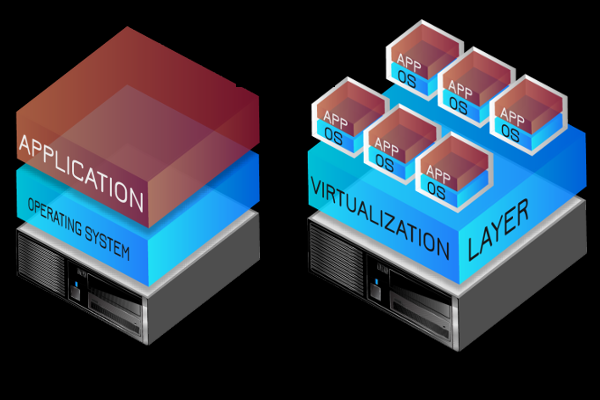
Techno Virtualisatie



Arthur Lanting

500673495

Inhoudsopgave

[‘simple’ subject: Linux IP-tables 3](#_Toc532216995)

[‘simple’ subject: Libvirt 5](#_Toc532216996)

[‘simple’ subject: Shodan.io 6](#_Toc532216997)

[‘moderate’ subjects: Diffie Hellman 8](#_Toc532216998)

[‘moderate’ subject: Docker 10](#_Toc532216999)

[‘hard’ subject: Rainbow tables 11](#_Toc532217000)

# ‘simple’ subject: Linux IP-tables

Iptables is onderdeel van netfilter en speciaal ontwikkeld voor Linux. Binnen Iptables kunnen IP-adressen, zowel over het werk als lokaal gemanipuleerd en gefilterd worden. Iptables worden geconfigureerd in de terminal. In dit script ga ik de startup configureren en de networkmanager.

Rusty Russell startte het project netfilter / iptables in 1998; hij had ook de voorganger van het project, ipchains, geschreven. Naarmate het project groeide, richtte hij in 1999 het Netfilter Core Team (oftewel coreteam) op. De software die zij produceren (hierna netfilter genoemd) maakt gebruik van de GNU General Public License (GPL) -licentie en werd in maart 2000 samengevoegd in versie 2.3. x van de Linux-kernel mainline.

Een van de belangrijke functies die bovenop het Netfilter-framework is gebouwd, is het volgen van verbindingen. Connection Tracking stelt de kernel in staat om alle logische netwerkverbindingen of sessies bij te houden en daarmee alle pakketten te relateren die deze verbinding kunnen vormen. NAT vertrouwt op deze informatie om alle gerelateerde pakketten op dezelfde manier te vertalen, en iptables kunnen deze informatie gebruiken om te fungeren als een stateful firewall.

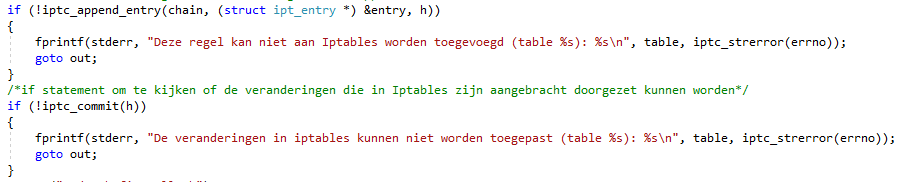
De verbindingsstatus is volledig onafhankelijk van elke status op het hoogste niveau, zoals TCP- of SCTP-status. Een deel van de reden hiervoor is dat bij het alleen doorsturen van pakketten, d.w.z. geen lokale bezorging, de TCP-motor niet noodzakelijkerwijs hoeft te worden aangeroepen. Zelfs transmissies zonder verbindingsmodus zoals UDP, IPsec (AH / ESP), GRE en andere tunnelingprotocollen hebben, ten minste, een pseudo-verbindingstoestand. De heuristiek voor dergelijke protocollen is vaak gebaseerd op een vooraf ingestelde time-outwaarde voor inactiviteit, na het verstrijken ervan wordt een Netfilter-verbinding verbroken.

