposisjon under testen, som har Kali Linux og Reccoon installert. Målet med testen er at personene skal starte Reccoon helt på egenhånd og utføre en external network footprinting av et domene. Noen av personene har vært uten forkunnskaper om hva ekstern network footprinting omhandler, og de andre hadde slik forkunnskap. Måladressen fikk testpersonene velge helt selv, fordi gruppen ville se om de skrev inn adressen på riktig måte. Prosjektgruppen vil overvåke hvordan testperosnene håndterer oppgaven de har fått tildelt og hvor enkelt det er å forstå brukergrensesnittet. Personene vil gjennom hele prosessen få minimalt med veiledning om bruken av rammeverket. Når testen er ferdig så ble testpersonene spurt om de kunne svare på en spørreundersøkelse som omhandler brukervennligheten av rammeverket. Personene ble bedt om å svare ærlige på spørsmålene, uten å sminke på sannheten.

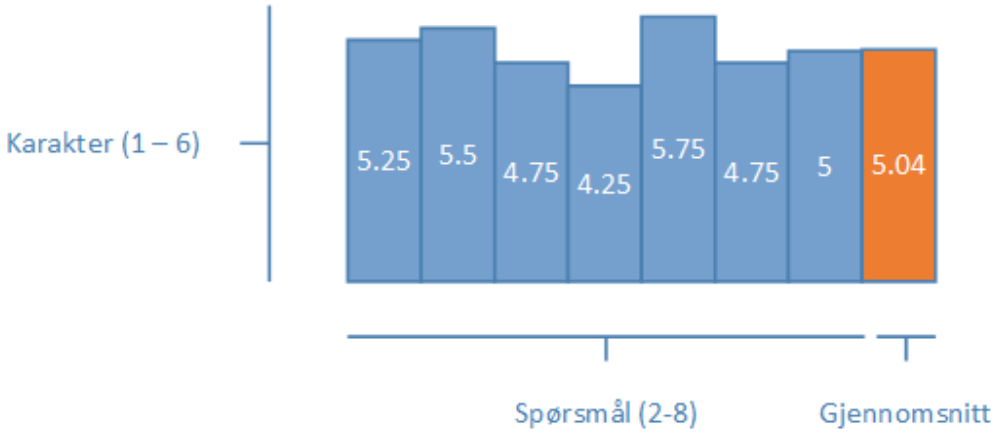
Deltest 2

Den siste deltesten ble utført av prosjektgruppen selv. Gruppemedlemmene vil velge to måladresser som det skal bli foretatt to metoder for ekstern network footprinting mot. Den første metoden er å manuelt gjennomgå verktøyene som Reccoon benytter, og aggregere data for å lage en rapport. Den andre metoden er å bruke Reccoon mot de to samme målene. Under begge metodene så skal tiden bli målt. Den manuelle metoden vil følge samme struktur som Reccoon bruker for å gjennomføre sine søk. Det betyr at personen bruker disse verktøyene i rekkefølge: Dmitry, Dnsenum, Dnsdict6, Fierce. Fem nettverksenheter ble skannet av Nmap. Og til slutt TheHarvester. Det ble ikke gjennomført geolokasjonsskanning og HaveIBeenPnwed-skanning i den manuelle testen. Når begge metodene var utført, så ble resultatene sammenlignet for å bekrefte at informasjonen stemmer.

6.3 Testresultater

Deltest 1

Enkeltheten og brukervennligheten kom veldig godt ut i spørreundersøkelsen, selv om bare halvparten av brukerne tildigere hadde erfaring med penetrasjonstesting. Det betyr at utgangspunktet for rammeverket er veldig godt, men det var noen felles feil som ble gjort av testpersonene. Fellesnevneren er at de fleste prøvde å bruke *www* foran i domenet, men at de skylte på seg selv for ikke å ha lest eksemplet i feltet før de skrev domenet. Testpersonene savnet en progresjonsbar som viste hvor langt i prosessen skanningen var. På denne måten kunne de vite at skanningen ennå pågikk. I tillegg savnet de muligheter for å pause, stoppe og ellers endre status på skanninger, og at meny-linjen var mer synlig under skrolling.
Figur 13 viser gjennomsnittskarakterene på hvert av spørsmålene som omhandler brukervennligheten av Reccoon (spørsmål 2 til 8 i spørreundersøkelsen). Søylen helt til høyre



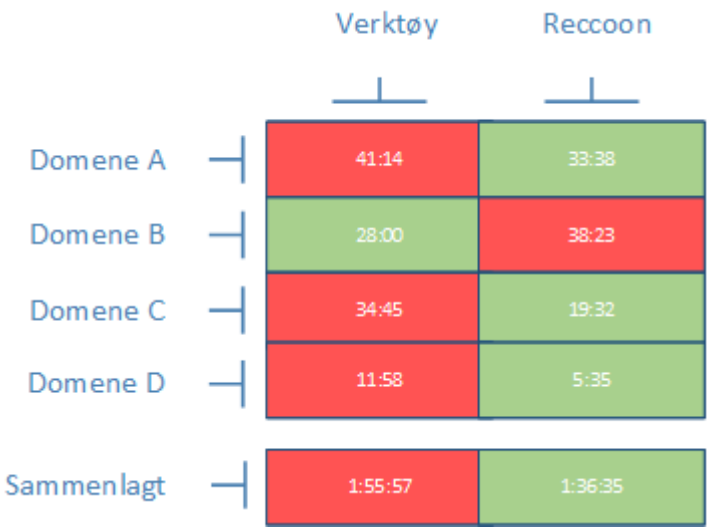
Figur 13: Karakteroversikt for spørreundersøkelsen.

representerer gjennomsnittskarakteren 5.04 (av 6 mulige). Noe som er et veldig bra resultatet. Vedlegg G inneholder svarene som testpersonene ga i spørreundersøkelsen.

I de siste delene av spørreundersøkelsen hadde testpersonene muligheten til å skrive tilbakemeldinger om rammeverket, og her har prosjektgruppen fått inn mange punkter som omhandler mulige forbedringer. Alle disse tilbakemeldingene går på små forbedringer av brukergrensesnittet. Det er med andre ord ingen som har hatt noen innvendinger angående rammeverkets funksjonalitet.

Deltest 2

Resultatene for deltest 2 vil baseres på tidsforskjellen mellom bruk av Reccoon og bruk av de enkeltstående verktøyene, og analyse om begge metodene returnerer identisk informasjon. Hvert av gruppemedlemmene har foretatt en skanning mot to forskjellige domener, både ved bruk av enkeltstående verktøy og med Reccoon.



Figur 14: Sammenligning av tidsbruk mellom manuelt arbeid og Reccoon.

Resultatene som er i figur 14 viser at det er tidsbesparende å benytte seg av rammeverket i stedet for manuelt arbeid. I tillegg til at det er tidsbesparende så var testresultatene identiske. Bruk av rammeverket går altså ikke ut over kvaliteten av den returnerte informasjonen. Det er også viktig å huske på at rammeverket automatisk presenterer resultatene i en oversiktelig rapport, noe som tar lenger tid å lage med de manuelle verktøyene. I tillegg så har rammeverket mulighet til å finne den geografiske lokasjonen alle enehente som blir funnet under operasjonen, samt sjekke alle epostadressene opp mot HaveIBeenPwned for å se om de kan være kompromitert. Dette ville vært en stor og repetitiv jobb om det skulle utføres i den manuelle gjennomgangen.

Testen viser at i ett av fire tilfeller var det faktisk raskere å benytte seg av de enkeltstående verktøyene fremfor rammeverket. Grunnen til dette avviket er uvisst. Gruppen prøvde å undersøke hvordan dette hadde oppstått uten resultater.

6.4 Analyse av testresultater

Oppsummering av testresultatene viser at rammeverket i stor grad gir det resultatet prosjektgruppen initielt sett håpet på. Programvaren har et brukergrensesnitt som viser seg å være så enkelt og forståelig som ønsket, også for personer uten tidligere kjennskaper til ekstern network footprinting. Kvaliteten på informasjonen som rammeverket henter inn holder mål, og tidsbruken som går med på å hente inn denne informasjonen går ned i forhold til tiden det ville tatt å utføre en manuell network footprinting.

I tillegg til testene som prosjektgruppen har gjennomført, har oppdragsgiver også fått muligheten til å teste rammeverket. Tilbakemeldingene som ble gitt av oppdragsgiver var utelukkende positive. Oppdragsgiver har testet rammeverket i produksjon, og har meddelt at det bidro til en merkbar tidsbesparing uten å gå utover kvaliteten. Dette er med på å understreke at resultatene vi har fått fra testfasen er korrekte.

7 Konklusjon

Dette kapitlet vil presentere en vurdering av hvorvidt prosjektgruppen har nådd målene som er satt for oppgaven. Dette inkluderer også muligheter for fremtidig utvikling av programmvaren etter prosjektperiodens slutt. Kapitlet avsluttes med en evaluering av gruppearbeidet og et svar til promlemstillignen gruppen har satt for oppgaven.

7.1 Måloppnåelse

Formålet med dette prosjektet var å utvikle et rammeverk for å forenkle gjennomføringen av external network footprinting. Resultatet av prosjektet er et rammeverk som benytter eksisterende programvare i Kali Linux for å gjøre nettopp dette. Prosjektgruppen har fått mye frihet til å utvikle rammeverket slik vi selv har ønsket. Men gruppen har likevel fulgt oppgavebeskrivelsen, vedlegg A, sine krav til rammeverket. Det betyr at rammeverket benytter verktøy som er listet opp i beskrivelsen og prøver å inkludere de fleste punktene fra PTES. Gruppen har fått frihet til å gjøre som vi ellers ønsket, og det har resultert i flere krav som blir innfridd av Reccoon. Det har resultert i en løsning som prosjektgruppen mener er av høy kvalitet, og i tillegg har flere gode muligheter for utvidelser. Reccoon har løst alle kravene som er blitt satt i kravspesifikasjonen, kapittel 2.

Oppdragsgiver har prøvd rammeverket to ganger. Den første gangen var bare for å teste og gi tilbakemeldinger til prosjektgruppen. Den andre gangen ble rammverket prøvd i reell produksjonssammenheng. Resultatet var at rammeverket hjalp oppdragsgiver med jobben for å finne external network footprint på ti minutter, en jobb som oppdragsgiver fortalte normalt ville tatt rundt en og en halv time.

Rammeverket presenterer et enkelt inntrykk av funksjonaliteter som er blitt implementert, og kan derfor gi den virkningen av at det ikke er blitt lagt mye arbeid i prosjektet. Men det er lagt til mye arbeid for å få denne enkelheten og brukervennligheten av Reccoon. Det er veldig mye som skjer i kulissene som ikke blir sett av sluttbrukeren. I tillegg er det lagt mye arbeid i å designe en grunnstruktur som gjør det lettere for nye utvidelser av rammeverket.

Gruppen har fulgt prosjektplanen og milesteinene veldig nøye i startfasen av prosjektet. Det ble startet tidlig med å gjennomføre forprosjektet, noe som resulterte i at gruppen kunne begynne rett på hovedoppgaven på begynnelsen av semesteret. Denne strukturen forsvant litt underveis, og gruppen arbeidet mer dynamisk enn det vi hadde forutsett. Det var ingenting negativt i dette, fordi gruppen viste at vi kunne gå tilbake til planen når som helst. Gruppearbeidet har fungert mye bedre enn initielle forventninger, og vi er fornøyd og stolte over å nå resultatene som ble satt.

7.2 Fremtidig utvikling

Reccoon har store forbedringsmuligheter med å legge til nye programmer og funksjonaliteter. Denne seksjonen beskriver egenskaper som prosjektgruppen kunne implementert, men som det ikke var nok tid til å implementere.

- Det er ikke en selvfølge at informasjonssikkerhetskonsulenter får ett domene som et mål, det vil derfor være til stor hjelp at rammeverket godkjenner IP og IP-ranger som legitime mål.
- Webgrensesnittet genererer statiske nettsider ved HTTP-henvendelser. Det betyr at sider med mye informasjon (som en skanningvisning med oversikt over alle data) vil bruke langt tid til å laste. Vi ønsker å redusere lastetiden hver side bruker med å benytte jQuery for å hente litt og mer eksakt data som brukeren spør om.
- Nettsider fungerer veldig bra for å få folk til å samhandle i et prosjekt. Prosjektgruppem ønsker at rammeverket skal benytte en brukerdatabase for innlogging og mulighet for brukerkontroll. Men gruppen måtte derfor sette større fokus på input validering, isolere skanninger fra andre brukere, og andre kritiske funksjonaliteter. Det positive er at nettsiden kunne flyttes over på en egen server, og da vil produksjonsmiljøet være mer stabilt og gi flere brukere tilgang til de samme skanningene.
- De to foregående forslagene er de mest nødvendigste for å utvikle nettsiden til å fungere på mobile enheter. Dette vil gjøre at informasjonssikkerhetskonsulentene kan starte skanninger fra en normal server og administere alt via Internett. Dette vil gjøre at konsulenten får større mobilitet enn når den lokale maksinen utfører jobben. Det vil også tillate at rammeverket blir cross-plattform og tilgjengelige i andre operativsystemer.
- I tillegg ønsket prosjektgruppen å implementere sårbarhetsskanning for å gjøre prioriteringen av interessante nettverksmål lettere. Da kunne prosjektgruppen prioritert nettverksenhetlisten for å vise hvilke nettverksenheter som var mest sårbare.
- Prosjektgruppen har lagt til en modul som kommuniserer med Metasploit, men har ikke fått tid til at denne overfører resultatene til Metasploit. Derfor endte det opp med en enkel eksporteringsmulighet til dette verktøyet.
- En logg som viser informasjon som funn av nye nettverksenheter, feilmeldinger, og generell tilbakemeldinger til sluttbrukeren.

