Basic Security

Natasja Vanherck

1. Inleiding Cryptografie ---- 2
2. Symmetrische cryptografie ---- 5
3. Symmetrische cryptografie - DES ---- 5
4. Asymmetrische cryptografie ---- 11
5. Hashing ---- 14
6. Digitale handtekening ---- 16
7. Hybride cryptografie ---- 17
8. Certificaten – PKI ---- 19
9. Web security ---- 24
10. E-mail (Applicatie laag) ---- 25
11. SET (Applicatie laag) ---- 26
12. SSL/TLS (Transport laag) ---- 27
13. IPSec (Netwerk laag) ---- 31
14. Firewall ---- 33
15. Virussen en wormen ---- 34
16. Trojan en Rootkit ---- 36
17. Sniffer – protocol analyzer ---- 37
18. Spoofing (TCP) ---- 38
19. DDos ---- 42
20. PortScanners ---- 45
21. Vulnerability scanner ---- 46
22. IDS - IPS ---- 47
23. Logging & SIEM ---- 49
24. Privacy ---- 51
25. Cookies ---- 53
26. XSS ---- 53
27. Banneradvertentie ---- 55
28. Webbug ---- 55

1. Inleiding Cryptografie

Classificatie cryptografische systemen

Op basis van **3 eigenschappen**

- Operaties om van X -> Y te gaan
  - Substitutie
  - Transpositie
- Gebruikte aantal sleutels
  - Symmetrisch – secret key - dezelfde sleutel
  - Asymmetrisch – public key - verschillende sleutels
- Operaties om van input -> X te gaan
  - Blokvercijfering
  - Stroomvercijfering

Steganografie

- **Informatie verbergen in een foto of geluid**
- Meestal in **LSB** (least significant bit)
- **Foto**
  - RGB waarden van een pixel (8, 8, 8 bits)
  - Laatste bit van iedere kleur vervangen door de bit van uw te verbergen data
- **Geluid**
  - Menselijk gehoor kan maar bepaalde frequenties waarnemen
  - Niet waarneembare frequenties gebruiken om data in te verstoppen

Verschillende type cijfersystemen

- **Substitutie**cijfersystemen
- **Transpositie**cijfersystemen

Substitutiecijfersystemen

Vervanging van tekens (confusion)

Caesar

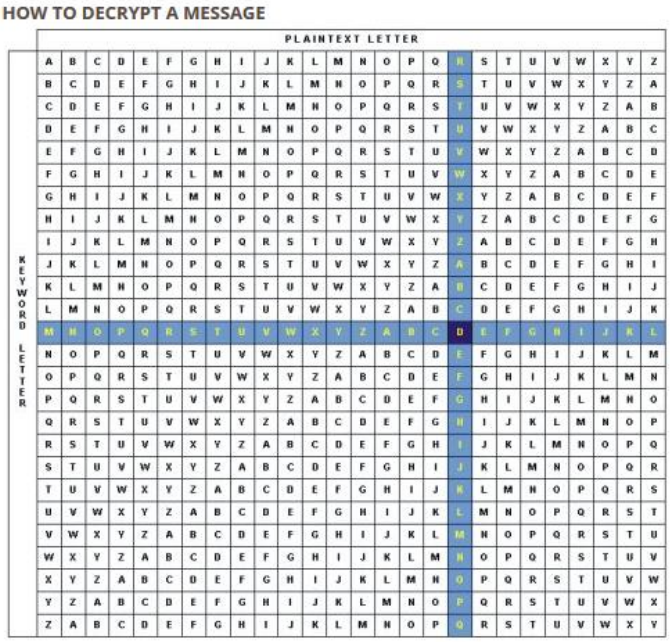
- Alfabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
- Caesar5: FGHIJKLMNOPQRSTUVWXYZABCDE
- X: DITISEENBOODSCHAP
- Y: INYNXJJSGTTIXHMFU

- Aantal plaatsen te verschuiven = sleutel
- Brute aanval-gevoelig! #sleutels = 26

℘ Vigenère

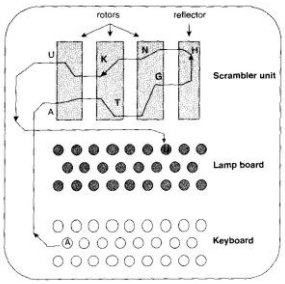
- Men kiest eerst een **geheim sleutelwoord**, bijvoorbeeld crypto.
- Dit schrijft men **onder de klare tekst**.
- Vervolgens zoekt men de **klare letter** op in het **verticale alfabet** en **de letter** van het **sleutelwoord** in het **horizontale alfabet.**
- De **kruising** van beiden is de **resulterende codeletter**.

Klare tekst:	DIT	IS	EEN
Key	CRY	CR	CRY
Cijfertekst	FZR	KJ	GVL



℘ Enigma

- Duitsers tijdens WOII
- 3 rotors met 26 tekens (alfabet)
- Initiële stand moet gekend zijn
- **Caesar met een twist**



transpositiecijfersystemen

Veranderen van volgorde (diffusion – permutation)

Voorbeeld kolomtranspositie

- De letters van de key worden **volgens alfabet genummerd, van links naar rechts**. (1342)
- Men schrijft **de klare tekst van links naar rechts** (rekening houdend met de lengte van de key) **en boven naar onder.**
- Vervolgens **lezen we de tekst af per kolom**, **beginnende met het kleinste nummer** van de key.

Woord: EEN VOORBEELD VAN KOLOMTRANSPOSITIE
Key: BLOK
Volgorde: 1342 (B komt het 1ste in het alfabet, dan K, …)

EENV
OORB
EELD
VANK
OLOM
TRAN
SPOS
ITIE

Cijfertekst: EOEV OTSI VBDK MNSE EOEA LRPT NRLN OAOI

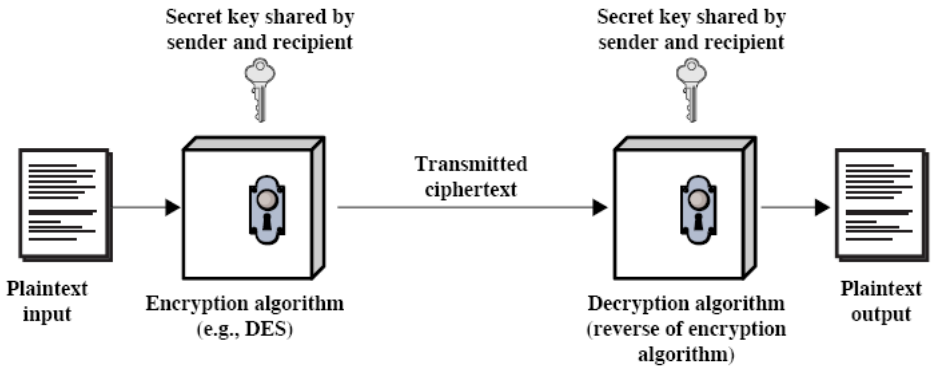
Scytale

Alleen met de **juiste diameter** van cilinder is het leesbaar!

2. Symmetrische cryptografie

Inleiding

- **Dezelfde sleutel voor encrypteren en decrypteren**
- **AES, DES, 3DES, IDEA**
- Ook wel **number cruncher** genoemd
    - Door de **kracht van de CPU** zal er substitutie en transpositie heel veel keer na elkaar gebeuren
    - **Heel veel iteraties na elkaar**
- **Avalanche effect**
    - Is de beveiliging van symmetrische cryptografie
    - Door de grote hoeveelheid **iteraties** zal een hacker die maar een **kleine foutje** maakt bij het decrypteren van een bericht, de kleine fout zo **groot** worden dat hij het bericht niet meer kan ontcijferen en ook niet weet dat hij eigenlijk zo dichtbij zat.



Voordelen en nadelen van symmetrische cryptografie

Voordelen

- **Zeer snel**

Nadelen

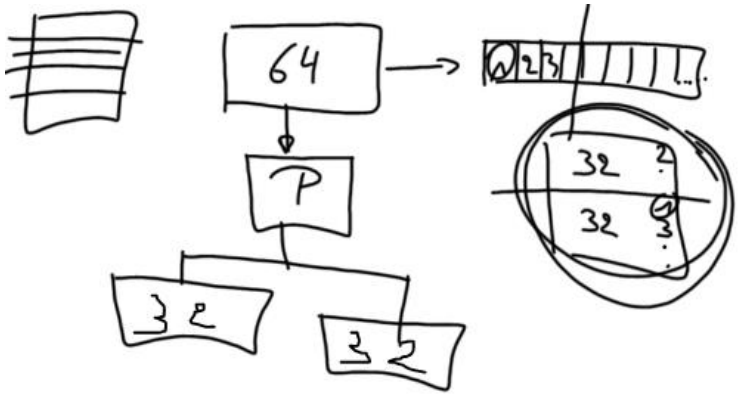
- **Sleutelbeheer, sleutelmanagement**
    - Hoe gaan we de **sleutel veilig communiceren** tussen persoon A en B?
    - **Aantal sleutels** ➔ per persoon die erbij komt zijn er meer sleutels nodig
    - 4 personen ➔ 6 sleutels
    - 5 personen ➔ 10 sleutels
    - 6 personen ➔ 15 sleutels
- **Oplossing:** asymmetrische cryptografie (RSA)

3. Symmetrische cryptografie - DES

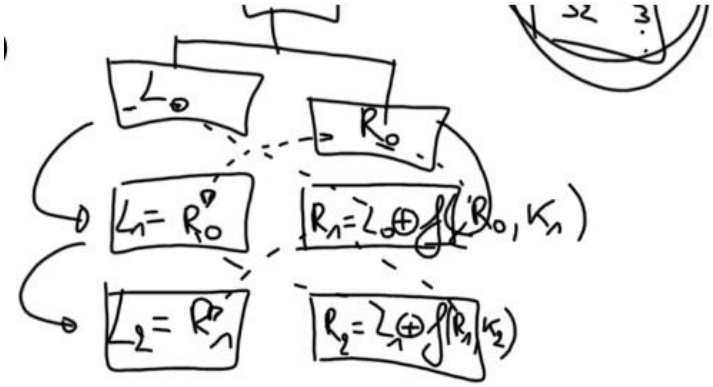
Wat is het?

- Data encryption standard
- Oud: 1977 standaard in USA
- Tekst wordt omgezet **naar binaire waarde** (0,1)
- Symmetrisch: encryptie en decryptie met **dezelfde sleutel**
- **Block cipher:** tekst wordt gesplitst in **blokken van 64 bits**
- **Decryptie = encryptie** (volgorde van bewerkingen worden omgedraaid)

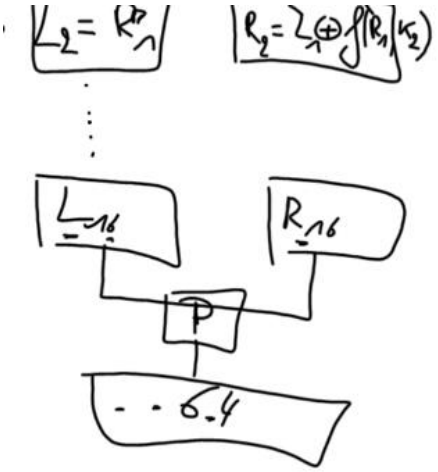
Werking DES



1. Uw datafile wordt **gekapt in stukken van 64 bits**
2. We nemen **1 stuk van 64 bits**
3. We sturen de blok van 64 bits door **een permutatie matrix (P)**
4. P zet de bits die op een bepaalde plaats staan, op een **andere plaats (= transpositie)** en maakt **2 blokken van 32 bits** (links en rechts)

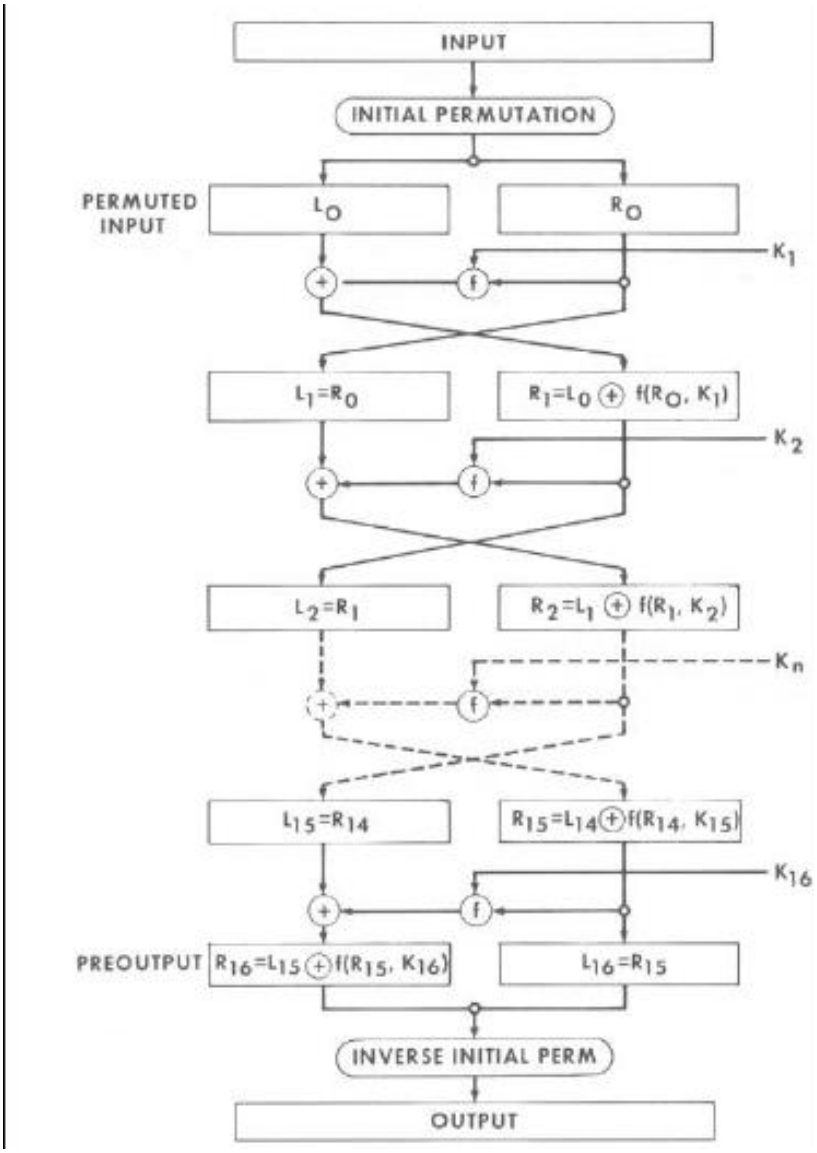


5. Er worden **16 iteraties** uitgevoerd
6. De **nieuwe linkerblok** is een <u>kopie van de vorige rechterblok</u>
7. De **nieuwe rechterblok** is de <u>vorige linkerblok XOR functie (= feistelfunctie)</u>
8. **Feistelfunctie** = vorige rechterblok (r_0), subkey (k_1)