Elke Netfilter-verbinding wordt uniek geïdentificeerd door een (laag-3 protocol, bron-adres, bestemmingsadres, laag-4 protocol, laag-4 sleutel) tuple. De laag-4 toets is afhankelijk van het transportprotocol; voor TCP / UDP zijn dit de poortnummers, voor tunnels kan dit de tunnel-ID zijn, maar anders is het gewoon nul, alsof het geen deel uitmaakt van de tuple. Om de TCP-poort in alle gevallen te kunnen inspecteren, worden pakketten verplicht gedefragmenteerd.

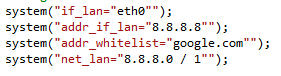
Netfilter-verbindingen kunnen worden gemanipuleerd met de connectie tussen gebruiker en spatie.

Iptables kunnen gebruik maken van het controleren van de informatie van de verbinding zoals toestanden, statussen en meer om pakketfilterregels krachtiger en gemakkelijker te beheren te maken.

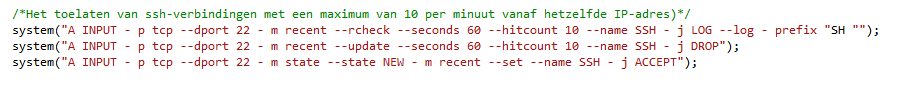
Ik heb een c programma gemaakt met een script daarin verwerkt. In code referentie 1 heb ik twee if statements gemaakt, die moeten controleren of de regels die moeten worden uitgevoerd daadwerkelijk succesvol zijn uitgevoerd. Als deze regels niet kunnen worden uitgevoerd krijgt de gebruiker een of twee meldingen te zien, mochten de regels wel succesvol zijn uitgevoerd, dan gaat het script gewoon door naar de volgende regel.



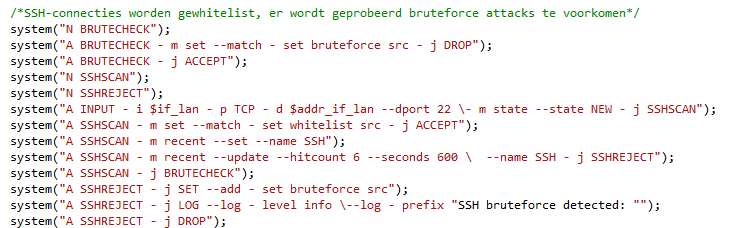
In codereferentie 2 whitelist ik de server van google:



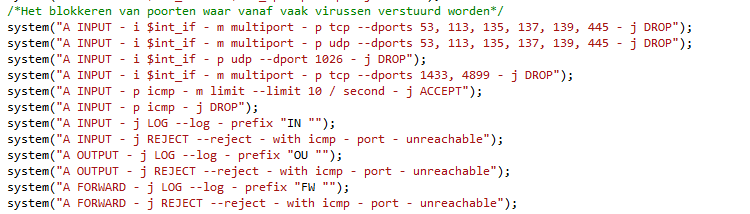
Codereferentie 3 gaat over het toelaten van SSH-verbindingen. Ik heb ervoor gekozen, om maximaal 10 pogingen per minuut toe te laten. Dit om te voorkomen dat het netwerk overbelast raakt. Als de verbinding instabiel is kan er na 6 seconden weer opnieuw verbinding gemaakt worden.



Codereferentie 4 gaat over het proberen te voorkomen van bruteforce attacks door SSH te scannen en een check te doen op bruteforce attacks.



Codereferentie 5 gaat over het blokkeren van poorten, waarvan vaak virussen worden verzonden. Als de ICMP-poort wordt geblokkeerd.



