

Project Information Security:

Scapy + Rpi Packet Generator

Inhoud

[1. Opgave 3](#_Toc440585698)

[2. Gebruikte technologie 4](#_Toc440585699)

[3. Webpage 4](#_Toc440585700)

[3.1. PHP code 5](#_Toc440585701)

[3.2. Tekst file 6](#_Toc440585702)

[3.3. Python script 6](#_Toc440585703)

[3.4. Rasberry Pi 7](#_Toc440585704)

[3.4.1. Netwerk adapter 7](#_Toc440585705)

[3.4.2. Ubuntu mate 7](#_Toc440585706)

[3.4.3. Instaleren van LAMPP 8](#_Toc440585707)

[3.5. Virtualbox 8](#_Toc440585708)

[4. Scapy 9](#_Toc440585709)

[4.1. Scapy in ons project 9](#_Toc440585710)

[4.2. Ping script 10](#_Toc440585711)

[4.3. Conclusie 11](#_Toc440585712)

# Opgave

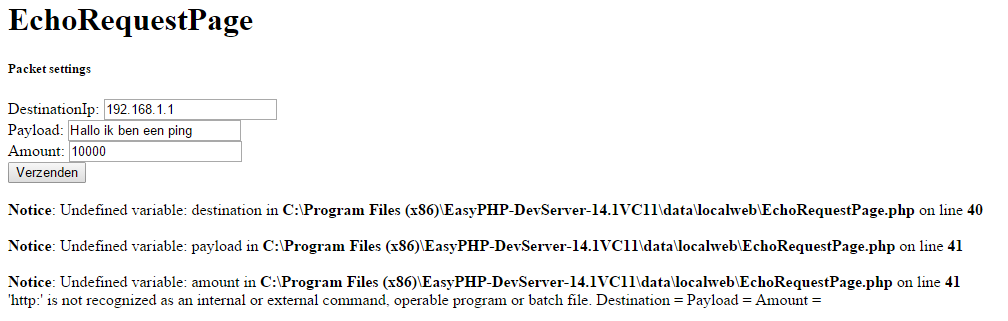
De opgave van dit project was om te leren omgaan met scapy, een zeer handige library op python die het gemakkelijk maakt om een internet-pakketje op te stellen en te versturen. Het was de bedoeling om een ui te maken waarin de gebruiker kan kiezen uit verschillende soorten pakketjes die hij wenst op te stellen en te versturen. Vervolgens worden er aan de gebruiker enkele parameters gevraagd die essentieel zijn voor het opbouwen en afhandelen van deze pakketjes. Zo wordt de gebruiker afgeschermd van alle rare/ingewikkelde code achter het pakketje, maar krijg hij toch resultaat te zien. Het uiteindelijk doel is dat deze handige tool zodanig uitgebouwd kan worden dat docenten op deze wijze enkele simpele of ingewikkelde pakketjes kunnen sturen binnen een labo omgeving, om zo een test-omgeving met internettrafiek te genereren waarin studenten met wireshark kunnen werken om bepaalde pakketjes te zoeken, uitlezen of andere dingen.

# Gebruikte technologie

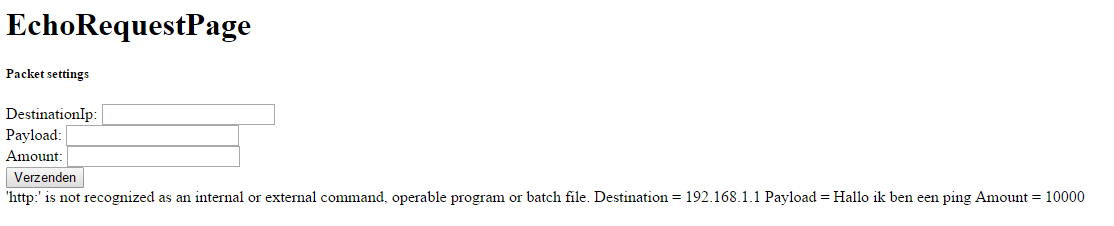
* Voor de webpage:
  + Html
  + PHP
* Voor het genereren van de pakketten:
  + Python (Scapy)
  + Rasperry pi

# Webpage

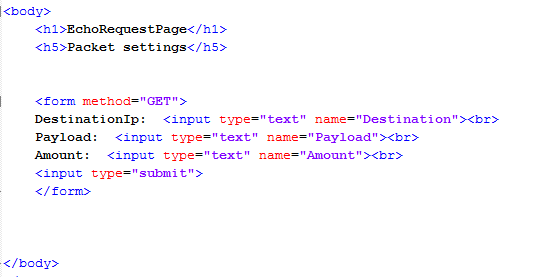
We hebben een simpele web pagina gemaakt. We hebben onze focus gelegd op het meest simpele protocol ICMP(internet control message protocol) echo request.