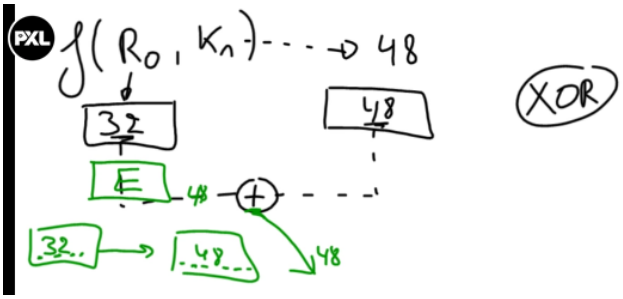
9. Dit wordt **16x herhaalt**
10. De 2 blokken worden terug **samengevoegd door een permutatie matrix**
11. Nu heb een blokje van 64 bits dat je hebt **geëncrypteerd met DES**
12. Dit doe je nu voor alle blokjes van uw datafile

Decrypteren werkt hetzelfde als encrypteren maar dan wordt eerst K_{16} gebruikt i.p.v. K_1

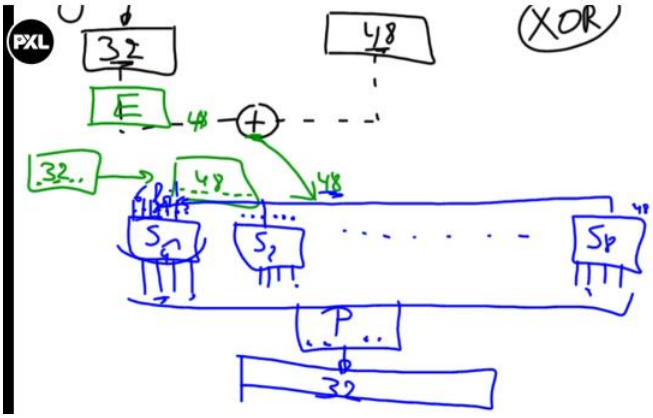


Afbeelding die we op het examen kunnen krijgen

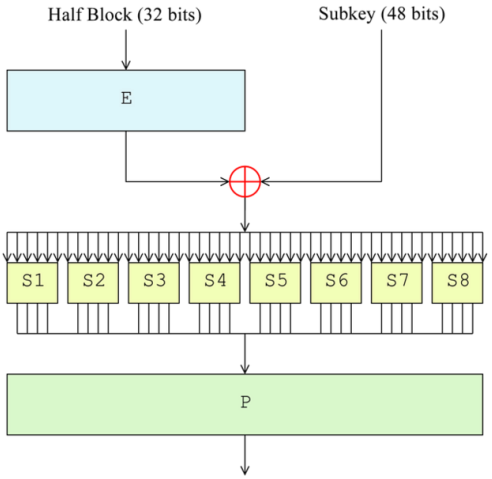
⌘ *Werking Feistelfunctie*



1. Feistelfunctie = vorige rechterblok (r_0), subkey (k_1)
2. Feistelfunctie = **32 bits, 48 bits**
3. Deze 2 blokken willen we **samen gaan XOR-en** maar dan moeten **beide blokken dezelfde aantal bits** hebben.
4. We sturen de blok van **32 bits** door een **expansie matrix**, E
5. E zet de bits op een **andere plaats** en neemt ook **bepaalde bits dubbel** om zo **48 bits** te krijgen.

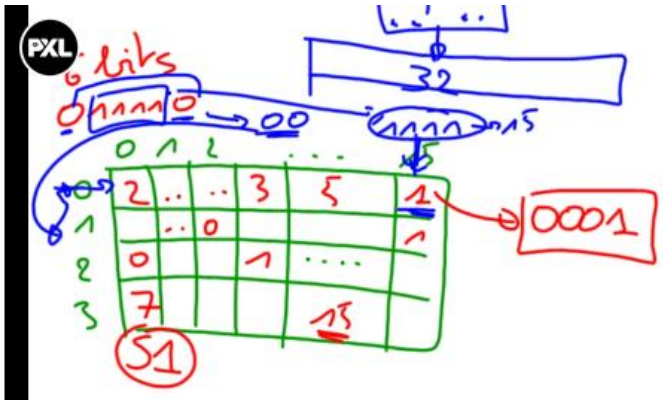


6. Uitkomst van XOR (**48 bits**) gaan we verspreiden over **8 S-boxen**
7. Deze bits worden gelijkmatig verdeeld dus niet op een andere plaats gezet
8. Elke S-box heeft **6 bits als input** en **4 bits als output**
9. De uitkomst van alle S-boxen worden weer door een **permutatie matrix** gestuurd
10. Hierdoor krijg je weer een **32 bit output**, de output van de functie.



Afbeelding die we op het examen kunnen krijgen

∞ Werking s-box



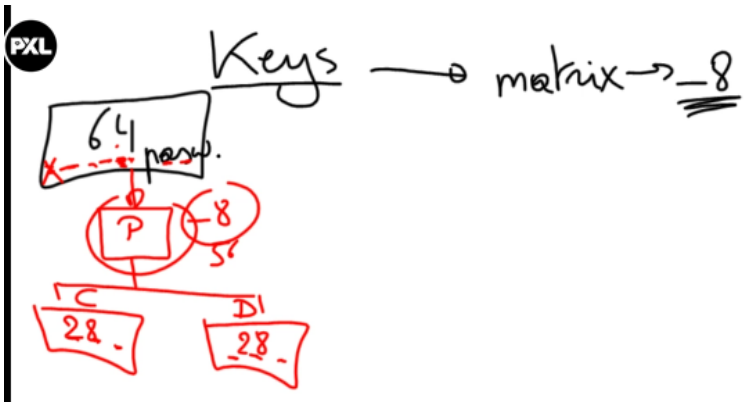
1. 6 bits input, 4 bits output
2. Een S-box is een **tabel met 4 rijen en 16 kolommen**
3. In de tabel staan allemaal getallen tussen **0 en 15**
4. De **eerste en de laatste bit** neem je samen, dit getal bepaald in **welke rij** je zit
5. De **middelste 4** neem je samen, dit getal bepaald in **welke kolom** je zit.
6. Nu nemen we het **getal** die in de juiste rij en kolom zit en wordt terug **binair omgevormd**

S-Box 1: Substitution Box 1

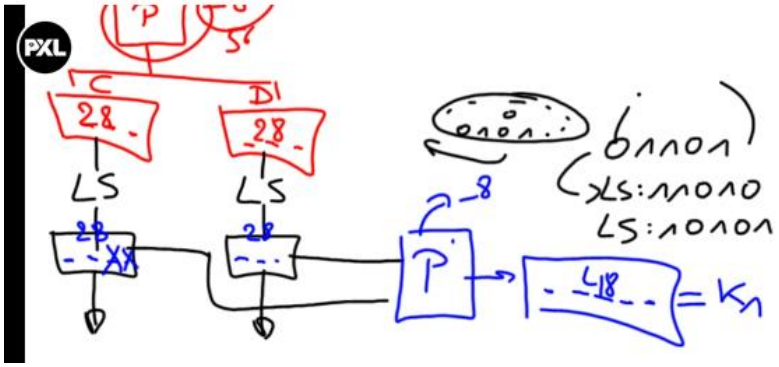
Row / Column	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Afbeelding die we op het examen kunnen krijgen

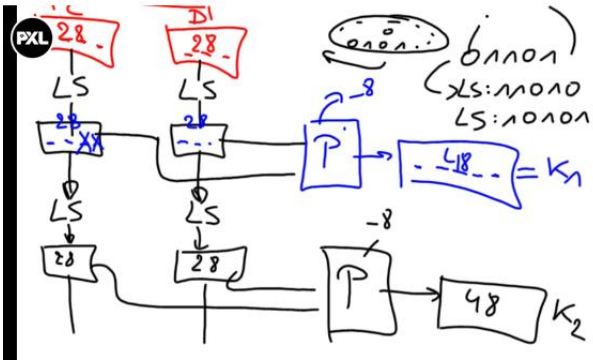
Werking subkeys



1. Masterkey; passwoord is **64 bits**
2. Als we bij de berekening van keys, de keys **door een matrix** sturen, zullen er **8 bits weggesmeten** worden.
3. De begin 64 bits worden door een **permutatiematrix** gestuurd.
4. De matrix **smijt 8 bits weg** en verplaatst de andere bits
5. Hierdoor krijg je **2 delen van 28 bits**, **C en D**



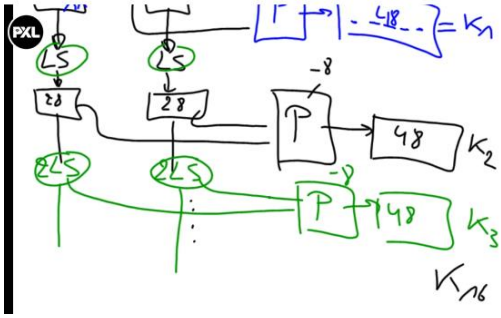
6. Er worden **16 iteraties** gedaan om **16 subkeys** te bekomen.
7. 1 iteratie **= Left Shift (LS) op beide blokken**.
    a. alle bits worden op een cirkel gezet en 1 plaats naar links verschoven
    b. 01101 wordt 11010
8. De **uitkomst** wordt naar **rechts** en naar **onder** gestuurd
9. Rechts: wordt door **een permutatie matrix** gestuurd en er worden **8 bits weggegooid**
10. Rechts: hierdoor krijg je uw eerste **subkey van 48 bits (k_1)**



11. Onder: **nieuwe iteratie**, terug LS enz.
12. Dit doe je 16 keer tot dat je **16 subkeys** hebt.

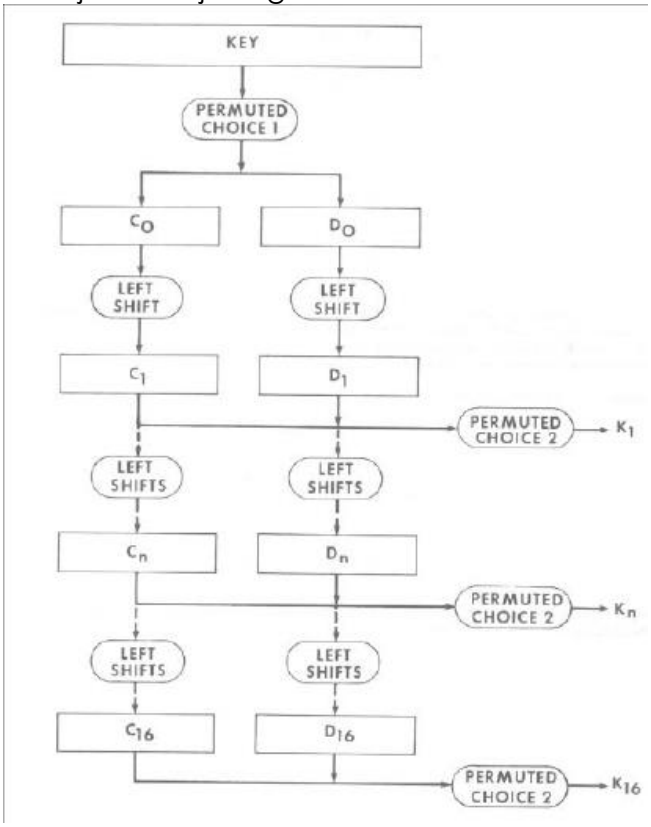
∞ *Details subkeys*

Bij sommige iteraties moet je **2x een LS** doen



Weak keys

Bijvoorbeeld uw C heeft 28 nullen ➔ 0000…
Als er een LS opgedaan wordt dan blijft dit nullen ➔ 0000…
Hetzelfde met allemaal 1ne ➔ 1111…
Subkeys blijven hetzelfde, zelfs bij 2x vercijfering



Afbeelding die we op het examen kunnen krijgen

3DES

- **Brute force** attack op DES is doenbaar!
  - Brute force: **alle** mogelijkheden uittesten
- 3 x DES met **3 verschillende sleutels** = onkraakbaar
- DES wordt in **elektronische chips** gemaakt
  - Voor 3DES kan men gemakkelijk dezelfde DES-chip gebruiken
- Encryptie simkaarten, encryptie routerverkeer tussen routers
- **Nadeel:** veel trager dan 1x DES

AES

- Advanced Encryption Standard
- **Opvolger van DES** – 2001
- Symmetrisch
- Is eigenlijk **Rijndael (2 Vlaamse onderzoekers)**
- **Sleutels van 128/192/256 bits** – onkraakbaar
- Op X (128 bits) wordt eerst een XOR functie toegepast met een subkey
- Dan volgen meestal 10 rondes, inclusief een eindronde

4. Asymmetrische cryptografie

Waarom bestaat het?

