SURFnet Intrusion Detection System

K. Trippelvitz en H.J. Blok

Maandag 4 juli 2005

Inhoudsopgave

Managementsamenvatting

SURFnet heeft binnen het SURFworks Next Generation jaarplan 2005 een project geformuleerd waarin SURFnet een Intrusion Detection Systeem (IDS) van sensornetwerken gaat opzetten. Dit IDS zal uit verschillende sensoren bestaan die verspreid kunnen worden binnen de netwerken van de aangesloten instellingen. Door deze verspreiding kan er duidelijk worden geanalyseerd hoe een bepaald type kwaadaardig verkeer zich heeft verspreid binnen SURFnet en welke aangesloten instellingen er in meer of mindere mate hinder van ondervinden. De gegevens van deze sensoren zullen centraal opgeslagen en geanalyseerd worden. Vervolgens zullen de aangesloten instellingen geïnformeerd worden over de analyses.

Het IDS systeem zal uit 3 onderdelen bestaan, namelijk de sensor, de tunnel en de centrale server. De sensoren zullen bij de instellingen in het netwerk geplaatst worden. Hier ontvangt de sensor verkeer van laag 2 en hoger. Dit verkeer wordt vervolgens via een tunnel doorgestuurd naar de centrale server. De sensor zelf zal zo simpel mogelijk gehouden worden. Deze kan bijvoorbeeld draaien op (oude) hardware die nog beschikbaar is bij de instellingen die een sensor willen plaatsen. De software en het OS van de sensor zal op een USB stick gezet moeten worden waar de sensor vanaf kan booten. Indien de hardware geen mogelijkheid heeft om vanaf USB te booten zal er een boot CD gebruikt moeten worden om vanaf USB te kunnen booten. Als tunnel oplossing is gekozen voor het gebruik van OpenVPN vanwege de eenvoudigheid van het protocol, de goede encryptie en het feit dat OpenVPN laag 2 verkeer kan doorsturen. De centrale server zal worden ingericht als honeypot om interactie te verzorgen met het binnenkomende verkeer op de sensoren. Door het gebruik van een honeypot kan een goede analyse gemaakt worden van het kwaadaardige netwerkverkeer binnen een instelling.

In het vervolgonderzoek zal nog onderzoek gedaan moeten worden naar methodes om de sensor beter onderhoudsvrij te maken. Daarnaast zal nog onderzoek gedaan moeten worden naar mogelijke tools die op de centrale server gebruikt kunnen worden om de informatie te verwerken.

2. Inleiding

SURFnet heeft binnen het SURFworks Next Generation jaarplan 2005 een project geformuleerd waarin SURFnet een Intrusion Detection Systeem (IDS) van sensornetwerken gaat opzetten. Dit IDS zal uit verschillende sensoren bestaan die verspreid kunnen worden binnen de netwerken van de aangesloten instellingen. Door deze verspreiding kan er duidelijk worden geanalyseerd hoe een bepaald type kwaadaardig verkeer zich heeft verspreid binnen SURFnet en welke aangesloten instellingen er in meer of mindere mate hinder van ondervinden. De gegevens van deze sensoren zullen centraal opgeslagen en geanalyseerd worden. Vervolgens zullen de aangesloten instellingen geïnformeerd worden over de analyses.

SURFnet is een not-for-profit organisatie die actief is op het gebied van ICT voor onder andere onderzoek en educatieve doeleinden. De aangesloten instellingen, voornamelijk instellingen als universiteiten en hogescholen, hebben invloed op het beleid dat SURFnet voert. SURFnet prefereert daarom om met de klanten samen te werken op het gebied van nieuwe ontwikkelingen. De klanten zijn vaak betrokken bij de projecten die SURFnet voert. Bij dit IDS project zijn twee pilot instellingen betrokken, namelijk de Fontys hogeschool en de technische Universiteit van Delft.

In hoofdstuk drie wordt omschreven uit welke onderdelen het SURFnet IDS project bestaat en wat wel en wat niet tijdens het research project onderzocht wordt. Het vierde hoofdstuk gaat in op de hardware die gebruikt wordt voor de sensoren en gaat in op de installatie van de software op deze sensoren. Ook wordt er kort beschreven hoe er een proof-of-concept gemaakt is voor de installatie van de sensoren. Het vijfde hoofdstuk beschrijft een wijziging in de gevolgde strategie en de technische consequenties daarvan. Hoofdstuk zes beschrijft hoe de infrastructuur er uit moet komen te zien en in hoofdstuk zeven wordt ten slotte beschreven hoe een proof-of-concept gebouwd is van de hele opstellingen. Het achtste hoofdstuk geeft de conclusie van het project gevolgd door aanbevelingen voor vervolg onderzoek in hoofdstuk negen. Daarna volgen nog een aantal bijlagen die de technische realisatie van de proof-of-concept opstellingen beschrijven.

3. Het project

3.1. Projectfases

3.1.1. Ontwerp sensoren en infrastructuur

Als eerste zal er een sensor ontwikkelt moeten worden die SURFnet eenvoudig bij de aangesloten instellingen kan installeren (plug en play). Het is belangrijk dat deze sensor onderhoudsvrij is. De sensor moet het (kwaadaardige) verkeer verzamelen en doorsturen naar een centrale server. De sensor moet de mogelijkheid hebben om ruwe netwerkdata te kunnen verzamelen. Aangezien bandbreedte geen bottleneck is kan de sensor al het verkeer doorsturen. Hierbij wordt door de sensor geen onderscheid gemaakt tussen goedaardig en kwaadaardig verkeer. Het is niet de bedoeling om de sensor in te richten als honeypot.

Parallel aan het ontwerp van de sensor zal er een ontwerp gemaakt moeten worden voor de infrastructuur. Hierbij moet onder andere nagedacht worden over de connectie tussen de sensoren, de server en de data-opslag. De infrastructuur moet schaalbaar zijn en meer dan 100 sensoren kunnen ondersteunen. De sensoren zullen geplaatst worden in de bestaande infrastructuur van de bij SURFnet aangesloten instellingen (bijvoorbeeld tussen servers).

3.1.2. Centrale analyse

De sensoren zullen de gegevens centraal opslaan op de server. Om deze gegevens te kunnen analyseren moet er een centrale analyse tool ontwikkeld worden. Deze centrale analyse tool dient per aangesloten instelling te registeren welke data vanaf welke sensor binnenkomt en hier een aantal statistieken van bij te houden. De data die vanaf de sensor binnenkomt moet in een gangbaar formaat worden opgeslagen. De aangesloten instellingen zouden zelf inzage willen in de statistieken van hun sensor(en). De statistieken kunnen hiervoor bijvoorbeeld per faculteit worden onderverdeeld.

3.1.3. Testen

De laatste fase in het project is het testen van de sensor, de infrastructuur en de centrale server.

3.2. Afbakening

Voor het uitvoeren van het research project RP2 is maar een maand beschikbaar, daarom is het niet mogelijk om het hele SURFnet IDS project uit te voeren, hier is meer tijd voor nodig. Een aantal onderdelen van het SURFnet IDS project zullen dus niet gerealiseerd kunnen worden tijdens het research project. De onderdelen die binnen het project vallen zijn:

- Het ontwerpen van de sensor.
- Het ontwerpen van de infrastructuur.
- Het ontwerpen van de centrale analyse.
- Het bouwen van een proof-of-concept van de sensor.
- Het bouwen van een proof-of-concept van de infrastructuur.
- [Eventueel] Een proof-of-concept bouwen voor de centrale analyse.

De onderdelen die buiten het project vallen zijn:

- Het ontwikkelen van de analysetool voor de centrale server.
- Het volledig implementeren van het systeem.

3.3. De uitvoering

Het SURFnet IDS bestaat uit sensoren die data doorsturen naar een centrale server. Voor het ontwerp van deze componenten zijn verschillende mogelijkheden. Hoe de componenten het beste ingericht kunnen worden, is onderzocht door literatuurstudie en door het bouwen van een proof-of-concept.

4. De installatie van de sensor

Voor het SURFnet Intrusion Detection Systeem zullen bij de op SURFnet aangesloten instellingen sensoren geplaatst worden. Zoals omschreven in hoofdstuk 3.1 worden er aan de sensor een aantal eisen gesteld. Kort samengevat moet de sensor:

- plug en play zijn
- onderhoudsvrij zijn
- het (kwaadaardige) verkeer doorsturen
- de mogelijkheid hebben om ruwe netwerkdata verzamelen
- niet als honeypot ingericht worden

Voor het ontwerp van de sensor is er allereerst gekeken naar de hardware die als basis voor de sensor gebruikt kan worden. Daarna is er gekeken hoe de software op de sensor geïnstalleerd moet worden.

4.1. De hardware

Er kan verschillende hardware gebruikt worden als basis voor de sensor. In principe gelden voor de hardware dezelfde eisen als voor de sensor. Een aantal eisen zijn echter niet relevant voor de keuze van de hardware. De eisen die over blijven zijn:

- plug en play
- onderhoudsvrij
- uitgerust met één of twee netwerkkaarten (afhankelijk van de gekozen software)

Als basis voor de sensor is het allereerst mogelijk om de binnen de instelling beschikbare PC hardware te gebruiken. Hierdoor hoeft er geen nieuwe hardware aangeschaft te worden waarder er kosten worden bespaard. Voorwaarde is wel dat de hardware uitgerust is met één of twee netwerkkaarten (afhankelijk van de gekozen softwareoplossing), maar meestal is dat wel het geval. Andere eisen aan de hardware (zoals eisen met betrekking tot het geheugen, de harde schijf en de processorsnelheid) hangen af van de gekozen software. Aangezien de software op de sensor relatief eenvoudig kan blijven, zal bijna alle hardware aan deze eisen voldoen. Het plaatsen van een sensor bestaat uit het aansluiten van de sensor en het installeren van de software daarop. Het aansluiten van de sensor (aansluiten op het netwerk en de stroomvoorziening) is niet ingewikkeld en daarom zal het van de gebruikte software afhangen of de sensor plug en play is. Of de sensor onderhoudsvrij is wordt weer grotendeels bepaalt door de gebruikte software. Het enige onderhoud waar een instelling verantwoordelijk voor blijft is het vervangen van defecte hardware. Als de installatie van de software eenvoudig is, dan kan de hele sensor vervangen worden en hoeft er geen kennis aanwezig te zijn voor het vervangen van één specifiek onderdeel.

