One-time passwords

Ultieme beveiliging van je pc

Klassieke wachtwoorden hebben één grote zwakheid: worden ze gestolen of afgeluisterd, dan kan een cracker er onmiddellijk misbruik van maken. Er bestaan echter ook systemen van eenmalige wachtwoorden. Je gebruikt zo'n wegwerpwachtwoord één keer en daarna is het niet meer geldig. We gingen ermee aan de slag onder Linux. *Koen Vervloesem*

Heb je je laptop toevallig niet bij je en wil je op een computer in een internetcafé op je server inloggen, dan zit je met een probleem. Vertrouw je erop dat er op die computer geen keylogger staat geïnstalleerd die je ssh-wachtwoord onderschept? Inloggen via een wachtwoord is dus uit den boze. Inloggen via een rsa-sleutel dan? Het probleem is dat je dan je privésleutel op een usb-stick moet meenemen en in de computer steken. Vertrouw je er dan op dat die computer niet automatisch de inhoud van je usb-stick kopieert? In sommige gevallen zijn beide manieren van inloggen dus niet aan te raden.

Onderscheppen is zinloos

De oplossing: een one-time password (otp). Zoals de naam al aangeeft, is dat maar één keer geldig. Staat er een keylogger op de computer waarop je het wachtwoord ingeeft, dan kan de hacker die daarvoor verantwoordelijk is, niets doen met het wachtwoord dat hij onderschept. Dus als je met zo'n eenmalig wachtwoord op je blog inlogt, dan kan een hacker evenmin iets met het wachtwoord dat hij onderschept door het netwerkverkeer af te luisteren. Wordt een otp onderschept, dan is het bovendien niet mogelijk om hier het volgende otp uit af te leiden.
Een project dat ondersteuning voor one-time passwords aan Linux en FreeBSD toevoegt, is OPIE (one-time passwords in everything). Dit werkt conceptueel als volgt: geef je een passphrase in een generator in, dan krijg je een volgnummer en een *challenge*. De computer waarop je met OPIE wilt inloggen, houdt dit volgnummer en de challenge bij. Wanneer je nu wilt inloggen, geef je op een andere computer je passphrase, een volgnummer en de challenge in een otp-calculator in. Dit programma genereert op basis van deze parameters een uniek en slechts eenmalig geldig wachtwoord, dat je dan op de server ingeeft. Wil je ook kunnen inloggen zonder een vertrouwde computer bij de hand, dan laat je een tiental wachtwoorden genereren (elk met een ander volgnummer), die je dan op een papiertje neerschrijft en telkens doorstreept als je ze hebt gebruikt.

Installeren en configureren

Dat is de theorie. Nu de praktijk. We gaan ervoor zorgen dat we op een Ubuntu 9.10-server alleen via ssh kunnen inloggen met een eenmalig wachtwoord. Voor andere Linux-distributies of andere releases werkt dit vergelijkbaar. Installeer eerst de OPIE-server en -client:

```
$ sudo apt-get install opie-server
$ sudo apt-get install opie-client
```

Nu gaan we van het Pam-systeem (Pluggable Authentication Modules) van Linux gebruikmaken om OPIE voor de authenticatie in te schakelen als we via ssh inloggen. Hiervoor moeten we in het bestand /etc/pam.d/sshd zijn. We hebben nu twee keuzes. Ofwel laten we zowel normale wachtwoorden als eenmalige wachtwoorden toe, ofwel enkel eenmalige wachtwoorden.
De eerste optie is bijvoorbeeld zinvol als je op je eigen vertrouwde computers op de server wilt kunnen inloggen met je normale wachtwoord, en slechts de - toch iets omslachtiger - eenmalige wachtwoorden wilt gebruiken als je op een niet-vertrouwde computer werkt.

