LiNSEP een nieuw email protocol

Dennis Brentjes

01-07-2010

Inhoudsopgave

1 Inleiding.

Voor ons onderzoek in kader van het vak Research & Development hebben wij onderzocht of het mogelijk is om een spamvrij emailprotocol te schrijven. Hierin hebben we onderzocht wat er precies moet gebeuren om dit te realiseren en wat de makkelijkste manier is om het ook daadwerkelijk te implementeren. Ook hebben we onderzocht of het nieuwe protocol niet te traag is en of dit wel echt spamvrij is. Hierover staat in de methode beschrijving, resultaten en conclusie meer uitleg. Verder staat er in het theoretisch kader nog meer uitleg en onderzoek naar rand onderwerpen die we gebruiken of aanstippen in de rest van ons verslag.

2 Theoretisch Kader.

2.1 Inherente problemen van e-mail.

Er zijn een aantal problemen met de huidige implementatie van e-mail. Dit ligt aan een aantal factoren. Sommige zijn uit te sluiten, andere niet. Een van deze factoren is de natuurlijke "slechtheid" van de mens om spam te versturen en daardoor de maatschappij op te zadelen met extra kosten en frustratie. Hier kan natuurlijk niks aan gedaan worden, maar is natuurlijk wel een idee om de spammers zelf aan te pakken in plaats van spam af te vangen. Hier gaan we het later uitgebreid over hebben.

Een tweede probleem is de mail infrastructuur oftewel de servers die de mail vesturen. Deze zijn in de meeste gevallen "relay stations". Dit wil zeggen er komt een email aan en wordt doorgestuurd naar de volgende server. Dit kan weer een relay of ontvangende server zijn. Hierdoor ontstaat een onveilige keten waarbij als een link zwak is er een grotere kans is dat de Spam mail wel aankomt bij de consument. Bovendien is dit niet erg lastig om te realiseren. Iedereen die het wil kan namelijk gewoon een SMTP (het huidige e-mail verzend protocol) server aanmaken, en als je een beetje handig bent deze aanpassen om afwijkend gedrag te vertonen. Namelijk het toelaten van gespoofte emails of aangepast headers. Dit probleem is op zichzelf ook niet op te lossen. Hiervoor zou je mensen moeten dwingen om een eerlijke server te maken, maar dit is niet te realiseren omdat SMTP protocol te goed gedocumenteerd.

Een derde probleem is de leeftijd van het protocol. Het SMTP protocol stamt uit de jaren ‘60 en is onstaan in ARPAnet (de voorganger van het internet). Het is een samenraapsel van een aantal messenging protocollen die toen der tijd door elkaar gebruikt werden om informatie te verzenden. Deze zijn dus gebundeld en eind jaren ‘80 opgenomen als officieel protocol. Het feit dat er geen spam bestond in de jaren dat het protocol tot stand kwam heeft ervoor gezorgd dat er geen faciliteiten zijn om spam te bevechten in het protocol zelf. In reactie hierop zijn er een aantal uitbreidingen gekomen in de loop van de jaren ‘90 en ‘00 (namelijk eSMTP en SMTP-auth). Deze zorgen voor betere identificatie en controle van e-mail verzenders, maar door deze "relay" structuur en "backwards compatibility" voor SMTP(1.0) gaat het nog steeds mis. Nu ook de spammers botnets gebruiken is het identificeren van de verzenders zelfs geen oplossing meer en bovendien bied het geen oplossing voor het spoofen van e-mail adressen.

In ons project proberen we dus de bovenstaande problemen het hoofd te bieden door een beter e-mail protocol te bedenken met ingebouwde spampreventie.

2.2 Huidige spam preventie.

Er worden vandaag de dag twee types spam preventie toegepast. De een is "content-based" de andere is "sender-based". De eerste is denk ik de meest bekende onder een breed publiek. Dit zijn namelijk de huidige spam filters die je zelf kan installeren of door je e-mail provider worden aangeleverd. Deze spam

filters kijken naar de inhoud en meestal sleutelwoorden om te beslissen of een bericht spam is of niet. Dit was een goede oplossing in het verleden maar met de opkomst van betere internetverbindingen en de komst van reclame e-mail met plaatjes niet in alle situaties toepasbaar. Ook is deze aanpak vervelend voor de gebruiker want deze mail komt gewoon binnen in de spam box en ik kan me dus indenken dat je nog steeds in contact komt met deze mailtjes.