De tweede mogelijkheid is om een kant-en-klare sensor te plaatsen bij de instellingen. Als basis voor de kant-en-klare sensor kan bijvoorbeeld gebruik gemaakt worden van thinclients die uitgerust zijn met een eenvoudig moederbord, een processor, geheugen, een flash-disk en één of twee netwerkkaarten. Ook voor de kant-en-klare sensor is relatief eenvoudige hardware te gebruiken, omdat de software op de sensor relatief eenvoudig kan blijven. Het onderhoud van de sensor hangt weer af van twee dingen: het onderhoud van de software en het onderhoud van de hardware. De verantwoordelijkheid voor het onderhoud van de hardware (het vervangen van eventuele defecte onderdelen) zal bij een kant-en-klare sensor bij SURFnet komen te liggen. Vaak zal dit dus betekenen dat er iemand van SURFnet naar de instelling moet om defecte hardware te vervangen. De software bepaalt dus uiteindelijk of de sensor voor de instelling onderhoudsvrij is. Ook of de sensor plug en play is wordt door de software bepaalt. Het installeren van de software op deze kant-en-klare sensor kan al voor het plaatsen gedaan worden (door SURFnet). De instelling hoeft dus alleen nog maar de sensor aan te sluiten en te configureren. Hoe ingewikkeld het configureren van de sensor is, hangt af van de gebruikte software.

Een derde optie is om een virtuele sensor te maken. Dit zou bijvoorbeeld kunnen door op de router van een instelling een virtuele interface aan te maken die al het verkeer via een tunnel [3] doorstuurt naar de centrale server. Deze optie is niet verder onderzocht omdat niet alle instellingen de kennis en de mogelijkheid hebben de configuratie van de router te wijzigen. Daarnaast zijn instellingen die wel de kennis en de mogelijkheid hebben niet altijd bereid om wijzigingen op de router door te voeren. Dit komt omdat het aanpassen van de router

configuratie impact kan hebben op de productie omgeving.

4.2. De software

De sensoren die gebaseerd zijn op PC hardware (zowel de kant-en-klare sensor als de sensor die gebaseerd is op binnen de instelling beschikbare hardware) zullen voorzien moeten worden van een OS waarop de sensorsoftware geïnstalleerd kan worden. Niet alle eisen die voor de sensor als geheel gelden zijn ook van toepassing op de installatie van de software op de sensor. Alleen de eisen dat de software onderhoudsvrij en plug en play moet zijn, zijn van toepassing. De software bepaalt voor een groot deel of aan deze eisen voldaan wordt, de gebruikte hardware is slechts voor een klein deel bepalend.

4.2.1. Onderhoudsvrije software

In het ideale geval is de software op de sensor onderhoudsvrij. Er hoeven dus geen updates van de software plaats te vinden. Toch zijn er redenen waarom de software op de sensor een update nodig heeft.

1. De eerste reden is omdat er een bug in de geïnstalleerde software zit. Door deze bug kan deze sensor minder goed functioneren of kan een inbreker mogelijkerwijs toegang krijgen tot de sensor.
2. De tweede reden is een update in de sensorsoftware. Snort is een voorbeeld van software die afhankelijk is van updates van de Snort rules.

Om de sensor zo onderhoudsvrij mogelijk te houden is het dus belangrijk dat er zo weinig mogelijk software geïnstalleerd is. Als er op de sensor geen services aangeboden worden blijven er nog twee catagorieën bugs over die misbruikt kunnen worden om op de sensor in te breken.

De eerste mogelijkheid is om via een bug in de TCP/IP stack toegang te krijgen tot de sensor. Voor de Linux kernel [4] is onderzocht hoe vaak er een bug in de TCP/IP stack van de kernel voorkomt. In het overzicht [5] van de bugs is gezocht op bugs met de impact 'high' of 'blocking'. In totaal worden er 72 bugs gevonden in ongeveer 30 verschillende kernelversies. De meeste bugs hiervan gaan over het niet goed functioneren van de TCP/IP stack of over het ontbreken van functionaliteit maar gaan niet over een veiligheidslek. In de TCP/IP stack van een kernelversie zitten gemiddeld dus twee bugs.

De tweede mogelijkheid is dat er in de gebruikte sensorsoftware een bug zit waardoor een inbreker toegang kan krijgen. Een voorbeeld hiervan is een bug in TCPdump [6]: door een speciaal geformeerd pakketje zou TCPdump kunnen crashen.

4.2.2. Live CD

Er zijn drie mogelijke methoden voor het installeren van de software op de sensor. De eerste mogelijkheid is om een Live CD te maken. Een Live CD is een UNIX distributie waarbij het OS volledig in RAM geheugen draait. Voor een Live CD is er dus geen lokale hardeschijf nodig. Een voorbeeld van een Live CD is Knoppix, dit is een Live CD gebaseerd op Debian (Linux). Het is mogelijk om een bestaande Live CD te 'remasteren' en om op die manier een 'sensor OS' met alleen benodigde applicaties te maken.

Deze methode is ontzettend eenvoudig voor de instellingen waar de sensor geplaatst wordt. Het enige wat nodig is om een sensor te installeren is om een PC op te starten vanaf de Live CD. Voorwaarde is wel dat de PC op kan starten vanaf CD, maar dit ondersteunen tegenwoordig bijna alle PC's. Natuurlijk moet de PC aangesloten worden op het netwerk waarin de sensor geplaatst moet worden. Eventueel moeten de netwerkinstellingen van de sensor handmatig geconfigureerd worden na het opstarten. Het nadeel van een Live CD is dat er geen instellingen op de sensor bewaart kunnen worden. Als de sensor om wat voor reden dan ook opnieuw opstart moet een eventuele configuratie opnieuw aangemaakt worden. Instellingen waarbij de configuratie van de netwerkinstellingen handmatig plaatsvinden moeten dus na elke reboot deze configuratie opnieuw uitvoeren. Kortom deze oplossing is voor de meeste instellingen plug en play tenzij de configuratie van de netwerkinstellingen handmatig plaatsvindt.

Het updaten van de sensorsoftware kan op twee manieren, allereerst is het mogelijk om een nieuwe CD op te sturen (per post) naar de instelling. Het nadeel van deze oplossing is dat de instellingen verantwoordelijk worden voor het updaten van de sensor. Een update die eenvoudig lijkt (nieuwe CD erin en rebooten) wordt dan ineens vrij complex. Bij de instellingen moet er een verantwoordelijke worden aangewezen voor het updaten van de

sensoren, de instelling moet kennis in huis halen over het updaten van de sensoren (waar staan ze en hoe voer ik een update uit) en er moet door SURFnet gecontroleerd worden of de updates daadwerkelijk uitgevoerd zijn. Kortom deze methode van updaten is niet onderhoudsvrij wat wel een eis voor de sensoren is. De tweede mogelijkheid is om de sensor periodiek te laten controleren of er een update van de gebruikte software beschikbaar is. Als er een nieuwere versie is, dan wordt deze gedownload en vervolgens geïnstalleerd. Na een reboot zou de sensor dan weer volledig up-to-date zijn. Maar omdat de sensor volledig vanuit RAM geheugen werkt is deze update direct na de reboot weer verdwenen. Daarom is deze methode van updaten niet geschikt voor een Live CD.

4.2.3. USB

De tweede methode om de sensor te voorzien van een OS met de benodigde applicaties is door middel van een USB stick. Door de PC op te starten vanaf de USB stick wordt het 'sensor OS' geladen met alle benodigde applicaties.

Net zoals bij de Live CD is het installeren van een sensor met behulp van deze methode eenvoudig: het opstarten van een PC vanaf de USB stick. Ook hier geldt dat de PC moet worden aangesloten op een netwerk en dat er eventueel een aantal instellingen geconfigureerd moeten worden (bijvoorbeeld netwerkinstellingen). Helaas heeft alleen de nieuwere hardware de mogelijkheid om op te starten vanaf een USB stick. Niet alle hardware kan dus gebruikt worden voor deze variant. Dit probleem zou opgelost kunnen worden door een CD met bootloader te gebruiken die vervolgens de USB stick boot. Eventuele instellingen kunnen op de USB stick bewaart worden waardoor het niet nodig is om de configuratie opnieuw uit te voeren na een reboot. Deze oplossing is dus plug en play te noemen, alhoewel de methode iets ingewikkelder wordt als er een CD gebruikt moet worden om vanaf de USB stick op te starten.

De eerste mogelijkheid om de sensor te updaten is door middel van het opsturen (per post) van een nieuwe USB stick. Deze methode heeft dezelfde nadelen als bij het opsturen van een nieuwe Live CD en is niet geschikt omdat deze methode niet onderhoudsvrij is. Een tweede mogelijkheid voor het updaten van de sensor is het periodiek controleren op nieuwere versies. Na het downloaden, installeren en rebooten van de sensor, is de sensor weer volledig up-to-date. Omdat de USB stick de update kan opslaan is deze methode van updaten het meest geschikt voor de USB methode. Daarnaast is deze methode voor de instelling volledig onderhoudsvrij. Ook als er een CD gebruikt wordt om vanaf de USB stick te kunnen booten is deze laatste methode onderhoudsvrij te noemen. Op de CD staat namelijk alleen maar software die het mogelijk maakt om vanaf USB te booten. De CD bevat geen software die onderdeel vormt van de sensor (het OS of de applicaties).

4.2.4. Netboot

Netboot is een derde methode om de sensoren te voorzien van software. De sensor laadt bij het opstarten het 'sensor OS' en de benodigde applicaties vanaf een centrale server. Het OS draait volledig in het RAM geheugen waardoor er voor de sensor geen harde schijf nodig is.

Het installeren van de sensor bestaat weer uit het aansluiten van de PC op het netwerk, het opstarten en vervolgens eventueel het handmatig invoeren van de configuratie. Netboot wordt, net zoals het opstarten vanaf een USB stick, niet door alle PC hardware ondersteund. Dit probleem is weer op te lossen door gebruik te maken van een CD die vervolgens het netboot-image laadt. Aangezien het OS volledig in het RAM geheugen werkt worden de eventuele instellingen niet door de sensor bewaart. Technisch is het mogelijk om bijna alle instellingen te bewaren op de centrale server, alhoewel dit gecompliceerd is. De enige instellingen die niet op de centrale server zijn op te slaan, zijn de instellingen van het netbooten zelf. Dit vormt alleen een probleem bij hardware waar een CD nodig is om te kunnen netbooten en dan alleen als de netwerkinstellingen handmatig geconfigureerd moeten worden. Het ophalen van het netboot-image vanaf de centrale server gebeurt via het netwerk, om dit mogelijk te maken zullen een aantal speciale poorten opengezet moeten worden. Niet alle instellingen laten het openzetten van deze poorten toe. Ook zijn niet alle instellingen in staat dit zelf uit te voeren. Zelfs al zou een instelling dit toelaten en kunnen dan vereist dit nog steeds veel organisatorisch werk. Op dit gebied is de netboot variant dus niet plug en play.