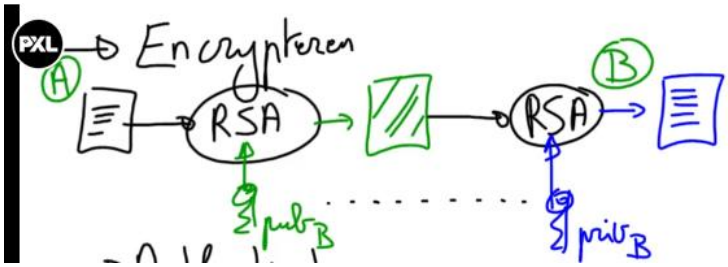
- **Door de nadelen van symmetrische cryptografie**: slechte sleutelmanagement

Inleiding

- **RSA**
- Werkt met **2 sleutels**: **public key en private key**
- Public key: mag iedereen weten, staat **op het internet**
- Private key: mag **alleen de eigenaar** weten
- **je kunt nooit de ene sleutel uit de andere afleiden**
- De 2 sleutels hebben een wiskundig verband met elkaar
  - **Wiskundige magic/trickery: priemgetallen & modulus**

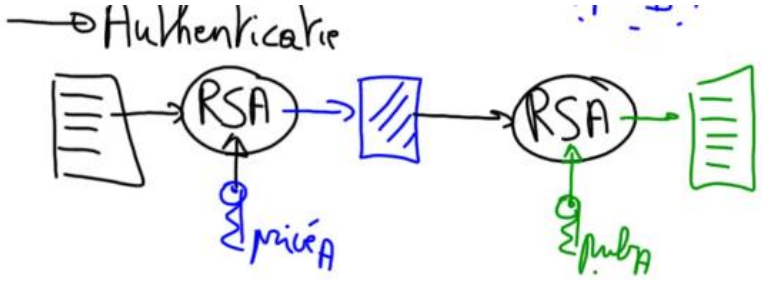
Werking - 2 basisberekeningen mogelijk

Encrypteren



- Een bestand wordt **door een algoritme** gestuurd (bv RSA)
- Bij het algoritme wordt de **publieke sleutel** van de **ontvanger (B)** gebruikt
- Nu krijgen we een geëncrypteerd bestand
- Om te decrypteren sturen we het terug door RSA
- We gebruiken de tegenhangede sleutel, **de private sleutel van de ontvanger (B).**
- Nu krijgen we terug het oorspronkelijk bestand
- **Als we iets gaan encrypteren met de publieke sleutel van de ontvanger kunnen we dit dus alleen maar decrypteren met de private sleutel van de ontvanger.**

Authenticatie



- **Bewijzen dat het bestand afkomstig is van de zender**
- We sturen een bestand door het RSA algoritme
- We gebruiken **de private sleutel van de zender (A)**
- Nu krijgen we een geëncrypteerd bestand
- Om te decrypteren sturen we het terug door RSA
- We gebruiken **de publieke sleutel van de zender (A)**
- Nu krijgen we het oorspronkelijk bestand
- **Iedereen kan dus dit bestand decrypteren omdat de publieke sleutel online staat maar door deze manier ben je wel zeker dat het bestand echt van de zender komt.**

4 belangrijke categorieën

Principes van wiskundige 'trickery'

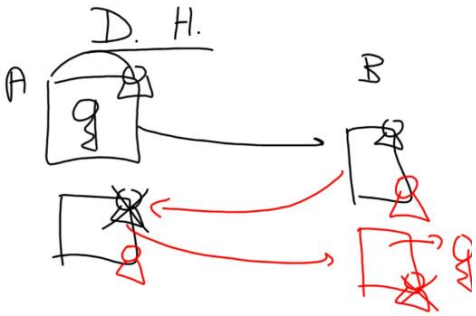
1. RSA ➔ factorisatie
2. D-H ➔ discrete logaritme
3. Knapzakprobleem
4. Elliptische curve cryptografie

Factorisatie

- Ik geef u een heel groot priemgetal
- **Zoek mij 2 kleinere priemgetallen dat als je deze 2 gaat vermenigvuldigen, je het heel groot priemgetal uitkomt**
- 21 = 7 * 3
- Bijna **onmogelijk** om beide kleine getallen te vinden
- **RSA** is gebaseerd op factorisatie

D-H

- Diffie-Hellman
- Berekenen van **discrete logaritme**
- **Eerste** public key cryptosysteem
- Nodig voor key management bij de symmetrische crypto **(DES-uitwisseling van secret key)**
- **Versturen van een geheime sleutel via open kanaal**



- **Een sleutel veilig van A naar B overbrengen**
- Een **slot van A** op de koffer zetten en naar B sturen
- **B** zal daar zijn eigen **slot** op zetten en terug naar A sturen
- **A** gaat zijn **slot eraf** halen en terug naar B sturen
- **B** gaat zijn **slot eraf** halen en kan nu de **koffer open** doen

Aanval op D-H

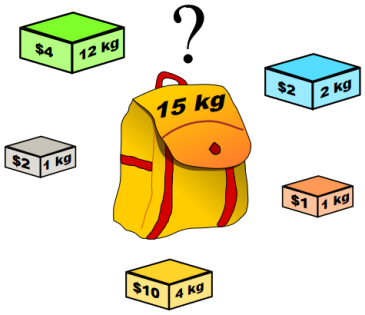
Man in the middle-attack (MITM)

- **C stelt zich tussen A en B op**
- We krijgen nu communicatie tussen A-C en C-B
- A-C spreken nu sleutel K1 af, en C-B spreken sleutel K2 af
- A en B hebben op die manier niet door dat iemand hun **berichten onderschept en eventueel kan veranderen**

Daarom **best authenticatie-methode** zoals digitale handtekeningen **gebruiken**

Knapzakprobleem

- Warenhuis vol met **items** die een **bepaald gewicht en waarde** hebben
- Je hebt een **rugzak** die maar een **bepaald gewicht aankan** (bv 5kg)
- Aantal items worden in de zak gestoken
  - **Geheim = welke objecten in de zak zitten**
  - **Bekendmaking = gewicht + lijst van items**
- **Opdracht:** zoek de beste combinatie van items zodat de rugzak het meeste waard is maar dat hij niet zal scheuren; onder het gewicht zal blijven.
- Zoveel items ➔ te veel combinaties om uit te rekenen

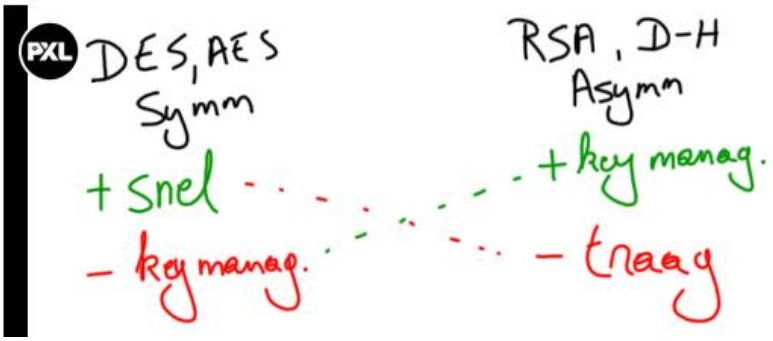


- Mag niet meer gebruikt worden, **niet meer veilig:** wiskundige zwakheid
- Er zit een achterdeur om de private key van de public key af te leiden
  - Lijst met elementen met uniek gewicht (public)
  - Enkele elementen in zak (private)
  - Totaalgewicht doorsturen (public)

Elliptische curve cryptografie

- ECC
- **Zeer snelle** berekeningen
- **Kortere sleutels**, kleinere rekentijd, hoge beveilig
- **Hoge beveiligingsniveaus**
- Gebaseerd op **ellipsen** met hun brandpunten
- Zeer wiskundig
- De toekomst

Voordelen en nadelen



5. Hashing

Inleiding

- **Belangrijkste algoritmes**
  - **MD5**
    - Aangeraden om niet meer te gebruiken ➔ collision met behulp van supercomputers
    - MD = Message Digest
  - **SHA1**, 2, …
- Heeft altijd een **vaste lengte**. Bv 256 bits
- **Bedoeling:** niet om bericht onleesbaar te maken, enkel **integriteit** waarborgen

Werking

- Sturen een **bestand door een algoritme** (MD5)
- Hierdoor krijgen we een getal, **hash-getal**
- Maakt niet uit hoe groot het bestand is, het getal blijft altijd 256 bits lang

Begrippen

Fingerprint

- Als je **1 bit verandert** aan het origineel bestand, dan krijg je een **volledig ander getal**
- **Integriteit waarborgen:** kunnen controleren of er iets veranderd is aan een bestand of niet

1-way-function

- Het is redelijk gemakkelijk om van een bestand naar een getal van 256 bits te gaan
- Je kunt **NOOIT** van het **256 bit** getal terug **naar** het **bestand** gaan

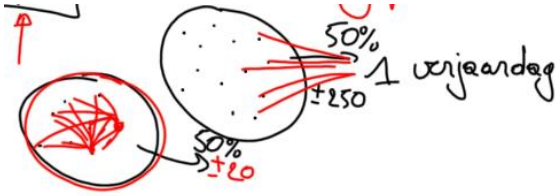
Collision

- **Oneindige hoeveelheid paswoorden en bestanden**
- Bij een **hash** van 256 bit zijn er 2^{256} **mogelijkheden**, wat veel is
- Maar **oneindig > 2^{256}**
- Dus we weten met zekerheid dat er **2 of meerdere bestanden** bestaan dat als we daar de hash van berekenen, ze **hetzelfde hash-getal** hebben.
- **Praktijk:** bijna onmogelijk om een collision te vinden, **collision-free**.

Paswoorden

- Er wordt een database bijgehouden met **username en hash**, **nooit de plain-tekst van een paswoord**
- Van het moment dat je uw **paswoord** ergens **ingeeft**, wordt er een **hash op uw ingaven berekent** en die wordt **gecontroleerd** met de hash in de **database**.
- Beste manier om te controleren of je het correct password hebt ingegeven
  - als er 1 bit aan het paswoord zou veranderen, heb je een heel andere hash.
- **Hackers** die in de database kunnen kijken vinden alleen de hash en zijn hier niks mee ➔ 1-way-function.

Birthday paradox



- **Hoeveel personen** moeten er van straat gehaald worden om **50% kans** te hebben dat er **1 iemand** tussen ziet die **dezelfde verjaardag** heeft **als u**?
  - **253 personen**
  - **1 op 1 matching**
- **Hoeveel personen** moeten er van straat gehaald worden om **50% kans** te hebben dat er tussen deze personen **2 personen** zijn die **dezelfde verjaardag** hebben?
  - **23 personen**
- **Conclusie**: het vinden van een exacte collision is zeer moeilijk, maar een willekeurige collision is iets minder moeilijk.

Birthday attack

- Het is bijna **niet te doen** om een 1 op 1 relatie te vinden, een **directed attack**.
- De kans dat je een **match** vindt bij **1 persoon** is **heel klein**
- De kans dat je een **match** vind in **heel de database** is **veel groter**

Wat is een match?

- Hackers gebruiken een dictionary (**dictionary attack**)
- In deze dictionary staan de **meest voorkomende paswoorden met hun hash**
- Ze gaan in een database met paswoorden de hash **vergelijken** met de hash van de dictionary
- Als er een hash **overeenkomt** dan ziet men in de dictionary welk password het is.

Rainbow table

- **Een hele grote dictionary die zo optimaal mogelijk bijgehouden wordt**
- Men neemt een paswoord ➔ berekent daar de hash op ➔ berekent daar vervolgens de inverse hash op ➔ hierop wordt weeral de hash en inverse hash berekent zodat er een hele keten gevormd wordt
  - Doelstelling: als je het eerste en laatste getal bijhoudt, dan kan je de hele keten opnieuw berekenen

Salting

- **Beveiliging tegen dictionary attacks**
- Als je voor het **eerste keer inlogt** wordt er een **random salt** gegenereerd
- Een getal van bv 256 bits
- Deze salt wordt in **plain-tekst bijgehouden** in de database
- **Password + salt = hash** die bijgehouden wordt in de database
- **Hackers:**
  - Door het principe van salting moeten hackers op elke mogelijke paswoord uit de dictionary **opnieuw de hash samen met de salt berekenen**
  - Omdat iedereen een andere salt heeft moeten ze het voor **elke gebruiker opnieuw doen**
  - Dit duurt zeer lang en is dit gelijk de **direct attack**

6. Digitale handtekening

= **om zeker te zijn van de identiteit van de verstuurder, zoals een geschreven handtekening**
= authenticatie

Voorwaarden

Niet vervalsbaar
- Alleen afzender kan zijn eigen handtekening zetten

Authentiek
- Ontvanger is zeker dat het van de afzender komt

Niet herbruikbaar
- Is een deel van de boodschap, kan niet onder een ander bericht gezet worden

Niet veranderbaar
- Na ondertekening van de boodschap is deze niet meer te veranderen

Niet verloochenbaar
- Afzender kan niet ontkennen dat de boodschap van hem kwam

Werking

Gaat ook met RSA ➔ **TRAAG**
Oplossing: digitale handtekening met hash

Digitale handtekening met RSA

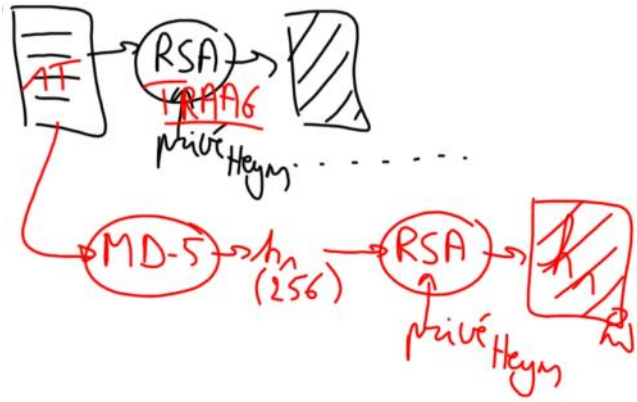
Een encryptie op het bericht met je **private key**
- Iedereen kan het decrypteren met je **publieke sleutel**, maar ze zijn er wel zeker van dat het bericht van jouw kwam.

⇨ **Traag!**

Digitale handtekening + encryptie met RSA

Geeft authenticiteit + vertrouwelijkheid
- A zet handtekening met **private key van A**
- A vercijferd dit met **public key van B**
- A verstuurt bericht
- B ontcijfert met **private key van B**
- B doet handtekening weg met de **public key van A**
- **Super traag!**

Digitale handtekening met hash

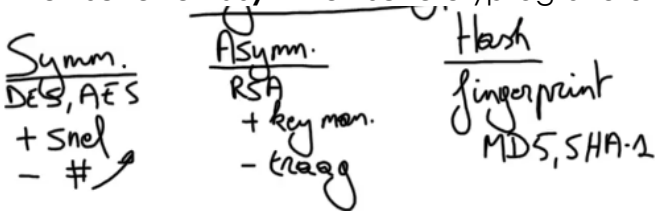


- Bestand door een hash algoritme sturen bv **MD5**
- Hierdoor krijg je een **hash**, zeer klein bestand
- Dit klein bestandje kan wel snel door een **RSA** gestuurd worden
- Hierdoor krijg je een gesigneerde hash, **geëncrypteerd hash**
- **Veel sneller!**

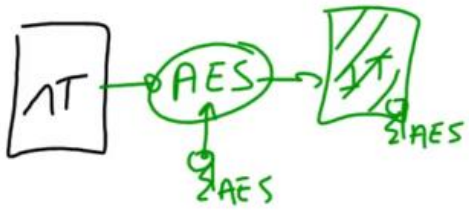
7. Hybride cryptografie

Waarom bestaat het?