Challenge en response

Wil je via ssh kunnen inloggen door middel van otp of een normaal wachtwoord, dan vervang je in /etc/pam.d/sshd de regel *@include common-auth* door:

```
auth sufficient pam_unix.so
auth sufficient pam_opie.so
auth required pam_deny.so
```

De eerste regel zorgt voor de normale login, terwijl de tweede regel de mogelijkheid van inloggen via OPIE toevoegt. Wil je alleen maar kunnen inloggen via OPIE, dan verwijder je die eerste regel gewoon.
Naast de Pam-configuratie moet ook de configuratie van de ssh-server worden aangepast. De otp-wachtwoorden vormen namelijk een challenge-response-authenticatie, en die mogelijkheid is standaard uitgeschakeld in OpenSsh. Specificeer daarom in /etc/ssh/sshd_config het volgende:

```
ChallengeResponseAuthentication yes
```

Herstart dan de ssh-server om de wijzigingen door te voeren:

```
$ sudo service ssh restart
```

Gebruik je overigens FreeBSD, dan is deze installatie en configuratie niet nodig. Dit besturingssysteem ondersteunt OPIE al out-of-the-box.

Geheime passphrase

Je computer is nu klaar voor OPIE, maar we moeten de eenmalige wachtwoorden nog initialiseren met een passphrase. Typ daarom op de computer het volgende in:

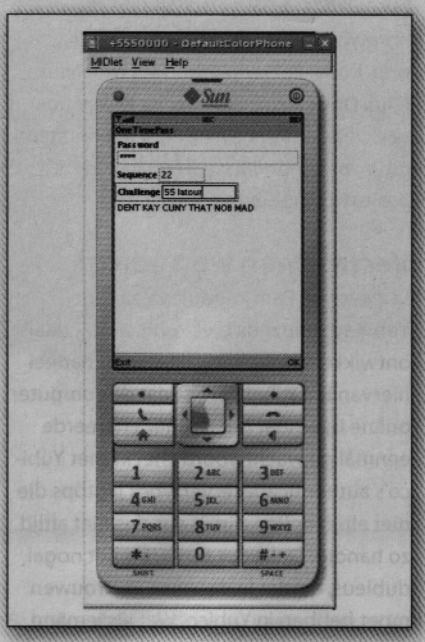
```
$ opiepasswd -c
```

De generator vraagt je nu om een geheime passphrase in te geven. Die wordt als parameter gebruikt om de eenmalige wachtwoorden aan te maken. Je kunt het best een passphrase van tenminste twaalf tekens lang gebruiken. Nadat je deze twee keer hebt ingegeven, krijg je een antwoord als het volgende:

```
ID koan OTP key is 499 vm7427
NEON CAGE RICK JOEY ACID ALP
```

Op de eerste regel zie je jouw loginnaam (hier *koan*), een volgnummer *(499)* en een challenge *(vm7427)*. Op de tweede regel krijg je het eenmalige wachtwoord te zien dat bij het volgnummer en de challenge

Afbeelding 1: Bereken eenmalige wachtwoorden op je mobieltje (bron: Jan-Frode Myklebust, J2ME-OTP)



Als je toch een vertrouwde computer hebt, waarom log je daarop niet in op je server

hoort, bestaande uit zes woorden. Zoals je ziet, worden deze eenmalige wachtwoorden zo opgesteld dat ze ondanks hun lengte met een beetje moeite nog wel zijn te onthouden.

Testen maar!