Voor het updaten van het 'sensor OS' en de applicaties hoeft er alleen maar een nieuw netboot-image op de centrale server gezet te worden. Na een herstart van de sensor is deze volledig up-to-date. Het probleem bij het updaten zit dus niet in distributie van de software. Wat wel een probleem vormt is het rebooten van de sensor.

Uit schaalbaarheidsoogpunt is het niet mogelijk om de sensor vanaf afstand te herstarten. Hiervoor zou namelijk exact bijgehouden moeten worden welke sensor wel en welke sensor niet gereboot is. Daarnaast zouden de sensoren remote toegang moeten geven waarvoor de router en/of firewall van een instelling aangepast moet worden. Zoals al eerder is aangegeven is dit niet wenselijk. Gelukkig is dit probleem op te lossen door de sensoren periodiek te laten controleren of er een nieuwe netboot-image beschikbaar is, en te herstarten als dat het geval is. Deze methode is dus onderhoudsvrij als de sensor zelf controleert op nieuwere versies, ook als er een CD gebruikt wordt om te kunnen netbooten. Deze CD bevat namelijk (net zoals voor de USB variant) alleen software die het mogelijk maakt om te netbooten en vormt geen essentieel onderdeel van de sensor (het OS en de applicaties).

4.2.5. Brede hardware ondersteuning

De software die geïnstalleerd wordt moet natuurlijk een goede ondersteuning bieden voor de hardware die gebruikt wordt als basis voor de sensor. Als dit niet het geval is, is de sensor namelijk niet plug en play. Als ervan uitgaan wordt dat het OS wat op de sensoren geïnstalleerd wordt een UNIX variant is dan kan er ruwweg gekozen worden uit een Linux distributie of een BSD variant. De meeste hardware ondersteuning (plug en play) wordt dan geboden door Linux, wat daarom een voor de hand liggende keuze is. Een BSD variant zou daarentegen wel wat onderhoudsvrijer kunnen zijn, omdat er BSD varianten zijn waar relatief weinig bugs in voorkomen. Updates blijven echter noodzakelijk waardoor dit geen punt van overweging vormt.

4.3. Proof-of-concept

Er zijn twee proof-of-concepts gebouwd voor de installatie van de sensoren. Beide proof-of-concepts waren op Knoppix (Debian) gebaseerde Live CD's. Achteraf is gebleken dat de keuze voor een Live CD als installatie medium voor de sensor niet de beste keuze was omdat het onderhoud hiervan problemen opleverde. Toch is er met het bouwen van dit proof-of-concept ervaring opgedaan die ook als basis kan dienen voor het maken van een bootable USB stick. Om deze reden zijn de ervaringen met deze Live CD's in dit verslag opgenomen.

4.3.1. Knoppix remaster

Het eerste proof-of-concept waar aan gewerkt is, is het remasteren van een Knoppix CD. Het voordeel van Knoppix is dat deze Linux distributie al heel veel hardware ondersteunt. Het grootste probleem van Knoppix is het feit dat het nog redelijk groot is. De sensor software heeft in principe heel weinig applicaties nodig, veel van de bij Knoppix meegeleverde tools zijn daarom overbodig. Naast het feit dat de overbodige tools ruimte innemen, vormen ze ook een beveiligingsrisico. Alle overbodige packages moeten dus verwijderd worden. Het moeilijkste gedeelte van de remaster is dan ook het beslissen welke tools nodig zijn en welke niet. In bijlage B staat beschreven hoe de Knoppix remaster is gemaakt.

4.3.2. DSL als basis

Het tweede proof-of-concept waar parallel aan gewerkt is, is een op DSL gebaseerde Live CD. DSL [7] is een gestripte versie van Knoppix. Van DSL zijn alleen de kernel en de bootscripts gebruikt om op te kunnen starten vanaf CD. Voor de applicaties die op de Live CD moesten komen te staan is gebruik gemaakt van BusyBox [8]. BusyBox compileert een aantal standaard Linux applicaties met basis functionaliteit. In bijlage A staat beschreven hoe de Live CD samengesteld is.

Er waren eigenlijk twee problemen bij het samenstellen van deze CD. Het eerste probleem was dat er tools ontbraken die nodig waren om te booten. Dit probleem was relatief eenvoudig op te lossen door deze tools toe te voegen. Het tweede probleem was dat er in eerste instantie gebruik gemaakt zou worden van TCPdump als sensorsoftware. TCPdump zat niet in BusyBox en moest daarom handmatig gecompileerd worden. TCPdump is afhankelijk van libpcap, die beide afhankelijk zijn van een aantal libraries. Daarom is TCPdump en libpcap statisch gecompileerd zodat de binaries niet afhankelijk zijn van libraries. Bij het testen van de CD bleek dat libpcap afhankelijk is van een aantal modules in de kernel. Deze modules zaten niet standaard in de bij DSL gebruikte kernel waardoor er een andere kernel als basis gebruikt moet worden. Er is geprobeerd de kernel en de bootscripts van de Knoppix CD als basis te gebruiken in plaats van de kernel en bootscripts van DSL. Ook is een poging gedaan om zelf een kernel te compileren. Beide pogingen waren zonder resultaat, vooral omdat er te weinig tijd was om dit proof-of-concept verder uit te werken.

5. Wijziging strategie

In eerste instantie was het de bedoeling dat de sensor alleen het (kwaadaardige) verkeer verzamelde en dit doorstuurde naar de centrale server. Het was niet de bedoeling dat de sensor een honeypot werd en dus interactie pleegt met een eventuele aanvaller. Na overleg met de TU Delft, één van de instellingen die pilot willen zijn voor het project, bleek dat er niet zozeer een behoefte is aan een 'statische' sensor. De statische sensor kan alleen een beperkte set van aanvallen detecteren omdat er geen interactie met een eventuele aanvaller gepleegd wordt. Deze sensor zou dus vooral bruikbaar zijn voor SURFnet brede statistische gegevens en zou weinig nieuwe inzichten bieden voor de aangesloten instellingen. Daarom is er meer behoefte aan een sensor die wel interactie pleegt met een eventuele aanvaller.

De andere eisen die aan de sensor gesteld worden blijven nog steeds gelden:

- plug en play
- onderhoudsvrij
- het (kwaadaardige) verkeer doorsturen
- de mogelijkheid hebben om ruwe netwerkdata verzamelen

5.1. De sensor als honeypot

De eerste mogelijkheid om interactie te plegen met een eventuele aanvaller is door de sensor in te richten als honeypot. De sensor biedt een aantal fake-services aan in het netwerk waar de sensor zich bevindt waarvan de activiteiten gevolgd kunnen worden. Maar het plaatsen van een honeypot in het netwerk heeft een aantal consequenties.

Allereerst moeten de activiteiten op de honeypot nauwkeurig in de gaten gehouden worden. Dit kan door het doorsturen van de logs naar de centrale server. Als deze logs via een standaard beschikbaar protocol doorgestuurd worden dan blijft de sensor plug en play. Toch is het raadzaam om af en toe op de honeypot in te loggen om handmatig te controleren of er niets vreemds gebeurt. Van de instellingen waar de sensor geplaatst wordt is niet te verwachten dat zij deze taak op zich nemen, daarom zal dit gedaan moeten worden door SURFnet. SURFnet zal dus toegang moeten hebben tot de sensor. Om dit mogelijk te maken moeten er poorten opengezet worden op de router of firewall, iets wat voor veel instellingen een probleem vormt door het gekozen beleid of door ontbrekende kennis. Deze methode is dus niet plug en play. Daarnaast kan de configuratie van de sensor wijzigen waardoor de router of firewall opnieuw aangepast moet worden om de beheerder vanaf afstand toegang te geven.

Ten tweede moet de software op de sensor goed up-to-date gehouden worden om eventuele inbraken te voorkomen. In hoofdstuk 4.2.1 was al gebleken dat updates van de sensor noodzakelijk waren. Het up-to-date houden van de sensor is dus geen nieuw probleem.

Als de sensor als honeypot ingericht wordt is het niet direct mogelijk om het (kwaadaardige) verkeer door te sturen of om de ruwe netwerkdata te verzamelen. Om dit mogelijk te maken zullen speciale aanpassingen gemaakt moeten worden aan de sensor.

5.2. De centrale honeypot

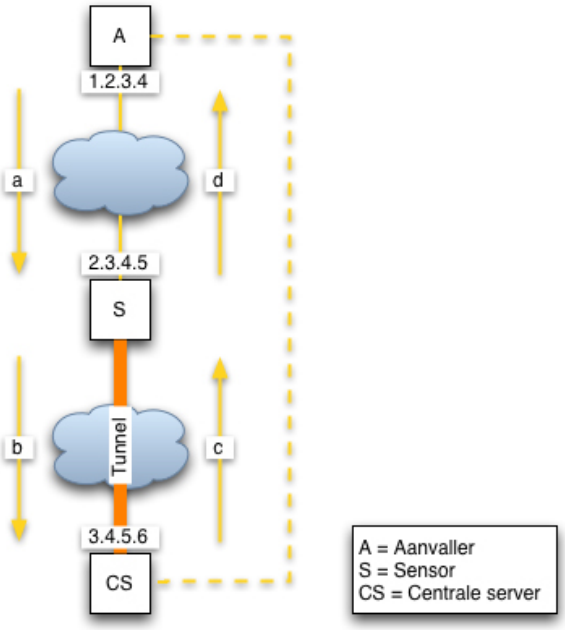
De tweede mogelijkheid om interactie met een eventuele aanvaller te plegen is om een centrale honeypot in te richten. Op deze centrale honeypot vindt de daadwerkelijke interactie met de aanvaller plaats en de sensoren bij de instellingen zorgen alleen maar voor het transport van het verkeer van en naar de centrale honeypot.

Aangezien de sensor alleen voor het transport zorgt is het dus niet meer noodzakelijk dat een beheerder op de sensor in kan loggen om de activiteiten op de sensor te controleren. Ook hoeven er geen logs meer doorgestuurd te worden omdat de analyse van het verkeer centraal plaatsvindt. Het blijft natuurlijk wel noodzakelijk om de sensor te voorzien van updates, maar hier zijn voor de instelling onderhoudsvrije methodes voor te ontwerpen.

5.3. De technische realisatie

Voordat er begonnen is met het bouwen van een centrale honeypot, is de situatie eerst geanalyseerd om enigszins

te bepalen of het technisch realiseerbaar is. De situatie komt er als volgt uit te zien.