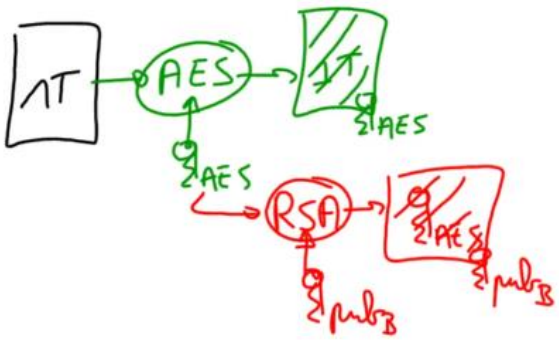
Om de **voordelen** van **symmetrische- en asymmetrische** cryptografie en **hashing** te **combineren**.



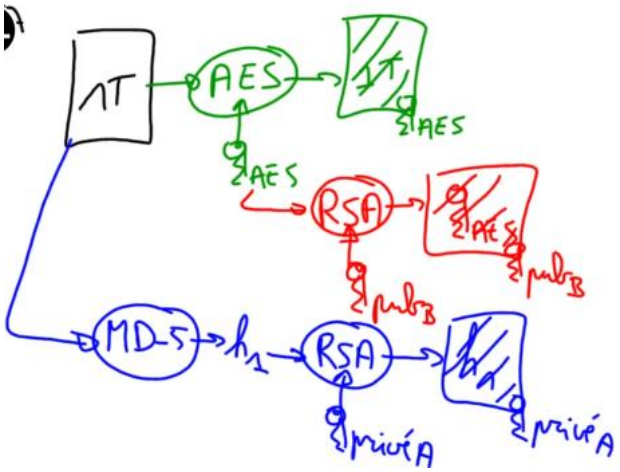
Werking - encrypteren



1. We nemen 1 groot bestand, 1TB
2. Dit sturen we door **AES met een AES-sleutel**
3. Hierdoor krijgen we een **bestand geëncrypteerd** met AES



4. De **AES-sleutel** sturen we door **RSA**
    a. Dit is een klein bestandje, dus het maakt niet uit dat RSA traag is.
5. We gebruiken de **publieke sleutel van B**
6. We krijgen een **geëncrypteerde AES-sleutel**

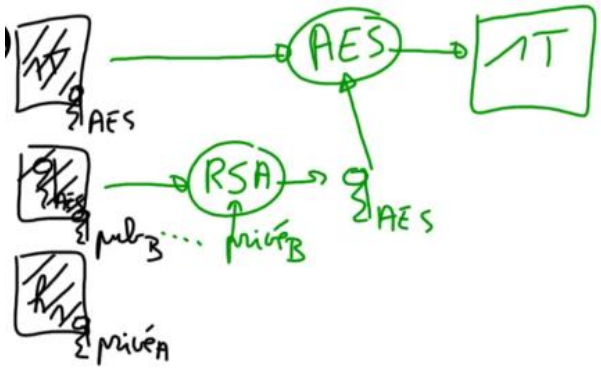


7. We sturen het **1TB bestand door een hashing algoritme**, MD5
8. Hier krijgen we een hash
9. Deze hash sturen we door **RSA**
10. We gebruiken de **private sleutel van A**
11. We krijgen een **geëncrypteerde hash**

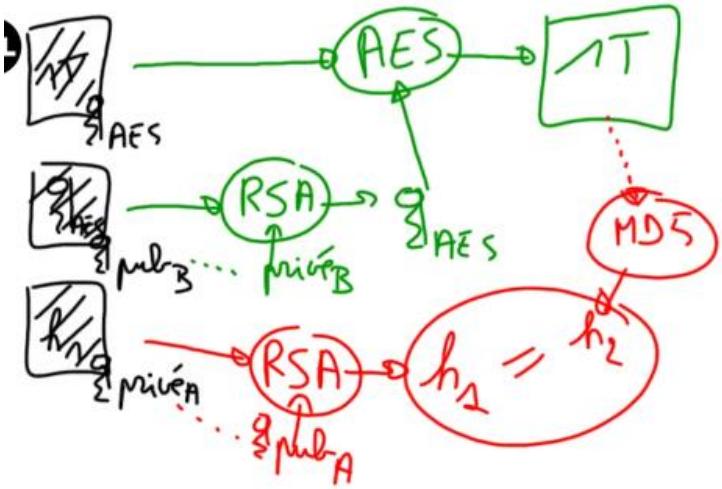
Het bestand, de sleutel en de hash sturen we alle drie naar B.

Werking – decrypteren

1. We sturen de **geëncrypteerde sleutel** terug door **RSA**
2. We gebruiken de **private sleutel van B**
3. We krijgen terug de **AES-sleutel**



4. Het **geëncrypteerde bestand** terug door **AES** sturen
5. We gebruiken de **gedecrypteerde AES-sleutel**
6. We krijgen het **oorspronkelijk bestand**



7. De **geëncrypteerde hash** wordt terug door **RSA** gestuurd
8. We gebruiken de **publieke sleutel van A**
9. We krijgen terug **het hash getal**
10. Op het **oorspronkelijk bestand** wordt een **MD5 berekend**
11. Hierdoor krijgt hij **een hash getal**
12. **2 hash getallen** moeten **overeenkomen**

Principes

- B is de enige die het bestand terug kan lezen **(private key B)**
- B is zeker dat het van A afkomstig is **(public key A)**
- B is zeker dat er onderweg niks veranderd is **(hash komt overeen)**
- A kan niet meer ontkennen dat A het bestand naar B heeft gestuurd **(hash komt overeen)**

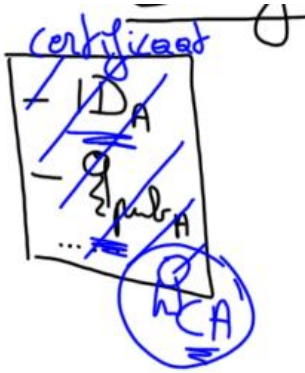
8. Certificaten – PKI

PKI = public key infrastructure
De verzamelnaam voor de uitgifte en beheer van certificaten.

Veiligheidsprobleem hybride cryptografie

- **Alle publieke sleutels** staan op het **internet**
- Je weet niet met echte zekerheid of de publieke sleutel echt van de juiste persoon is (= **geen koppeling**)
- **Oplossing** = certificaten (koppeling wordt gegarandeerd)
    - Certificaat verbindt een publieke sleutel aan en individu of organisatie
    - = elektronische identiteitskaart

Certificaten



Bevat 2 belangrijke items

- Een persoon zijn **ID**
- Een persoon zijn **public key**

Belangrijkste eigenschap

- Certificaat **vertrouwen**
    - Gebaseerd op de X.509v3-standaard, ze zijn dus gestandaardiseerd.
- Wordt altijd uitgereikt door een **CA**
    - Certification Authority
    - Een bedrijfje dat certificaten uitdeelt tegen betaling
- Dit bedrijf **signeert het certificaat** (= digitale handtekening)
- Certificaat geëncrypteerd met **CA zijn private key**

Soorten certificaten

Persoonsgebonden certificaten

- Gekoppeld aan 1 persoon

Rolgebonden certificaten

- Rol die een persoon vervult binnen een organisatie

Toepassingsgebonden certificaten

- Applicaties die zelf gegevens (veilig) versturen, gaan die gegevens ondertekenen
- Bv een server die log-info iedere dag doorstuurt naar een aantal admins

SSL-servercertificaat

- Beveiligde verbinding tussen client en server
- Identificatie van het IP-adres + domain name van de server
- Gebruiker zit op de juiste website en gegevens die hij invult kunnen niet door derden worden afgeluisterd

Signing certificaat

- Ondertekening van software ➔ code is afkomstig van ondertekenaar

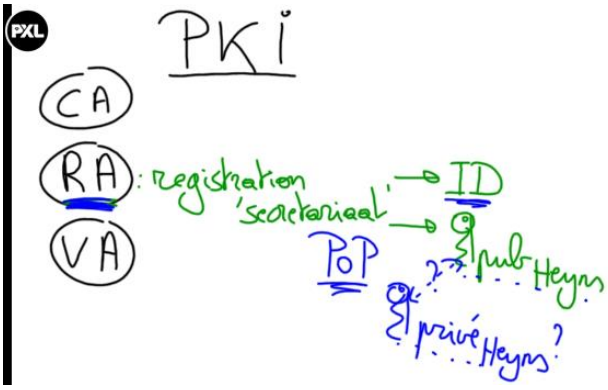
PKI componenten

PKI = De verzamelnaam voor de uitgifte en beheer van certificaten.

RA

= registration authority

- **Registratie van gebruikers**
- **Vaststellen identiteit van gebruikers**



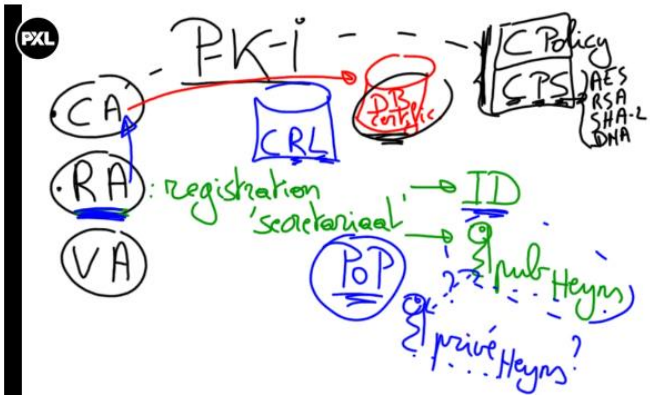
Hier ga je naartoe als je een **certificaat wilt hebben**

- Er moeten **2 items gecontroleerd** worden
  - ID
  - Public key
- **ID** kan op **verschillende manieren** gecontroleerd worden
  - Pas, vingerafdruk, irisscan, DNA-scan, …
  - Hangt af van het niveau van het certificaat (beveiligingsniveau)
- **Public key** wordt gecontroleerd aan de hand van **POP**
  - Proof of possession
  - Ben ik in bezit van de tegen hangende private key?
  - Er wordt een simpel bestand geëncrypteerd met de publieke sleutel die gegeven was
  - Als je deze kan decrypteren met uw private key is er bewijs dat je eigenaar bent van de public key
- **De RA geeft deze gegevens door aan de CA**

CA

= Certification authority

- **Aanmaak certificaten**
- **Uitgave en distributie**
- **Intrekken van certificaten, …**



- Gaat een **echte certificaat maken**
- Deze certificaat wordt in **een databank gestoken**
- Er bestaan **2 beleidsdocumenten** waarin de **standaarden** staan hoe deze certificaten gemaakt moeten worden

CP

- Certification policies
- **Higher level management richtlijnen**
- Regels en normen die aan CA wordt gesteld
- Beleidsniveau
- Gestandaardiseerd document: hoe willen ze uitgifte aanpakken, certificaat formaat, contractuele bepalingen.

CPS

- Certification practice statements
- Door IT-afdeling gebruikt
- **Hoe gaan we certificaten maken om de policies te realiseren?**
- AES, RSA, SHA2, DNA voor ID, …

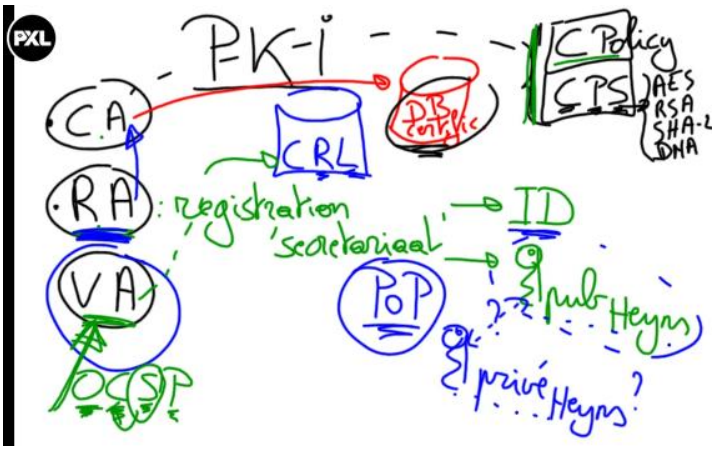
CRL

- Certificate revocation list
- **Een databank** met certificaten die op de **blacklist** staan

VA

= validation authority

- **Verifiëren of certificaat al dan niet ingetrokken of opgeschort is**

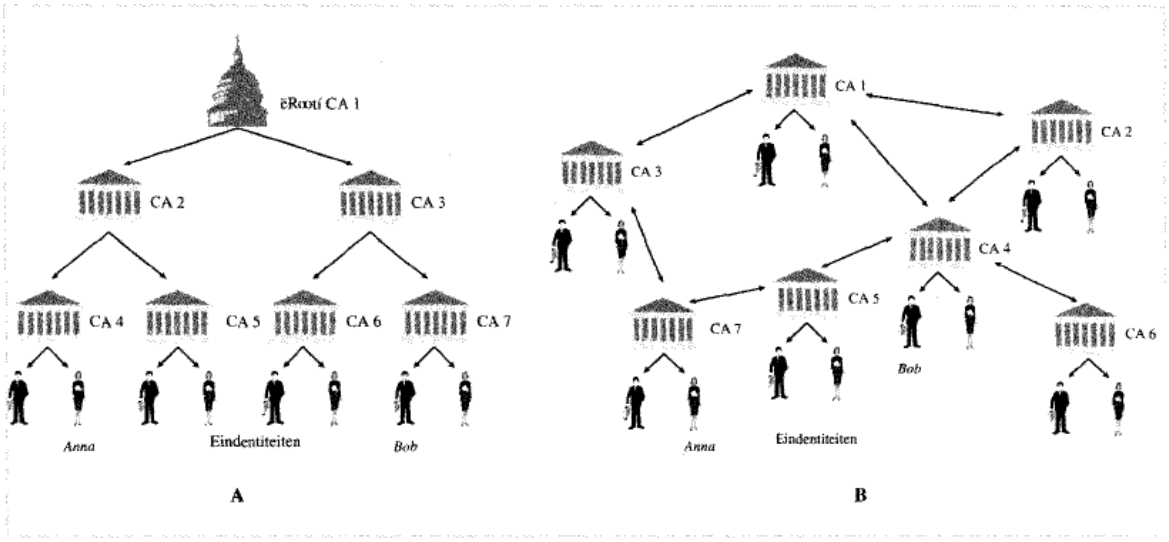


- **Checken van certificaten**
- **Antwoord**: ok, ongeldig certificaat of ik weet het niet
- Gebruikte protocol: **OCSP**
  - Online certificate status protocol
  - Online realtime controle van certificaat (freshness)
  - Good: niet op CRL
  - Revoked: op CRL
  - Unknown: OCSP kent CA niet