Zoals te zien op de afbeelding hebben we de webpage echt tot het minimum en simpel gehouden. Het enige wat de gebruiker moet invullen is het destinationip adres, de payload en de hoeveelheid van ping dat de gebruiker wilt sturen naar het ip adres naar keuze.



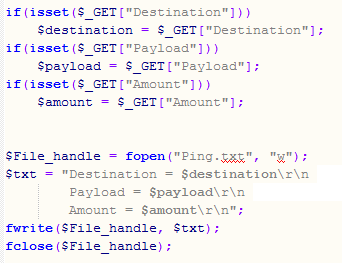
Om de verzonden data te kunnen nakijken of deze correct wordt weg geschreven laat ik via een php functie deze terug laten zien. Zoals te zien op de afbeelding word de data correct weggeschreven naar de text file.



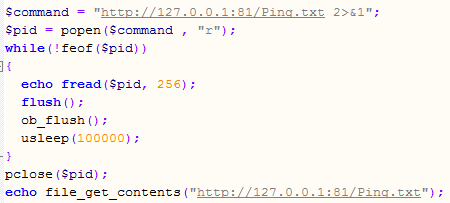
We hebben een simpele form gebruikt voor de ingegeven data aan de PHP code door te geven. Het door geven van de gebeurt met een "GET" methode.

## PHP code

Het eerste plan was om de data ineens in een python script te schrijven en deze dan te gebruiken maar het probleem was hierbij dat elke keer de file overschreven wordt en het script terug geschreven moest worden. Dus hebben we gekozen om een tekst file als middle man te gebruiken, dus eerst wordt de data naar de tekst file geschreven en opgeslagen en dan gaan we met een python script deze tekst file terug uit lezen voor dan te gebruiken met scapy.

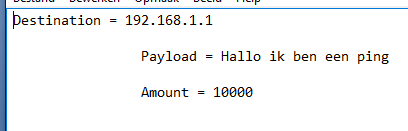


Dit is de PHP code die de verzonden data wegschrijft naar de tekst file. Eerst gaan we kijken met de isset functie of de variabelen effectief zijn ingevuld. Dan maak ik een variabele $file\_handle aan en open de tekst file "Ping.txt" met " fopen" als deze niet bestaat zal deze van zelf aangemaakt worden. De content van de file wordt ingevuld in de variabelen $txt. Het schrijven naar de file wordt gedaan door de Fwrite functie te gebruiken en dan file\_handle en de txt variabelen hier met mee te geven en dan het schrijven naar de file te sluiten met de fclose functie.



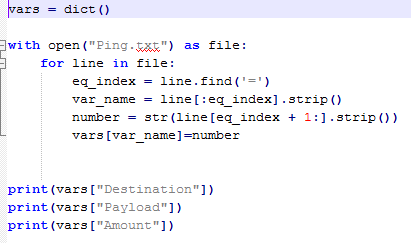
Met deze functie laat ik de content van de tekst file terug zien op de webpage. ik maak de variable $command met hier in de locatie van de tekst file in dit geval is het de localhost. Deze functie ga checken of de file er wel is en checken naar de content er van. Dan met de echo functie wordt de gevonden content in de tekst file weergegeven op de webpage.

## Tekst file



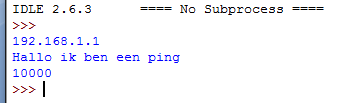
Dit is de inhoud van de tekst na dat we deze hebben gegenereerd met de PHP code. Dit bewijst dat de data correct wordt weg geschreven.

## Python script



Om de tekst file terug te kunnen uitlezen met een python script. Dit gebeurt door de tekst file lijn per lijn uit te lezen. Eerst wordt de tekst file opgeroepen en dan wordt er met een for loop elke lijn gelezen om te zoeken naar een "=" teken als hij deze gevonden heeft gaat hij de rest van de zin weg doen zo dat alleen wat achter het "=" staat overblijft. Dan om te kijken of alles correct wordt weergeven print ik elke variabele terug af. Omdat de gevonden tekst in de array van "vars" wordt gezet kunnen deze op de bovenstaande manier worden afgeprint

Resultaat van de print:



Voor het testen van het python script hebben we IDLE gebruikt. IDLE is een tool die mee geïnstalleerd wordt bij de installatie van python. Zoals te zien op de afbeelding worden de ingegeven waarden van de webpage correct uitgelezen.