Bronvermelding:

<https://gist.github.com/Atem18/4695539>

<https://bani.com.br/2012/05/programmatically-managing-iptables-rules-in-c-iptc/>

<https://www.howtoforge.com/bash-script-for-configuring-iptables-firewall>

<https://en.wikipedia.org/wiki/Netfilter>

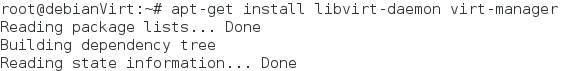
# ‘simple’ subject: Libvirt

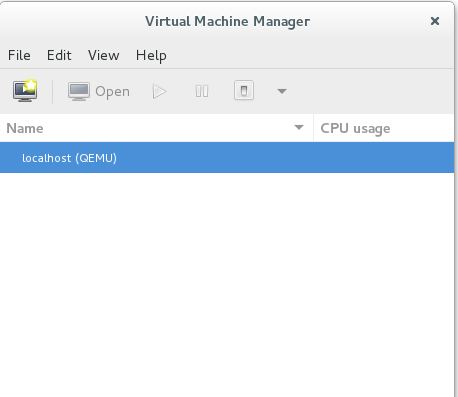
Libvirt is een managementtool voor virtualisatie. Het kan worden gebruikt voor onder andere KVM en VMware ESXi. Libvirt bestaat uit een C-library, die gecombineerd is met andere programmeertalen, zoals Python, Java en Javascript.

Kernel-gebaseerde Virtual Machine (KVM) is een virtualisatie-infrastructuur voor de Linux-kernel die er een hypervisor van maakt. Het werd samengevoegd in de Linux kernel mainline in kernelversie 2.6.20, die werd vrijgegeven op 5 februari 2007. KVM vereist een processor met hardware-virtualisatie-uitbreidingen. KVM is ook geporteerd naar FreeBSD en illumos in de vorm van laadbare kernelmodules.

QEMU is een gehoste virtuele machinemonitor: het emuleert de processor van de machine door middel van dynamische binaire vertaling en biedt een reeks verschillende hardware- en apparaatmodellen voor de machine, waardoor deze een groot aantal gastbesturingssystemen kan uitvoeren. Het kan ook met KVM worden gebruikt om virtuele machines met een bijna-native snelheid te laten werken (door gebruik te maken van hardware-uitbreidingen zoals IntelVT). QEMU kan ook emulatie uitvoeren voor processen op gebruikersniveau, waardoor toepassingen die zijn gecompileerd voor de ene architectuur, op een andere kunnen worden uitgevoerd.

Eerst heb ik in een virtuele machine Libvirt geïnstalleerd.



Virtual machine manager is geïnstalleerd: 

Bron:

<https://stackoverflow.com/questions/8723059/how-to-generate-xml-file-dynamically-using-c-on-linux-platform>

<https://xmodulo.com/use-kvm-command-line-debian-ubuntu.html>

<https://www.linuxhelp.com/how-to-create-and-manage-kvm-virtual-machine-through-command-line-in-linux>

create\_kvm\_vm\_cli.pdf

http://xmodulo.com/disable-network-manager-linux.html

# ‘simple’ subject: Shodan.io

Shodan is een soort zoekmachine, waarmee je onder andere webcams, routers, servers, verkeerslichten, beveiligingscamera's, huisverwarmingssystemen en besturingssystemen voor waterparken, benzinestations, waterplanten, elektriciteitsnetten en kerncentrales die verbonden zijn met het internet kan vinden. SHODAN is in 2009 gelanceerd door een programmeur genaamd, John Matherly.