7.3 Evaluering av gruppearbeid

Gruppen møtte ikke de potensielle problemene som ble forutsett i forprosjekt kapittel 5 (vedlegg D). Arbeidet mellom to gruppemedlemmer har vært veldig dynamisk. Det har ført til at vi i liten grad har fulgt GANTT-planen, selv om hovedtrekkene ble fulgt. Gruppemedlemmene har klart å utfylle hverandre, og hatt komplementære egenskaper for å gjennomføre prosjektet. Det har i liten grad vært behov for en gruppeleder og rollefordelinger, fordi begge har gått inn i diskusjoner med den andre før noen avgjørelse ble tatt.

Gruppen har vært flinke til å fordele arbeid mellom gruppemedlemmene. Gruppen har lagt vekt på at hvert gruppemedlem har fått prøvd seg på forskjellige deler av arbeidsprosessen, selv om noen har bidratt mer på enkelte områder. Det har vært god kontakt mellom gruppen og oppdragsgiver. De har vært støttende under hele prosjektet, og det er gledelig at rammeverket har fungert etter oppgavebeskrivelsen. Det var særlig under implementeringen av rammeverket at gruppen oppdaterte en blogg gjevnlig for å informere oppdragsgiver. Dette gav dem en mulighet for å komme med hyppige tilbakemeldinger.

Prosjektgruppen siktet litt for bredt i starten av prosjektet. Gruppen ønsket for eksempel å inkludere sårbarhetsskanning inn i rammeverket. Men gruppen har vært flinke til å prioritere hvilke funksjonaliteter som skulle initielt inn i rammeverket, og ikke overvurdert tidsbruken. Gruppen har vært smidige og vært åpne til forbedringer av rammeverket, og synes det er strålende at gruppen traff blink nesten med en gang. Oppdragsgiver har vært flinke til å begrense arbeidsmengen i forhold til tidsbruken vi har hatt. De har i tillegg hatt en forståelse for at det er denne rapporten som skal karaktersettes.

Prosjektgruppen har hatt gjevnlige møter med veileder og oppdragsgiver, der de har kommet med tilbakemeldinger om at arbeidsmetoden har i stor grad vært selvstendig. Dette har gjort gruppen selvsikre på at de var på riktig vei og ville oppnå målene som vi tidligere satt. Gruppen har på et tidspunkt vært foran prosjektplanen under utviklingen av rammeverket. Alle ting tar tid, og det var særlig testingen som tok lenger tid enn planlagt. I innspurten av innleveringen har gruppen nok med tid til å levere rapporten et par dager før fristen.

7.3.1 Individuell evaluering

Kasper Kristoffersen

Jeg er veldig fornøyd med både arbeidesprosessen vi har hatt, og resultatet vi sitter igjen med etter bachelorperioden. Vi har kun vært to studenter på gruppen og selv om dette gjør det vanskeligere å ha en en fast gruppestruktur, mener jeg det har ført til at vi arbeidet dynamisk og effektivt. I forprosjektet planla vi å ha rullerende ansvarsområder, noe vi også har realisert til en viss grad gjennom hele bachelorperioden.

Jeg synes vi har gjort en veldig god jobb under planleggingsfasen av prosjektet. Dette har ført til at vi kom raskt igang med oppgaven og har arbeidet jevnt gjennom hele perrioden, og dermed ungått å lengre arbeidsdager og stressende situasjoner den siste tiden. Det har også vært et godt arbeidsmiljø i gruppen.

Vi har arbeidet med et tema som jeg synes er utrolig spennende. Det å få lov til å lage et verktøy som skal bli benyttet i penetrasjonstester av Watchcom er inspirerende i seg selv. Når dette er sagt har jeg aldri vært best i klassen når det kommer til programmering. Det har derfor vært utrolig bra å være på gruppe med Jan William, som har veldig god kontroll på dette området. Da er det alltid hjelp å få for å komme seg vidre. Dette har ført til at min kompetanse innen programmering har økt drastisk iløpet av perrioden, og også motivasjonen min for programmering.

Det er veldig morsomt at Watchcom virker fornøyd med resultatet vi har produsert. Det

er også morsomt å se hva vi kan få til ved å kombinere kunnskap vi har tilegnet oss gjennom de forskjellige emnene på Høgskolen i Gjøvik.

Jan William Johnsen

Jeg og Kasper ble veldig tidlig, høsten året før, enige om å arbeide på bacheloroppgavegruppe sammen. Jeg har tidligere jobbet med Kasper på et objekt-orientert programmering prosjekt, og var klar over at jeg mulig ville få hovedansvaret for programmeringen hvis vi gikk for en programmeringsoppgave. Dette syntes jeg ville være helt greit.

Watchcom hadde dette halvåret en presentasjon i IMT3501 Programvaresikkerhet, og i avsluttningen presenterte de oppgaven for klassen. Allerede her fikk jeg en visjon om hvordan jeg ønsket å løse oppgaven. Det tok ikke lang tid før oppgaveforslagene var tilgjengelige for alle, og til vi hadde bestemt oss for dennne oppgaven og fått reservert den hos oppdragsgiver. Vi valgte ut denne oppgaven fordi vi syntes det var den morsom, utfordrende og ville gi oss kunnskaper som kunne bli brukt senere i arbeidslivet.

Jeg er fornøyd med både arbeidsfordelingen og Reccoon prosjektet sin utvikling. Vi har klart å holde en gjevn fart på arbeidet denne våren, selv om jeg synes jeg kunne arbeidet litt mer på rapportskriving. Skrivingen har fort blitt veldig kjedelig, og jeg har bestemt at ved slike prosjekter senere så skal programmering og skriving bli mer fordelt.

7.4 Konklusjon og svar på problemstilling

Oppgaven var å utvikle et rammeverk som samler informasjon om et eksternt nettverk. Denne informasjonen skulle settes sammen for å danne et oversiktlig bilde av nettverket, sett utenfra. På grunn av rammeverkets omfang og et halvt år med programmering og rapportskriving, så er prosjektgruppen veldig fornøyde med det endelige resultatet. Reccoon har gjort det enklere for oppdragsgiver å utføre tidligere tidkrevende oppgaver.

Reccoon oppfyller kravene som er blitt spesifisert i kravspesifikasjonen 2, og rammeverket er blitt et godt og stabilt produkt. Etter tilbakemeldinger fra oppdragsgiver, innfrir rammeverket oppdragsgivers forventinger til effektivitet og funksjonalitet. Oppdragsgiver blir sitert på at de brukte ti minutter på en jobb de brukte en og en halv time før de fikk rammeverket.

Testpersonene som prøvde Reccoon hadde alle tilbakemeldinger om at det er enklere å benytte seg av rammeverket enn hvert enkeltstående verktøy. Automatiseringen har effektivisert denne arbeidsprosessen. Funksjonaliteten Reccoon tilbyr vil redusere arbeidstimer relatert til informasjonsinnhenting av external network footprinting.

I introduksjonen av denne rapporten fastsatte prosjektgruppen en problemstilling (1.10) for oppgaven. Gjennom prosjektperioden har gruppen funnet svar på denne problemstillingen. Den tekniske informasjonsinnhentingsdelen av external network footprinting kan i veldig stor grad automatiseres. Dette er noe prosjektgruppen har bevist med utviklingen av Reccoon. Under testingen av rammeverket fikk prosjektgruppen også vist at denne automatiseringen fører til en merkbar tidsbesparing, uten at dette går ut over kvaliteten av informasjonen.

Konklusjonen for dette prosjektet er at det har vært utfordrende og læringsrikt. Det har vært et veldig godt prosjekt for å vise noen av kunnskapene vi har tilegnet oss på Høgskolen i Gjøvik.

Bibliografi

[1] Engebretson, P. 2011. *The Basics of Hacking and Penetration Testing*. Syngress, USA, 1st edition.

[2] Penetration testing execution standard. web 14. mai 2014 http://pentest-standard.org/index.php/Intelligence_Gathering.

[3] Watchcom security group as. web 14. mai 2014 http://www.watchcom.no/.

[4] Tomes, T. oktober 2012. Web 13. mai 2014 https://bitbucket.org/LaNMaSteR53/recon-ng/.

[5] Rapid7. Web 13. mai 2014 http://www.metasploit.com/.

[6] Micallef, S. Web 13. mai 2014 http://spiderfoot.net/.

[7] Python software foundation. web 14. mai 2014 https://docs.python.org/2.7/.

[8] Wikipedia. Web 14. mai 2014 http://en.wikipedia.org/wiki/Use_case.

[9] Wikipedia. Web 14. mai 2014 http://no.wikipedia.org/wiki/Superbruker.

[10] Justis- og politidepartementet. web 14. mai 2014 http://www.regjeringen.no/nb/dep/jd/dok/regpubl/otprp/2008-2009/otprp-nr-22-2008-2009-/2/8/1.html?id=540240.

[11] Guzel, B. Web 14. mai 2014 http://code.tutsplus.com/articles/sql-for-beginners-part-3-database-relationships--net-8561.

[12] Wikipedia. Web 14. mai 2014 http://no.wikipedia.org/wiki/Normalisering.

[13] Penetration testing execution standard - footprinting. web 14. mai 2014 http://pentest-standard.org/index.php/Intelligence_Gathering#Footprinting.

[14] AS, U. N. Norid. web 17. mai 2014 http://www.norid.no/.

[15] Stuart McClure, Joel Scambray, G. K. 2009. *Hacking Exposed: Network Security Secrets and Solutions, Sixth Edition*. McGraw-Hill Osborne Media, USA, 6th edition.

[16] kali.org. Kali linux documentation web 16. mai 2014 http://docs.kali.org/introduction/what-is-kali-linux.

[17] Zurb foundation 5. web 15. mai 2014 http://foundation.zurb.com/docs/.

[18] Django web framework 1.6. web 16. mai 2014 https://docs.djangoproject.com/en/1.6/.

Oppdragsgiver:

Watchcom Security Group AS
Postboks 1861, Vika
N-0124 Oslo

A Oppgaveforslag

Kontaktperson:

Bjørn Richard Watne
bjorn.watne@watchcom.no
918 12 177

Morten Gjendemsjø
morten.gjendemsjo@watchcom.no
480 23 131

Eivind Utnes
eivind.utnes@watchcom.no
400 09 040

Security- and penetration testing -
Information correlation and reporting framework

Information gathering boils down to using the Internet to collect as much information as possible to about the target. Regardless of whether the information gathering is being done with offensive or defensive goals in mind, the value of the intelligence depends on coverage, quality of the analysis and how it is made accessible. While information gathering usually employ both technical and non-technical methods, this project is solely concerned with the former.

Security professionals employ numerous well-known tools when footprinting networks. Tools like Nmap provide both port scanning and host identification, while other tools like dnsenum, fierce and dnsmap can be used to provide insight into the domain. Metagoofil and Nessus can be used to extract metadata and scan for vulnerabilities, respectively.

All of the aforementioned tools have their strengths as well as restrictions and limitations. The common denominator is that they each provide a piece of the puzzle. The proposed project is to create a tool that automates the process of external network footprinting, by combining the pieces into a more complete view of a network's exposure on the Internet. A large part of this project will be to analyze the different tools' output, i.e. removing duplicate data and extracting the most valuable and interesting information from each tool. That is, an automated tool that employs widely used information gathering tools, aggregates and analyzes, and reports the result in an appropriate manner.

The aggregated data should be presented in a machine-friendly format, e.g. JSON or XML. In addition, the tool should also produce a report aimed at security professionals incorporating a high degree of technical information about the target

network. The data should be presented in an appropriate format, e.g. HTML, PDF or similar.
The students should strive to include data from each of the "Footprinting" approaches mentioned in PTES [1].

The following list provides examples of tools that the students may choose to incorporate in data gathering. Please note that this list is by no means complete, and students are encouraged to add or remove tools as they see fit.

- nslookup
- dnsenum/fierce
- traceroute
- telnet (banner grabbing)
- Nessus / Arachni
- Nmap
- Nikto
- Metagoofil
- Search engines (google hacking)
- Vendor/Equpment specific tools, e.g. cisco ios analysis.

This project requires knowledge of Linux as well as programming skills.

[1] http://www.pentest-standard.org/index.php/Intelligence_Gathering

B Prosjektavtale

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Watchcom Security Group AS

(oppdragsgiver), og

Jan William Johnsen og Kasper Kristoffersen

(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 03/01 - 2014 til 05/06 - 2014.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra HiG å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstillelse av den skriftlige besvarelsen vedrørende prosjektet.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter, taper mv. som inngår som del av eller vedlegg til besvarelsen, gis det en kopi av til HiG, som vederlagsfritt kan benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.