Architectuur

2 soorten architectuur

- Hiërarchisch
- Mesh



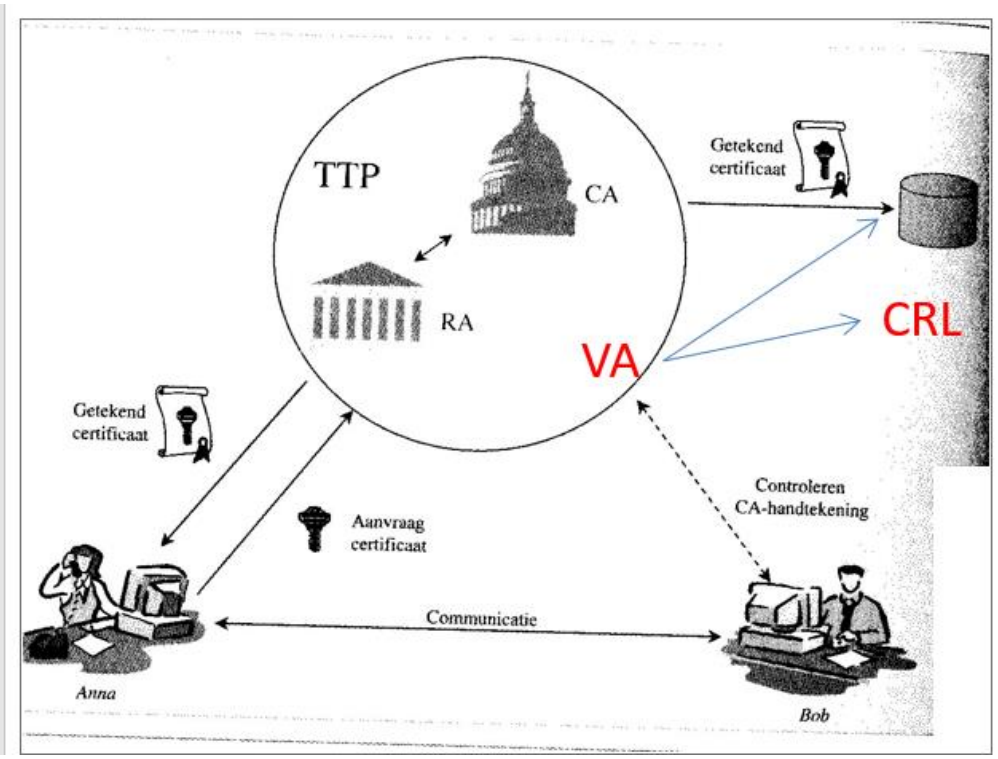
Hiërarchische structuur

- Veel toegepast
- **Iedereen kent public key van root CA**
- **Root CA:**
  - Signeert CA's voor onderliggende level
  - Die dan signeren voor de level daaronder
  - En uiteindelijk het certificaat kunnen uitreiken op het laatste level
- **Voordeel:** duidelijkheid in opbouw vertrouwensrelatie
- **Nadeel**: als de vertrouwen tussen 2 CA's wegvalt ➔ hele structuur valt in elkaar

Mesh structuur

- Alle CA's hebben **gelijkwaardige relaties** met elkaar
- **Geen root of hiërarchie**
- Iedereen **kent de public key van de lokale CA** waar hij bij aangesloten is
- Verificatie kan langs verschillende paden
- **Cross-certificering!**
  - Erkenning van 2 CA's in een verschillende omgeving (= zij vertrouwen elkaar)
  - op basis van CPS
- **Voordeel:** flexibiliteit, uitvallen van CA is geen probleem
- **Nadeel:** niet duidelijk hoe de vertrouwensrelatie juist is

Certificaat aanvragen



1. A geeft haar **public key + persoonlijke gegevens aan RA**
2. Controle ➔ ok? Volgende stap
3. Certificaat voor A wordt **aangevraagd bij CA**
4. CA keurt dit goed en **maakt de certificaat aan + ondertekend deze**
5. De certificaat wordt **teruggestuurd naar A**
6. De certificaat van A wordt **in de databank gestoken**

7. B is een gebruiker en **krijgt de certificaat van A** binnen
8. B **controleert de certificaat met de public key van CA** om te kijken of het echt van CA komt
9. B **kijkt of dat de certificaat van A geldig is via VA**
10. **Nu heeft B de public key van A** en is hij zeker dat de key echt van A komt

Certificaat intrekken

Wanneer?

- Vervallen certificaten
- Certificaten die gehackt zijn geweest

Gevolg

- Certificaten worden uit de databank gehaald
- Deze worden in de **CRL** gestoken
- Intrekking wordt gepubliceerd door CA

Key Escrow

= het beheren van private keys door CA

VS ziet crypto als een **militair aspect**

- Verbiedt de uitvoer van sterke crypto-systemen
- Zwakke crypto mag wel (korte sleutels)
- Sterke crypto is soms toegestaan, maar dan moet de **gebruiker zijn private sleutel** ook **laten beheren door CA**
- Als ze een **gerechtelijk bevel** hebben moet CA de private key geven ➔ zo kunnen ze communicatie bekijken. **(= big brother toestanden)**
    - nodig in de strijd tegen terrorisme
- Nooit de private signeringskey uit handen geven!

9. Web security

Soorten bedreigingen

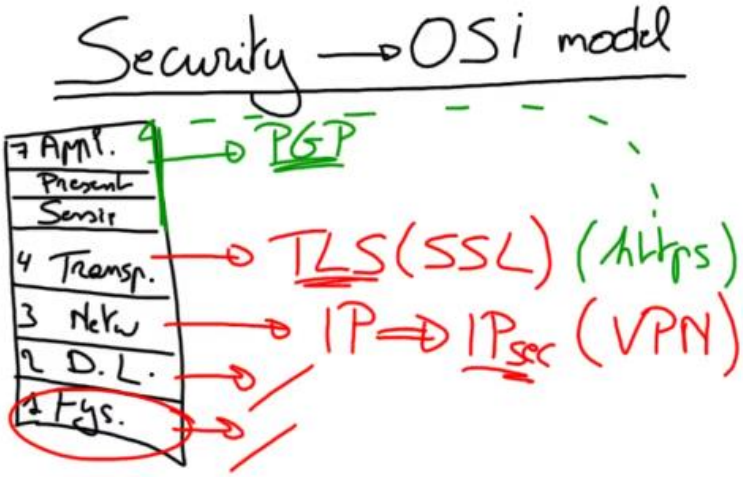
	Threats	Consequences	Countermeasures
Integrity	•Modification of user data •Trojan horse browser •Modification of memory •Modification of message traffic in transit	•Loss of information •Compromise of machine •Vulnerabilty to all other threats	Cryptographic checksums
Confidentiality	•Eavesdropping on the Net •Theft of info from server •Theft of data from client •Info about network configuration •Info about which client talks to server	•Loss of information •Loss of privacy	Encryption, web proxies
Denial of Service	•Killing of user threads •Flooding machine with bogus requests •Filling up disk or memory •Isolating machine by DNS attacks	•Disruptive •Annoying •Prevent user from getting work done	Difficult to prevent
Authentication	•Impersonation of legitimate users •Data forgery	•Misrepresentation of user •Belief that false information is valid	Cryptographic techniques

Web verkeer beveiligen in het OSI model

Applicatie laag
- PGP, SET, S/MIME

Transport laag
- SSL/TLS

Netwerk laag
- IPsec



10. E-mail (Applicatie laag)

- Een veel gebruikte toepassing op het internet
- **2 systemen**
  - S/MIME
  - PGP
- Beiden gebruiken het principe van **hybride cryptografie** om e-mails te versturen
- **Verschil:** gebruik van certificaten

S/MIME

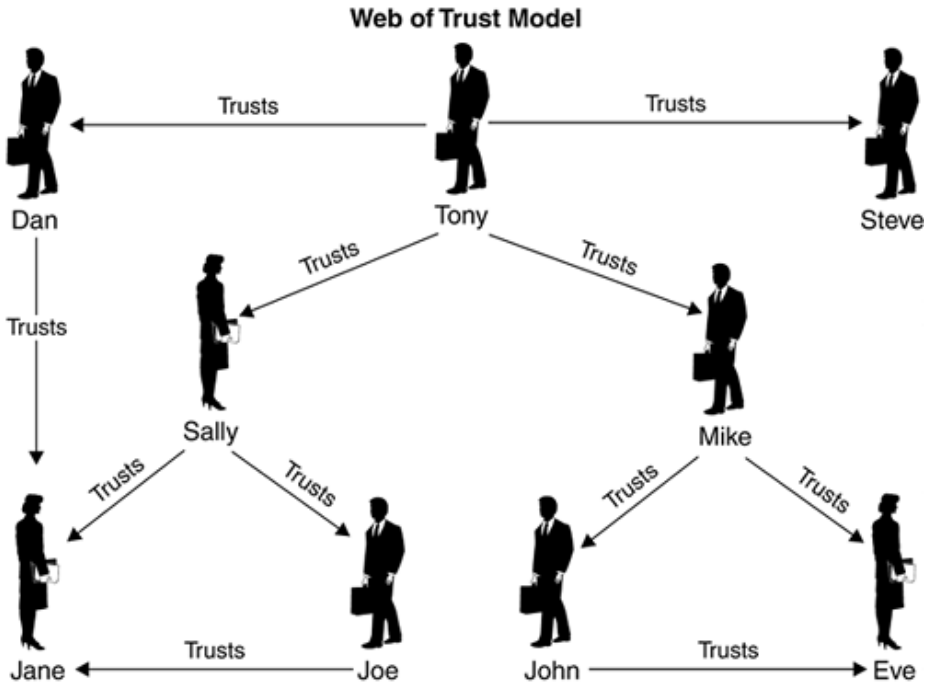
- Secure Multipurpose Internet Mail Extension
- Gebruikt certificaten die uitgereikt zijn door een **CA**
- Wordt meestal gebruikt in **web-based** mailing systemen

PGP

- Pretty Good Privacy
- Gebruikt zelfgemaakte certificaten die vertrouwd worden door een **WoT**
- Wordt meestal door **IT'ers** gebruikt

WoT

- Web of Trust
- Een gedecentraliseerde manier om certificaten te vertrouwen
- **Niet afhankelijk van een CA**
- 'Als x-aantal van uw vrienden een certificaat vertrouwen, dan vertrouw jij dit ook'
- Als iemand een certificaat vertrouwt dan gaat hij dit certificaat singen.
- Als er 1 rotte appel in het netwerk zit, valt heel het netwerk uiteen.



11. SET (Applicatie laag)

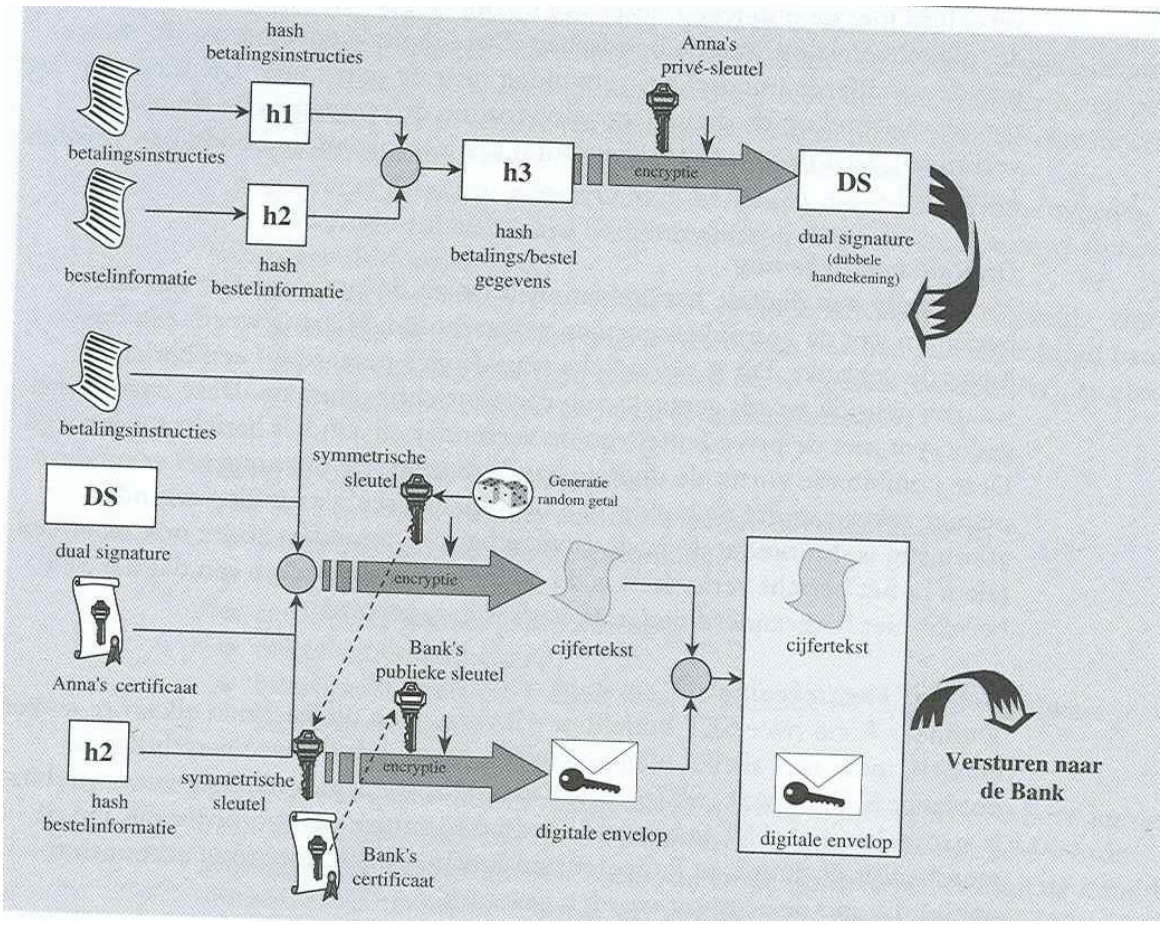
<Niet in de les behandelt maar staat wel in de PowerPoint>

- Secure electronic transaction
- Standaard voor **betalingen via creditcard over het internet**
- Zowel **verkoper als koper** moeten over een **certificaat** dat uitgegeven is door **CA** beschikken.
- **Encryptie:** hybride cryptografie
- **Dubbele handtekening**

Dubbele handtekening

De zender wilt 2 berichten versturen in 1 envelop, maar die mogen beiden niet gelezen worden door 1 persoon.