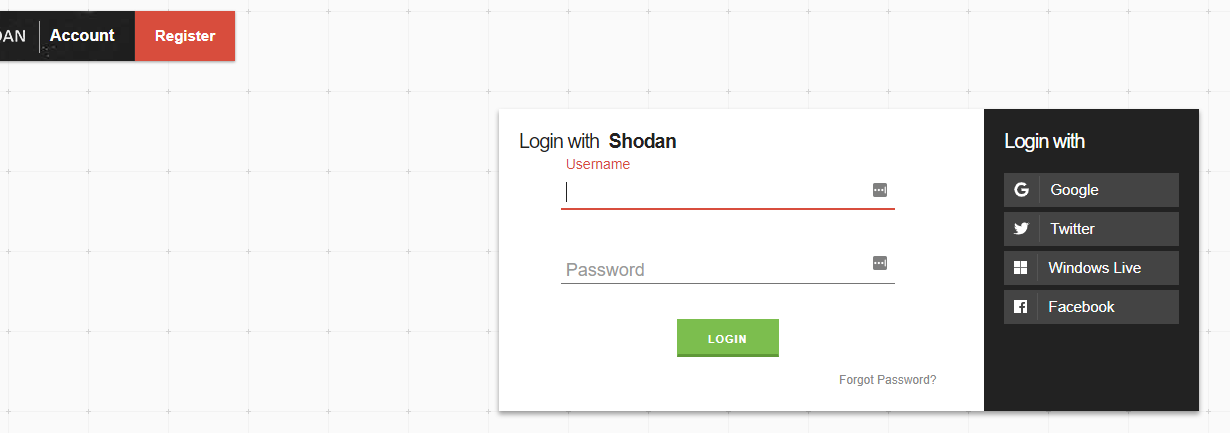
De meeste informatie van shodan komt van webservers (poort 80, 8080, 443, 8443) en via FTP, SSH, Telnet, SNMP, IMAP, SMTP, SIP en het Real Time Streaming Protocol. Het Real Time Streaming Protocol kan worden gebruikt om toegang te krijgen tot webcams en een videostream.

De meeste apparaten zijn slecht beveiligd. Ze hebben vaak ‘admin’ als gebruikersnaam en ‘1234’ als wachtwoord en de enige software die nodig is om verbinding met ze te maken, is een webbrowser.

SHODAN Diggity is een gratis tool, waarmee je een handige lijst met 167 zoekopdrachten die klaar zijn in een vooraf gemaakt dictionarybestand kan zien. Dit dictionarybestand helpt verschillende technologieën te targeten, waaronder webcams, printers, VoIP-apparaten, routers, broodroosters, schakelaars en zelfs SCADA / Industrial Control Systems (ICS) om er maar een paar te noemen.

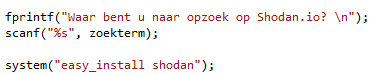
Voor deze paper heb ik een programma in C gemaakt, waarin ik de API van shodan aanroep en commando’s gebruik, die bepaalde zoekresultaten geven. De API is zo ontwikkeld dat deze aangeroepen kan worden vanuit de Linuxterminal. Het is ook mogelijk om dit te verwerken in een C-programma door System te gebruiken.

Om gebruik te maken van de website van shodan moet je je eerst registreren.

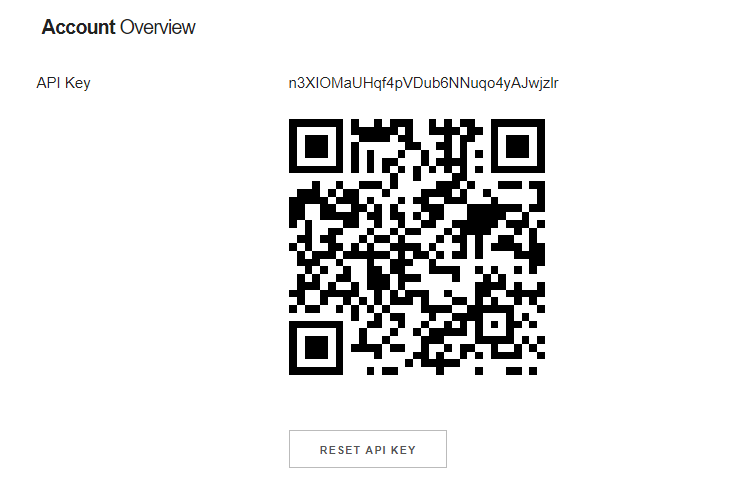


Dit kan door een account aan te maken, of door in te loggen met een van de andere opties, zoals google en facebook.