   Besvarelser med karakter C eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på bibliotekets del av skolens internett-sider. Dette avhenger av at studentene skriver under på en egen avtale hvor de gir biblioteket tillatelse til at deres hovedprosjekt blir gjort tilgjengelig i papir og nettutgave (jfr. Lov om opphavsrett). Oppdragsgiver og veileder godtar slik

offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og dekan om de i løpet av prosjektet endrer syn på slik offentliggjøring.

5. Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

6. Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer oppgavebesvarelsen med vedlegg (pdf) i Fronter. I tillegg leveres et eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan/prodekan som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere regulerer forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.

   Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.

10. Når HiG også opptrer som oppdragsgiver trer HiG inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Lasse Øverlier

Oppdragsgivers kontaktperson (navn): Morten Gjendemsjø

Student(er) (signatur): Jan William Johnsen dato 19/12-13

Kasper Kristoffersen dato 19/12-13

_____ dato _____

_____ dato _____

Oppdragsgiver (signatur): _____ dato 19/12/13

IMT Dekan/prodekan (signatur): _____ dato _____

Versjon Januar 2011hb

C Gruppekontrakt

Gruppekontrakt

til bacheloroppgave

Gruppemedlemmer:
Johnsen, Jan William - jan.johnsen@hig.no - 930 71 707
Kristoffersen, Kasper - kasper.kristoffersen@hig.no - 913 48 424

Tidsperiode: 01. jan 2013 til og med 05. jun 2013.

Oppmøte og møtetider: Vi har avtalt å faste møter, se listen under, for å arbeide med oppgaven. Andre møter og aktiviteter blir avtalt på foregående møter eller med å opplyse de andre medlemmene.

- Mandag kl. 09:00 - 17:00
- Onsdag kl. 09:00 - 17:00
- Torsdag kl. 09:00 - 17:00

Plikter: Alle i gruppen skal skrive en personlig daglig timelogg for arbeid som er blitt gjennomført. Alle i gruppen skal, så langt det lar seg gjøre, forsøke å legge ned 30 timer i uka på prosjektet. Alle medlemmer må delta på samtlige gruppemøter som blir avtalt, levere avtalt arbeid til avtalt tid og møter for gruppeveiledning med veileder. Ved sykdom eller annen gyldig fraværsgrunn skal de andre medlemmene på gruppen kontaktes slik at møtet kan utsettes, arbeid bli omfordelt, el. Alle har plikt til å melde fravær, manglende arbeid, og forfall så tidlig som overhodet mulig.

Arbeidsfordeling, samarbeid og egenarbeid: Alt arbeid skal deles så likt og retterferdig som mulig, alle har rett og plikt til å påse og påpeke forhold de mener er urettferdige. Dersom det oppleves for mye tidspress på en person pga for eksempel private forhold bør denne personen si i fra til det andre gruppemedlemmet (eller veileder dersom dette ikke er ønskelig), slik at eventuelle arbeids oppgaver kan omfordeles.

Utgifter: Alle utgifter til gruppearbeidet skal deles likt mellom medlemmene. Kvittering bør forvises.

Uenigheter: Alle bør argumentere for å oppnå enighet. Dersom det ikke blir en enighet i gruppen, skal veileder kontaktes. Veileder har siste ordet ved uenigheter.

Avskjediget: Dersom et gruppemedlem gjentatte ganger ikke møter opp til fastsatte tider (uten gyldig grunn), eller ikke bidrar i gruppeprosessen, til tross for advarsler. Kan gruppen oppløses og gruppemedlemmene vil individuelt jobbe videre med prosjektet. Veileder skal bli kontaktet for å gjøres klar over separasjonen.

Underskrifter: Når jeg undertegner er jeg enig i gruppekontrakten, og i at jeg vil gjøre mitt beste for å følge den. Jeg er klar over at alvorlige og gjentatte regelbrudd kan føre til eksklusjon fra gruppen. Undertegningen fant sted ved Høgskolen i Gjøvik, den 09/01/2014.

Jan William Johnsen

Kasper Kristoffersen

D Forprosjekt

FORPROSJEKT:

Rammeverk for automatisk nettbasert rekognosering og informasjonsinnhenting

FORFATTERE:
Jan William Johnsen
Kasper Kristoffersen

DATO:
2014

Innhold

Figurer

1 Introduksjon

For å få godkjent bachelorgrad innen informasjonssikkerhet ved Høgskolen i Gjøvik (HiG)[1], må en bacheloroppgave verdt 20 studiepoeng utføres i utdanningens 6. semester. Vi valgte oppgaveforslaget levert av Watchcom Security Group fordi dette var begge gruppemedlemmenes førsteønske. Oppgaven virket interessant, utfordrende og ikke minst relevant for videre karriere. Prosjektet går i grove trekk ut på å utvikle et rammeverk for å kunne utføre automatisk rekognosering og informasjonsinnhenting over internett. Tanken med dette er at Watchcom skal få et verktøy de kan benytte seg av når de utfører penetrasjonstester for sine kunder.

En penetrasjonstest er en metodikk for å teste system- og nettverkssikkerheten til en bedrift. Dette skjer ved at en vennlig aktør tar på seg rollen som fiendtlig angriper for å avdekke reelle svakheter i systemene. Penetrasjonstestenes rammer kan varriere avhengig av hva kundene har spurt etter. Det finnes også en rekke forskjellige penetrasjonstestmodeller, men vi vil forholde oss til metologien *Zero Entry Hacking* [2, s. 13] som blir undervist ved HiG i faget IMT3491 Ethical Hacking and penetration testing[2].

Av de fire fasene i denne metologien skal vårt produkt kunne fungere som et komplett hjelpemiddel for utførelse av de to første:

- Recconaissance: Her foregår selve informasjonsinnhentingen, man søker igjennom Internett for å finne teknisk informasjon om målet og dets ansatte.
- Scanning: Her skannes mål og personer som blir oppdaget under informasjonsinnhentingen. Med skanning menes identifisering av systemets enheter og åpne tjenester. Det skannes også etter svakheter i systemene som oppdages.
- Exploitation: Her utnyttes de svakhetene som oppdages under den foregående fasen for å la angriperen få tilgang til målets ressurser.
- Maintaining Access: Når målet er komrimert setter angriperen opp en bakdør til systemet for å lett få tilgang ved en senere anledning.

Den fulle oppgaveteksten for Watchcoms oppgave ligger i vedlegg **??**.

[1] `http://www.hig.no`
[2] `http://www.hig.no/content/view/full/28009/language/nor-NO`

2 Mål og rammer

Det ble nevnt i vårt første møte med Watchcom, den 19. desember 2013 refarat i vedlegg **??**, at rammeverk med GUI og Metasploit[1] rapportering er et pluss, selv om dette ikke er noe krav i forhold til oppgaven. Veileder ga oss deretter et tips om å bygge moduler til Armitage[2], som er et GUI administrasjonsverktøy til Metasploit. Vi diskuterte dette i gruppen, men kom frem til at vi fort kunne ha blitt for opptatt av at all data må passe sammen for at Armitage skal benytte seg av det under exploitation-fasen. Dette prosjektet skal i hovedsak ha fokus på rekognosering- og skanningfasen.

Vi fant et annet verktøy som allerede var et rammeverk for rekognoseringfasen, dette verktøyet er kalt Recon-ng[3]. Den originale planen var å bygge moduler til Recon-ng, men dette ble litt vanskelig på grunn av to ting: Få av modulene lagret data den fant, så vi måtte endre på veldig mange moduler hvis de blir brukt. Det vil også bli vanskelig å automatisere et verktøy som ikke lagrer data som blir innhentet. I tillegg måtte vi sette oss inn i en kode som mange har skrevet.

Hovedfokuset vårt vil ligge på hvilke elementer som må samles inn for å få en god dekning av målet, og hvilke verktøy som vil hjelpe oss i denne innsamlingen. Vi vil utvikle et eget program, men vi tror at vi vil benytte Recon-ng som et eksempel på hvordan man henter inn informasjon. I dette prosjektet fikk vi veldig mye frihet til å velge det vi selv ønsker. Resultatmålet reflekterer dette.

2.1 Effektmål

Rammeverket vi skal utvikle vil føre til at Watchcom kan utføre raskere og bredere informasjonsinnhenting. Resultatet av det automatiserte rammeverket skal være en rapport som senere kan brukes i den neste fasen av en penetrasjonstest. Rammeverket vil også gi de samme effektene til andre aktører i samme bransje som har interesse for det.

2.2 Resultatmål

Dette rammeverket skal utvikles for Kali Linux[4], som er en Debian-basert Linux distribusjon. Vi valgte Kali Linux på grunnlag av det brede utvalget av verktøy og det faktum at de er installert på standard lokasjoner. Dette vil føre til at rammeverket er lettere å installere på nye enheter og oppdatere til nyere versjoner.

Rammeverket skal være terminalbasert, og bruke en IP-adresse, en IP-range eller en URL som parameter. Andre parameter vil bli avgjort hvis de trengs senere. Rammeverket vil bruke denne inputen til å samle personell- og systemdata fra målet. Innsamlingen av personelldata er for å identifisere individer, ikke-listede telefonnummere, brukernavn (og om mulig passord), data fra sosiale nettverk og metadata fra filer. Innsamlingen av systemdata er for å identifisere aktive systemer, åpne porter og service-versjoner, operativsystemer, domener, IP blokker, og opplisting av DNS-informasjon.

[1] `http://www.metasploit.com`
[2] `http://www.fastandeasyhacking.com`
[3] `https://bitbucket.org/LaNMaSteR53/recon-ng/`
[4] `http://www.kali.org`

Nettsider, som f.eks. Google, benytter seg av et antall henvendelser per sekund før man blir utestengt fra tjenesten. Rammeverket vil være nødt til å ta hensyn til disse restriksjonene fra tredjepart tjenester.

2.2.1 Oppsummering av nøkkelfunksjoner

- Utviklet for Kali Linux og terminal basert
- Bruker IP-adresse, IP-range og URL som parametre
- Samler personelldata
- Samler system- og skanning data
- Resultatet eksporteres til XML og Metasploit

2.3 Rammer og avgrensinger

Vi har noen fastsatte datoer for oppgaven:

- 27. januar - Levere forprosjekt og prosjektavtale.
- 19. mai - Levere ferdigstilt prosjekt
- uke 22 - Levere timelogg og refleksjonsnotater.

Presentasjon av oppgaven er på enten 3., 4. eller 5. juni. Oppdragsgiver vil inviteres til å være tilstede under presentasjonen.

3 Oppgavebeskrivelse

Oppgaven er levert av Watchcom Security Group, og omhandler utviklingen av et rammeverk for nettbasert rekognosering og informasjonsinnhenting. I tillegg legges skanning av målet til, for at resultatet skal kunne brukes i den neste fasen av en penetrasjonstest. Derfor skal rammeverket generere en rapport til brukeren og dele informasjonen som er blitt hentet inn til andre verktøy, som f.eks. Metasploit.

Rammeverket skal samle inn så mye relevant informasjon som mulig. Både fra målets eget nettverk og fra åpne kilder på internett. Dette betyr at sannsynligheten er stor for gjentakelser av informasjon. Vi må ta ta hensyn til hvilke verktøy som dupliserer informasjon, og filtrere dette vekk fra rammeverket. I tillegg skal informasjon som er mer interessant enn andre ikke forsvinne i mengden.

Oppdragsgiver har satt opp noen forslag til verktøy som de kunne tenke seg at vi inkluderer i rammeverket, men vi står fritt til å velge verktøy selv. Vi synes at listen med verktøy er grei, men at vi vil velge verktøy først etter vi har oversikt over hvilke elementer som skal innhentes. Det er også oppgitt i oppgaveteksten at vi bør streve etter å inkludere data fra hvert punkt under *Footprinting* som er nevnt i PTES [1]. Oppdragsgiver har ellers satt få føringer i oppgaveforslaget, og i tillegg nevnt at vi har mye frihet i hvordan vi løser oppgaven.

Vi ønsker å lage et open source rammeverk. Det er fordi produktet da kan bli videreutviklet av andre, i stedet for at det blir liggende i en skuff. Vi blir nødt til å designe rammeverket på en ordentlig måte, slik at andre kan gjenoppta arbeidet vårt uten for mye stev. Designet vil også være offentlig. Vi ønsker at rammeverket skal være modulbasert. På den måten kan vi garantere at rammeverket fungere selv om vi legger til nye verktøy i inkrementer.

Rammeverket blir utviklet i Python. Implementasjonen vil gå raskere i Python, men det er vanskeligere å debugge koden. Dette er også et argument for at vi ønsker å utvikle rammeverket inkrementelt, fordi vi da kan debuggen koden for hvert inkrement. Python passer også bra til databaseprogrammer, og kan enkelt eksportere sine data til XML-format takket være de mange åpne modulene.