- A maakt van 2 berichten een hash (h1, h2)
- h1 en h2 worden samengevoegd tot h3
- h3 wordt vercijfert met de private key van A
    - = dubbele handtekening
    - RSA
- Er worden 2 geheime sleutels gemaakt DES1 en DES2
- Betalingsinstructies, dubbele handtekening, certificaat van A en h2 worden versleutelt met DES1 (X1)
- Bestelinformatie, dubbele handtekening, certificaat van A en h1 worden versleutelt met DES2 (X2)
- A stuurt naar Bank (= acquirer)
    - X1 + DES1 versleutelt met de public key van de bank
- A stuurt naar de merchant
    - X2 + DES2 versleutelt met de public key van de merchant



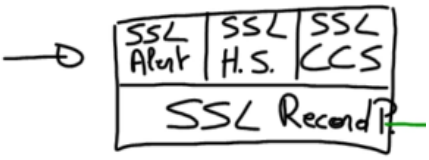
3Dsecure

- 3 Domains Secure
- Het principe van **SET** is technisch enorm goed, zeker met de dubbele handtekening, maar wordt **in de praktijk niet meer gebruikt**.
- **Visa** is met 3Dsecure afgekomen, wat technisch **minder goed** in elkaar zit **dan SET**
- Door de grote kracht van de **Visa garantie** hebben de banken gekozen voor dit systeem t.o.v. het SET-systeem.
- In de praktijk, indien je een e-shop met betalingen wilt opzetten, ga je waarschijnlijk via Ogone werken die het 3Dsecure systeem voor u gaat uitwerken zodat je onder de Visa garantie valt.
- Het werkt ongeveer als SET, je gebruikt niet echt certificaten maar een **creditcard met een lezer om handtekeningen te plaatsen.**

12. SSL/TLS (Transport laag)

Inleiding

- Secure socket layer / transport layer security
- SSL is ontwikkeld door Netscape
- **Beveiligt verbinding tussen webbrowsers en webservers**
  - **Https** = http + SSL
- Bestaat uit 2 lagen van protocollen
  - Bestaat uit 3 hulpprotocollen
    - **SSL Alert protocol**
    - **SSL Handshake protocol**
    - **SSL Change Cipher Spec protocol**
  - **SSL record protocol**



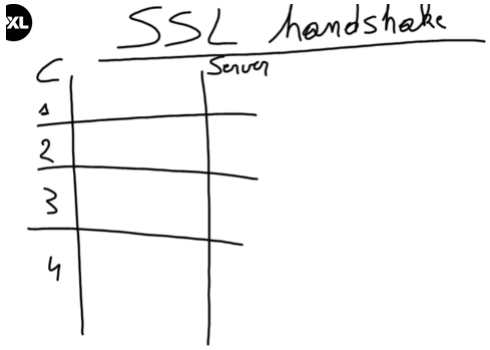
SSL Alert

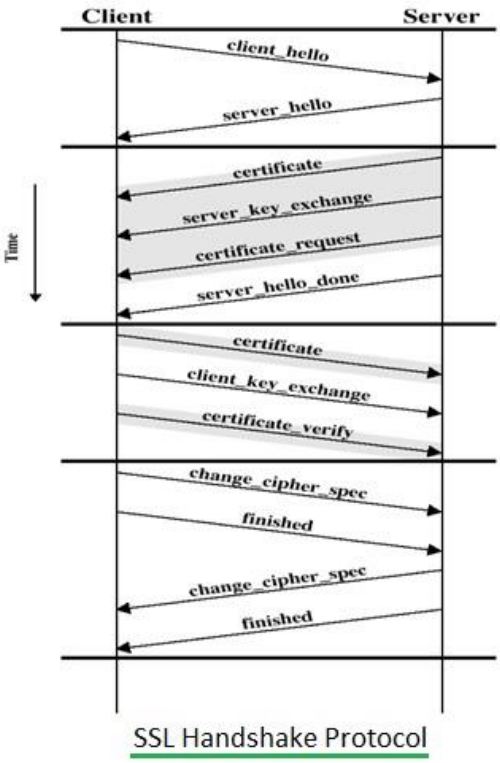
- Van het moment dat er iets **mis is gelopen** tijdens de communicatie wordt er een alert gestuurd.
- Om **communicatie af te breken of om verder te beslissen** wat er gedaan moet worden.

SSL CSS

- Change cipher spec
- In de handshake wordt op een bepaald moment gezegd dat de client en de server vanaf nu **secure gaan communiceren.**

SSL Handshake Protocol

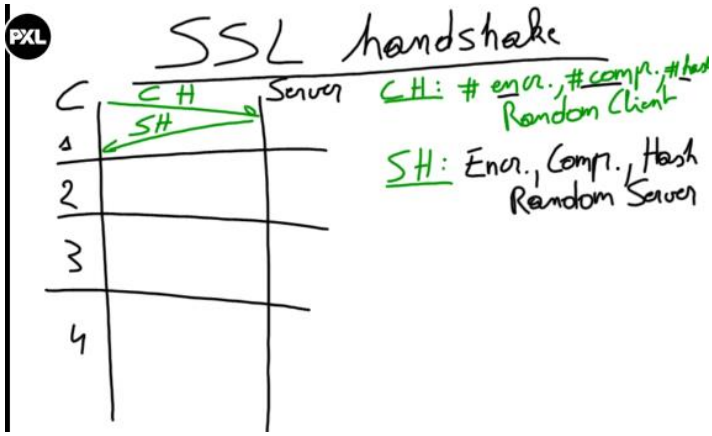


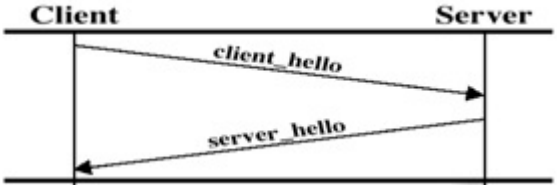


SSL Handshake Protocol

- **Authenticatie + onderhandelen** over te gebruiken cryptoalgorithme
- Moet in orde zijn **voordat** er beveiligde communicatie kan bestaan
- 4 fasen

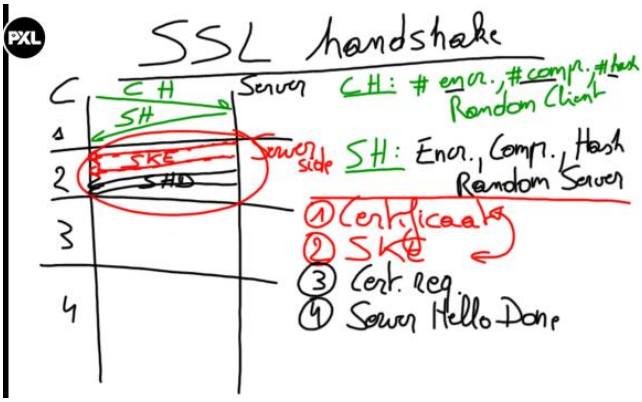
Fase 1





1. Client wilt veilig gaan communiceren met de server en stuurt een **client hello**
2. In een client hello zit:
    a. Welke vormen van **encryptie** de client kan doen
    b. Welke soorten van **compressie** de client kan
    c. Welke soorten van **hashing** de client kan doen
    d. Een **random getal** van de client
3. Server reageert met een **server hello**
    a. In een server hello zit **welke soort** encryptie, compressie, hashing de server heeft **gekozen**
    b. En een **random getal** van de server

Fase 2

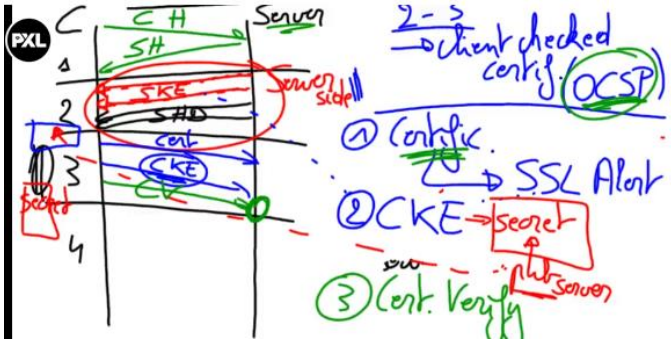


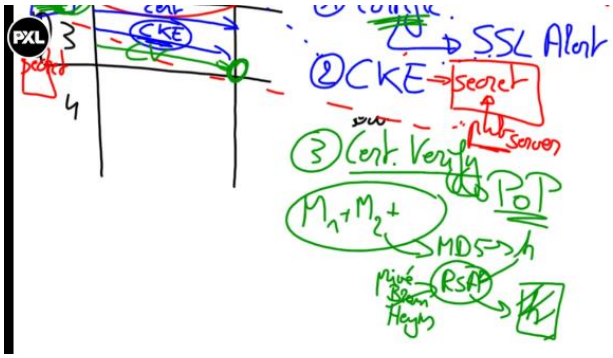
1. De server stuurt zijn **certificaat** door
2. Als de server **geen certificaat** heeft, stuurt de server een **server key exchange**
    a. Uitwisselen van de publieke sleutel van de server
3. De server vraagt naar het **certificaat van de client**, **certificate request**
4. Stuurt een **server hello done**
    a. Laat de client weten dat de server klaar is en dat de client een bepaalde tijd heeft om te antwoorden want anders zal de server de connectie verbreken (tegen DOS aanvallen)

Tussen fase 2 en 3

1. De client **controleert het certificaat** van de server
    a. Gebruik van **OCSP** (online certificate status protocol)

Fase 3





1. De client stuurt zijn **certificaat**
    a. Geen certificaat? ➔ **SSL Alert**
2. De client stuurt een **client key exchange**
    a. De client maakt een geheim
    b. Uit dit geheim kan **een master secret** gemaakt worden waaruit de encryptie- en hashingsleutels gehaald worden.
    c. De client **encrypteert het geheim met de public key van de server**

3. De client stuurt een **certificate verify** bericht
    a. De server wilt het certificaat van de client controleren op geldigheid maar wilt hier **geen OCSP** voor toepassen omdat dit veel te lang duurt
    b. Dit wordt gedaan aan de hand van een **PoP (= proof of possesion)**; weten dat de persoon bij de publieke sleutel de bijhorende private sleutel heeft
    c. De client berekend een **hash op alle berichten** die tot nu toe uitgewisseld zijn
    d. Deze hash wordt **door RSA** gestuurd met de private sleutel van de client
    e. Hierdoor krijgen we een **gesigneerde hash wat het certificate verify bericht is**
    f. **De server berekent ook een hash** van alle uitgewisselde berichten
    g. De server haalt de public key van de client uit het certificaat dat de client gestuurd had. Hiermee kan de server de gesigneerde **hash** decrypteren en **vergelijken met zijn eigen berekende hash**.

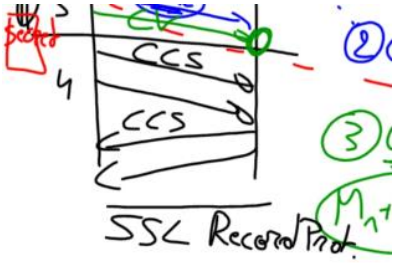
Master secret

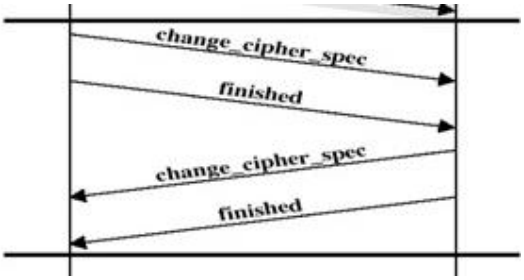
1. Het **geheim** uit de client key exchange wordt **gecombineerd** met de **random getallen** van de client en server
2. Hierdoor krijgen we een **master secret**
3. Uit deze master secret halen we de **AES en HMAC sleutels**
4. **De random getallen worden ook gebruikt om replay-attacks tegen te gaan**

Replay-attacks

- Ook wel **bandrecorderaanval** genoemd
- Client A heeft communicatie met server B
- Een **tussenpersoon neemt de communicatie op**
- Nu gaat de tussenpersoon zich als persoon A voordoen en communicatie maken met de server
- **De tussenpersoon kan exact hetzelfde zeggen als persoon A**. bijvoorbeeld een pizza bestellen
- **Door de master secret** kan deze attack **niet meer** gebeuren omdat de **server altijd** een **ander random getal geeft aan de client**. Hierdoor **klopt** de **opgenomen communicatie niet meer.**

Fase 4



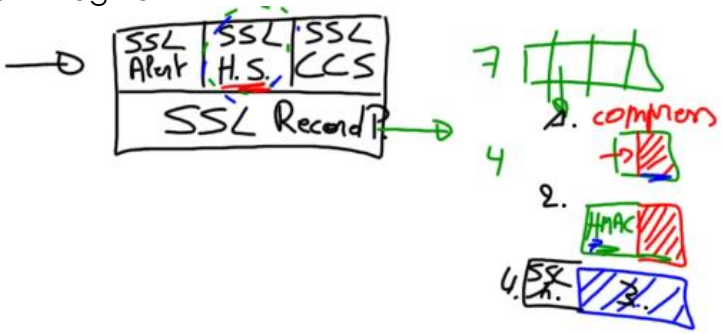


1. Client zegt: **change cipher spec**
2. Client zegt: **finished**
3. Server zegt: change cipher spec
4. Server zegt: finished

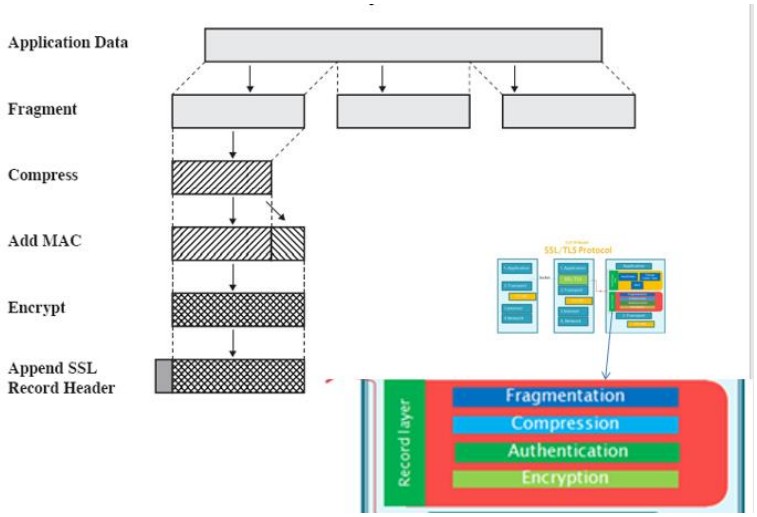
SSL Record Protocol komt nu in actie

SSL Record Protocol

Vertrouwelijkheid en berichtintegriteit



- Data van laag 7 wordt doorgestuurd naar laag 4 waar SSL aan het werken is
- Laag 4 gaat de **data in stukjes kappen**
- Elk stukje wordt **gecomprimeerd**
- Elk stukje krijgt een **HMAC**
- Dit volledig stuk wordt **geëncrypteerd**
- Bij dit stuk wordt een **SSL header** aan toegevoegd. Hierin staat hoe het geëncrypteerd is.
- Dit gebeurt voor elk stukje wat nadien ook volledig geëncrypteerd wordt.