Om shodan te kunnen gebruiken in de terminal moet het pakket eerst geïnstalleerd worden in de terminal. Hiervoor kan het commando easy\_install shodan gebruikt worden. Ik heb dit commando verwerkt in mijn programma.



Code referentie 1, het uitvoeren van een zoekopdracht en het installeren van shodan, om het mogelijk te maken om opzoek te gaan op de website. Ook heb je een API key nodig. Deze key kan je vinden door naar developers te gaan en dan op ‘Install CLI’ te klikken en daarna op ‘Get your API Key’.



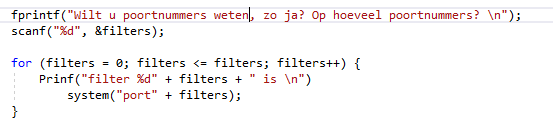
De API key heb ik gebruikt in het programma om mezelf te identificeren voor de website, ook vraag ik het huidige IP-adres op om te valideren. Dit is code- referentie 2:



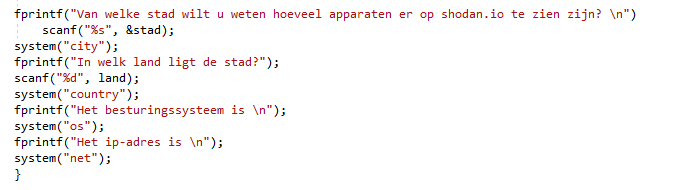
Om een lijst van gehackte websites te laten zien, gebruik ik meerdere commando’s, die deze lijst opvragen. De lijst is een .json-bestand en wordt gezocht via poort 23. Dit is codereferentie nummer 3:



Om een lijst met poortnummers aan te vragen heb ik een for loop gemaakt, daarna wordt afhankelijk van het aantal keer dat de gebruiker aangeeft geloopt. Hiervoor gebruik ik het commando’s port.



In de codereferentie vraag ik op van welke stad de persoon wilt zien hoeveel apparaten er op shodan.io te zien zijn. Er wordt ook informatie over het besturingssysteem verwerkt, en het IP-adres wordt ook getoond na het uitvoeren van het commando ‘net’.



Bronnen: <https://danielmiessler.com/study/shodan/>.

<https://gist.github.com/ddouhine/e643247bf3752d186047>.

https://github.com/TerribleDev/Shodan.Net/blob/master/src/Shodan.Net/IShodanAsyncClient.cs.

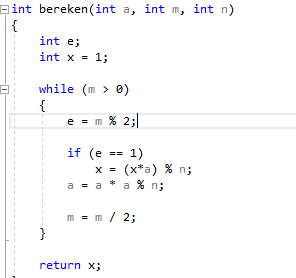
<https://www.linuxquestions.org/questions/programming-9/c-program-code-to-run-a-linux-command-line-63859/>

https://en.wikipedia.org/wiki/Shodan\_(website)

# ‘moderate’ subjects: Diffie Hellman

Het Diffie Hellmanprotocol is een protocol, waarbij er tussen twee clients die geen informatie van elkaar hebben via een onbeveiligd communicatiekanaal een geheime encryptiesleutel kunnen uitwisselen. De sleutel kan worden gebruikt om communicatie tussen de deelnemers te versleutelen. Deze methode werd voor het eerst beschreven 1976 door Whitfield Diffie en Martin Hellman.

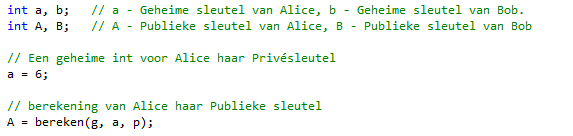
Codereferentie 1: In dit stuk code wordt een voorberekening gedaan voor later bij het berekenen van de publieke sleutel.



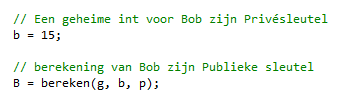
Codereferentie 2: Het vaststellen van de standaardwaarden voor het uitwisselen van de publieke en de privésleutel.