Et siste krav er at rammeverket skal eksportere all rådata til enten XML eller JSON. I tillegg blir en grunndig rapport generert til sikkerhetsekspertene, samt et format til Metasploit.

4 Organisering og ansvarsområder

Det er viktig at roller og ansvarsområder er delt mellom gruppemedlemmene. Dette vil gjøre implementasjonen lettere ved å redusere misforståelser, fremme samarbeid og gi klarhet om hvem som er ansvarlig for hva.

Det er bare to medlemmer i denne prosjektgruppen. Vi har bestemt oss for å ha en person som har oversikt over hele prosjektet og dets fremgang, og en referent som tar detaljerte notater fra møter og håndterer prosjektet her og nå. Jan William var den som startet på forprosjektet i juleferien som var, og det blir naturlig å velge han som personen som starter med å ha oversikt over hele prosjektet. Kasper får rollen som referent. Vi vil bytte på disse arbeidsoppgavene omtrent i "Verktøy til innhenting av systemdatafasen, se Gantt-skjema i figur 3 for tidspunkter. Dette vil gi begge erfaringer fra de forskjellige rollene.

Begge gruppemedlemmene er ansvarlig for å føre timelogg for arbeid som er blitt gjennomført, skriving av bacheloroppgaven, og presentere resultatet i juni. Vi vil bytte på oppgaven å skrive oppdateringer på prosjektets hjemmeside.

Vi vil prøve å komme til en enighet ved konflikter. Hvis dette ikke går, så vil vi kontakte vår veileder. Detaljert gruppekontrakt er vedlagt i vedlegg B.

4.1 Kontaktinformasjon til gruppemedlemmer

Jan William Johnsen
jan.johnsen@hig.no
+47 930 71 707

Kasper Kristoffersen
kasper.kristoffersen@hig.no
+47 913 48 424

4.2 Kontaktinformasjon til veileder

Lasse Øverlier
lasse.overlier@hig.no
+47 402 22 022

4.3 Beslutningspunkter

For å holde oversikt over progresjonen av prosjektet, vil vi holde timelogger over arbeidet som er blitt gjennomført. Vi vil holde et statusmøte når vi avslutter arbeidet hver torsdag. På disse møtene vil vi kunne omprioritere vårt arbeid til det neste møtet. Resultatet fra disse møtene vil bli publisert på prosjektets nettside. Dette vil gjøre at kontaktpersonene fra Watchcom blir oppdaterte på vår progresjon, og kan gi tilbakemeldinger. I tillegg vil det bli brukt av veileder.

5 Risikoanalyse

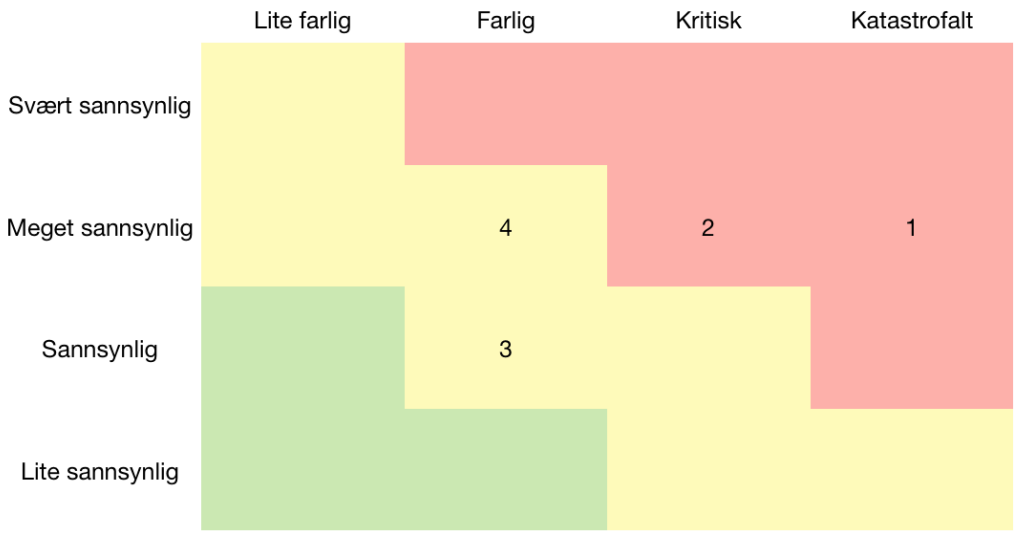
Figur 1 inneholder risikovurdering av hvert punkt uten tiltak. Figur 2 inneholder risikovurdering der tiltak er på plass.

1. Miste prosjektfiler: Det er en sjanse for å miste noen eller alle prosjektfilene. Vi kan være heldige å få gjenopprettet dem, eller være uheldige og miste de for alltid. Det er også en risiko for at vi overskriver feil filer eller informasjon når vi oppdaterer et felles arbeidsområde. **Tiltak:** *Vi vil benytte oss av et versjonskontroll-repository på serverene til høgskolen. Etter hver dag vi har arbeidet med prosjektfilene (sammen eller alene), vil vi slå sammen endringene til repository og ta en backup av vår lokale arbeidsområde på enten en ekstern hard disk eller til skyen.*

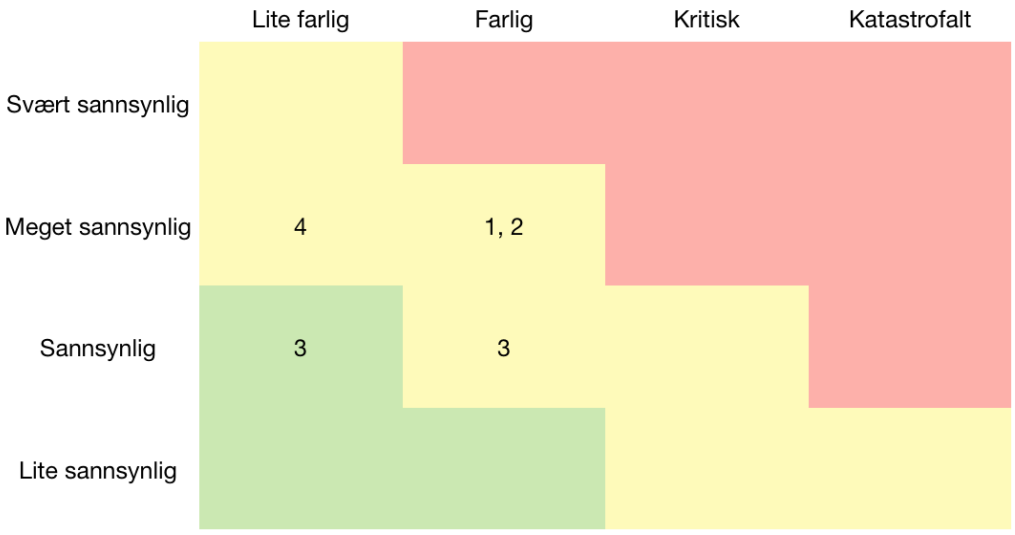
2. Sykdom over lenger tid: Folk er veldig mottakelig for forkjølelse om våren. En eller flere gruppemedlemmer kan bli syke. **Tiltak:** *Vi vil gjøre vårt beste for å fullføre arbeidsoppgavene. Vi bør være veldig syke før vi kan utsette dem. Hvis det blir bestemt at vi utsetter arbeidet, så bør vi i det minste sørge for at det er nok tid for å fullføre dem eller at vi kan jobbe ekstra for å fullføre dem senere.*

3. Uenigheter: Uenigheter vil alltid oppstå i prosjekter med flere personer. **Tiltak:** *Vi vil lage en grundig gruppekontrakt mens vi er venner, og benytte denne kontrakten når vi er uvenner. Vår siste mulighet for å løse krangler er å kontakte vår veileder.*

4. Inkludere unødvendige funksjonaliteter: Det finnes mange verktøy for teknisk informasjonsinnhenting. Dette øker sjansene for å velge et unødvendig verktøy. **Tiltak:** *Vi vil bruke de første ukene av prosjektet til å skaffe en oversikt over verktøy vi ønsker å inkludere til rammeverket. Hvis vi finner et nytt verktøy i perioden etter dette, så vil vi først analysere det for å finne ut om det gir oss ny informasjon. Vi ønsker å levere et lite produkt enn å levere et produkt som inneholder mye men som ikke fungerer.*



Figur 1: Risikoanalyse uten tiltak



Figur 2: Risikoanalyse med tiltak

5.1 Ressurskrav

Informasjonen vi skal innhente er offentlig. Vi trenger å sette opp en liste over bedrifter vi kan benytte som rekognoseringsmål, og hvilke resultater vi fikk om dem fra forskjellige verktøy. Vi har derfor ikke behov for mange ressurser for å utføre denne oppgaven:

- Våre egne bærbare PC-er (vi kan også benytte oss av skolens data-labber)
- En programvare for virtualisering (VMware eller VirtualBox) med både Kali Linux og servere vi kan forsøke å skanne
- Biblioteket ved HiG og i Gjøvik sentrum
- Tilgang til bibliotekets litteraturdatabaser

Vi har allerede alt vi trenger av ressurser. Vi kan også få gratis VMware- og Windows Server lisenser fra IT-tjenesten ved HiG.

6 Kvalitetskontroll

Vi vil benytte oss av et repository med versjonskontroll for både implementasjonsfilene og oppgavefilene våre. Dette vil føre til at vi har muligheten til å rette eventuelle feil på en enkelt måte. Vi må være kritiske til kilder og data vi vil benytte oss av, og vi burde avgrense kildene våre til vitenskaplig litteratur. Når vi skriver oppgaven vil vi benytte oss av bachelor *gucthesis* malen gitt av HiG.

Rammeverket skal kunne automatisk skanne et mål. For at vi ikke skal komme i trøbbel for denne aktiviteten, vil vi lage et testmiljø med servere av diverse operativsystemer. Disse vil vi kunne skanne lokalt. For å sikre at vi har implementert verktøy riktig, så er det viktig at vi tester hverandres kode. Dermed unngår vi å se oss blind på vår egen kode, og vi kan gjøre endringer på uklare kodeområder.

For å forsikre oss om at informasjonen som er innhentet fra rammeverket er riktig, så skal vi benytte oss av en liste over bedrifter som er mål. I den listen kan vi inkludere informasjon som er tidligere hentet fra verkøyene, og sammenligne med det som er samlet inn fra rammeverket. Da kan vi sjekke om rammeverket henter inn den samme informasjonen.

7 Plan for gjennomføring

Dette kapittelet inneholder beskrivelser av de forskjellige fasene av prosjektet som vist i Gantt-skjemaet, figur 3. Nummererte faser er hovedfaser for prosjektet.

Fase 0 - Forprosjekt
Forprosjektrapport
Denne fasen må inneholde: Mål, tema med avgrensinger, ansvarsforhold, deltakere, ressursbehov, fremdriftsplan, og prosjektavtale. Endelig kortnavn til prosjektet ble *reccoon*. Beregnet innlevering er 09.01.2014.

Opprette nettside
Vi setter opp en enkel nettside i Wordpress, denne vil bli brukt for å legge ut oppdateringer om alt vi har gjort opp til siste innlegg. Nettsiden vil bli benyttet hyppig for å oppdatere både veileder og Watchcom om progresjonen.

Fase 1 - Design
Utvikle modell
Her skal vi lære oss metoder for å innhente informasjon, samt få en oversikt over hvilken informasjon som vi ønsker å hente. Vi skal også designe tabeller til databasen. Vi vil også gjennomgå et par guider for å lære oss Python. Vi designer også rammeverket.

Velge verktøy
Vi setter opp en prioritert liste med verktøy i den rekkefølgen vi ønsker å implementere dem. Noen verktøy krever at vi har funnet informasjon som blir funnet av et annet verktøy. Verktøyene blir valgt fra resultatet fra forrige fase.

Fase 2 - Implementasjon
Grunnfunksjonaliteter
Her implementerers grunnfunksjonaliteter for rammeverket: XML-parser, eksport til Metasploit, og kanskje muligheter til eksportering i JSON hvis det er tid til det. Oppretting av databasen, inkludert funksjoner for kommunikasjon med den.

Verktøy til innhenting av systemdata
I denne fasen vil vi legge til verktøy for innhenting av system- og skanningdata.

Verktøy for innhenting av personalldata
I denne fasen vil vi legge til verktøy for innhenting av personalldata.

Sluttføre rammeverket og automatisere verktøyene
I denne fasen sluttfører vi rammeverket, dvs at resultatet blir eksportert og verktøyene som er blitt lagt til blir automatisert.

Fase 3 - Rapportskriving
Første utkast
Vi begynner å skrive rapporten med en gang vi har fått oversikt over verktøy. Se vedlegg C for disposisjon av den endelige rapporten.

Møte med veileder for første utkast
Her vil vi få tilbakemeldinger på hvordan vi ligger ann med rapporten, og hvilke endringer vi bør gjøre.

Ferdigstille rapport
Her vil vi gjøre de endringene som blir anbefalt av veileder, samt rettskrive rapporten.

Levere ferdig prosjekt
Rapporten og alt annet spesifisert i Fronter-rommet vil bli levert i Fronter.