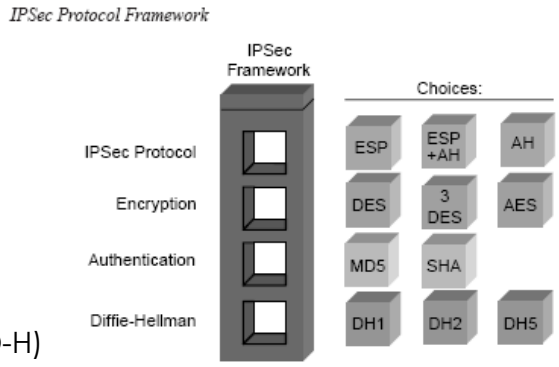


13. IPSec (Netwerk laag)

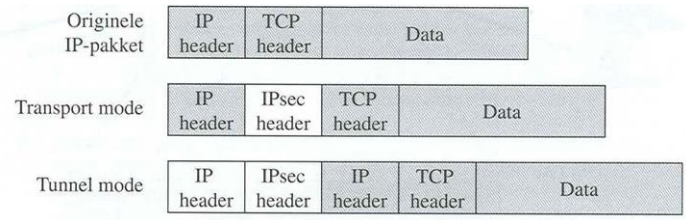
VPN

- Virtual private network
- **Eigen afgeslote netwerk dat gebruik maakt van het internet**
  - Remote access van medewerker tot bedrijf
  - E-commerce: gedeelte van het netwerk openstellen voor klanten
  - Extranet: partners die mee kunnen inloggen om bv voorraad op te vragen
- **Private network:** beveiligd door allerlei mechanismen, dus privé
- **Virtual:** geen directe lijn maar over het internet door gebruik te maken van **tunneling** techniek

IPSec

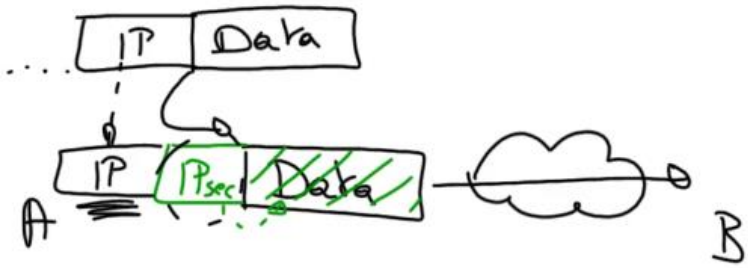


- Is een **framework** van open standaarden
- Wordt **vooral gebruikt** om **VPNs** te maken
- Keuze tussen verschillende algoritmes (4 keuzes)
  - Welke **IPSec protocol**
  - Welke **symmetrische encryptie**
  - Welke **authenticatie**
  - Hoe gaan we de **gedeelde sleutel** afspreken (D-H)
- **2 verschillende modes**
  - Transport mode
  - Tunnel mode



Transport mode

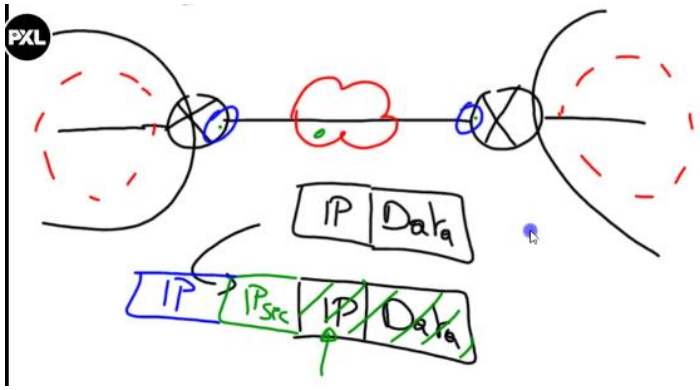
Het behouden van het originele IP header



- We nemen het stukje **data** die we volledig gaan **encrypteren**
- Daarvoor zetten we een **IPsec header**
- Deze header zegt hoe dat we het datastuk hebben geëncrypteerd
- Voor de IPsec header zetten we een **header met de IP-adressen**
- **Mensen van buitenaf** kunnen **niet zien** welke **data** je aan het sturen bent maar **wel waar het vandaan komt en waar het naartoe gaat** (door de IP-adressen)
  - **Veilig naar data, niet veilig naar privacy**

Tunnel mode

Alles, inclusief IP header, verbergen en het een nieuwe IP header geven



- Gemaakt om van de **ene gateway naar de andere gateway te communiceren**
- De oorspronkelijke **data met de IP-adressen** worden **geëncrypteerd**
- Daarvoor zetten we een **IPsec header**
- Deze header zegt hoe dat we het datastuk met de hebben geëncrypteerd
- Voor de IPsec header zetten we **een header met de IP-adressen van de gateways**
- **Veilig naar data en privacy toe**

14. Firewall

Wat is een firewall?

- **Niet volledig waterdicht**
- Het is een programma dat de **doorgang tussen het internet en de computer** vormt.
- Met een firewall kun je het **internetverkeer** van je eigen **computer bewaken** en de computer beschermen van verschillende gevaren zoals virussen
- **Een onderneming** met een intranet installeert een firewall met **zijn eigen regels** om ervoor te zorgen dat de werknemers van het bedrijf niet op bepaalde sites kunnen
- **Controleert meestal op:**
  - Bronadres
  - Doeladres
  - Poortnummers
- Extra functionaliteiten in sommige firewalls:
  - Filteren van inhoud (blokkering van toegang tot bepaalde sites)
  - Virtual Private Networks
  - Network Address Translation
  - Load Balancing

Soorten firewalls

Algemeen 3 categorieën

1. Firewalls die **pakketten filteren**
   a. **L4 OSI model**
2. **Stateful packet-filter** firewalls
   a. **L5 OSI model**
3. Firewalls die als **proxy** fungeren
   a. **L7 OSI model**

L4 firewall

Packet-filter firewall

- **Meestal routers met pakketfilterfuncties**
- **Voordeel**
  - Gemakkelijk te implementeren, meestal heeft men al een router
- **Nadelen**
  - Niet echt voorbereid op **DoS-aanval**
  - Kunnen meestal **geen sessietoestanden** bijhouden
    - **Poorten boven 1024** moeten open blijven staan om de TCP-sessies en de onderhandelingen netjes te laten verlopen
    - **Geen goed idee om open poorten te hebben**
  - Grote ACLs kunnen bij grote belasting **netwerkprestaties verminderen**

L5 firewall

Stateful Packet-filter Firewall

- Dit is meestal het **minimumvereiste** voor een firewall voor een netwerk
- Doet **pakketfiltering + houdt sessies** en verbindingen **bij** in **tabellen**
  - Speciaal ontworpen **tegen DoS attacks**
  - Kunnen nu de **poorten** boven 1024 standaard **gesloten** laten en alleen openen als het nodig is.

Verschil tussen L4 en L5

- Een ACL is een lijst met allemaal regels; wie mag wat?
- **Standaard** wordt alle verkeer dat **van het internet** komt en dat **naar het intern netwerk** wilt, **geblokkeerd.**

Oplossing van gewone packetfilters (L4)
Alle verkeer dat van het internet naar binnen wilt en een destination poort heeft dat groter is dan 1024, toelaten.

Oplossing van stateful packetfilters (L5)
Houden de sessie bij.
Laten enkel het verkeer binnen als de sessie gestart was in het interne netwerk.

L7 firewall

Proxyfirewalls

- Inspecteert het verkeer niet alleen op **netwerk- en sessieniveau** maar ook nog eens op **toepassingsniveau**
- **Deep packet inspection**
  - HTTP pakket arriveert ➔ doorgegeven aan het HTTP-proxyprocedure
  - FTP pakket arriveert ➔ doorgegeven aan het FTP-proxyprocedure
- Is in principe dus veiliger want **het begrijpt** ook de **toepassingsprotocols** (HTTP, FTP, SMTP, POP, …)
- **Problemen**
  - Meestal trager (belangrijk bij zware belasting op het netwerk)
  - Nieuwe protocol wordt uitgevonden ➔ geen procedure voor

Tekortkomingen van firewalls

- **Hoe hoger de beveiliging, hoe minder de functionaliteit**
  - Het gewone gebruik van het internet kan in gedrang komen
- **Firewalls vormen soms een vals gevoel van veiligheid**
  - Ze zijn niet absoluut onfeilbaar, zorg ervoor dat het niet de enige 'line of defence' is.
  - Kijk goed de firewall logs na

15. Virussen en wormen

Wat is een computervirus?

- **Een programma dat zich kopieert door andere programma's te infecteren**
- **Replicatie** is het grootste kenmerk van een virus
- Infectie:
  - **Het virusprogramma hecht zich aan 1 of meerdere andere programma's op het doelsysteem**.
  - Het programma start op ➔ virus activeert
  - Booten kan al genoeg zijn om het virus te activeren
- De meeste virussen volgen **de route:**
  - Gebruiker start een legitiem programma **(gebruikersinteractie)**
  - Virus, die zich genesteld had in het instructiereeks van dat programma, wordt uitgevoerd i.p.v. het originele programma
  - Viruscode wordt beëindigd en geeft de controle terug aan legitieme programma

Wat is een computerworm?

- Een worm doet ook een **replicatie** maar de manier van hechting is anders dan een virus
- Een worm **verspreidt zich via netwerken** / systemen **zonder** zich ergens aan te **hechten**
- Er is dus ook **geen gebruikersinteractie** nodig
- Een computer wordt dus **besmet door zijn omgeving i.p.v. specifieke objecten** zoals bestanden

In the wild

- **Als een virus is ontsnapt** of vrijgegeven is
- **Als het grote publiek er dus mee te maken krijgt**, niet de onderzoekers in de afgesloten labo's
- **WildList Organisation (WLO)**
    - Lijst bijhouden van virussen die momenteel circuleren 'in the wild'

BSI

Boot Sector Infectors

- Een boot sector bevat info over de bestandsstructuur van een schijf; wat zich waar bevindt + extra info om te booten.
- Bij een BSI **vervangt een virus op een usb de normale bootrecord met zijn eigen code**
- Wanneer een HD geïnfecteerd is, zullen **alle usb-sticks** die via die PC gebruikt worden, **geïnfecteerd worden met het virus.**
- De meeste bootsector virussen **kopiëren de originele bootsectorcode** naar een andere plaats
    - Nadat het zijn viruscode heeft uitgevoerd, gewoon verder booten

Memetische virussen

- Virussen zonder code, **nepvirussen**, virus van de geest, **hoax**
- Meestal in de vorm van een **kettingbrief**
- *"Search your PC for the folder System32. If you find it, delete it, because it's a virus. Please share…"*

Viruskenmerken

Stealth

- Alle virussen zijn min of meer stiekem
- Proberen **aanwezigheid te verbergen** om zo kans op **ongemerkte verspreiding te vergroten**
- **Verdachte springladingen worden meestal vermeden**, het wordt op onregelmatige tijdstippen verspreid
- Meeste virussen **bewaren originele code**, voor als het OS bepaalde info opvraagt, of een simpele virusscanner
    - Een scanner moet dus anti-stealth technieken hebben

Polymorfie

- **Vroeger** infecteerden virussen door een **bijna exacte kopie van zichzelf** aan een ander object te hechten
    - **Gemakkelijk te detecteren** door een virusscanner
- Daarom polymorf: hecht **iedere keer een iets andere virusvorm** aan het object, een **geëvolueerde versie**
    - Bv veranderen van instructievolgorde, toevoegen van lege bytes of nepinstructies, …
- Meeste **virusscanners** moeten dus een **soort van patroon herkennen** en niet gewoon identiek dezelfde code

16. Trojan en Rootkit

Definitie trojan

- Een Trojaans paard is een programma dat **beweert een wenselijke of noodzakelijke functie uit te voeren** (dat het soms ook echt doet), maar daarnaast ook een functie(s) uitvoert die **onverwacht en ongewenst** is door de gebruiker
- **Belangrijkste verschil tussen virussen, wormen en trojans**
  - Virussen en wormen zijn replicerende programma's
  - Trojans zijn **statische code** (altijd geassocieerd met **misleiding**)
- Moeilijk om op een 'echte Trojan' te scannen, want de definitie van Trojan houdt in dat er een **verschil is in wat het programma doet, en wat de gebruiker verwacht** ➔ moeilijk te bepalen

Backdoor Trojans

RAT / Application-level backdoor trojan

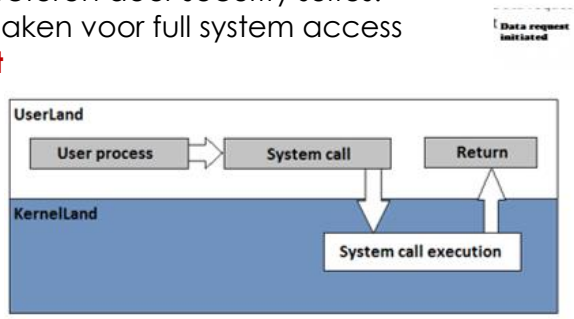
- **Remote Access Trojan**
- **Alternatieve weg om toegang te krijgen** tot een applicatie/systeem
- Meestal een **bypass van de authenticatie**
- Een Trojan om toegang te krijgen zonder het medeweten van de victim
- De hacker heeft controle over het systeem
- Client – server
- Op de victim wordt een server applicatie geïnstalleerd
- De hacker heeft een client om op die server te connecteren