Een aanvaller stuurt een pakket (a) naar de sensor, als bestemmingsadres heeft het pakket de sensor (2.3.4.5), als bronadres heeft het pakket de aanvaller (1.2.3.4). De sensor moet het pakket nu niet verwerken maar doorsturen naar de centrale server (b). Als de sensor dit pakket 'gewoon' verder routeert, dan blijven deze adressen gelijk, ook als er gebruik gemaakt wordt van een tunnel.

Het pakket van de aanvaller komt vervolgens aan op de centrale server die het pakket verwerkt en een antwoord op dit pakket genereert (c). Normaal gesproken zou dit pakket als bronadres het adres van de centrale server hebben (3.4.5.6) en als bestemmingsadres het adres van de aanvaller (1.2.3.4). Na het ontvangen van dit pakket zal de sensor vervolgens dit pakket doorsturen naar de aanvaller, waarbij de adressen niet gewijzigd worden (d).

Het eerste probleem hierbij is dat de aanvaller geen pakket met het bronadres van de sensor terugkrijgt, maar een pakket met het bronadres van de centrale server (3.4.5.6). Het tweede probleem is dat de server niet weet dat hij het pakket terug moet routeren via de tunnel naar de sensor. Het pakket kan dus via een andere route teruggaan naar de aanvaller (de stippellijn). Op deze manier wordt er geen gewone server of een gewoon werkstation gesimuleerd. Het derde probleem is het feit dat de server niet weet dat het pakketje voor hem bestemd is, omdat het een andere bestemming heeft (namelijk de sensor).

Deze problemen kunnen op twee manieren opgelost worden. Allereerst kan de sensor het pakket aanpassen zodat het pakket het juiste bronadres heeft. Dit kan bijvoorbeeld door gebruik te maken van NAT. Op deze manier weet de centrale server nog steeds niet waar het pakket heen gerouteerd moet worden. Dit probleem is op te lossen door op de centrale server een virtuele interface aan te maken met het IP adres van de sensor (2.3.4.5) en vervolgens source based routing te gebruiken om te bepalen waar het pakket heen gestuurd moet worden. De situatie wordt dan als volgt.

De aanvaller stuurt een pakket naar de sensor (a), de sensor doet niets anders dan dit pakket doorsturen naar de centrale server (b). De centrale server ontvangt dit pakket op de virtuele interface (2.3.4.5), verwerkt dit pakket en stuurt dit vervolgens via dezelfde virtuele interface terug (c). Dit pakket heeft als bronadres het adres van de virtuele interface. Op basis van dit bronadres bepaalt de server via welke route dit pakket teruggestuurd moet worden, in dit geval via de tunnel naar de sensor. De sensor ontvangt dit pakket en routeert het pakket op basis van bestemming naar de aanvaller terug. De sensor hoeft dus geen NAT te gebruiken om het juiste bronadres aan het pakket mee te geven.

Om problemen met de routering te voorkomen kan er het beste een tunnel tussen de sensor en de centrale server opgezet worden. Er moet hiervoor onderzocht worden welke tunneltechniek het meest geschikt is voor deze situatie. Vervolgens kan er een proof-of-concept gebouwd worden om uit te testen of het daadwerkelijk in de

praktijk te realiseren is.

6. Infrastructuur

Om het verkeer van de sensor naar de centrale server te kunnen sturen zal er een tunnel opgezet moeten worden. De verschillende tunnel mogelijkheden [3] kunnen in twee categorieën opgedeeld worden. Ten eerste zijn er de tunnels met sterke encryptie:

- IPsec [2]
- SSH
- SSL
- OpenVPN [1]

Ten tweede zijn er de tunnels zonder encryptie, namelijk:

- PPTP
- L2TP
- GRE [9]
- IP-IP

Om een onderbouwde keuze te maken zullen de eisen die aan de tunnel gesteld worden eerst duidelijk moet zijn. Wanneer de centrale server als een honeypot wordt ingericht, dan stuurt de sensor alleen het verkeer door en doet zelf verder niets met het verkeer. De honeypot op de centrale server simuleert dan een werkstation in het netwerk van de instelling. Om dit goed te doen moet deze gesimuleerde computer zowel op laag 2 als op laag 3 pakketten versturen en ontvangen.De eerste eis aan de tunnel is dat het mogelijk is om zowel laag 2 als laag 3 verkeer te kunnen transporteren. De volgende keuze is het wel of niet gebruiken van encryptie. Aan de ene kant is het verkeer dat de sensor ontvangt geen regulier verkeer met belangrijke data, er draaien immers geen echte services op de sensoren. Aan de andere kant kan een aanvaller, als er maar één netwerkkaart in de sensor zit en er geen encryptie wordt gebruikt, wel zien dat zijn eigen verkeer weer verder wordt gestuurd door een tunnel. Dit laatste is niet van toepassing op virussen en wormen. Daarnaast kan met behulp van encryptie ook de integriteit van het verkeer, dat naar de centrale server wordt gestuurd, gewaarborgd worden. Het gebruik van encryptie bij het doorsturen van het verkeer door de tunnel is dus wenselijk. Verder is het belangrijk dat de tunneltechniek aan de eisen voldoet die aan de sensor gesteld worden. Ten eerste moet de tunnel plug en play zijn. Het moet voor de instelling makkelijk te installeren zijn. De tunnel moet in principe zonder handmatige acties op de sensor opgezet kunnen worden. Daarnaast moet de tunnel onderhoudsvrij zijn. Dat wil zeggen dat de instelling zelf geen onderhoud hoeft te plegen aan de tunnel, dit betekend voornamelijk dat de tunnel zonder aanpassingen aan de configuratie van de firewall(s) en router(s) van de instelling opgezet kan worden.

Het is mogelijk tunnels weer in andere tunnels te tunnelen, waardoor meerdere lagen tunnels ontstaan. Een tunneltechniek die bijvoorbeeld wel laag 2 verkeer kan tunnelen maar geen mogelijkheid heeft voor encryptie kan je vervolgens inpakken in een tunnel die wel encryptie heeft. Het nadeel van deze oplossing is dat er steeds minder ruimte in de tunnel overblijft voor het echte verkeer. Daarom is het wenselijk om zo weinig mogelijk lagen tunnels te gebruiken.

6.1. IPsec tunnel

De eerste mogelijkheid om een tunnel op te zetten is door gebruik te maken van IPsec. De voordelen van het gebruik van IPsec liggen vooral in de goede authenticatie en encryptie. Hierbij wordt ofwel gebruik gemaakt van certificaten ofwel van een pre-shared secret. Het gebruik van IPsec als tunnel methode heeft ook een aantal nadelen. IPsec maakt gebruik van een aantal poorten en protocollen die meestal niet aan staan op de routers en firewalls van de instellingen. Hierdoor moeten de betreffende instellingen aanpassingen doen aan hun routers en/of firewalls. Dit is geen wenselijke situatie, zoals al eerder is aangegeven. Op deze manier werkt IPsec namelijk niet plug en play. Een andere reden waarom IPsec niet voldoet aan de eisen van de tunnel is het feit dat IPsec zelf geen laag 2 verkeer door de tunnel heen kan sturen. Om dit te kunnen bereiken zal een ander protocol in combinatie met IPsec gebruikt moeten worden.

6.2. SSH/SSL

SSH en SSL zijn methodes die allebei een tunnel opzetten op laag 4. Zowel SSH als SSL zijn methodes die

makkelijk geïnstalleerd en gebruikt kunnen worden. Toch zijn beide methodes niet direct plug en play, omdat er gebruik wordt gemaakt van unieke keys of certificaten. Deze zullen op de sensor gegenereerd moeten worden om ervoor te zorgen dat ze uniek zijn. Het is echter mogelijk om deze keys of certificaten automatisch te genereren en bekend te maken aan de centrale server waardoor toch aan de plug en play eis voldaan kan worden. SSH en SSL kunnen meestal zonder veranderingen aan router of firewall configuraties gebruikt worden. Het grootste nadeel van beide methodes is het feit dat deze methodes op zichzelf geen mogelijkheid bieden voor het versturen van laag 2 verkeer door de tunnel. Om dit toch te kunnen realiseren zou gebruik gemaakt moeten worden van een tweede protocol. Hiervoor moet een tunnel gebruikt worden die wel in staat is laag 2 verkeer te versturen, zoals bijvoorbeeld L2TP. In dit geval zorgt SSH of SSL voor de beveiliging van de tunnel.

6.3. OpenVPN

OpenVPN is eigenlijk geen tunnel protocol maar een programma dat een tunnel protocol implementeert. OpenVPN is een flexibel softwarepakket wat ondersteund wordt door veel verschillende besturingssystemen, onder andere Linux en verschillende BSD varianten. Daarnaast biedt OpenVPN de mogelijkheid voor twee verschillende soorten van encryptie, namelijk encryptie gebaseerd op statische sleutels of encryptie gebaseerd op certificaten. OpenVPN maakt gebruik van UDP als onderliggend transport protocol.

OpenVPN is in principe een methode om een tunnel op te zetten waarbij zowel encryptie als het versturen van laag 2 verkeer mogelijk is. OpenVPN maakt gebruik van SSL om de tunnel op te zetten. Dit betekent dat er op de sensor een shared secret moet staan of een certificaat gegenereerd moet worden wat vervolgens door de centrale server ondertekend moet worden. Dit is, net zoals bij SSH en SSL, in principe te automatiseren en hoeft geen belemmering te zijn. Het voordeel van OpenVPN is dat er in de tunnel geen extra tunnellaag gebruikt wordt om het laag 2 verkeer te versturen, het (laag 2) verkeer wordt direct over de SSL laag gestuurd. OpenVPN voldoet in principe aan alle eisen die gesteld zijn aan de tunnel. De kanttekening hierbij is dat er op de sensor een aantal acties geautomatiseerd moeten worden en dat er goed nagedacht moet worden over de structuur om de certificaten te ondertekenen.

6.4. PPTP

PPTP is een tunnel protocol dat laag 3 en hoger kan tunnelen. Het is origineel bedoeld om IP verkeer te tunnelen over point-to-point links en is relatief eenvoudig op te zetten. PPTP encapsuleert PPP pakketten aan de hand van een variant op het GRE protocol. Hierdoor kan PPTP ook andere protocollen dan IP tunnelen, zoals bijvoorbeeld IPX. PPTP maakt gebruik van PAP of CHAP voor de authenticatie. De encryptie die gebruikt word bij het PPTP protocol is niet sterk en wordt meestal aangevuld met andere protocollen, zoals bijvoorbeeld IPsec. Daarnaast is PPTP niet in staat om direct laag 2 verkeer te versturen zonder gebruik van een ander protocol.