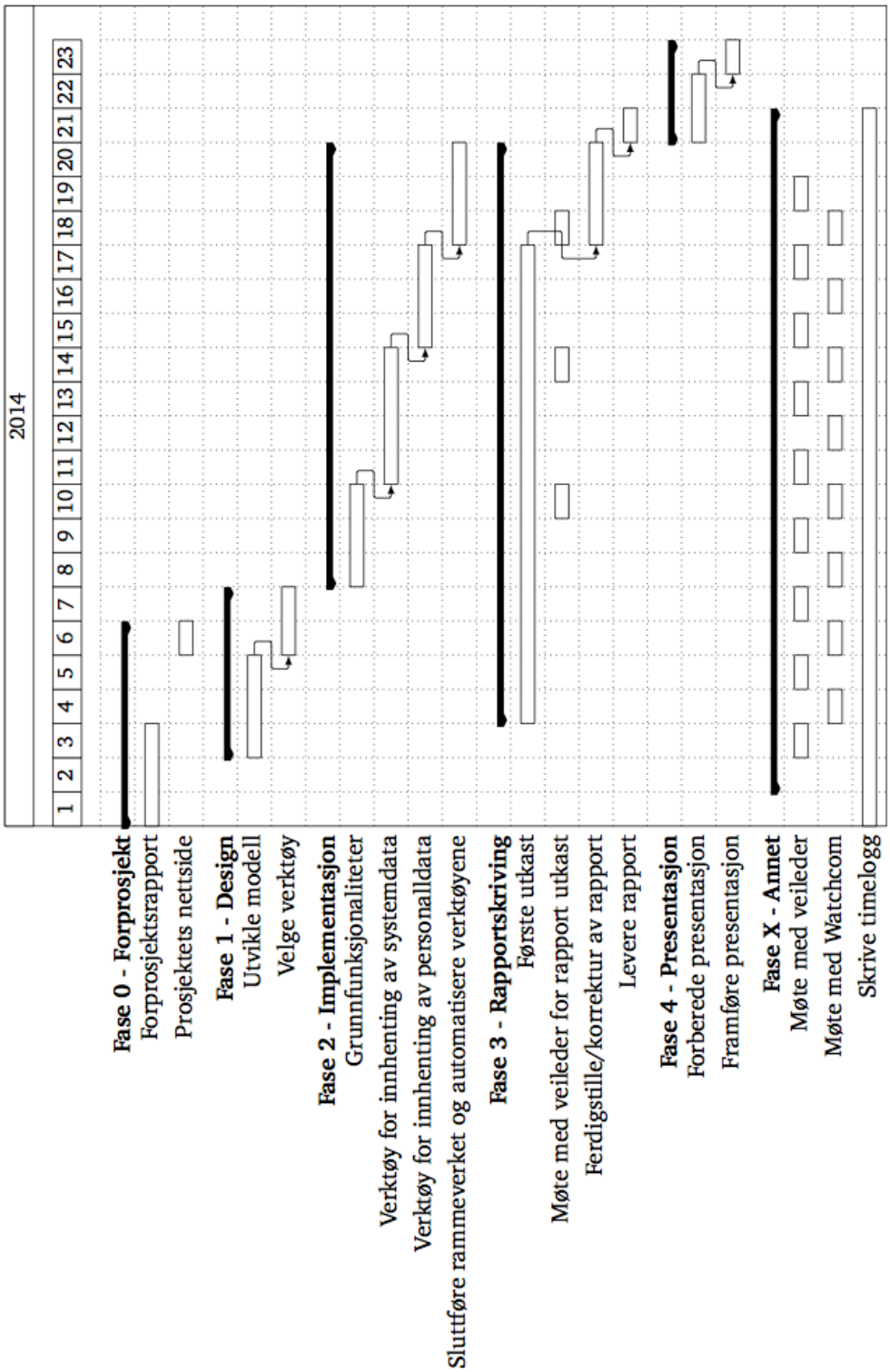
Fase 4 - Presentasjon
Forberede presentasjon
Vi må lage en plakat som skal bli hengt opp for å informere om prosjektet vårt. Vi skal også lage PowerPoint presentasjon med manus, dette vil bli brukt for å gjennomføre øvingspresentasjoner.

Presentere oppgave
Vi presenterer oppgaven enten 3., 4., eller 5. juni. Watchcom kontaktene vil få en invitasjon om å delta på denne presentasjonen.

Fase X - Annet
Møte med veileder
Avtalte møter med veileder er på mandager, klokkeslett blir avtalt senere. Vi vil også benytte e-post og Skype for kommunikasjon med veilederen.

Møte med Watchcom
Watchcom har et ønske om å ringe oss opp annenhver uke for å komme med tilbakemeldinger. Møtet avhenger om de finner det nødvendig etter å ha lest oppdateringene fra nettsiden. Møtetidspunkter blir avtalt senere.

Skrive timelogg
Vi skal loggføre tid som er brukt på prosjektarbeid. Loggen skal inneholde: dato, start klokkeslett, slutt klokkeslett, og arbeid utført.



Figur 3: Gantt-skjema

Bibliografi

[1] The Penetration Testing Execution Standard, *Intelligence Gathering*, http://www.pentest-standard.org/index.php/Intelligence_Gathering, 9. januar 2014.

[2] Patrick Engebretson, 2011, *The Basics of Hacking and Penetration Testing*
1st edition, Waltham, USA

[3] HolisticInfoSec *Toolsmith: Recon-ng* http://holisticinfosec.blogspot.no/2013/05/toolsmith-recon-ng.html, 3. mai 2013, 9. januar 2014

8 Disposisjon av endelig rapport

Dette vedlegget inneholder disposisjonen av den endelige rapporten, inkludert hva vi tenker at de enkelte underpunktene skal inneholde. Dette vil være med, men er ikke beskrevet: Sammendrag, forord, innholdsfortegnelse, ord definisjoner, referanser, vedlegg.

Innledning
Problemområde
Hvilke utfordringer det er om informasjonsinnhenting i dag.

Oppgavedefinisjon
Veldig kort om krav til funksjonaliteter til rammeverket.

Avgrensing
Vår avgrensing av oppgaven, hvem rammeverket er beregnet for, og andre avgrensinger vi har.

Formål
Kort beskrivelse av fremtidig situasjon som oppnås ved å gjennomføre prosjektet.

Bakgrunn
Vår egen bakgrunn og kompetanse. I tillegg til hva som må læres.

Målgruppe
Først og fremst Watchcom, men også andre som interesserer seg for IT-sikkerhet og penetrasjonstesting. Dette rammeverket kan også brukes til utdanningssammenheng.

Valg av utviklingsmodell og organisering av arbeid
Hvilken utviklingsmodell vi benytter, og hvorfor denne ble valgt. Hvilke spørsmål vi tok i betrakting for å velge modell. Beskrivelse av de forskjellige fasene og hva vi skal gjøre i dem. Og arbeidsfordelingen i gruppen.

Recon-ng
Beskrive hvordan Recon-ng er laget, brukes, hvem det er laget av, hovedfunksjoner.

Verktøy
Hvilke verktøy som vi har benyttet oss av: versjonskontroll og SVN.

Kravspesifikasjon
Beskrivelse av hvilke verktøy (prioritert liste med input og output) i den rekkefølgen vi ønsker å implementere dem.

Implementering
Beskrivelse av valgt lisens, utviklingsmiljø, det vi har brukt (operativsystem, python, SQL, osv), valg av XML parser, filstruktur, ressursbruk, osv.

Testing og kvalitetssikring
Beskrivelse av hvilken testmetode vi benytter for å sikre at rammeverket fungerer, og hvordan vi sjekker dette opp til kravspesifikasjonen, sikre kildekode lesbarhet og konsis-

tens, teste at funksjonalitetene er riktige, kompabilitet med andre maskiner.

Resultat/avsluttning
Diskusjon
Konklusjon
Fremtidig arbeid

E Møtereferater

MØTEREFERAT:

Møtereferater for bacheloroppgave

FORFATTERE:
Jan William Johnsen
Kasper Kristoffersen

DATO:
2014

0.1 19.12.2013

- **Møte med oppdragsgiver:**
  Dette er en oppgave som vi får gjøre stort sett det vi selv ønsker. De hadde ingen særlige ønsker om verktøy som skal inkluderes eller hvordan det se ut. *Kommentar: Vi går ut i fra listen i oppgaveforslaget i starten, og inkluderer eller fjerner verktøy etter behov.* De nevnte at det å importere den informasjonen som blir funnet til Metasploit og at verktøyet er GUI basert som pluss. Programmeringsspråk velger vi selv, de nevnte Python, men vi får bruke det vi føler oss komfortable med. Effekt- og resultatmål samt rammer var OK. Eneste kommentaren var om *zone transfer* da denne kan bryte norsk lov. Watchcom har for sakens skyld lov til å samle inn alt de ønsker når de inngår kontrakt. *Kommentar: Dette vil redusere hvordan konfigurasjonen av programmet blir.*

  Programmet bør bli utviklet til Kali, den har allerede en mengde verktøy installert og i tillegg til standard plasseringer av dem. Det bør også kunne håndtere IP-range, URL, og IP som input. De foreslo at vi skulle benytte oss av inkrementell utviklingsmodell, og få et solid modulbasert rammeverk for så å legge verktøy til dette. *Kommentar: Vi er enige i denne vurderingen, og tror at inkrementell er det beste å bruke. Det gjør oss i stand til å lage et bra produkt som fungerer, i stedet for å lage noe som er stort men som ikke fungerer.*

  De anbefalte at vi lager en liste over verktøy vi vil ha, og se hvilken input og output de gir. Det gjør at vi for oversikt over hvordan vi hopper fra det ene til det andre med informasjon. Vi må også vurdere hvor mye programmet kan gjøre parallelt og være effektivt. F.eks. de ønsker ikke at programmet skal få dem utestengte fra nettverktrafikk fordi den genererer for mye trafikk. De poengterte at vi lager noe nytt, så vi bør begrense mengden verktøy og at verdien av programmet ligger i den informasjonen vi samler inn og resultatet som blir generert.

  Under demonstrasjonen:
  De bruker Maltego for å hente informasjon om et selskap (i tillegg til en rekke andre programmer). De verifiserer også med whois, at det er et mål de får lov til å fange informasjon om. Deretter går de frem og tilbake i det som er blitt funnet for å finne mer informasjon og mer detaljert informasjon. De fjerner også redundant informasjon og uvesentlig informasjon. De verktøyene som blir mest brukt er Metagoofil, dnsenum, theharvester, og recon-ng.

  De anbefalte at vi ikke benyttet oss av Maltego. *Kommentar: Dette tror vi er lurt, fordi vi ikke har noen lisens til Maltego.*

  Vi har avtale om:

  - Sende tidsfrister og kontaktinformasjon til oss og veileder til Watchcom.
  - De vil ta kontakt ca annenhver uke for oppdateringer.
  - Avtale møtetidspunkter.

Kommentar: Hvis vi får lagt ut mye detaljrik informasjon på den nettsiden som opprettes til dette prosjektet, så kan Watchcom bli oppdatert via den og gi oss tips videre med oppgaven før de ringer oss.

0.2 08.01.2014

- **Gruppemøte:**
- Vi har kommet godt i gang med det administrative rundt oppgaven. SVN og diverse dokumenter er på plass.
- Vi har arbeidet med forbedring av gruppekontrakt (satt antall arbeidstimer i uka fra 24 til 30 timer), dette betyr at vi også endre møtetidspunktene fra 09-15 til 09-17.
- Funnet opp kortnavn (Reccoon), og outsourcet logo-design.
- Lagt inn avsnitt om ressursbruk vs. utestengelse fra nettverk under Resultatmål i forprosjekt.
- Lagt til seksjonen Oppgavebeskrivelse i forprosjekt.
- Begynt å utvikle en disposisjon til bacheloroppgave-rapporten.

0.3 09.01.2014

- **Gruppemøte:**
- Vi har lagt til og endret masse punkter i forprosjektet, og er klare for korrekturlesing og ferdigstilling.
- Tildeling av moduler i Recon-ng. Gå igjennom listen som ligger i "Annetmappen i SVN. List opp navn på modul, hvilke mulige parametere, output, kommentar. Hovedkolloner for oppsett av informasjonark om de forskjellige Recon-ng-modulene er: IP, IP-range og URL + andre inputs som er mulig (for eksempel antall, brukernavn etc). Sett opp mest mulig i kolonner og kryss av for aktuell funksjon til modulene.
- Tillegg: Vær opmerksomme på informasjon som er viktigere enn annen for oppgavens helhet. Hvor aktiv må vi være (og mot hvem) for å få tak i dette, samt hvordan dette kan hentes.

0.4 13.01.2014

- **Møte med veileder:**
- Første prioritet: Vi må tenke over hvilken informasjon vi vil få frem i rapporten programmet vårt genererer, og hvilke verktøy vi må benytte oss av for å hente inn denne. Sett opp en liste på grunnlag av dette. Tenk på hva Watchcom ønsker seg.
- Vi bør bruke en database, og bruke åpne moduler i Python.
- Bruke recon-ng som inspirasjon. Se på hvordan de henter sin informasjon.
- Skrive disposisjon til rapport: Skriv fortløpende inn punkter som dukker opp i disposisjonen. Lasse vil at vi begynner med å arbeide på rapporten tidlig.
- Hvis vi bestemmer oss for å bygge på recon-ng må vi dokumentere hvilke funksjoner

vi har lagt til.