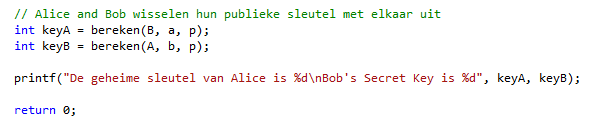
Codereferentie 3:



Codereferentie 4:



Codereferentie 5:



<https://github.com/cfgj/diffie-hellman/blob/master/src/lib/diffieHellman.ts>

<https://sublimerobots.com/2015/01/simple-diffie-hellman-example-python/>

https://www.geeksforgeeks.org/implementation-diffie-hellman-algorithm/

# ‘moderate’ subject: Docker

Docker is gemaakt door Solomon Hykes als een intern project in Frankrijk van dotCloud, met bijdragen van andere programmeurs van dotCloud, waaronder Andrea Luzzardi en Francois-Xavier Bourlet. Jeff Lindsay was ook betrokken als een onafhankelijke medewerker. Docker vertegenwoordigt een evolutie van de eigen technologie van dotCloud, die zelf is gebouwd op eerdere open-sourceprojecten zoals Cloudlets. In 2013 werd docker uitgebracht en vrijgegeven als open source software.

Docker is primair ontwikkeld voor Linux, waarbij het gebruik maakt van de resource-isolatiefuncties van de Linux-kernel zoals cgroups en kernel namespaces, en een union-geschikt bestandssysteem zoals OverlayFS en anderen om onafhankelijke "containers" binnen Linux, waarbij de overhead van het starten en onderhouden van virtuele machines (VM's) wordt vermeden. De ondersteuning van de Linux-kernel voor naamruimten isoleert meestal de weergave van een toepassing van de besturingsomgeving, inclusief procesbomen, netwerk-, gebruikers-ID's en gekoppelde bestandssystemen, terwijl de c-groepen van de kernel bronbeperking bieden voor het geheugen en de CPU. Sinds versie 0.9 bevat Docker de libcontainer-bibliotheek als een eigen manier om direct virtualisatiefaciliteiten van de Linux-kernel te gebruiken, naast het gebruik van geabstraheerde virtualisatie-interfaces via libvirt, LXC en systemd-nspawn.

Voortbouwend op de faciliteiten die de Linux-kernel biedt (voornamelijk cgroups en naamruimten), vereist een docker-container, in tegenstelling tot een virtuele machine, geen afzonderlijk besturingssysteem. In plaats daarvan vertrouwt het op de functionaliteit van de kernel en wordt gebruikgemaakt van bronisolatie voor CPU en geheugen, en afzonderlijke naamruimten om de weergave van de toepassing van het besturingssysteem te isoleren. Docker geeft toegang tot de virtualisatiefuncties van de Linux-kernel, hetzij rechtstreeks met behulp van de bibliotheek libcontainer, die beschikbaar is vanaf Docker 0.9, of onrechtstreeks via libvirt, LXC (Linux Containers) of systemd-nspawn.

Software: de Docker-daemon, dockerd genaamd, is een permanent proces dat Docker-containers beheert en containerobjecten verwerkt. De daemon luistert naar verzoeken die worden verzonden via de Docker Engine API. Het Docker-clientprogramma, docker genoemd, biedt een opdrachtregelinterface waarmee gebruikers kunnen werken met Docker-daemons.

Objecten: Docker-objecten zijn verschillende entiteiten die worden gebruikt om een ​​toepassing in Docker samen te stellen. De hoofdklassen van Docker-objecten zijn afbeeldingen, containers en services.

Een Docker-container is een gestandaardiseerde, ingekapselde omgeving waarin toepassingen worden uitgevoerd. Een container wordt beheerd met behulp van de Docker API of CLI.

Een Docker-afbeelding is een alleen-lezen sjabloon dat wordt gebruikt om containers te bouwen. Afbeeldingen worden gebruikt voor het opslaan en verzenden van applicaties.