6.5. L2TP

L2TP is een tunnel protocol met de mogelijkheid om laag 2 verkeer door te sturen. In dit opzicht voldoet L2TP aan de eisen voor de tunnel. Daarentegen biedt L2TP weinig mogelijkheden om encryptie toe te passen. Hierdoor is dit protocol niet echt veilig te noemen. Dit is ook de reden dat L2TP vaak in samenwerking gebruikt wordt met andere protocollen, zoals IPsec, om de beveiliging toe te passen. Het L2TP protocol voldoet in dit geval alleen aan de eis om laag 2 verkeer door te kunnen sturen en zal dus altijd in combinatie met een andere tunnel techniek gebruikt moeten worden om encryptie van het verkeer te realiseren. Het gebruik van IPsec hiervoor is geen optie vanwege eerder genoemde redenen. SSL is daarentegen wel een optie om te gebruiken voor de encryptie van het verkeer.

6.6. GRE

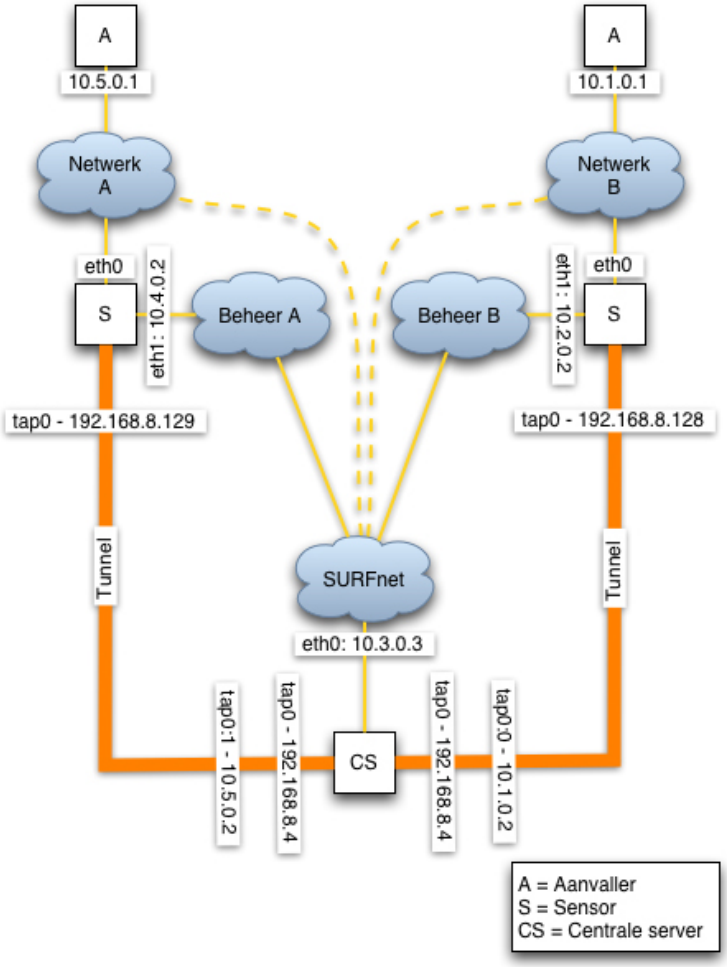
GRE is een protocol dat in principe een packaging protocol is. Dat wil zeggen dat het in staat is elk pakketje te encapsuleren in een generiek data pakketje. Op deze manier kan GRE veel verschillende protocollen door een tunnel heen sturen. Het GRE protocol wordt voornamelijk gebruikt bij het tunnelen over IP, maar is hiertoe niet gelimiteerd. Het GRE protocol is een protocol wat vrijwel al het verkeer kan doorsturen in een tunnel. Dat wil zeggen dat er ook laag 2 verkeer verstuurd kan worden. GRE heeft zelf geen encyptie en zal dus gecombineerd moeten worden met een andere tunneltechniek om encryptie te realiseren.

6.7. IP-IP

IP-IP is een tunneling protocol dat alleen laag 3 en hoger kan tunnelen. Hierbij wordt simpelweg gebruik gemaakt van een dubbele encapsulatie van het IP pakket. Op deze manier wordt er een tunnel opgezet zonder encryptie. Om toch encryptie toe te kunnen passen op de pakketjes wordt vaak IPsec of een dergelijk protocol gebruikt voor de encryptie (en/of authenticatie). Ook hier is het duidelijk dat een tweede protocol noodzakelijk is voor het verzorgen van de encryptie en voor het versturen van laag 2 verkeer.

7. Proof-of-concept van de hele opstelling

Om de hele opstelling te testen is er een proof-of-concept gebouwd van een centrale server, twee sensoren en twee aanvallers. De twee aanvallers zijn gesimuleerd met één machine met twee netwerkkaarten. De gebouwde opstelling is getekend in het onderstaande plaatje.



Voor het opzetten van de tunnel tussen de sensor en de centrale server is gebruik gemaakt van OpenVPN. Op de centrale server draait de OpenVPN server en op de sensor de OpenVPN client. De sensor heeft twee netwerkkaarten, één netwerkkaart (`eth1`) is aangesloten op een beheersnetwerk en wordt gebruikt om de tunnel op te zetten naar de centrale server. De andere netwerkkaart (`eth0`) wordt verbonden met het netwerk waar de sensor het eigenlijke werk moet doen. Bij het opzetten van de tunnel maakt de sensor lokaal een `tap0` (netwerk)device aan. Al het verkeer wat naar dit device gestuurd wordt, wordt via de tunnel naar de centrale server gestuurd. Het andere eindpunt van de tunnel is de centrale server. Vóór het opstarten van de OpenVPN server wordt hier een `tap0` device aangemaakt waar alle tunnels vanaf de sensoren op 'uitkomen'.

Door op de centrale server een virtuele interface aan te maken, die deel uitmaakt van het netwerk waarin de sensor 'zijn werk doet', kan de centrale server zich voordoen als onderdeel van dat netwerk waarin de sensor zich bevindt. De sensor moet al het verkeer doorsturen wat voor deze virtuele interface bestemt is. Om dit technische mogelijk te maken moet op de sensor het `tap0` device samen met `eth0` in een bridge worden gehangen. De sensor functioneert dan puur als bridge en zal al het verkeer wat bestemt is voor de virtuele interface op de centrale server naar de centrale server doorsturen en omgekeerd. De tunnel wordt dus als het ware een virtuele netwerkkabel van de virtuele interface op de centrale server naar het netwerk waar de sensor geplaatst is.

Bij het testen van de opstelling met twee sensoren bleek dat er toch pakketten die bedoeld waren voor netwerk A terecht kwamen in netwerk B. De pakketten die in netwerk B terecht kwamen waren broadcasts uitgestuurd door de centrale server. Hier moet dus een oplossing voor bedacht worden. Het filteren van broadcasts is geen oplossing omdat deze noodzakelijk zijn voor de interactie tussen aanvaller en centrale honeypot. Hoe dit probleem opgelost moet worden zal nog verder uitgezocht moeten worden.

In bijlage C staat beschreven hoe de configuratie van deze testopstelling eruit ziet.

8. Conclusie

Tijdens het project is er door literatuurstudie en door het bouwen van proof-of-concepts onderzocht wat de beste methode is voor het bouwen van het SURFnet Intrusion Detection Systeem. De eisen die gesteld worden aan het IDS systeem zijn:

- Plug en play
- Onderhoudsvrij
- Het (kwaadaardige) verkeer doorsturen naar de centrale server.
- De mogelijkheid hebben om ruwe netwerkdata verzamelen.

Aangezien een statische sensor weinig nieuwe inzichten biedt voor de instellingen waar de sensoren geplaatst worden is er gekozen om toch voor een honeypot constructie te kiezen. Aan het inrichten van de sensor als honeypot zitten een groot aantal nadelen. De sensor werkt namelijk niet direct plug en play en is ook niet onderhoudsvrij voor de instelling waar de sensor geplaatst wordt. Dit komt omdat er vanaf afstand toegang moet zijn tot de sensor om beheerstaken uit te voeren. Wanneer de sensor als honeypot ingericht wordt zullen de gebruikte tools nog verder ontwikkelt moeten worden om aan de andere eisen te voldoen. Daarom is het beter om de centrale server als honeypot in te richten en om het verkeer van de sensor door te sturen naar de centrale server. Deze methode voldoet namelijk wel aan de gestelde eisen. Hiervoor is het belangrijk dat de sensor zowel het laag 2 als het laag 3 verkeer kan doorsturen naar de centrale server. Tevens kan hiermee voldaan worden aan de eis dat er vanaf de centrale server ruwe netwerkdata te verzamelen is van de sensoren. Een analyse van de situatie waarin de centrale server functioneert als honeypot laat zien dat dit technisch ook haalbaar is.

8.1. De sensor

Om aan de gestelde eisen voor de sensor te voldoen maakt het eigenlijk niet veel uit welke hardware variant gekozen wordt. Alleen de virtuele sensor is duidelijk geen optie. De keus om bij de instelling beschikbare hardware te gebruiken heeft als voordeel de lagere kosten terwijl de kant-en-klare sensor volledig werkend opgeleverd kan worden en alleen nog maar aangesloten hoeft te worden.

Als de drie methoden voor de installatie van de software op een rij gezet worden dan is te concluderen dat de meeste methoden niet direct onderhoudsvrij en ook niet altijd direct plug en play zijn. Vaak is het echter wel realiseerbaar om deze methodes onderhoudsvrij en plug en play te krijgen. Uiteindelijk is de Live CD methode plug en play maar niet onderhoudsvrij, de USB methode zowel plug en play als onderhoudsvrij en, ten slotte, de Netboot methode wel onderhoudsvrij maar niet plug en play. De meest geschikte methode voor de installatie van de sensor is dus om gebruik te maken van de USB methode, eventueel aangevuld met een CD om vanaf USB op te kunnen starten.

Door bugs in de gebruikte software en door updates van de sensorsoftware zal de sensor afhankelijk blijven van updates. Het mechanisme wat zorgt voor de updates zal nog ontwikkelt moeten worden, in hoofdstuk negen wordt een mogelijk updatemechanisme beschreven.

Verder kan er geconcludeerd worden dat een Linux distributie de beste keuze is als OS voor de sensoren omdat Linux de beste ondersteuning heeft voor de hardware. Hierdoor heeft de sensor een hoger 'plug en play gehalte'. Op het gebied van onderhoud maakt het niet veel uit of er gekozen wordt voor Linux of voor BSD.