De andere aanpak is "sender-based" deze methode probeert om de spam te stoppen voordat het aankomt bij de persoon waar deze naar gestuurd wordt. Hiervoor zijn een aantal aanpakken denkbaar. Zo kun je notoire spam verstuurders blacklisten zodat je hier geen spam meer van ontvangt, maar met de komst van botnets is dit niet meer van toepassing. De reden hiervoor is dat deze botnets bestaan uit duizenden Pc's van normale mensen die geïnfecteerd zijn door een bepaald virus waardoor de spammers via deze computers spam kunnen versturen. De eigenaar van deze computer is natuurlijk niet schuldig aan het versturen van spam. Verder merkt deze persoon hier ook vrij weinig van, omdat de Pc's van tegenwoordig in verhouding veel rekenkracht hebben. Met als gevolg dat je heel erg veel adressen moet blacklisten wil dit effect hebben. Bovendien als deze persoon toevallig jouw wil bereiken kan dit vervolgens niet meer en wordt deze persoon gedupeerd.

Een andere mooie "sender-based" systeem die we tegen zijn gekomen is heel erg gesofisticeerd. Hier is een klein stukje uit hun artikel.

> In particular, we believe our work offers three contributions: First, we produce a general approach for inferring e-mail generation templates in their entirety, subsuming prior work focused on particular features (e.g., mail header anomalies, subject lines, URLs). Second, we describe a concrete algorithm that can perform this inference task in near real-time (only a few seconds to generate an initial high-quality regular expression, and fractions of a second to update and refine it in response to subsequent samples), thereby making this approach feasible for deployment. Finally, we test this approach empirically against live botnet spam, demonstrate its effectiveness, and identify the requirements for practical use. [1]

wat deze aanpak in principe doet is kijken hoe verschillende spam mailtjes van hetzelfde botnetwerk op elkaar lijken. Deze worden namelijk gemaakt vanuit een template. Dit is een email met variabele velden in bijvoorbeeld het From veld of een achternaam in de body van de mail. Zodat de spammers email kunnen versturen met dezelfde inhoud maar telkens net iets anders dat de "content-based" systemen de mail niet als spam klasseren. Dit anti-spam systeem maakt dan (bijna) realtime een reguliere expressie waar deze mailtjes allemaal op matchen en dus veilig kunnen worden gefilterd. Dit allemaal met bijna geen 'false positives', dus niet-spam die als spam wordt geïdentificeerd. je moet jezelf wel afvragen of je überhaupt 'false positives' wil hebben, want stel dat je hierdoor een belangrijk mailtje mist, zoals een herrinering aan een tentamen of een kamernet reactie. Natuurlijk als dit systeem ooit geperfectioneerd wordt zou dit een van de betere

oplossingen zijn voor het probleem. Ook zeggen ze zelf dat het systeem zoals het nu is beter is als aanvulling op de andere oplossingen dan als alleen oplossing.

2.3 Onze visie op spam preventie.

De manier waarop wij de problematiek willen aanpakken is als volgt. We willen zorgen dat mail die verstuurd wordt vanaf een botnet wordt tegenhouden. We doen dit door een Captcha ("Completely Automated Public Turing-test to tell Computers and Humans Apart") te gebruiken. Hierdoor kunnen botnets niet massaal automatisch mailtjes te versturen vanaf niets vermoedenden thuis pc's. Hierdoor komt er natuurlijk een probleem met andere "goede" geautomatiseerde mail zoals nieuwsbrieven en validatie mailtjes. Hiervoor zal een server implementatie het waarschijnlijk handig vinden om een concept van black- en whitelisting toe te passen. zodat mensen (en misschien de server zelf) bepaalde afzenders, zoals de RU mail accounts, toe te laten zonder dat deze afzenders een Captcha moeten invullen. Iets wat onmogelijk was voor geautomatiseerde systemen, dit nemen we in ieder geval aan. Dit is misschien een beetje extra werk voor de gebruiker maar weegt niet op tegen de gemakken voorzien door de afname van spam in het algemeen.