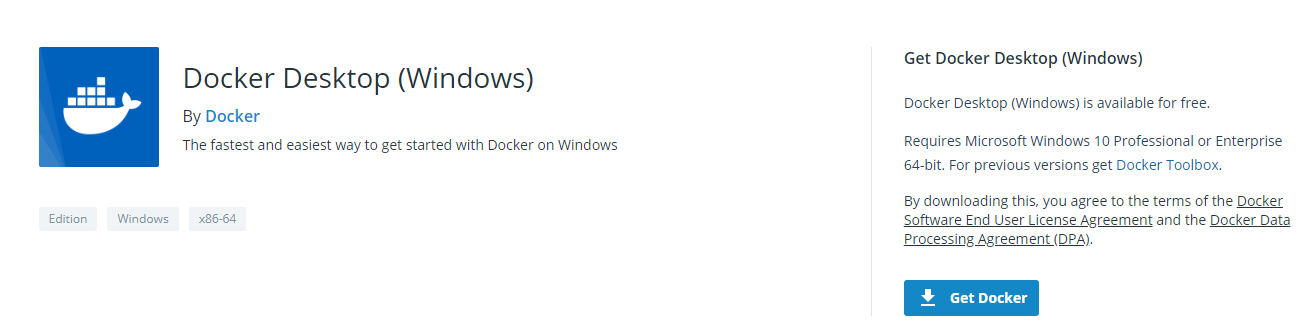
Met een Docker-service kunnen containers over meerdere Docker-daemons worden geschaald. Het resultaat staat bekend als een "zwerm", een reeks samenwerkende daemons die communiceren via de docker-API.

Registries: een Docker-register is een opslagplaats voor Docker-afbeeldingen. Docker-clients maken verbinding met registers om afbeeldingen te downloaden ("trekken") voor gebruik of om afbeeldingen ("push") die ze hebben gebouwd te uploaden. Registers kunnen openbaar of privé zijn. Twee belangrijke openbare registers zijn Docker Hub en Docker Cloud. Docker Hub is het standaardregister waar Docker naar afbeeldingen zoekt.

Hulpmiddelen

Docker Compose is een hulpmiddel voor het definiëren en uitvoeren van Docker-toepassingen met meerdere containers. Het maakt gebruik van YAML-bestanden om de services van de applicatie te configureren en voert het creatie- en opstartproces uit van alle containers met een enkele opdracht. Met het docker-compose CLI-hulpprogramma kunnen gebruikers opdrachten op meerdere containers tegelijk uitvoeren, bijvoorbeeld door afbeeldingen te maken, containers te schalen, containers te laten stoppen die zijn gestopt en meer. Opdrachten met betrekking tot beeldmanipulatie of gebruikersinteractieopties zijn niet relevant in Docker Compose omdat ze betrekking hebben op één container. Het bestand docker-compose.yml wordt gebruikt om de services van een toepassing te definiëren en bevat verschillende configuratie-opties. De buildoptie definieert bijvoorbeeld configuratie-opties, zoals het Dockerfile-pad, met de opdrachtoptie kunt u standaard Docker-opdrachten overschrijven en meer. De eerste openbare versie van Docker Compose (versie 0.0.1) werd uitgebracht op 21 december 2013. De eerste productierijpere versie (1.0) is beschikbaar gesteld op 16 oktober 2014.

Docker Swarm biedt native clustering-functionaliteit voor Docker-containers, die een groep Docker-engines in één virtuele Docker-engine veranderen. In Docker 1.12 en hoger is de Zwermmodus geïntegreerd met Docker Engine. Met het hulpprogramma zwerm CLI kunnen gebruikers zwermcontainers uitvoeren, detectietokens maken, knooppunten in het cluster weergeven en meer. Met het CLI-hulpprogramma voor dockerknooppunten kunnen gebruikers verschillende opdrachten uitvoeren om knooppunten in een zwerm te beheren, bijvoorbeeld door knooppunten in een zwerm te vermelden, knooppunten bij te werken en knooppunten uit de zwerm te verwijderen. Docker beheert zwermen met behulp van het Raft Consensus Algorithm. Volgens Raft moeten de meeste knooppunten van Swarm het eens worden over de update om een ​​update uit te voeren.