Op basis van de ervaring met de proof-of-concepts valt te concluderen dat het remasteren van Knoppix geen goede optie is voor het samenstellen van het 'sensor OS'. De Knoppix distributie zit vol met onnodige packages en het is veel werk deze er allemaal uit te halen, als dat al mogelijk is door de afhankelijkheid van sommige packages van andere packages. De andere optie is om een aantal onderdelen van de Knoppix distributie te gebruiken voor het samenstellen van een geheel eigen distributie. Het concept wat gebaseerd is op DSL is hier een voorbeeld van. Alleen is DSL waarschijnlijk geen goede basis omdat er modules ontbreken in de kernel die voor de sensorfunctie noodzakelijk zijn.

8.2. De tunnel

Om al het (kwaadaardige) verkeer van de sensor naar de centrale server te kunnen sturen kan er het beste gebruik gemaakt worden van een tunnel, voornamelijk om problemen met routering te voorkomen en om encryptie toe te

kunnen passen. Deze tunnel moet voldoen aan een aantal eisen. De tunnel moet ten eerste laag 2 en hoger verkeer door kunnen sturen aangezien de honeypot dit verkeer nodig heeft om interactie te simuleren. Daarnaast is dit ook nodig om de ruwe netwerkdata op te kunnen vangen op de centrale server. Verder is het wenselijk om encryptie van het verkeer toe te passen, aangezien dit een goede methode is om de integriteit van het verkeer te waarborgen. Gebaseerd op de eisen die gesteld worden aan de sensor moet de tunnel eenvoudig opgezet kunnen worden (plug en play) en zonder menselijke interactie kunnen opstarten en herstarten (onderhoudsvrij).

Na de verschillende tunnel protocollen beoordeeld te hebben op deze eisen vallen er direct al een aantal af vanwege de slechte encryptie of de afwezigheid van encryptie in de tunnel protocollen. IP-IP, L2TP, PPTP en GRE bieden geen of slechte encryptie, daarom zal er altijd een tweede tunnelprotocol gebruikt moeten worden om encryptie te realiseren. IPsec heeft wel de mogelijkheid voor encryptie maar kan in de meeste gevallen niet zonder aanpassingen aan de router of firewall van een instelling functioneren, daarnaast is het ook niet in staat om laag 2 verkeer door de tunnel te sturen. De keuze is uiteindelijk op OpenVPN gevallen omdat SSH en SSL op zichzelf geen laag 2 verkeer door de tunnel kunnen sturen. Bij OpenVPN hoeft er geen tweede tunnelprotocol gebruikt te worden om het mogelijk te maken om laag 2 verkeer te versturen. OpenVPN voldoet dus aan de eisen die gesteld zijn aan de tunnel. Een tunnel met OpenVPN is relatief eenvoudig op te zetten. De encryptie wordt opgezet met behulp van SSL certificaten en het biedt ook de mogelijkheid om laag 2 verkeer door de tunnel te sturen. OpenVPN is hierdoor een voor de hand liggende keuze. Wel moeten er nog een aantal acties geautomatiseerd worden om OpenVPN echt plug en play te maken.

8.3. De centrale server

Op de centrale server zal een honeypot moeten draaien die ervoor zorgt dat er interactie mogelijk is met de aanvallers. Zoals al eerder is uitgelegd is deze interactie nodig om echt interessante informatie te kunnen verzamelen. Aan het onderzoek hoe de informatie op de centrale server geanalyseerd moet worden is nog geen tijd besteed, dit is dus iets voor het vervolg onderzoek.

9. Vervolg onderzoek

Het vervolgonderzoek kan op een aantal punten verder gaan waar bij het huidige onderzoek geen tijd meer voor was. Er is een keuze gemaakt met betrekking tot de sensorsoftware, de tunnelmethode en de centrale server, maar nog niet alle punten zijn volledig uitgewerkt.

Ten eerste moet de sensor nog volledig onderhoudsvrij gemaakt worden. Dat betekent dat de sensor onder andere automatisch een SSL certificaat moet kunnen genereren, het IP adres waarop de sensor moet functioneren door moet geven naar de centrale server en in staat zijn automatisch updates te downloaden en te installeren. Tijdens het project is er een concept gemaakt waarmee automatisch een SSL certificaat gegenereerd wordt (zie bijlage C.2). Er is nog geen methode bedacht om het IP adres door te geven aan de centrale server. Voor het update van de sensor is wel een methode bedacht. Het updaten van de sensor zou kunnen door gebruik te maken van DNS en HTTP. In een tekst veld van DNS kan de huidige softwareversie worden gezet. De sensor controleert regelmatig of het versienummer in het DNS tekst veld verandert. Als het tekst veld verandert, is er een nieuwere versie van de software beschikbaar en worden de updates vervolgens via HTTP gedownload. Als er gebruik gemaakt wordt van HTTPS in plaats van HTTP dan kunnen de downloads worden geauthenticeerd.

Als tweede is het wenselijk dat een sensor, voordat hij 'gebruik kan maken' van de (centrale) honeypot zich moet authenticeren. Dit kan gebeuren op basis van het gegenereerde SSL certificaat. Alleen sensoren met certificaten die door de centrale server vertrouwd worden, mogen een OpenVPN verbinding opzetten. Om dit mogelijk te maken moet het door de sensor gegenereerde certificaat ondertekend worden door een voor de centrale server vertrouwde partij. Dit is in de testopstelling al gerealiseerd (zie ook bijlage C.2) maar het ondertekenen van het door de sensor gegenereerde certificaat vind nog handmatig plaats. Dit ondertekenen zou geautomatiseerd kunnen worden, maar dit heeft als nadeel dat alsnog iedereen gebruik kan maken van de honeypot. Het zou daarom het beste zijn dat het door de sensor gegenereerde certificaat pas ondertekend wordt als de instelling hier toestemming voor geeft. Dit toestemming geven kan bijvoorbeeld op een website plaatsvinden waar de contactpersoon van de instelling na het plaatsen van de sensor in kan loggen. Na deze toestemming wordt het certificaat ondertekend en kan het door de sensor worden opgehaald. Hierna kan de sensor een OpenVPN verbinding opzetten en gebruik maken van de honeypot.

Ten derde moet er nog onderzocht worden hoe het broadcast verkeer (bijvoorbeeld ARP) kan worden afgevangen zodat broadcast verkeer niet meer via alle tunnels (en dus naar alle op de sensoren aangesloten netwerken) wordt gestuurd, maar alleen naar de betreffende sensor.

Verder is er nog weinig tijd aan het ontwerp van de centrale analyse besteed. Het is mogelijk door de gebruikte tunnel techniek om centraal een honeypot te draaien, maar de analyse hiervan moet nog verder uitgewerkt worden. Punten die hierbij meegenomen kunnen worden zijn:

1. Het controleren of de sensor het nog doet (op de centrale server).
2. Het kunnen verwijderen van informatie van een bepaalde sensor. Bijvoorbeeld omdat er teveel informatie van één sensor is.

Geraadpleegde bronnen

1. OpenVPN - http://www.openvpn.net/
2. Het secure internet protocol (IPsec) - http://www.surfnet.nl/innovatie/ipsec/secureip/
3. Types of network tunnels - http://www.netheaven.com/TunnelTypes.html
4. De Linux kernel - http://kernel.org/
5. Overzicht van Linux kernelbugs - http://bugzilla.kernel.org/
6. Beschrijving bug in TCPdump - http://www.securiteam.com/exploits/5GP012KG0S.html
7. Damn Small Linux (DSL) - http://www.damnsmalllinux.org/
8. Busybox - http://www.busybox.net/
9. Generic Routing Encapsulation (GRE) - http://www.faqs.org/rfcs/rfc2784.html
10. Knoppix remaster - http://www.knoppix.net/wiki/Knoppix_Remastering_Howto
11. OpenVPN in bridge mode - http://openvpn.net/bridge.html
12. Distributed Intrusion Detection platform - http://www.os3.nl/2003-2004/ANP/reports/JvB+TN-distributed-intrusion-detection-platform.pdf

A. Bouw eigen Live CD

In deze bijlage staat uitgelegd hoe we zelf een distributie (een Live CD) hebben gebouwd. Op het systeem waarop we deze distributie bouwen hebben we Debian Sarge geïnstalleerd (kernel 2.6.8-2-386). We hebben hier drie partitie's op aangemaakt.

- [hda1] 1.2 GB swap partitie.
- [hda2] 8 GB /remaster partitie.
- [hda3] 10.8 GB / partitie.

A.1 Basis voor de distributie

Als basis voor de eigen distributie hebben we gebruik gemaakt van DSL [7] en van BusyBox [8].

A.2 Stap 1 - de basis voor de Live CD

Als eerste maken we de basis aan voor de Live CD. Het samenstellen van de CD doen we op de `/remaster` partitie. De basis voor de Live CD bestaat uit een een bootloader (isolinux) en een kernel. Hiervoor gebruiken we de bootloader en de kernel van DSL (versie 1.2).

```
mkdir -p /remaster/newcd
mount /dev/cdrom
cp -a /cdrom/boot /remaster/newcd
```

A.3 Stap 2 - het filesysteem en de applicaties

De volgende stap is om een image te maken van het filesysteem met daarin alle benodigde applicaties. BusyBox is een compacte set applicaties (in GNU stijl) die we in dit filesysteem gaan gebruiken. Met BusyBox worden de meeste benodigde applicaties meegeleverd. Eventuele extra applicaties zullen handmatig gecompileerd moeten worden. De sensorsoftware die we voorlopig gaan gebruiken, TCPdump, is een voorbeeld van een applicatie die niet in BusyBox aanwezig is. Het samenstellen van het filesysteem met de applicaties voor het image doen we in `/remaster/source`. We beginnen met het compileren van BusyBox (versie 1.00).

```
mkdir -p /remaster/source
mkdir -p /remaster/software/busybox
wget http://www.busybox.net/downloads/busybox-1.00.tar.gz
make menuconfig
```

Via `menuconfig` bepalen we welke applicaties en op wat voor manier we BusyBox compileren. Dit wordt vervolgens opgeslagen in `.config`. De totale `.config` is hieronder weergegeven. Belangrijke punten hierin zijn:

- Support voor devfs.
- Compileren als static binary.
- De installatie directory specificeren (`/remaster/source`).

```
# /remaster/software/busybox/busybox-1.00/.config
HAVE_DOT_CONFIG=y
CONFIG_FEATURE_BUFFERS_GO_ON_STACK=y
CONFIG_FEATURE_VERBOSE_USAGE=y
CONFIG_FEATURE_DEVFS=y
CONFIG_FEATURE_DEVPTS=y
CONFIG_STATIC=y
EXTRA_CFLAGS_OPTIONS=""
PREFIX="/remaster/source"
CONFIG_CP=y
CONFIG_EXPR=y
CONFIG_LN=y
CONFIG_LS=y
```

```
CONFIG_FEATURE_LS_FILETYPES=y
CONFIG_FEATURE_LS_FOLLOWLINKS=y
CONFIG_FEATURE_LS_RECURSIVE=y
CONFIG_FEATURE_LS_SORTFILES=y
CONFIG_FEATURE_LS_TIMESTAMPS=y
CONFIG_FEATURE_LS_USERNAME=y
CONFIG_FEATURE_LS_COLOR=y
CONFIG_MKDIR=y
CONFIG_RM=y
CONFIG_TEST=y
CONFIG_TTY=y
CONFIG_FEATURE_AUTOWIDTH=y
CONFIG_FEATURE_HUMAN_READABLE=y
CONFIG_AWK=y
CONFIG_FEATURE_AWK_MATH=y
CONFIG_SED=y
CONFIG_GREP=y
CONFIG_FEATURE_GREP_EGREP_ALIAS=y
CONFIG_FEATURE_GREP_FGREP_ALIAS=y
CONFIG_FEATURE_GREP_CONTEXT=y
CONFIG_INIT=y
CONFIG_FEATURE_USE_INITTAB=y
CONFIG_FEATURE_INITRD=y
CONFIG_FEATURE_EXTRA_QUIET=y
CONFIG_HALT=y
CONFIG_POWEROFF=y
CONFIG_REBOOT=y
CONFIG_FEATURE_IPV6=y
CONFIG_HOSTNAME=y
CONFIG_IFCONFIG=y
CONFIG_FEATURE_IFCONFIG_STATUS=y
CONFIG_PING=y
CONFIG_FEATURE_FANCY_PING=y
CONFIG_PING6=y
CONFIG_FEATURE_FANCY_PING6=y
CONFIG_ROUTE=y
CONFIG_WGET=y
CONFIG_FEATURE_WGET_STATUSBAR=y
CONFIG_FEATURE_WGET_AUTHENTICATION=y
CONFIG_FREE=y
CONFIG_KILL=y
CONFIG_KILLALL=y
CONFIG_PIDOF=y
CONFIG_PS=y
CONFIG_TOP=y
FEATURE_CPU_USAGE_PERCENTAGE=y
CONFIG_UPTIME=y
CONFIG_FEATURE_SH_IS_ASH=y
CONFIG_ASH=y
CONFIG_ASH_JOB_CONTROL=y
CONFIG_ASH_ALIAS=y
CONFIG_ASH_MATH_SUPPORT=y
CONFIG_ASH_MATH_SUPPORT_64=y
CONFIG_ASH_OPTIMIZE_FOR_SIZE=y
CONFIG_FEATURE_COMMAND_EDITING=y
CONFIG_FEATURE_COMMAND_HISTORY=15
CONFIG_FEATURE_COMMAND_SAVEHISTORY=y
CONFIG_FEATURE_COMMAND_TAB_COMPLETION=y
CONFIG_FEATURE_SH_FANCY_PROMPT=y
CONFIG_SYSLOGD=y
CONFIG_FEATURE_ROTATE_LOGFILE=y
CONFIG_KLOGD=y
```

```
CONFIG_LOGGER=y
CONFIG_PIVOT_ROOT=y
CONFIG_MOUNT=y
CONFIG_UMOUNT=y
CONFIG_FEATURE_MOUNT_LOOP=y
```

Hierna kunnen we BusyBox compileren en installeren.

```
make dep
make
make install
```

In eerste instantie hadden we ervoor gekozen om TCPdump te gebruiken als sensorsoftware. TCPdump is afhankelijk van libpcap. Beide applicaties moeten dus toegevoegd worden op de CD. We hebben TCPdump en libpcap statisch gecompileerd zodat er niet nog diverse libraries aan de CD toegevoegd hoefden te worden. We hebben TCPdump en libpcap zelf uit source gecompileerd.

```
mkdir -p /remaster/software/tcpdump
cd /remaster/software/tcpdump
wget http://www.tcpdump.org/release/libpcap-0.8.3.tar.gz
wget http://www.tcpdump.org/release/tcpdump-3.8.3.tar.gz

cd libpcap-0.8.3
env CFLAGS=-static ./configure --prefix=/remaster/source/usr/local
make
make install

cd tcpdump-3.8.3
env CFLAGS=-static ./configure --prefix=/remaster/source/usr/local
make
make install
```

Er moeten nog een aantal bestanden en directories aan het filesysteem toegevoegd worden voordat het compleet is. Als eerste voegen we een `inittab` toe, deze `inittab` is gebaseerd op de bij BusyBox geleverde `inittab`.

```
mkdir -p /remaster/source/etc
  vi /remaster/source/etc/inittab
```

```
# SURFnet IDS /etc/inittab
  ::sysinit:/etc/init.d/rcS
  ::askfirst:-/bin/sh
  ::restart:/sbin/init
  ::ctrlaltdel:/sbin/reboot
  ::shutdown:/bin/umount -a -r
  ::shutdown:/sbin/swapoff -a
```

Vervolgens booten we naar DSL omdat we daar een aantal files uit nodig hebben. Vanuit DSL kopiëren we de kernel en een aantal devices.

```
sudo su
mount -rw /dev/hda2 /mnt # Mount /remaster op /mnt
mkdir -p /mnt/source/boot
cp -a /boot/* /mnt/source/boot/
```

```
mkdir -p /mnt/source/dev
cp -a /dev/null /mnt/source/dev/
cp -a /dev/console /mnt/source/dev/
```

A.4 Stap 3 - de image en het ISO maken

Als het filesysteem en de applicaties gereed zijn is het tijd om het image en de ISO te maken. Allereerst maken we het image.

```
mkisofs -R busybox_source | create_compressed_fs – 65536 >
newcd/KNOPPIX/KNOPPIX
```

Daarna maken we de ISO voor de CD.

```
mkisofs -no-pad -l -r -J -no-emul-boot -boot-load-size 4 -boot-info-table \
-b boot/isolinux/isolinux.bin -c boot/isolinux/boot.cat -hide-rr-moved \
-o SURFnetSensor-1.0.iso newcd
```

A.5 Stap 4 - testen

De laatste stap is natuurlijk het testen van de CD. Punten die nog niet werken:

- In de kernel zit geen ondersteuning voor libpcap.
- De opstartscripts voor o.a. het netwerk zijn nog niet geladen. De binaries die nodig zijn om het netwerk op te starten zijn ook nog niet aan de CD toegevoegd.

B. Knoppix remaster

Voor het remasteren van de Knoppix CD gebruiken we een Debian Sarge systeem, kernel 2.6.8-2-386. We hebben hier drie partitie's op aangemaakt.

- [hda1] 1.2 GB swap partitie.
- [hda2] 8 GB /remaster partitie.
- [hda3] 10.8 GB / partitie.

We hebben voor een 1.2 GigaByte grote swap gekozen omdat dit aangeraden werd in de Knoppix Remaster handleiding. Verder is er besloten een aparte partitie aan te maken voor het remasterproces. De knoppix remaster is aan de hand van een howto [10] gedaan. Hierbij is er geprobeert de packages die standaard bij Knoppix worden meegeleverd zoveel mogelijk eruit te halen. Alleen het broodnodige is overgelaten. Hierdoor krijg je een mooie kleine versie van Knoppix. Een aantal keuzes die we bij het maken van de remaster hebben gemaakt:

- Geen ondersteuning voor audio.
- Geen ondersteuning voor grafische omgevingen.
- Geen ondersteuning voor muis.
- Geen ondersteuning voor wireless.
- Geen ondersteuning voor pcmcia.
- Alleen de bash shell is aanwezig.

De overgebleven packages bieden voornamelijk ondersteuning voor verschillende hardware. Daarnaast zijn er nog een aantal packages overgebleven met tools die we mogelijkerwijs later nog kunnen gebruiken bij het opzetten van de sensor software en de tunnel naar de server. Uiteindelijk kan de Knoppix remaster nog kleiner worden wanneer de overige tools die niet gebruikt zijn ook weggehaald worden.

Een alternatieve manier om packages snel te verwijderen is met behulp van de volgende commando's. Het eerste commando maakt een lijst van alle packages die geïnstalleerd zijn en stopt ze in selectie.txt. Vervolgens kan het bestandje selectie.txt aangepast worden en de packages veranderd worden indien nodig.

```
dpkg --get-selections > selectie.txt
```

Om de veranderingen in selectie.txt door te voeren kan het volgende commando uitgevoerd worden:

```
cat selectie.txt | dpkg --set-selections
dpkg -a
```

Na deze commando's worden alle veranderingen uit het bestand selectie.txt uitgevoerd.

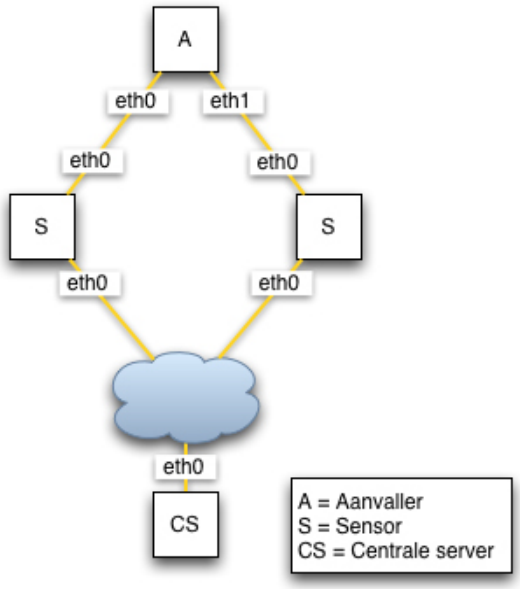
Aan de hand van ervaringen opgedaan tijdens het remaster proces van Knoppix is gebleken dat het remasteren van Knoppix geen goed uitgangspunt is om te gebruiken voor de sensor. De standaard Knoppix livecd is veel te uitgebreid en heeft allerlei onnodige applicaties. Daarentegen heeft het wel veel ondersteuning voor verschillende hardware en protocollen, zoals wireless, pcmcia, ppp, netbooting, etc. Het enige voordeel van de Knoppix distributie is het feit dat er goed is nagedacht over het kernel image en de boot structuur. Dit feit kan als leidraad gebruikt worden voor het maken van een eigen kernel voor het sensor OS. De keuze voor een Knoppix remaster als sensor distributie zou alleen nuttig zijn wanneer er een grote verscheidenheid aan software, protocollen en hardware ondersteund moet worden.