Want zelfs als de Captcha's niet helemaal veilig zijn zullen de spammers meer moeite moeten doen om hun spam te verzenden. Hierdoor zullen de kosten oplopen en het gewin minder zijn, doordat het meer tijd kost om dezelfde hoeveelheid mail te versturen. Daarom zou een verdubbeling van de tijd om een mailtje te versturen door een computer (iets wat makkelijk haalbaar is door een computer tekst herkenning te laten doen) zou er al voor zorgen dat de winst marge van de spammer halveert. Ook zou de door het botnet geïnfecteerde thuiscomputer meer last ondervinden van de infectie en zal de gebruiker eerder stappen ondernemen.

2.4 Ons protocol en de internet wereld.

Om ons protocol te laten werken hebben we een transfer protocol nodig, namelijk UDP of TCP. Om hier tussen te kiezen hebben we naar de voor en nadelen gekeken. Hier is een stukje van de specificatie van TCP.

> A connection is fully specified by the pair of sockets at the ends. A local socket may participate in many connections to different foreign sockets. A connection can be used to carry data in both directions, that is, it is "full duplex". [3]

Dit wil zeggen dat de connectie blijft bestaan ook al zend je er geen data overheen en ben je dus constant bezig om die socket te bedienen. De enige manier om de connectie te verbreken is door expliciet te zeggen dat de verbinding klaar is (of natuurlijk een time out). Iets wat met veel connecties tegelijk best zwaar kan zijn voor de server. Aan de andere kant willen we wel de mail meteen afhandelen. In sommige gevallen moet de gebruiker namelijk een Captcha invullen. Als we dit met TCP doen weten we meteen waar we dit naar toe moeten

sturen, zo niet moeten we bijhouden waar de gebruiker is en dan terug sturen wat natuurlijk ook de server weer extra belast.

als 2e keus hebben we UDP. UDP is een "loose connection" protocol. Hier volgt weer een stukje uit de specificatie.

> This protocol provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed. Applications requiring ordered reliable delivery of streams of data should use the Transmission Control Protocol (TCP) [4]

Dit is een protocol wat een pakketje opstuurt naar een doel computer en niet garandeert dat deze daar aankomt en ook geen uitspraak doet over de volgorde waarin meerdere pakketjes aankomen. Dit zijn beide problemen die op te lossen zijn met een software methode, maar dit is best moeilijk en erg latency afhankelijk. Bovendie voorzien wij een probleem als de client de verbinding verbreekt voordat de captcha is aangekomen, waardoor de client in de veronderstelling is dat de email verstuurd is, maar dit echter niet het geval is. Daarom kiezen we voor het TCP protocol

3 Methode.

3.1 Is ons nieuw protocol niet een te grote last voor de gebruiker?

Als eerste moeten we vast stellen wat "een te grote last" precies is en dus de vraag beantwoorden wat vinden mensen irritant als ze de computer gebruiken. Dit is juist heel belangrijk omdat ons systeem een "Sender-based" spam preventie is en dus heel erg veel plichten leggen bij de verzenders. Verder zijn mensen ongeduldig en als ze dus iets van de computer gedaan willen hebben moet dit snel gebeuren. Dit kun je makkelijk vast te stellen door te vragen aan vrienden of familie wat ze zo irritant vinden aan de computer. Vaak krijg je een antwoord in de vorm van "het ding doet niet wat ik wil" of "ik heb 20 minuten gedaan over een print taak". Dus nemen we in alle redelijkheid aan dat *tijd* een belangrijke factor zal zijn in ons onderzoek. Natuurlijk is een van deze irritaties, spam zelf, het ding dat wij proberen op te lossen dus mogen we eigenlijk geen nieuwe irritaties toevoegen. Helaas is al gebleken dat we een vorm van Captcha's ("Completely Automated Public Turing-test to tell Computers and Humans Apart") moeten gebruiken. Deze zijn ook een bron van irritatie onder studenten en jongeren. Zij komen deze turing tests regelmatig tegen en als ze slecht gemaakt zijn kosten ze ook erg veel tijd om langs te komen.