- Vi burde få en oversikt over hvordan Metasploit lagrer sin informasjons for å få en optimal eksportering.
- Rammeverket burde være open-source slik at det kan vidreutvikles av andre, og at det ikke blir puttet i en skuff".
- Vi må ta et valg: Basere oss på recon-ng eller lage noe nytt?
- Neste møte med Lasse: Mandag 27.01.2014, send inn disposisjon til rapport med oppdatert valg av eget rammeverk eller påbygg av recon-ng innen fredag 24.01.2014.

0.5 16.01.2014

- **Gruppemøte:**
- Vi har bestemt oss for å utvikle et eget rammeverket fra bunnen av.
- Vi har sett på hvordan vi kan sette opp hjemmeside i Ghost, og satt opp et forslag til hjemmeside.
- Vi har begynt å sette opp en liste med elementer vi vil at den genererte rapporten skal inneholde. Senere må vi sette opp hvilke verktøy vi skal bruke for å få til dette.
- Vi har begynt å se på outputen til noen verktøy, og hvordan vi skal eksportere/behandle disse.
- Vi har sett på automatisering av metasploit.
- Vi har fortsatt 2 uker igjen av design-perioden, men har også en del arbeid igjen.
- TO-DO: 1. Fortsette å lete etter elementer vi vil ha med i rapporten. 2. Vi må begynne å tenke på hvordan programmet skal se ut (tabeller, rammeverk etc).

0.6 22.01.2014

- **Gruppemøte:**
- VIKTIG: Utvikle disposisjon av endelig rapport m/ liste over data vi skal finne + begrunnelse for at vi velger å utvikle rammeverket fra bunnen av. Sendes til Lasse innen fredag 24.01.
- I TILLEGG: Gjøre ferdig kravspesifikasjon (lag diagrammer etc. etter at kravspesifikasjonen er i boks). Legge inn nytt design/logo på nettside.

0.7 23.01.2014

- **Gruppemøte:**
- Vi har utviklet en disposisjon for den endelige rapporten. Denne er også sendt til veileder.
- TO-DO: Utvikle tabeller til databasen.
- Vi har utviklet en skisse av rammeverket for Reccoon.

0.8 27.01.2014

- **Møte med veileder:**
- Lasse sa at listen med elementer vi vil finne så bra ut, men at det vil dukke opp flere ting vi vil ha med. Han nevnte at det kunne være lurt å ha informasjon om geolocation for personer, og at det er flere mål en kun dns. Han kunne også tenke seg at diverse informasjon, som for eksempel om det er funnet trådløse nett, blir lagret.
- Vi burde tenke på en enkel struktur for rammeverket, som gjør at vi enkelt kan legge til nye moduler.
- Det er lurt å se på hvordan recon-ng fungerer. Hvis vi blir inspirert av koden til recon-ng må vi dokumentere dette og gjøre rede for hvor vi har funnet koden og at den ikke er vår.
- Tenk på interface mellom logikken og modulene. Hvilke datastrukturer blir sendt etc.
- Vi kan begynne å få på plass bakgrunn og introduksjon i rapporten.
- Få igang noe som fungerer fort. Da er det letter å se hva som kan utbedres (bare fokuser på å få et par moduler til å fungere i starten).
- Vi burde komme igang med programmering og rapportskriving så tidlig som mulig.
- Les tidligere bacheloroppgaver for inspisrasjon til oppsett.
- Send en statusoppdatering til Lasse før helga, evt. spørsmål vi trenger svar på. (Blogg?).
- Vi fortsetter med telefonmøter annenhver mandag.

0.9 06.02.2014

- **Telefonmøte med oppdragsgiver:**
- Vi ga Watchcom en kort statusrapport og fortalte at vi er ca 90 prosent ferdig med introduksjon og kravspesifikasjon til hovedrapporten. Vi skal bare rettskrive og levere den til veileder før helgen, så vil vi legge den ut på nettsiden når vi har fått tilbakemeldinger på den.
- Vi fortale de også at vi har en "alpha-versjon"av selve rammeverket oppe og kjører, med MySQL-interaksjon og parsing av output fra Nmap.
- Vi sa at vi ville ha listen med verktøy klar i løpet av neste uke. Da vil vi også gjerne få tilbakemeldinger på denne listen.
- Vi spurte etter tilbakemeldinger for effektmålene som er satt opp, silk at vi kan utdype disse mer i hovedrapporten.
- Vi spurte også etter tilbakemeldinger på listen med data vi ønsker å samle inn. Vi skulle motta dette på epost.
- Ellers hørte det ut som om Watchcom er fornøyd med prosessen frem til nå, og vi skal bare 'gønne' på med rapportskriving, siden det er den vi blir dømt ut ifra.
- Watchcom ønsker at vi sender de en epost når vi har oppdatert hjemmesiden til prosjektet.
- Neste møte med oppdragsgiver er torsdag 20. februar.

- **Prosjektmøte/ToDo-liste:**
- Begge leser igjennom rapporten til fredag 07.02.2014, så sender vi den inn til Lasse.
- Kasper: Implementere database som satt opp i det konseptuelle klassediagrammet. Implementer delvis. Start med elementære deler.
- Jan: Leker seg med rammeverket slik det er nå. F.eks. begynne å lage XML-dokumenter og PDF-dokumenter.

0.10 10.02.2014

- **Telefonmøte med veileder:**
- Har sett på det vi sendte inn før helgen, og mener at det er en grei start.
- Konseptuelt klassediagram 'Hosts' sin 'IP addresse' kan ikke være unik. Samme gjelder 'Persons', med 'E-mail address'. Disse bør knyttes sammen på en annen måte. Det samme gjelder 'OS' til 'Vulnerabilities'.
- 'OS' og 'Ports' kan ha flere sårbarheter.
- Forklar det konseptuelle klassediagrammet bedre, for eksempel hva betyr bold skrift etc.
- Lasse vil at vi skal gå igjennom relasjoner i databasen. (Target List kan merges med Host Ports for eksempel siden begge er 1..1).
- Se på verktøy, input og output opp mot databasen. Se for oss hvert enkelt verktøy opp mot databasen.
- Ha en beskrivelse av hvordan hvert program passer inn i databasen.
- Hvordan analysere verktøy? Ta utgangspunkt i ønskede verltøy, se på hvordan verktøy henter inn informasjon. Kanskje noen henter inn noe noen andre allerede har hentet inn.
- Fokuser på informasjonsinnhenting, søk etter DNS, E-post informasjon etc. Port- og sårbarhetsskanning kan være ekstra hvis vi har god tid.
- Lasse vil ha tilsendt en liste med verktøy vi vil prioritere.
- Lag et flytskjema med hvilke verktøy som burde kjøre etter hverandre.
- Akademisk bidrag; lage en systematikk for informasjonsinnhentingen.
- List opp verktøy før databasedesign.

0.11 13.02.2014

- **Gruppemøte:**
- Etter forrige møte med veileder har vi fått satt opp en liste med verktøy vi ønsker å benytte oss av i rammeverket.
- Vi har begynt å se på hvilken input/output disse verktøyene har, og satt det opp mot punktene vi vil dekke i det konseptuelle klassediagrammet.
- Vi vil gå vekk ifra at rammeverket skal genererer en .pdg-fil. Vi ønsker heller at rap-

porten skal genereres i et HTML-format. Jan har satt opp en side for dette.

- **To-Do:**
- Jan fortsetter med siden for rapportering som nevnt i punktet over.
- Kasper fortsetter med å sette opp databasen etter de kravene som verktøylisten setter.
- Vi må se på normalisering av databasen, siden vi vil lagre data lengre en tidligere antatt.

0.12 20.03.2014

- **Møte med oppdragsgiver:**
- Vi har sendt testversjon nummer 2 av rammeverket til Watchcom, men de hadde ikke fått kjørt det. Vi skal zippe rammeverket og sende det på nytt med flere oppdateringer.
- Watchcom lurer på om det er noe de kan bidra med, men sier at det virker som om vi har god kontroll på oppgaven så langt. Vi må ikke nøle med å kontakte de om det er noe vi lurer på.
- Watchcom vil prøve å benytte rammeverket i en reell case, anonymisere data, og gi oss tilbakemeldinger. De vil også dobbeltsjekke resultatene opp imot de enkeltstående verktøyene, og se på tidsbruken i forhold til å gjøre hele prosessen manuelt.

0.13 10.04.2014

- **Møte med oppdragsgiver:**
- Oppdragsgiver har fått testet rammeverket et par ganger i produksjon, og de er veldig fornøyd. Ordene 'Stor suksess' og 'imponert' ble brukt. Oppdragsgiver er veldig fornøyd med at rammeverket returnerer korrekt informasjon, og er tidsbesparende.
- Oppdragsgiver er veldig fornøyd med endringene vi har gjort på rammeverket siden sist (mulighet for endring av parametere og redigering av hosts etc).
- Oppdragsgiver nevner at rammeverket er utmerket for å kartlegge infrastruktur, og er midt i blinken for de testene som ikke skal gå i dybden.
- Oppdragsgiver nevnte at tidsbruken av en scan ble redusert fra 1.5 time til 10 minutter.
- Oppdragsgiver ønsker seg en log-fil med operasjonene som rammeverket foretar seg.
- Prosjektgruppen stilte et spørsmål angående det juridiske aspektet rundt portscanning. Oppdragsgiver sa at det ikke er ulovlig i Norge, men heller ikke etisk.

0.14 29.04.2014

- **Epostutveksling med veileder:**
- Lasse har lest igjennom rapporten vår så langt. Tilbakemeldingene går på at vi ikke trenger å inkludere informasjon i rapporten som vi kan referere til andre steder. Infromasjon som burde være lett tilgjengelig kan legges til som vedlegg.
- Lasse mener vi har et bra utgangspunkt med rapporten, men han savner flere refe-

ranser.

- Lasse sier at det er bra at vi bruker mange figurer, og sier at skjermdumper også burde inkluderes.

0.15 14.05.2014

- **Telefonmøte med veileder:**
- Lasse mener at vi burde ha en kort introduksjon til hvert kappitel i starten av hvert kappitel.
- Lasse mener vi burde få en annen gruppe til å lese gjennom oppgaven vår, for å belyse punkter vi ”har sett oss blinde på”.
- Produktet vårt er det viktige, det er viktig av vi får dette frem i rapporten via at alle delene av rapporten bygger opp om rammeverket.
- Lasse synes vi har en god oppbygning av rapporten til nå.
- Lasse sa hvis vi har programmert noe bra, som vi gjerne vil skryte av, så burde vi legge det som vedlegg i rapporten og referere til dette.
- Det er viktig av rapporten får frem hva vi har gjort som er bra og unikt.

Spørreundersøkelse vedrørende testing av Reccoon

F Spørreundersøkelse

1. **Har du tidligere erfaring med penetration testing / external network footprinting?**
   Markér bare én oval.

   - ◯ Ja
   - ◯ Nei

2. **Hvordan var det å starte Reccoon?**
   Markér bare én oval.

	1	2	3	4	5	6	
Vanskelig	◯	◯	◯	◯	◯	◯	Enkelt

3. **Hvordan var det å starte en ny skanning?**
   Markér bare én oval.

	1	2	3	4	5	6	
Vanskelig	◯	◯	◯	◯	◯	◯	Enkelt

4. **Hvordan var det å gjøre flere operasjoner på en startet skanning, som f.eks. Nmap- og geolokasjonsskanning?**
   Markér bare én oval.

	1	2	3	4	5	6	
Vanskelig	◯	◯	◯	◯	◯	◯	Enkelt

5. **Hvor oversiktlig er resultatene som Reccoon produserer?**
   Markér bare én oval.

	1	2	3	4	5	6	
Lite ovesiktlig	◯	◯	◯	◯	◯	◯	Oversiktlig

6. **Hvordan var det å eksporter resultatene til en HTML-rapport?**
   Markér bare én oval.

	1	2	3	4	5	6	
Vanskelig	◯	◯	◯	◯	◯	◯	Enkelt

7. **Hvor oversiktlig er resultatene i HTML-rapporten?**
Markér bare én oval.

	1	2	3	4	5	6	
Lite oversiktlig	◯	◯	◯	◯	◯	◯	Oversiktlig

8. **Hva syntes du om den helhetlige bruken av Reccoon?**
Markér bare én oval.

	1	2	3	4	5	6	
Vanskelig	◯	◯	◯	◯	◯	◯	Enkelt

9. **Hvordan kan Reccoon eventuelt forbedre jobben med network footprinting? Finnes det eventuelt noen negative konsekvenser ved denne bruken? Begrunn svaret ditt.**
Kan du spare tid med innhenting og bearbeiding av data, benytte data i en annen delen av penetrasjonstest, m.m.

.....

.....

.....

.....

.....

10. **Kan bruken av Reccoon gå ut over kvaliteten av network footprinting?**
Markér bare én oval.