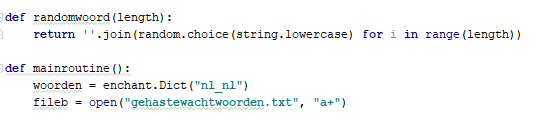


# ‘hard’ subject: Rainbow tables

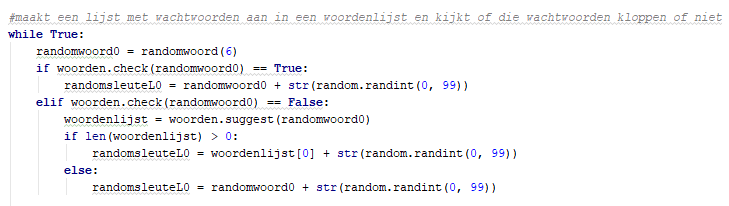
Rainbow tables zijn tabellen met mogelijke wachtwoorden en hashes van de wachtwoorden. De techniek is veel sneller dan de brute force-techniek, omdat bij deze techniek de hashes van de wachtwoorden moet worden berekend. Het maken van een hashtabel kost veel rekentijd neemt veel ruimte in beslag, maar als de tabel berekend is kan de hash snel worden omgezet naar een wachtwoord.

Eén LM-hash werkt met wachtwoorden tot 7 tekens lang. Als een wachtwoord 7 tot 14 tekens lang is zijn er twee hashes nodig, enzovoort. Een manier om een systeem te beveiligen tegen rainbowtable-aanvallen is door salt toe te voegen aan de passwordhash.

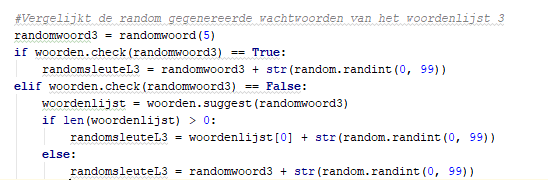
Codereferentie 1: het aanmaken van een lijst met woorden, die gebruikt worden om als wachtwoord te gebruiken.



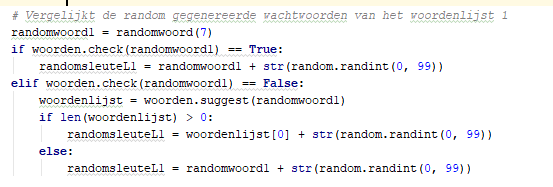
Codereferentie 2:In dit stuk code wordt er gekeken of de randomwoorden kloppen als wachtwoord, bij woordenlijst 0.



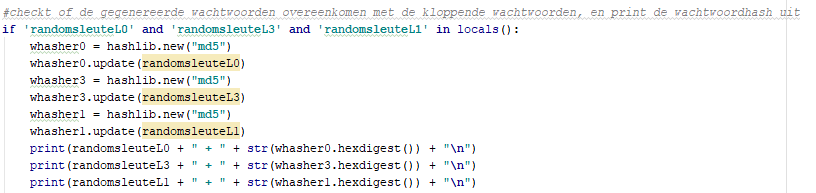
Codereferentie 3:In dit stuk code wordt er gekeken of de randomwoorden kloppen als wachtwoord, bij woordenlijst 3.



Codereferentie 4:In dit stuk code wordt er gekeken of de randomwoorden kloppen als wachtwoord, bij woordenlijst 1.



Codereferentie 5: In dit stuk code wordt er gekeken of de wachtwoorden kloppen en in de database voorkomen, daarna wordt de bijbehorende hash op het scherm uitgeprint.



https://redvice.wordpress.com/2015/04/19/python-code-to-make-rainbow-tables/