dus ons doel hier is een spambot te maken voor ons protocol en deze vervolgens mailtjes te laten versturen en kijken hoelang deze hier gemiddeld over doet. Hiervoor moeten we wel ons paradepaardje (de Captcha) laten varen en 1 vaste Captcha meegeven in plaats van een variabele aangeleverd door een server. Omdat we mailtjes willen sturen met een geautomatiseerd systeem en geen black- of whitelist hebben geïmplementeerd.

3.2 Werkt ons protocol wel?

Wat ons betreft 2e zaak waar we naar moeten kijken, als ons protocol te onhandig is wordt deze toch niet gebruikt, is of het protocol alle spam tegen kan houden. Dit ligt aan de definitie van spam en de manieren waarop deze worden verzonden. De definitie voor spam die wij gebruiken luid als volgt "email berichten die naar veel meer mensen gestuurd wordt dan een mogelijk geïnteresseerde groep." Dit wil zeggen dat de mail verstuurd wordt door botnets en dus geautomatiseerde mail verstuurders. We moeten dus kijken of het wel onmogelijk is voor computers om een mail te versturen tenzij de ontvangende partij deze geautomatiseerde verstuurder kent en toelaat. Dit doen we door onze methode op een heel erg formele manier in natuurlijke taal te verwoorden. Hierdoor kunnen we dan een conclusie trekken.

4 Resultaten.

4.1 Stress test.

Hier zijn de resultaten van de "stress test" van onze test implementatie.

Testnummer	aantal seconden / 100 mailtjes.
1	52.43 s
2	49.88 s
3	51.79 s
4	48.72 s
5	54. 75 s
6	50.54 s
7	51.23 s
8	48.15 s
9	50.90 s
10	52.04 s

4.2 Korte bewijsvoering.

Als er een Captcha moet worden ingevuld

- De captcha kan niet door een ander persoon opgelost worden
  - De captcha kan niet doorgestuurd door zelf een email te ontvangen. Het antwoord zelf wordt namelijk niet doorgestuurd, alleen een bewijs (md5sum) dat de oplosser ook de oplossing heeft
  - Via high traffic sites bezoekers laten oplossen. Dit is echter onwaarschijnlijk/zeer weinig [5]

Als de zender op een whitelist van de gebruiker of server staat

- Dan heeft de gebruiker zelf toestemming gegeven, om alle email van deze zender in de inbox te bezorgen
- Dan is het zijn eigen domein en zegt van zich zelf dat hij geen spam verstuurd.

Als de zender op een blacklist van de gebruiker of server staat

- Dan heeft deze zender blijkbaar al eerder geprobeerd te spammen of heeft een slechte reputatie. En is er dus geen probleem.

5 Analyse

5.1 De responsie tijd van ons protocol

Als we de verwerkingstijd nemen van 100 mailtjes dus het gemiddelde van de gevonden waardes namelijk 51.04 seconden. Vervolgens delen we dit gemiddelde door 100 dit levert ons een verwerkingssnelheid op van 0.510 seconden per mailtje. Verder was ons mailtje ~1,5 MB groot namelijk een 1.4 MB Captcha en nog wat "plain text". En samen met een gemiddelde netwerk snelheid van ongeveer 3.5 MB/s levert dit ons een minimum verwerkingstijd van 0.42 seconden op. onze implementatie die dus puur en alleen data verwerkt in een bepaalde volgorde op een rijtje zet is verrassend inefficiënt. Verder zijn deze resultaten niet te vergelijken met die van SMTP omdat daar nog een encryptie overheen gaat en deze over relay stations meestal pas na 2 of 3 stappen op de plaats is waar deze moet zijn.