## Rasberry Pi

Dus voor dit project was het de bedoeling dat de webpage gehost zou worden op de rasberry pi. We hebben er voor gekozen voor kali linux als OS te kiezen voor op de rasberry pi te zetten. Dit heeft als reden dat Kali linux scapy standaard ondersteunt. We hebben veel problemen onder vonden voor een OS stabiel te laten lopen op onze rasberry pi, hier van als oorzaak is een slecht micro SD kaart.

Er zijn verschillende problemen opgedoken toen we met de rasberryaant werken waren zoals:

### Netwerk adapter

We hebben een netwerk adapter gekocht maar geraakten we niet met op het netwerk op de campus omdat de beveiliging WPA-2 enterprice is en deze dit niet toelaat. Hier is wel een oplossing voor door handmatig in de command line dit te verhelpen. Dit hebben we meerdere malen geprobeerd maar dit zonder succes. Dit probleem komt niet voor op een thuis netwerk door dat de beveiliging van het netwerk anders. Dit maal de adapter verbonden met de rasberry maar de adapter weigerde deze dienst. Daarom hebben we dit geprobeerd met een andere netwerkadapter TP-Link WN821N hier lukte het wel voor met het thuis netwerk te verbinden.

### Ubuntu mate

Omdat Kali linux een vrij zware OS is voor een rasberry pi hebben voor een andere OS gekozen namelijk ubuntu mate maar deze is meerdere malen gecrash waarschijnlijk door het slechte micro SD kaartje. Zelf bij het branden van de image naar de sd kaart kregen we meerder foutmeldingen.

Door dat we de deadline naardere hebben er voor gekozen om de rasberry pi links te laten liggen en verder te werken op een virtual machine gehost op virtualbox.

### Instaleren van LAMPP

Dit was het grootste probleem dat we onder vonden hebben met het werken aan dit project. In principe om een LAMPP op te zetten in een linux omgeving is niet moeilijk maar als we dit aan proberen waren op de rasberry bleekt dit toch een probleem te zijn door dat de image die op de SD kaart gebrand was maar 4gig groot is een de OS zelf is 3.8 dit laat iets meer dan 200mb over om de modules van de LAMPP te installeren dit bleek niet voldoende te zijn. We hebben geprobeerd het geheugen te vergroten omdat de gebruikte sd kaart 32gig groot is maar dit was zonder succes.

# Scapy

Scapy is een heel machtig interactief programma om op zelfstandige basis een volwaardig internet pakketje aan te maken en deze te versturen. Scapy is een library gebouwd op python, wat het zeer eenvoudig van gebruik maakt omdat men geen Type checking doet. Dit maakt het eenvoudiger om met verschillende variabelen te werken, die elks een verschillend doel/type hebben.

Het concept van scapy is zeer eenvoudig uit te leggen voor iedereen die over een basiskennis van het OSI-model beschikt. Het is namelijk mogelijk om op zelfstandige basis een pakketje op te stellen volgens elke laag van dit model, gaande van de “payload” in de Application-layer tot zelf het kiezen van hoe men het signaal verstuurd op fysiek niveau in de Physical-Layer van het OSI-model, elk klein detail kan men manipuleren. Binnen deze lagen kan de programmeur dan ook elke mogelijke eigenschap of parameter aanpassen naar wens. Dit geeft de programmeur dan ook ontzettend veel macht in het internet gebeuren.

Scapy kan gebruikt worden om heel eenvoudige dingen te maken zoals een simpel ping-pakketje, maar voor zij die zich verdiepen kent scapy geen grenzen. Als men weet welke parameters/eigenschappen van bepaalde protocollen moet veranderen in een bepaalde laag van dit OSI-model, is het mogelijk om extreem vieze dingen te gaan doen.

Voor zij die graag weten of zelf bepalen hoe hun “attack” of “tool” zijn werk doet is dit zeer interessant omdat men dit hier simpelweg allemaal gewoon zelf kan instellen. En voor de mensen die graag een cadeautjes achterlaten is het ook mogelijk om hun “tag” in hun pakketjes te steken.

Waar het op neer komt is *“Alles kan! (maar niet alles mag)”* zolang men weet wat men wil en een beetje verstand van de opbouw van een pakketje heeft, kan men doen wat men wil, gaande van ping-pakketjes sturen tot TCP-connecties, ARP-tools, Sniffing scriptjes schrijven, MITM-attacks, en zoveel meer, de mogelijkheden zijn maar zo beperkt als uw kennis.