- ◯ Ja
- ◯ Nei

11. **Er det noe som kunne vært gjort annerledes i Reccoon? Gjerne begrunn svaret ditt.**

.....

.....

.....

.....

.....

Kasper Kristoffersen
Rediger dette skjemaet

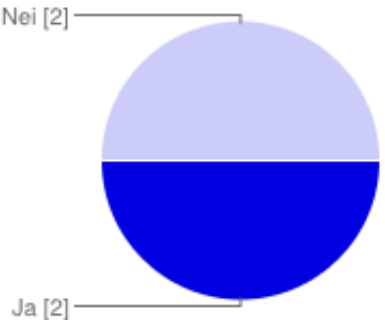
4 svar

G Spørreundersøkelse svar

Publiser analytics

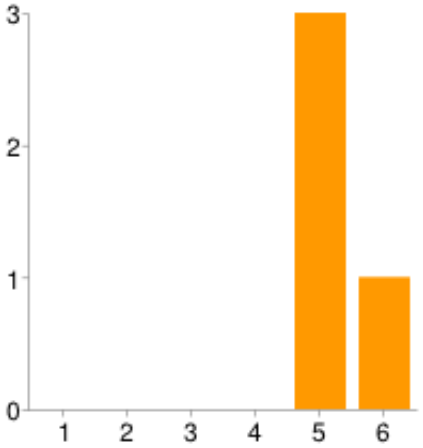
Sammendrag

Har du tidligere erfaring med penetration testing / external network footprinting?



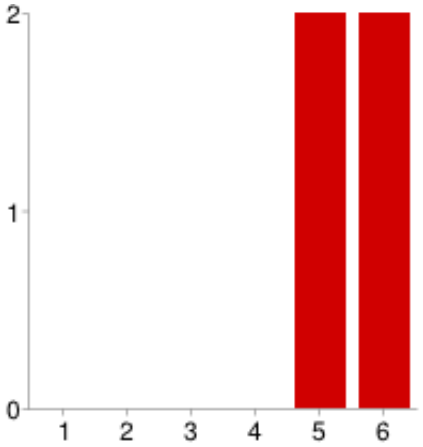
Ja	**2**	50 %
Nei	**2**	50 %

Hvordan var det å starte Reccoon?



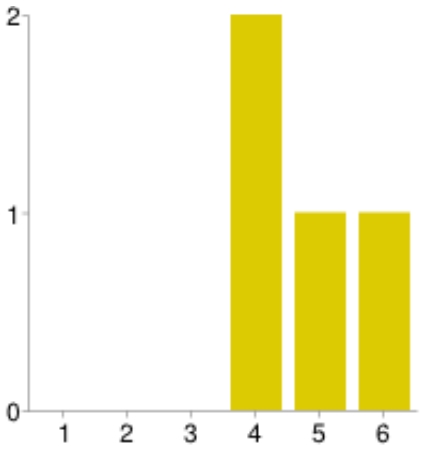
1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**0**	0 %
5	**3**	75 %
6	**1**	25 %

Hvordan var det å starte en ny skanning?



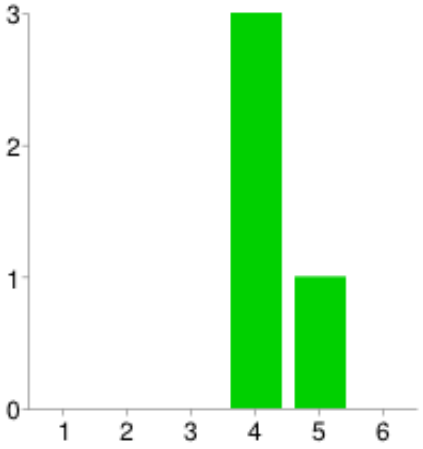
1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**0**	0 %
5	**2**	50 %
6	**2**	50 %

Hvordan var det å gjøre flere operasjoner på en startet skanning, som f.eks. Nmap- og geolokasjonsskanning?



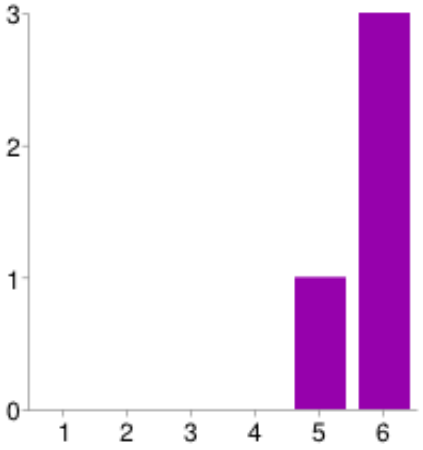
1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**2**	50 %
5	**1**	25 %
6	**1**	25 %

Hvor oversiktlig er resultatene som Reccoon produserer?



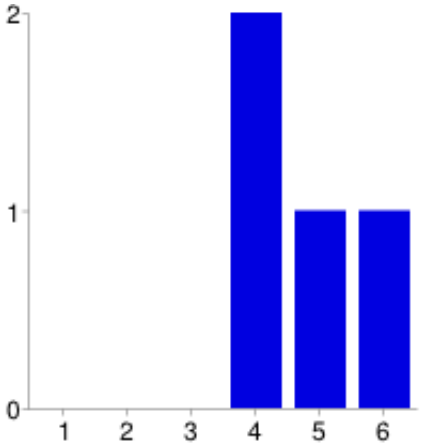
1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**3**	75 %
5	**1**	25 %
6	**0**	0 %

Hvordan var det å eksporter resultatene til en HTML-rapport?



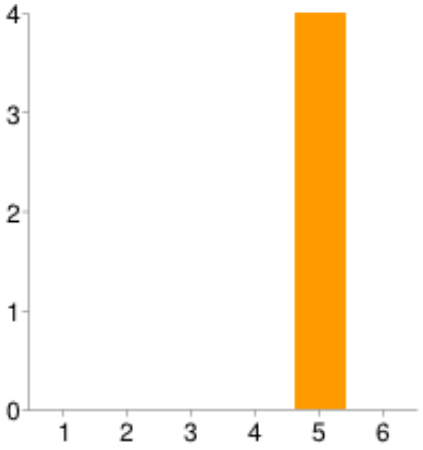
1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**0**	0 %
5	**1**	25 %
6	**3**	75 %

Hvor oversiktlig er resultatene i HTML-rapporten?



1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**2**	50 %
5	**1**	25 %
6	**1**	25 %

Hva syntes du om den helhetlige bruken av Reccoon?

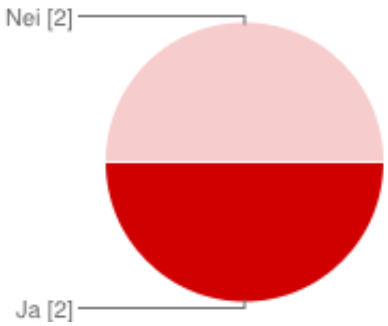


1	**0**	0 %
2	**0**	0 %
3	**0**	0 %
4	**0**	0 %
5	**4**	100 %
6	**0**	0 %

Hvordan kan Reccoon eventuelt forbedre jobben med network footprinting? Finnes det eventuelt noen negative konsekvenser ved denne bruken? Begrunn svaret ditt.

Dette kan være en tidsbesparende løsning sammenlignet med å bruke de bakenforliggende programmene igjennom terminalen. Her samles flere verktøy i et enkelt og pent brukergrensesnitt. Den negative konsekvensen er at et slikt verktøy gjør det lettere for script kiddies og andre med feil intensjoner å kartlegge domener og nettverk. Fin pils

Kan bruken av Reccoon gå ut over kvaliteten av network footprinting?



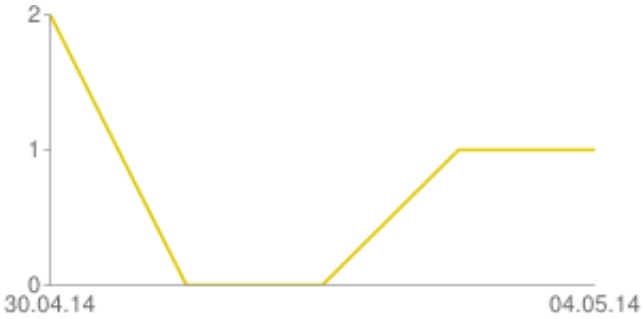
Ja	**2**	50 %
Nei	**2**	50 %

Er det noe som kunne vært gjort annerledes i Reccoon? Gjerne begrunn svaret ditt.

(gui ting) Delvis loading av sider scann knappen for langt ned når du hadde mange targets feste hovedmenyen til siden så den ikke forsvinner ved scrolling, så flytt den eller lim den til bunnen av skjermen så den har en konstant posisjon til skjermen uansett scrolling endre "start scann" knappen til " start additional scans automatisk status uppdate

Funskjonalitet for avbrytning av scanning. "Load bar" flere steder i programmet. Bruker tutorial bør være tilgjengelig, her bør funksjonalitet i programmet presenteres. Mer synlig meny. Slik at programmet blir mer intuitivt, og brukeren slipper å lete.

Antall svar per dag



H Elementer

System-elementer

- Report name
  - Customer IP range
  - Customer IP
  - Customer URL
- Domain Name System
  - Domain name
  - Domain holder name
  - Registrar handle
  - Legal-c handle
  - Tech-c handle
  - Name server handle*
  - Created (YYYY-MM-DD)
  - Last updated YYYY-MM-DD)
  - NORID handle
  - Type
  - Name
  - ID type
  - ID number
  - Registrar handle
  - Post address
  - Post code
  - Post area
  - Country
  - Phone number
  - Email address
- Target list (Prioritert rekkefølge)
  - Host*
    - IP address
      - Country

    - Country code
    - Region
    - City
    - Latitude
    - Longitude
    - ISP
- Port*
    - Port ID (Number)
    - Port protocol (tcp/udp)
    - State (Open/Closed/Filtered)
    - Service name
    - Version
    - Vulnerabilities*
        - Common Vulnerability Scoring System (CVSS)
        - OpenVAS Network Vulnerability Test (NVT)
        - CVE
            - CVSS score
            - ID
            - Name
            - Family
            - Vulnerability type
            - Version
            - Created
            - Last modified
            - Confidentialy impact
            - Integrity impact
            - Availability impact
            - Access impact
            - Authentication
            - Gained access
            - Summary
            - CVSS base
            - Risk factor
            - CVSS base vector

    - Operating system
        - Name
        - Version
        - Common Platform Enumeration (CPE)
        - Vulnerabilities*
            -

Personell-elementer

- Person list (Prioritert rekkefølge)
- Person*
    - First name
    - Last name
    - Email address
    - Position
    - Username
    - Password
    - Password hash
    - Hash type
    - Telephone number
    - Leaks (haveibeenpwned)

I Statusrapporter

Statusrapportene er hentet fra prosjektets hjemmeside, og bilder er utelatt i denne kopien. Disse statusrapportene er i første omgang ment til å gi oppdragsgiver et innblikk i hvor vi ligger i prosjektet, det er derfor mange innlegg under programmeringsperrioden, mens det blir mindre utover i rapportskrivingen.

Første oppdatering! (20.01.2014)

Da er forprosjektet levert og hjemmesiden er oppe og går! Vi vil forsøke å oppdatere denne siden ofte utover i bacheloroppgaveperioden.

Til nå har vi arbeidet mye med det organisatoriske rundt oppgaven, som for eksempel å opprette et felles versjonkontrollsystem for alle prosjektfiler, ordne prosjektavtaler og gruppereglement, og annen generell planlegging for gjennomføring av oppgaven. Dette har resultert i et forprosjekt som vi er veldig fornøyd med.

Siden vi kom i gang ganske tidlig med forprosjektet har vi rukket å gjort noen vurderinger for hvordan vi vil ha rammeverket. Vi vurderte først å basere rammeverket vårt på 'recon-ng', men vi har valgt å heller utvikle noe eget fra bunnen av for å ha full kontroll over hva slags data som lagres, og hvordan den lagres. Vi har utarbeidet en liste over data vi vil at den automatisk genererte rapporten skal inneholde, og vi har begynt å tenke på hvilke verktøy vi vil inkludere i rammeverket for å fremstille alle disse dataene på en god måte.

Nå er vi er klare for å virkelig begynne på oppgaven!

Captain's log, stardate uke 4 (23.01.2014)

Siden siste oppdatering har vi jobbet litt med videreutvikling av hjemmesiden, og vi tror at den snart holder til sitt bruk (selv om det antageligvis blir gjort noen små endringer i nærmeste fremtid). Noen andre ting vi har arbeidet med er:

- Kravspesifikasjon: Vi har begynt å se på funksjonelle og operative krav for reccoon, med forskjellige 'use-case'-beskrivelser på forskjellige nivåer.
- Liste over data: Vi har satt opp en liste (nederst i denne posten) med data vi ønsker at reccoon skal finne på målnettverket og lagre i databasen sin. Vi planlegger å starte prosessen med å velge ut hvilke verktøy vi skal bruke i uke 6.
- Databasedesign: Vi har begynt å sette opp et konseptuelt klassediagram for rammeverkets database. Denne tar utgangspunkt i listen med data som er nevnt i punktet over
- Arkitekturskisse: Vi har satt opp en skisse med hovedelementene vi mener rammeverket er helt avhengig av for å fungere.