C. Bouw proof-of-concept opstelling

Voor de bouw van de proof-of-concept hebben we vier Debian (kernel 2.6.8-2-386) machines gebruikt. Hierop installeren we OpenSSL, OpenVPN, bridgeutils en TCPdump. Dit laatste pakket is niet noodzakelijk voor de definitieve opstelling maar maakt het controleren van het netwerk mogelijk.

```
apt-get install openssl
apt-get install openvpn
apt-get install bridge-utils
apt-get install tcpdump
```

Vervolgens hebben we alle netwerken aangesloten. Normaal doe je dit met crosskabels maar je kunt hier ook switches voor gebruiken. Voorwaarde is wel dat elke verbinding zijn eigen switch heeft. De aanvaller (met twee netwerkkaarten) verbinden we direct met de twee sensoren. De sensoren verbinden we via een switch met de centrale server. De aanvaller geven we alvast een IP adres, `eth0` krijgt 10.5.0.1, `eth1` krijgt 10.1.0.1. Het aansluitschema staat hieronder weergegeven.



C.1. De centrale server

Vervolgens configureren we de server als OpenVPN server. Details over het installatie proces van de OpenVPN server staan op de website van OpenVPN [1]. We hebben de Ethernet Bridge [11] handleiding als uitgangspunt gebruikt. Welke stappen we daaruit gevolgd hebben staan in deze bijlage, eventueel wordt hierin een stap toegelicht. Allereerst moeten op de OpenVPN server certificaten aangemaakt worden. Standaard worden er met OpenVPN hier een aantal scripts voor meegeleverd. Deze staan in de map `/usr/share/doc/openvpn/examples/easy-rsa`. De stappen die we volgen zijn:

1. De `vars` file aanpassen en uitvoeren.
2. Het aanmaken van een CA.
3. Het server certificaat aanmaken.
4. De Diffie-Helman parameters aanmaken.

```
# /usr/share/doc/openvpn/examples/easy-rsa/vars
export D=`pwd`
export KEY_CONFIG=$D/openssl.cnf
export KEY_DIR=$D/keys
export KEY_SIZE=1024
export KEY_COUNTRY=NL
```

```
export KEY_PROVINCE=UT
export KEY_CITY=Utrecht
export KEY_ORG="SURFnet"
export KEY_EMAIL="mail@mailhost.nl"
```

```
cd /usr/share/doc/openvpn/examples/easy-rsa/
. ./vars
./build-ca
./build-key-server server
./build-dh
```

Vervolgens maken we de OpenVPN server configuratie aan en kopiéren we de benodigde bestanden naar `/etc/openvpn`.

```
cp /usr/share/doc/openvpn/examples/easy-rsa/keys/ca.crt /etc/openvpn/
cp /usr/share/doc/openvpn/examples/easy-rsa/keys/dh1024.pem /etc/openvpn/
cp /usr/share/doc/openvpn/examples/easy-rsa/keys/server.crt /etc/openvpn/
cp /usr/share/doc/openvpn/examples/easy-rsa/keys/server.csr /etc/openvpn/
cp /usr/share/doc/openvpn/examples/easy-rsa/keys/server.key /etc/openvpn/
```

De volgende stap is om de server configuratie aan te maken. Deze configuratie slaan we op als `/etc/openvpn/server.conf`.

```
# /etc/openvpn/server.conf
port 1194
proto udp
dev tap0
ca ca.crt
cert server.crt
key server.key
dh dh1024.pem
server-bridge 192.168.8.4 255.255.255.0 192.168.8.128 192.168.8.200
ifconfig-pool-persist ipp.txt
keepalive 10 120
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
status openvpn-status.log
verb 3
```

Voordat we de OpenVPN server kunnen starten moeten we nog een `tap0` device aanmaken en moeten we het IP adres van de server (`eth0`) nog instellen. Om dit te doen hebben we een klein scriptje geschreven (`/etc/openvpn/scripts/tap-start`.

```
#!/bin/bash

###################################
# /etc/openvpn/scripts/tap-start
# Set up tap device on Linux
# Kees en HJ
###################################

tap="tap0"

tap_ip="192.168.8.4"
tap_netmask="255.255.255.0"
tap_broadcast="192.168.8.255"

eth="eth0"
eth_ip="10.2.0.3"
```

```
eth_netmask="255.255.255.0"
eth_broadcast="10.2.0.255"

openvpn --mktun --dev $tap

ifconfig $tap $tap_ip netmask $tap_netmask broadcast $tap_broadcast
ifconfig $eth $eth_ip netmask $eth_netmask broadcast $eth_broadcast
```

C.2. De sensoren

Het enige wat op de sensor moet gebeuren is het aanmaken van een certificaat en het aanmaken van de OpenVPN configuratie. Voor het maken van het certificaat hebben we en script gemaakt gebaseerd op de bij OpenVPN geleverde scripts. Omdat het certificaat door de server gesigneerd moet worden wordt het certificaat naar de centrale server gekopieerd. Als de server het certificaat gesigneerd heeft wordt dit weer opgehaald door de sensor. Dit laatste moet nog geautomatiseerd worden op de server. Het script voor het aanmaken van het certificaat slaan we op in `/etc/sids/generate_certificate.sh`. Dit script heeft nog een speciale `openssl.cnf` nodig voor het aanmaken van het certificaat, die we opslaan als `/etc/sids/sids_certificate.conf`.

```
#!/bin/sh
#
# /etc/sids/generate_certificate.sh
#
# Script om het sensor certificaat te maken voor de OpenVPN tunnel
# Gebaseerd op de bij OpenVPN geleverde scripten en configuraties
#

# Unieke key naam
KEY_NAME=`date +%s`

# Waar is de OpenSSL configuratie
export KEY_CONFIG=/etc/sids/sids_certificate.conf
export KEY_DIR=/etc/openvpn
export KEY_SIZE=1024

export KEY_COUNTRY=NL
export KEY_PROVINCE=NH
export KEY_CITY=Utrecht
export KEY_ORG="SURFnet IDS"
export KEY_EMAIL="hjblok@os3.nl"
export KEY_COMMONNAME=$KEY_NAME
export KEY_UNITNAME="SURFnet IDS"

openssl req -days 3650 -nodes -new -keyout /etc/openvpn/sensor.key -out
/etc/openvpn/sensor.csr -config /etc/sids/sids_certificate.conf -batch
chmod 0600 /etc/openvpn/sensor.key

scp /etc/openvpn/sensor.csr kees@10.2.0.3:~/$KEY_NAME.csr

while [ ! -e "/etc/openvpn/sensor.crt" ]; do
  scp kees@10.2.0.3:~/$KEY_NAME.crt /etc/openvpn/sensor.crt
  sleep 30
done
```

```
# /etc/sids/sids_certificate.conf
HOME                    = .
RANDFILE            = $ENV::HOME/.rnd
oid_section             = new_oids
[ new_oids ]
[ ca ]
default_ca = CA_default         # The default ca section
```

```
[ CA_default ]
dir         = $ENV::KEY_DIR         # Where everything is kept
certs         = $dir                 # Where the issued certs are kept
crl_dir         = $dir                 # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir                 # default place for new certs.
certificate = $dir/ca.crt          # The CA certificate
serial         = $dir/serial                 # The current serial number
crl         = $dir/crl.pem                 # The current CRL
private_key = $dir/ca.key          # The private key
RANDFILE = $dir/.rand         # private random number file
x509_extensions = usr_cert         # The extentions to add to the cert
default_days = 3650                 # how long to certify for
default_crl_days= 30                 # how long before next CRL
default_md = md5                  # which md to use.
preserve = no                  # keep passed DN ordering
policy         = policy_match
[ policy_match ]
countryName         = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName          = supplied
emailAddress          = optional
[ policy_anything ]
countryName         = optional
stateOrProvinceName = optional
localityName          = optional
organizationName = optional
organizationalUnitName = optional
commonName          = supplied
emailAddress          = optional
[ req ]
default_bits          = $ENV::KEY_SIZE
default_keyfile         = privkey.pem
distinguished_name = req_distinguished_name
x509_extensions = v3_ca # The extentions to add to the self signed cert
string_mask = nombstr
[ req_distinguished_name ]
countryName                 = Country Name (2 letter code)
countryName_default         = $ENV::KEY_COUNTRY
countryName_min                 = 2
countryName_max                 = 2
stateOrProvinceName         = State or Province Name (full name)
stateOrProvinceName_default = $ENV::KEY_PROVINCE
localityName                 = Locality Name (eg, city)
localityName_default         = $ENV::KEY_CITY
0.organizationName         = Organization Name (eg, company)
0.organizationName_default = $ENV::KEY_ORG
organizationalUnitName         = Organizational Unit Name (eg, section)
organizationalUnitName_default = $ENV::KEY_UNITNAME
commonName                  = Common Name (eg, your name or your server\'s
hostname)
commonName_default          = $ENV::KEY_COMMONNAME
commonName_max                  = 64
emailAddress                  = Email Address
emailAddress_default          = $ENV::KEY_EMAIL
emailAddress_max         = 40
[ req_attributes ]
unstructuredName          = An optional company name
[ usr_cert ]
basicConstraints=CA:FALSE
```

```
nsComment                  = "OpenSSL Generated Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
[ server ]
basicConstraints=CA:FALSE
nsCertType                  = server
nsComment                  = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
[ v3_ca ]
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
basicConstraints = CA:true
[ crl_ext ]
authorityKeyIdentifier=keyid:always,issuer:always
```

```
/etc/sids/generate_certificate.sh
```

Vervolgens maken we de OpenVPN configuratie aan die we opslaan als `/etc/openvpn/client.conf`.

```
# /etc/openvpn/client.conf
client
dev tap
proto udp
remote 10.2.0.3 1194
resolv-retry infinite
nobind
user nobody
group nogroup
persist-key
persist-tun
ca ca.crt
cert sensor.crt
key sensor.key
up ./bridge_add_tap.sh
comp-lzo
verb 3
```

Voor het toevoegen van `tap0` aan de bridge na het opzetten van de tunnel hebben we een script geschreven (`/etc/openvpn/bridge_add_tap.sh`).

```
#!/bin/sh
# /etc/openvpn/bridge_add_tap.sh

brctl addif br0 tap0
```

NB: niet alle files staan direct op de sensor. Van de server hebben we ook nog het CA certificaat nodig (genaamd `ca.crt`).