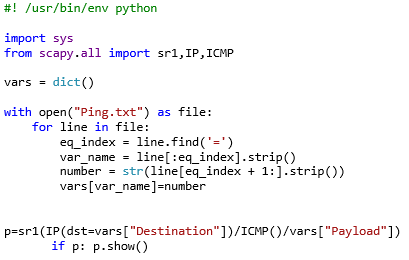
## Scapy in ons project

Wij moesten voor dit project scapy gebruiken om pakketjes op te bouwen op eenvoudige wijzen. Het oorspronkelijke idee is dat men een simpele basislayout in een webpagina toont aan een gebruiker. De gebruiker wordt gevraagd om aantal parameters in te vullen afhankelijk van het pakketje of handeling dat hij wenst te verrichten. Vervolgens als de gebruiker op ‘submit’ duwt zal er een pakketje naar zijn wensen gecreëerd en verstuurd worden. Dit maakt het makkelijker voor mensen men een beperkte kennis over internet-protocollen en pakketjes om toch bepaalde acties te verrichten.

Omdat scapy ook nieuw is voor ons hebben wij slechts één functie in een scriptje kunnen schrijven namelijk een basis ICMP\_ECHO\_REQUEST, dit is het vakjargon voor ‘ping-pakketje’. Wij hebben ook onderzoek gedaan naar andere functionaliteiten, maar die vereiste meer interactie met de gebruiker wat veel moeilijk af te handelen is in een voorgemaakt scriptje.

De opbouw van een ping pakketje in wijn meest eenvoudige vorm bestaat slechts uit 1 essentiële laag nl. de netwerk laag. Binnen deze laag gaat men gebruik maken van het Internet Protocol (IP) en dit encapsuleren in het Intern Control Message Protocol (ICMP). De enige parameters die we in principe nodig hebben voor dit pakketje op te bouwen zijn slechts: het Destination-IP and eventueel indien men wil, een payload met een vriendelijk berichtje.

## Ping script



Eerst beginnen we met de standaard shebang om python module op te roepen, vervolgens importeren we ‘sr1’, ‘IP’ en ‘ICMP’ modules van de Scapy Library. Deze hebben we nodig om het pakketje te kunnen opbouwen en versturen. Vervolgens is er een hele hoop code geschreven die, om de uitleg zo kort mogelijk te houden, ervoor gaat zorgen dat een tekst bestand, waar de parameters, die de gebruiker heeft ingegeven, zijn opgeslagen, uit te lezen.

Nu komt het echte scapy werk. Hier stellen we het pakketje samen. Eerst beginnen we met het Internet Protocol te declareren d.m.v. IP(). Binnen deze ronde haakjes kan men alle parameters van de IP-protocol veranderen, in ons geval enkel het destinationip met de code *‘dst=”ipadress”*’ en omdat wij de variabele destinationip aan de gebruiker hebben gevraagd, lezen we deze uit de tekst file m.b.v. *vars[“Destination”]*. Vervolgens gaan we een laag dieper om binnen het IP-protocol, de ICMP protocol te creëren m.b.v. *ICMP()*, standaard zal de ICMP laag een ICMP\_ECHO\_REQUEST pakketje zijn. Als extra hebben we dan ook voorzien dat de gebruiken een payload kan ingeven als vriendelijk berichte voor de persoon die de ping ontvangt.

Het *‘sr1’* commando zal ervoor zorgen dat het pakketje, dat als parameter wordt meegegeven tussen de ronde haakjes, verstuurd zal worden, en zal er ook voor zorgen dat het antwoord opgevangen wordt en deze in de variabele *‘p’*, zoals hierboven aangetoond, wordt opgeslagen. Vervolgens zal de *‘if-functie’* nagaan of de variabele ‘*p’* wel werkelijk data bevat, m.a.w. of de ping-functie correct afgehandeld is en beantwoord is. Als *‘p’* data bevat zal ‘*p.show()’* ervoor zorgen dat deze data uitgeprint wordt indien men de ping-handeling wil analyseren.

## Conclusie

Wij zelf hebben meer gespeeld met scapy binnen de terminal om het wat te verkennen en hebben veel bijgeleerd. Scapy is zeker een zeer interessante omgeving om mee in te spelen, wij hebben er alvast veel in bijgeleerd.

Als laatste voorstel lijkt het me zeer interessant om scapy te integreren in de labo lessen omtrent OSI-model in de eerste jaren, omdat dit een zeer overzichtelijke omgeving is en men in praktijk kan zien hoe elke laag werkt en wat elke laag van het OSI-model afhandelt en hoe netwerktrafiek werkt. Zo krijgen studenten de kans om een beter inzicht te krijgen over OSI-model, door er gewoonweg wat mee te spelen, ons heeft het zeker geholpen sommige zaken toch beter te begrijpen.