Tijd om dit eens te testen. Log op een andere computer op je server in via ssh. Als je in de Pam-configuratie ook nog normale wachtwoorden hebt toegelaten, krijg je gewoon zoals anders de vraag *Password:* te zien, maar als je op Enter drukt, krijg je de vraag om een eenmalig wachtwoord in te geven.

```
otp-md5 498 vm7427 ext, Response:
```

We zien hier dus het volgnummer (*498*, bij elke login wordt één afgetrokken) en de challenge. Om het hierbij behorende eenmalige wachtwoord te krijgen voer je op een vertrouwde computer het commando *opiekey* uit met deze parameters, waarna je de geheime passphrase invult:

```
$ opiekey 498 vm7427
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase:
ARID FRET BONG TAB KUDO ARGO
```

Als resultaat geeft het programma het eenmalige wachtwoord, dat je ingeeft op je ssh-login. Dit wachtwoord kun je niet opnieuw gebruiken. De volgende keer dat je inlogt, toont de ssh-prompt je het volgende volgnummer, *497*. De server registreert namelijk de laatst gebruikte combinatie van volgnummer, challenge en eenmalig wachtwoord in /etc/opiekeys en trekt 1 van het volgnummer af, wanneer je de volgende keer succesvol inlogt. Uiteraard is dit procédé wat omslachtig. Als je immers toch een vertrouwde computer hebt, waarom log je daarop niet in op je server? Maar het leuke is dat je deze wachtwoorden op voorhand kunt genereren en schrijven of printen op een stukje papier dat je in je portefeuille meeneemt. Op de volgende manier genereer je bijvoorbeeld tien eenmalige wachtwoorden voor je server, te beginnen met het volgnummer *497*:

```
$ opiekey -n 10 497 vm7427
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase:
488: DIRT MONT JACK SAY WAD GUSH
489: TICK SAL TINY MOCK ANTI SOON
490: EAT DEFY RUDE MAIL MID FIT
491: JOKE SLUM SELF MINK NOON ROOF
492: ROSA AMOK TRUE ABED SLED CARE
493: COO BANG CHEW LAD KEG BERN
494: NOV MITE HESS MERT TEE GLOB
495: MILT BIDE SEE ROT REIN FUND
496: LICE NEE FLAK LAND ROME CULT
497: NED BAKE CODE GAME WHO CEIL
```

Er bestaat overigens ook een applicatie die je op je mobieltje draait en die het one-time password berekent dat bij een bepaalde passphrase, volgnummer en challenge hoort: J2ME-OTP.

Afbeelding 2: Bereken eenmalige wachtwoorden op je iPhone (bron: Rho)

Voor de iPhone is er een vergelijkbaar programma: 1Key, al betaal je daar wel 3,99 euro voor.
Een andere manier om eenmalige wachtwoorden te genereren is de YubiKey, een

We zijn paranoïde, dus we maken gebruik van de tweede manier: herinitialisatie van onze YubiKey

usb-sleutel die je in je computer steekt en die is voorzien van een knop. Het kleinood weegt slechts twee gram en kan aan een sleutelbos worden gehangen. Elke keer dat je op de knop drukt, stuurt de YubiKey een unieke sleutel naar de computer. Het mooiste van alles is dat dit op alle besturingssystemen zonder drivers werkt, doordat de YubiKey zich voordoet als een usb-toetsenbord. Een YubiKey kost 25 dollar.
Yubico's businessmodel is: verdien door het verkopen van de hardware, maar hou alle software open source. Zo is de broncode van de eigen authenticatieserver open, zodat je zelf eenvoudig een alternatief voor Yubico's service kunt opzetten. Verder is er ook broncode beschikbaar voor verschillende client-library's, onder andere in Java, C, C#, php, Ruby, Perl en Python en voor Unix Pam. Door al deze broncode zijn er ook verschillende externe projecten ontstaan die YubiKey-ondersteuning in andere projecten inbouwen.

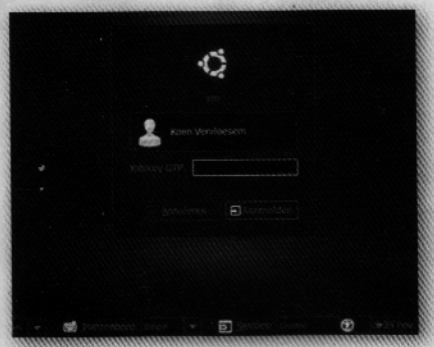
Vertrouwen we Yubico?