Als we dit toch even "off the record" vergelijken met SMTP. Deze doet er onder goede omstandigheden 2.1 seconden over. Dit geeft aan hoe veel meer SMTP doet ten opzichte van ons protocol want zelfs met "DNS lookups" en beperkte bandbreedte over het internet zou je geen verhouding van 1 staat tot 4 kunnen recht rekken. Er zijn dus die andere factoren die er voor zorgen dat ons protocol snel is maar daardoor onveilig of zelfs niet functioneel genoeg. [6]

5.2 Houden wij spam tegen?

- Als we uit gaan van de definitie Captcha's dan is het mogelijk om computers en mensen uit elkaar te houden in internetcommunicatie
- Als we van een gebruiker kunnen bepalen of deze is een computer of een mens is kunnen we computers weren uit ons systeem.
- Als we alle computers (als in computers zonder beheerders toestemning) weren uit ons systeem is het niet mogelijk om met botnets ons systeem te benaderen.
- Als geen botnets ons systeem kunnen benaderen is het niet mogelijk om spam te versturen volgens de door onze aangegeven definitie van spam (zie theoretisch kader).

6 Conclusie.

6.1 Hoe is de responsie tijd van ons protocol.

Ons onderzoek blijkt niet representatief te zijn. Als we onze resultaten erbij pakken is er geen probleem om de conclusie te trekken dat ons protocol snel genoeg is, maar er zijn een aantal problemen. ten eerste hebben we geen gebruik gemaakt van variabele Captcha's iets wat de hoofdzaak is van ons protocol. Ten tweede is er geen encryptie bij ons protocol en dit gebeurt ook niet in onze client. Iets wat nu wel gebeurt bij e-mail en dus bij ons minder tijd kost dan bij het oude manier van e-mail versturen. Ook is er niet getest in een internet omgeving maar een netwerk omgeving. Hier zijn de latency ook veel lager en zijn er geen vertragingen van bijvoorbeeld DNS servers of internet hardware van mensen thuis. Ook hebben we onze Black- en Whitelisting niet kunnen testen omdat we dan meerdere computers een client moesten laten draaien en we hebben dit thuis getest en niet op de universiteit, omdat er geen verbinding kon gemaakt worden op het universiteitsnetwerk.

Hierdoor kunnen we dus geen conclusies trekken over de last voor de gebruiker en dus verzender van dit protocol. We verwachten dat, de oplos tijd van de Captcha niet meegerekend, de last voor de gebruiker best wel meevalt. Hierdoor wordt het dus de vraag of je het erg vindt om mensen op te zadelen met Captcha's en dit is weer een ander onderzoek waar wij niet aan toegekomen zijn.

6.2 Zijn Captcha's genoeg?

In principe wel als we maar de eerste aanname kunnen hard maken. Dit is echter niet het geval. Er zijn computers die Captcha's kunnen lezen en dus vervolgens oplossen. Dit duurt wel lang en is geen foutvrij systeem. Dus dit is wel te detecteren. Als dan een client meerdere keren achter elkaar de Captcha niet correct invult kun je deze ook uitsluiten voor het systeem door deze toe te voegen aan de blacklist. Het is dus, zoals in de theoretisch kader al aangestipt, voldoende dat we de clients genoeg afremmen zodat ze in ieder geval minder schade aanrichten en minder winstgevend zijn.

Het is dus niet te concluderen dat ons protocol helemaal spam vrij zal zijn, maar het zal wel een goede stap in die richting zijn, maar er zijn ook andere systemen zoals Judo die op een betere manier deze spambot problematiek tegenhouden zonder het ongemak van een ander protocol.

Referenties

[1] cseweb.ucsd.edu/~voelker/pubs/judo-ndss10.pdf

[2] http://www.nytimes.com/2002/12/10/science/human-or-computer-take-this-test.html?sec=technology&pagewanted=all

[3] http://tools.ietf.org/html/rfc793

[4] http://tools.ietf.org/html/rfc768

[5] http://www.nytimes.com/2010/04/26/technology/26captcha.html?_r=1&src=me&ref=technology

[6] http://www.jcho.de/jc/Pubs/itc2002-col.pdf