Vi skal forsøke å komme med ukentlige oppdateringer på denne bloggen. `Notat: Her er listen over innsamlede elementer.`

Captain's log, stardate uke (01.02.2014)

Vi hadde et møte med veileder på mandag. Vi fikk tilbakemelding om at listen vi hadde laget var bra, men at vi helt sikkert vil finne andre elementer som gjør listen lengre. Vi ble tipset om å begynne og programmere på rammeverket så tidlig som mulig. På den måten kan vi finne strukturen som modulene bruker for å kommunisere seg imellom. Vi kunne f.eks. se på Recon-ng for å få noen inspirasjoner, men det er veldig viktig at vi dokumenterer det som ikke er vårt.

I slutten av forrige uke så sende vi en ferdig disposisjon av sluttrapporten. Siden vi blir vurdert på denne, så vil vi prioritere den. Når det gjelder rammeverket så fokuserer vi på å lage noe som fungerer, i stedet for et stort program som ikke blir brukt. Gjør vi det lett for andre å legge til moduler til rammeverket så vil ikke det kreve like stor jobb som å lage rammeverket.

Det vi har gjort denne uken er:

- Fortsatt på kravspesifikasjonen.
- Programmere rammeverket: Vi har laget et lite utkast av de mer generelle delene av rammeverket. Som dens arkitektur, database og XML klasser. Den starter med å skanne Metasploitable og parse XML filen Nmap gir oss. Et eksempel kan du se på bildet under. Dataen blir lagt inn i en liten tabell i databasen. Dette er i startfasen og hovedsak til å prøve oss frem/teste.

`Notat: Her er det et bilde med parsing av resultater fra Nmap.`

Dette skal vi gjøre til neste gang:

- Vise frem element listen og en liste med verktøy til oppdragsgiver.
- Bli ferdig med første (innledning) og andre (kravspesifikasjon) kapittel i sluttrapporten. Den vil bli send til veileder før helgen, slik at vi får tilbakemeldinger på møtet til mandag uken etter.
- Eventuelt bli ferdig med konseptuelt klassediagram, og implementere den.

Captain's log, stardate uke 6 (07.02.2014

Denne uken har vi jobbet med:

- Kravspesifikasjon: Vi har kommet godt igang med arbeidet på hovedrapporten og har (så godt som) ferdigstilt introduksjonen og kravspesifikasjonen. Dette inkluderer utviklingen av det konseptuelle klassediagrammet og de forskjellige brukermønstrene.
- Møte med oppdragsgiver: Torsdag 06.02 hadde vi et telefonmøte med Watchcom. Vi ga de en kort statusrapport og fortalte hvordan vi føler vi ligger an. Vi spurte også etter hvilken motivasjon (ønskede effektmål) de har for produktet vårt, og etter eventuelle tilbakemeldinger på listen av data vi vil samle inn.

Dette skal vi jobbe med videre:

- Verktøyliste: På grunn av arbeid med hovedrapporten har vi ikke rukket å sette opp en liste med verktøy vi vil inkludere denne uka.
- Verktøyanalyse: Vi må analysere verktøyene vi velger for å prioritere input og output som passer både rammeverket vårt og listen med data som vi vil samle inn. Vi må finne ut hvordan vi skal utføre denne analysen.
- Database-implementasjon: Vi vil begynne med å implementere elementer til databasen slik vi har satt det opp i det konseptuelle klassediagrammet. Dette vil gjøre at vi kan bygge på alpha-versjonen av rammeverket slik at det kan lagre mer data.

Captain's log, stardate uke 7 (13.02.2014)

Vi har lagt ut forprosjekt og foreløpig bacheloroppgave, det bør tas forbehold at det vil komme endringer i denne rapporten.

Denne uken har vi jobbet med:

- Veileder informerte oss på møtet at vi burde liste opp alle verktøy vi ønsker oss, for så å designe databasen. Vi har deretter gått grundig gjennom hvert enkelt verktøy Kali Linux har under "Information gathering", og analysert dem for hva de gir oss av informasjon. Vi har testet forskjellige domener og IP adresser for å se om noen av dem gav et bedre resultat. Veileder beskrev at port og sårbarhetsskanning går utenfor problemområdet vårt, og vi kan eventuelt gjøre det hvis vi har tid til overs.
- Jan William har sett på forskjellige løsninger for å produsere PDF rapporter. Python hadde flere moduler som kunne generere PDF dokumenter. Fellesnevneren ved disse er at de ikke fungerer særlig bra. Noen krever at det er en mal med et åpent format (ODS – Open Document Spreadheet) som legger inn verdier på faste steder i malen; andre krever at vi må spesifisere X og Y koordinater for hvor teksten skal inn. Nødløsningen er å installere LaTeX for å generere PDF rapporter. Vi har heller valgt å benytte en rapportering i HTML, beskrever mer under "dette skal vi jobbe med videre".
- Kasper har satt hvert enkelt verktøy opp mot databasen, og han har gjort endringer i konseptuelle klassediagrammet for at det skal godta nye data fra verktøyene.

Dette skal vi jobbe med videre:

- Jan William vil jobbe videre på HTML rapporten. Vi valgte å gå videre på HTML fordi oppdragsgiver får mulighet til å redigere data og legge til nye. Det gjør at vi bør normalisere databasen litt mer, siden vi ikke vil slette innholdet i den. Oppdragsgiver vil kunne håndtere sine kjøringer av Reccoon.
- Kasper vil fortsette å gjøre endringer på det konseptuelle klassediagrammet. Han vil deretter sette opp en funksjonell database, slik at vi kan begynne å implementere verktøyet og rammeverket.

`Notat: Her er det et bilde av brukergrensesnittet til Reccoon.`

Dette er den foreløpige listen som vi har av verktøy. De er i prioritert rekkefølge.
System:

- nslookup [kjører på input]
- dmitry [kjører bla. whois (-i -w -n -s)]
- fierce [fierce -dns]
- dnsdict6 [lagrer IPv6-adresser fra DNS]
- nmap [kjøres på resultater fra fierce]
- dnsrecon [kjører på domenet]
- dnsmap [teste at adresser er korrekte]
- xprobe2 [teste at OS er korrekt]
- telnet [banner grabbing på tjenester]

Personell:

- metagoofil [starte tidlig, kan potensielt ta lang tid]
- theharvester [kjøres på 'all' med domenet som søkeord]
- 'haveibeenpwned' [ser etter sårbare e-post adresser og brukernavn]
- recon-ng/contects/ [finner info om personell]
- recon-ng/creds/ [finner info om personell]

Sårbarhetsskanning:

- nessus [søker etter sårbarheter]
- arachni [søker etter web-sårbarheter]

Captain's log, stardate uke 8 (20.02.2014)

Denne uken har vi jobbet med:

- Vi begynte med en vanlig Apache-webside for rammeverket vårt, men vi har funnet et Python-webrammeverk vi vil benytte oss av, nemlig Django. Django lar oss abstraktere databasen vår. Så vi har valgt å implementere databasen vår via Django.
- Siden vi har gått tilbake til den orginale ideen vår med HTML-rapportering har vi mulighet til å lage et grafisk grensesnitt til Reccoon, istedet for å kun ha kommandolinje.
- Vi har begynt å utvikle noen av modulene til rammeverket. Disse modulene benytter verktøyene Host/Nslookup og Dmitry (deep magic information gathering tool), samt en enkel name-resolver.
- Vi har hatt mye problemer fra mandag (17.02) til torsdag (20.02) med debugging av Django, men nå kan rammeverket legge til og fjerne scanninger (noe ala nessus), og kan starte automatisk scan med en modul.
- Planen vår fremover er å fortsette å programmere på rammeverket, og vi håper på å levere noe som kan testes innen 2 uker.

Captain's log, stardate uke 9 (01.03.2014)

Ut i fra teksten i dette innlegget, så kan man sitte igjen med inntrykk av at vi ikke har gjort så mye denne uken. Men det vi ikke nevner, er at modulene er blitt lagt til rammev-

erket og starter automatisk ved nye skanninger. I følge tidsskjemaet som vi har satt for oss selv, så er vi i midten av Fase 2 – Implementasjon av grunnfunksjonaliteter. Det betyr at vi egentlig er minst to til tre uker foran planen.

Denne uken har vi jobbet med:

- Vi har laget flere moduler som bruker verktøyene: fierce, dnsdict6, og nmap. Vi har også fortsatt på metagoofil og theharvester, samt hoppet over verktøy som dnsrecon, dnsmap, xprobe2, og telnet for denne uken.
- Det grafiske grensesnittet har også mulighet til å vise resultater og redigere disse. Vi har i tillegg begynt å skrive videre i rapporten. Vi skriver for det meste om de forskjellige verktøyene som er blitt analysert.

Dette er planen fremover:

- Forbedre rammeverket slik at det kan bli testet av oppdragsgiver i slutten av neste uke.
- Fortsette å programmere moduler og skrive i rapporten.

Captain's log, stardate uke 10 (06.03.2014)

Denne uken har vi arbeidet med å få igang en testversjon av rammeverket. Dette har inkludert å forsøke en installasjon på en blank VM for å se om alt fungerer som det skal. Under denne prosessen fant vi noe småpirk som må utbedres. Den største feilen var at geolokasjonmodulen vår ikke fungerte som den skulle. Vi måtte dermed kutte ut denne fra testversjonen av rammeverket. Bortsett fra geolokasjonmodulen ser alt ut til å fungere bra, og vi mener selv at vi har et veldig godt utgangspunkt for videre arbeid.

En nedlastingslenke til testversjonen av Reccoon er sendt inn til oppdragsgiver, slik at de kan se hvordan vi ligger an frem til nå.
`Her er det et bilde som viser informasjonen om en host i rammeverket.`

I bildet over kan du seg hvordan en host fra en skanning blir fremstilt i rammeverket (med geolokasjon). Tabellen med åpne porter og tjenester kan skimtes nederst på bildet.

Ellers denne uken har vi vært på lynkurs om rapportskriving i regi av Høgskolen i Gjøvik. I tiden fremover vil vi jobbe med å få på plass mer av grunnfunksjonaliteten i rammeverket, og vi vil også fortsette arbeidet på sluttrapporten.

Captain's log, stardate uke 12 (20.03.2014)

Det ble dessverre ingen oppdatering av bloggen forrgie uke (uke 11), da begge gruppemedlemmene var opptatt med et sosialt arrangement sammen med klassen.

De siste to ukene har vi jobbet for å utbedre de punktene som ble tatt opp av oppdragsgiver etter den første testversjonen av Reccoon ble levert. Dette har inkludert:

- Muligheter for brukeren å selv bestemme hvilke parametere som blir sendt med til de fleste verktøyene.

- Muligheter for å la brukeren legge til, fjerne og redigere Hosts, Users, OS og Ports som returneres fra skanninger.
- Sørge for at duplikat informasjon ikke forekommer.
- Eksport til HTML, XML og Metasploit.
- Lagt til flere fungerende moduler til rammeverket. Disse er Nmap, Theharvester og en modul for geolokasjoner (ip-api.com).

Vi har tatt en avgjørelse om å ikke implementere Nessus og OpenVAS som moduler. Dette er fordi Metasploit allerede har mulighet for sårbharhetsskanning. Det blir mer naturlig prosess når det blir gjort gjennom Metasploit, enn at det blir gjennom en vårt rammeverk. Derfor har vi gjort det lettere for å starte denne prosessen med å importere våre data til Metasploit.

Denne utbedringen av rammeverket vi har foretatt de siste ukene, har resultert i at vi har kommet frem til "testversjon" nummer 2 av Reccoon. Denne er sendt inn til oppdragsgiver og veileder i dag, og vi gleder oss til å få tilbakemeldinger.

Neste uke vil vi gjøre det siste av finpuss på rammeverket, og vi vil forhåpentligvis sitte igjen med en nesten ferdig programvare. Vi kan dermed bruke mer tid til arbeid med sluttrapporten.

Captain's log, stardate uke 15 (10.04.2014)

Det har gått et par uker siden forrige oppdatering. Siden siste innlegg har vi:

- Skrevet videre på hovedrapporten. Dette har hovedsakelig bestått av arbeid med kravspesifikasjon-, analyse- og designdelene av rapporten. Et av punktene som har opptatt mye av tiden vår går på de juridiske og etiske aspektene ved å bruke Reccoon. Vi har vært i kontakt med både NSM og Datatilsynet for å prøve finne frem til noen konkrete lover mot bruken av port skanning.
- Oppdragsgiver har testet Reccoon v2 med stor suksess. De har opplevd at rammeverket returnerer korrekt informasjon og er betraktelig raskere enn å gjøre alt manuelt (fra 1,5 time til 10 minutter). Både oppdragsgiver og vi på gruppen er strålende fornøyd med resultatet.

I tiden fremover vil vi fortsette med rapportskrivingen. Vi håper også på å begynne planleggingen av en omfattende test av rammeverket.