Er zijn twee Pam-modules voor de YubiKey. Enerzijds is er YubicoPAM, dat is ontwikkeld door Yubico zelf. Het nadeel hiervan is dat het vereist dat de computer online is, omdat het het gegenereerde eenmalige wachtwoord checkt met Yubico's authenticatieserver. Voor laptops die niet altijd online zijn, is dat dus niet altijd zo handig. Ook voor servers is het nogal dubieus, omdat je daarmee vertrouwen moet hebben in Yubico. Wat als iemand inbreekt in de server met de authenticatiegegevens van alle YubiKeys? Je kunt wel een eigen validatieserver opzetten waarop de module verbindt, maar dat vereist toch heel wat configuratiewerk. Daarom hebben de ontwikkelaars van het SecurixLive-team een eigen Pam-module gemaakt die offline werkt zonder de authenticatieserver van Yubico: YubiPAM. Deze gebruiken we hier. Ten tijde van schrijven was de meest recente versie van YubiPAM versie 1.0.4.
We downloaden, compileren en installeren de module als volgt:

```
$ sudo su -
# apt-get install libpam0g-dev
# cd /usr/local/src
# wget http://www.securixlive.com/download/yubipam/YubiPAM-1.0.4.tar.gz
# tar xf YubiPAM-1.0.4.tar.gz
# cd YubiPAM-1.0.4
# ./configure
# make install
# addgroup yubiauth
# touch /etc/yubikey
# chgrp yubiauth /etc/yubikey /sbin/yk_chkpwd
# chmod g+rw /etc/yubikey
# chmod g+s /sbin/yk_chkpwd
```

Nu YubiPAM is geïnstalleerd, moeten we

Afbeelding 3: Log in met je YubiKey



aan de module de aes-sleutel van onze YubiKey doorgeven. Dat kan op twee manieren. De eerste manier is dat je aan de supportmensen van Yubico (mail naar support@yubico.com) vraagt om je toegang te geven tot de Yubico Management Service, een website waarop je je YubiKeys kunt beheren en de aes-sleutels kunt downloaden. Om te bewijzen dat je wel de rechtmatige eigenaar van de YubiKey bent, vragen ze je twee otp-sleutels van het kleinood.
Maar waarom doen we al die moeite voor een offline YubiKey Pam-module als we onze Aes-sleutel toch nog via Yubico ontvangen? We zijn paranoïde, dus we maken gebruik van de tweede manier. We herinitialiseren onze YubiKey, waardoor we onze eigen - voor anderen onbekende - aes-sleutel in de YubiKey stoppen. Deze laatste kan dan niet meer worden gebruikt voor alle online YubiKey-diensten, zoals de validatieserver, management-service en OpenID-server. We downloaden en compileren de YubiKey Personalization Tool voor Linux zo:

```
$ sudo su -
# apt-get install libusb-1.0-0-dev
# cd /usr/local/src
# wget http://yubico-c.googlecode.com/files/libyubikey-1.5.tar.gz
# tar xf libyubikey-1.5.tar.gz
# cd libyubikey-1.5
# ./configure
# make install
# cd ..
# wget http://yubikey-personalization.googlecode.com/files/ykpers-1.1.tar.gz
# tar xf ykpers-1.1.tar.gz
# cd ykpers-1.1
# ./configure
# make install
```

Afbeelding 4: Een eenmalig wachtwoord met een druk op de knop

Eerst genereren we een willekeurige Aes-128-sleutel. Dat kan bijvoorbeeld als volgt gebeuren:

```
$ dd if=/dev/urandom count=16 bs=1
2>/dev/null | od -x | tr -d ' ' |
sed -n '1s/^0000000//p'
```

Steek je YubiKey nu in een usb-aansluiting van je computer en voer het programma ypersonalize uit met als parameter de zojuist gegenereerde aes-sleutel en een vast prefix voor de sleutel. Bijvoorbeeld:

```
$ ykpersonalize -a
95aea736d37cd9cc1f43230c3f2c8f58
-ofixed=ccccccccfnlj
```

Valideer nu je sleutel om te zien of alles werkt. Dit kan met het commando *ykdebug*:

```
$ ykdebug <aeskey> <token>
```

Hier vul je je aes-sleutel in en daarna een token dat je door je YubiKey laat genereren. Je krijgt nu heel wat uitvoer, maar wat telt is de laatste regel. Staat daar *crc check: ok*, dan komen het token en de Aes-sleutel van dezelfde YubiKey. Staat er *crc check: fail*, dan klopt dit niet.
Nu hebben we alles klaar om in te loggen met YubiPAM. Eerst koppelen we de YubiKey aan onze account:

```
$ sudo ykpasswd -a --user <user>
-k <aeskey> -o <token>
```

Krijg je op het einde *Completed successfully* te zien, dan is alles in orde. We kunnen nu controleren of de YubiKey aan onze gebruiker is gekoppeld met het commando *ykvalidate:*

```
$ ykvalidate --user <user> <token>
OTP is VALID.
```

Het token is dus geldig. Nu moet je elke verwijzing naar de aes-sleutel van je computer verwijderen. De veiligheid van je YubiKey staat of valt immers met het geheimhouden van je aes-sleutel, die nu enkel op de YubiKey en in versleutelde vorm in het bestand /etc/yubikey is te vinden. Voor de Pam-configuratie kunnen we weer net zoals met *pam_opie* te werk gaan. Maar een YubiKey is vooral handig voor authenticatie van lokale logins, dus we pakken het hier iets anders aan. Om eenvoudigweg voor alle vormen van authenticatie op je Linux-computer van de YubiKey gebruik te maken, plaats je de volgende regel bovenaan /etc/pam.d/common-auth:

```
auth sufficient pam_yubikey.so
```

Hiermee geef je de via *ykvalidate* geregistreerde gebruikers de mogelijkheid via hun YubiKey in te loggen. Als er geen YubiKey aanwezig is, kunnen ze nog via de normale authenticatie *(pam_unix.so)* inloggen door bij de vraag naar de otp gewoon op Enter te drukken. Die laatste mogelijkheid kun je uitschakelen als je slechts inloggen via de YubiKey wilt toelaten. Dan becommentarieer je de *pam_unix.so*-regel door er een # voor te zetten. Verlies dan echter niet je YubiKey! Probeer nu ook maar eens of de Pam-module een otp dat je al hebt ingegeven, opnieuw accepteert. Je zult zien dat dit dan wordt geweigerd. *Mission accomplished!*

Ontgrendeling

Door de YubiPAM-module in /etc/pam.d/common-auth te plaatsen, zijn verschillende soorten logins via de YubiKey mogelijk - niet alleen een sudo- of een console-login, maar ook het loginvenster van GDM en de vergrendeling van de Gnome-screensaver. Op het Yubico-forum (zie kader) is er zelfs een leuk script te vinden dat je Gnome-sessie automatisch vergrendelt wanneer je je YubiKey verwijdert, en pas weer ontgrendelt wanneer je je YubiKey er opnieuw insteekt en een eenmalig wachtwoord genereert. «

Nuttige links

J2ME-OTP
http://tanso.net/j2me-otp

1Key
www.rho.cc/index.php/iphone-software/1key

YubiKey
www.yubico.com/home/index

YubiKey forum
http://forum.yubico.com

YubiKey Personalization Tool
http://code.google.com/p/yubikey-personalization

YubicoPAM
http://code.google.com/p/yubico-pam

YubiPAM
www.securixlive.com/yubipam

Vergrendel je sessie met YubiPAM
http://forum.yubico.com/viewtopic.php?f=11&t=246