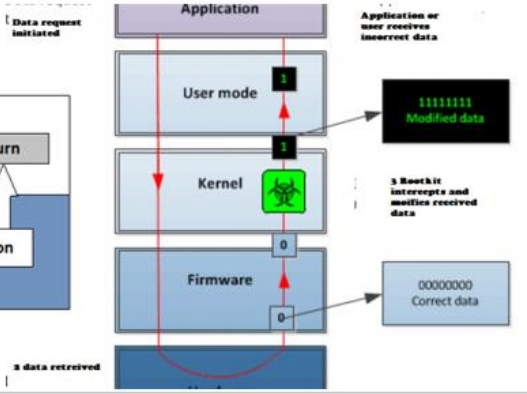
Traditionele rootkits

- **Very well hidden backdoor**
- Eigenlijk **zoals een RAT,** maar veel **'sneakier'**
- Gaat veel gebruikte en belangrijke executables aanpassen / vervangen **(key system components)**
  - Op deze manier wordt er een backdoor gemaakt om full access te geven
- '**rename essential services**'
  - ➔ wordt meestal in de **quarantaine** geplaatst omdat de scanner niet zeker weet of dat het een echte programma is of niet.
- Gaat zich echt **diep in je systeem verbergen**
- **User mode rootkit**

kernel-level rootkit

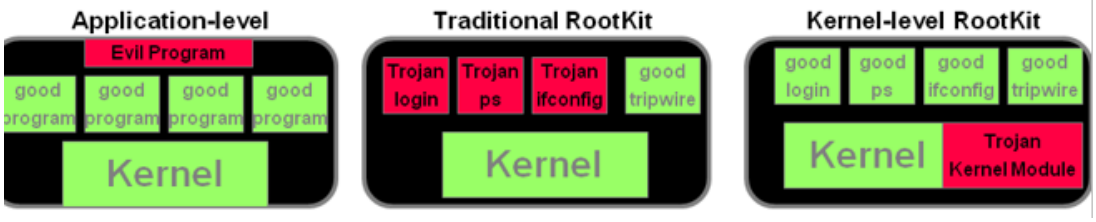
- **The sneakiest of the sneaky**
- **Vervangen/aanpassen van de kernel**
  - Dus het opvangen of veranderen van system calls
  - Zeer moeilijk te detecteren door security suites!
  - Zo een backdoor maken voor full system access
  - **Kernel based rootkit**





Trojan Horse Backdoors

Type of Trojan horse backdoor	Characteristics	Analogy	Example tools in this category
Application-Level Trojan Horse Backdoor	A separate application runs on the system	An attacker adds poison to your soup.	Sub7, BO2K, Tini, etc.
Traditional RootKits	Critical Operating System components are replaced.	An attacker replaces your potatoes with poison ones	Lrk6, T0rnkit, etc.
Kernel-Level RootKits	Kernel is patched.	An attacker replaces your tongue with a poison one.	Knark, adore, Kernel Intrusion System, rootkit.com, etc.



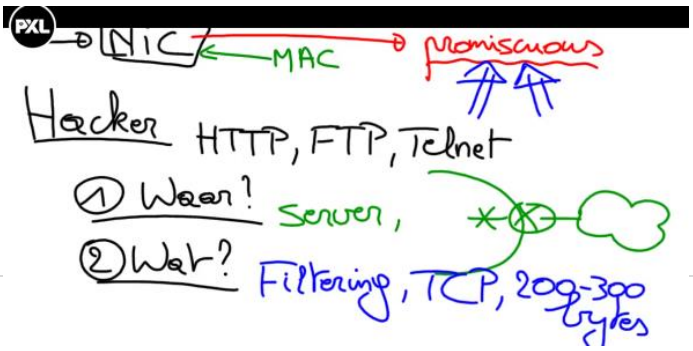
17. Sniffer – protocol analyzer

Inleiding

- Het meest gebruikte sniffer is **Wireshark**
  - **Kan al het dataverkeer opvangen en opslaan om daarna analyse op deze data te doen**
- **2 grote functies**
  - Voor de hackers (sniffer)
  - Voor beveiliging (protocol analyzer)
- **Basisvereisten**
  - **Netwerkkaart** (NIC) moet in **promiscuous** mode staan
    - Normaal gezien wordt alleen maar data van ons **eigen MAC-adres** ontvangen
    - Als de netwerkkaart in deze speciale modus staat ontvangt hij **alle** de data waarmee hij in contact komt

Hacker

- Hacker zal zeker naar de **plain-tekst data** moeten kijken (**HTTP, FTP, Telnet**)
- **Waar** gaat hij sniffen?
  - Dicht bij **servers**
  - Dicht bij de rand van het netwerk waar de meeste traffic is (dicht bij **gateway**)
- **Wat** gaat gij capturen?
  - Als hier geen aandacht aan besteed wordt dan gaat er een hele **grote capture file** gecreëerd worden. ➔ dit bestand **valt op**
  - **Filtering** wordt toegepast
  - **TCP filtering** ➔ meest nuttige informatie; login informatie
  - De meest nuttige informatie voor een hacker zijn de eerste **200-300 bytes** van een connectie



Defence

- **Zeer moeilijk! Een sniffer is passief**
  - Stuurt geen verkeer over het netwerk. **Luistert** gewoon naar het netwerk
- Er zijn 2 manieren van defence

℘ *Detect & destroy*

- **Programma's** die gaan **zoeken of er hosts** in het netwerk zijn waar de **promiscuous mode aan staat.**
- **Hash-code** berekenen van de **data van de servers**. Van het moment dat een hacker een **extra programma** op de server gaat zetten dan zal de **hash niet meer hetzelfde** zijn.
- **Topology changes** ➔ komt er een nieuw device in mijn netwerk

℘ *Protect data*

- **Encryptie**
- **Segmentatie gebruiken**
  - Sniffers kunnen alleen gegevens detecteren die hem passeren
  - toevoegen van **extra switchen** en routers in het netwerk
  - Een switch doet al **filtering op mac-adressen** ➔ hoe meer switchen hoe meer filtering
  - Omzeilen door een switch te laten functioneren door een **hub** ➔ mac-spoofing/flooding

18. Spoofing (TCP)

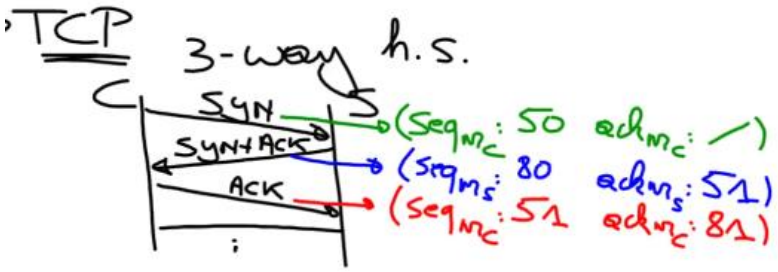
Inleiding

- Zich gaan **voordoen als iemand anders** zodanig dat je **de rechten van** die **iemand anders** kunt **gebruiken** om dingen te doen.
- **TCP** is het interessantste **protocol** voor een hacker omdat via deze protocol alle beveiligde communicatie gaat (**passwoorden** enz.)
- Werkt met een **3-way handschake**
  - **Getallen** van de handschake zijn belangrijk bij spoofing ➔ van zodra dat 1 getal niet klopt dan wordt alles **gereset**.
  -

3-way handschake

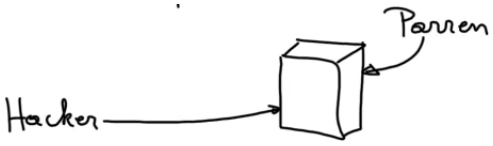
Wordt gebruikt om connectie tot stand te brengen

1. De client stuurt een **SYN** bericht naar de server
   a. Sequence number client: 50 (**random gekozen**)
   b. Acknowledge number: **leeg**
2. De server stuurt een **SYN + ACK** naar de client
   a. Sequence number server: 80 (**random gekozen**)
   b. Acknowledge number: **51** ➔ het volgende bericht dat de server van de client zal ontvangen (seqnr client + 1)
3. De client stuurt een **ACK** naar de server
   a. Sequence number client: **51**
   b. Acknowledge number: **81**
4. TCP-sessie start



TCP spoofing

- Er is een server met een administrator
- Een hacker wilt zich voordoen als de administrator zodat de hacker dezelfde rechten heeft op de server



1. Authenticatie
2. Soort TCP spoofing kiezen
3. Ervoor zorgen dat de administrator down is

Stap 1: Authenticatie

rhost file

- een file op de server waarmee je **authenticatie** kunt doen op **basis van IP-addressen**.
- Als je komt van deze IP, heb je deze, deze en deze rechten.
- **Niet veilig**. Iedereen kan tegenwoordig zijn eigen IP veranderen.

Stap 2: Soort TCP spoofing kiezen

1. Non-blind spoofing
2. Blind spoofing

Non-blind spoofing

- Als we als hacker een TCP-bericht sturen naar een server willen we ervoor zorgen dat de **server zijn antwoord terug stuurt naar de hacker.**
- Wij hebben het **antwoord van de server nodig** voor de nummers van de **3-way handshake.**
- Omdat wij het **IP-adres hebben van de administrator** kan het goed zijn dat de server zijn antwoord terugstuurt naar de administrator i.p.v. de hacker.

2 manieren om dit aan te pakken:

IP source routing

- In een IPv4-header zit onder andere een onderdeel genaamd opties
- In deze opties zit in **IP source routing optie**
- In deze optie kan je **definiëren hoe het bericht teruggestuurd moet worden.**
- Tegenwoordig worden de pakketten met IP source routing **automatisch geblokkeerd door de routers.**

ARP poisoning (spoofing)

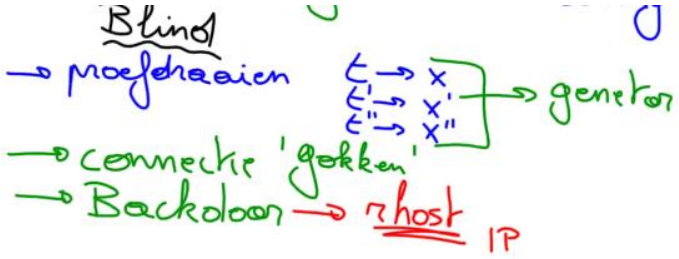
- **ARP** = een tabel, in de netwerkkaart, waar de mapping van alle MAC-adressen met de bijbehorende IP-addressen staan
- ARP is nodig omdat binnen een lan alle berichten worden gestuurd op basis van MAC-addressen.
- Bij ARP poisoning stuurt de hacker naar de server dat **een IP bij zijn MAC-adres hoort** (ARP responce) en de server zet dit dan vervolgens in zijn **ARP tabel.**
- Om deze aanval te doen moet je wel binnen **dezelfde lan** zitten
- De hacker moet **regelmatig ARP responces sturen** omdat de **cache** regelmatig wordt **vernieuwd**.

Verschillende methodes van ARP Spoofing
De meest gebruikte target is de default gateway

- **Passive sniffing**
  - Alleen de pakketjes bekijken en ze dan doorsturen naar de echte gateway.
- **Man-in-the-middel-attack**
  - Pakketjes veranderen en dan verder sturen naar de gateway
- **DoS-attack**
  - Pakketjes gewoon allemaal tegenhouden, niemand kan buiten LAN communiceren.
  - pingen naar de gateway is wel mogelijk, dus moeilijk te detecteren.

∞ *Blind spoofing*

- **Moeilijkste en meest voorkomende**
- Je kunt de **3-way berichten niet lezen**, de server stuurt de **SYN + ACK ergens anders** i.p.v. naar de hacker.
- Mogelijk **buiten lan**



Stap 1: proefdraaien

- Een dag op voorhand **heel veel TCP-connecties naar de server leggen**
- De server antwoord met SYN + ACK en de hacker zal dit allemaal **noteren**
- **Op dit tijdstip heeft de server getal x gekozen**
- Dit doen we omdat een random nummer niet echt bestaat. Het is altijd een **pseudo-random** nummer dat **gebaseerd is op de tijd.**
- Bijgevolg bestaan er **niet al te veel pseudo-random generators** omdat deze redelijk moeilijk zijn om te schrijven.
- We gaan de data die we opgeschreven hebben matchen met de generators zodat we **er achter komen welke generator de server gebruikt.**

Stap 2: connectie 'gokken'

- Omdat we nu de generator weten, kunnen we een **gok doen** naar welk **getal** de server zal kiezen.
- Dit zal niet altijd juist zijn maar de **kans is veel groter.**

Stap 3: backdoor installeren

- Zodat de hacker altijd **full access** heeft op de server en **niet** de vorige **stappen** telkens **opnieuw** moet doen.
- **Hacker zijn ip** toevoegen aan de **rhost file**

⌘ *Stap 3: Ervoor zorgen dat de administrator down is*

- Als de administrator van de server een SYN + ACK krijgt terwijl hij geen SYN heeft gestuurd dan stuurt de administrator naar de server dat hij de **handshake moet stop zetten**.
- Wordt gedaan met **DDOS** aanvallen ➔ **SYN-flood** (uitleg zie DDOS)

Volledig stappenplan spoofing

1. De hacker moet zijn **doelwit identificeren**
2. Moet de source waarvoor hij zich wil uitgeven **verdoven (bv. SYN-flooden)**
3. **Vervalsen** van het adres van de source
4. **Connectie maken** met server als die source
    a. Non-Blind ➔ berichtjes onderscheppen voor de Seq/Ack nummers
    b. Blind ➔ van te voren proefdraaien en daarna raden van het volgnummer waar de server om vraagt
5. Eenmaal binnen ➔ **backdoor** maken

Beveiliging tegen spoofing

- Heel goede **monitoring** en **logging** doen
    - DDOS en ARP poisoning kan heel goed gedetecteerd worden
- **Sterke authenticatie** voorzien
- **Egress filtering**
    - blokkeren van pakketten die van **interne LAN** komen met als **source IP** een **extern adres**
    - een hacker heeft waarschijnlijk die computer overgenomen
- **Ingress filtering**
    - Blokkeren van pakketten die van **buiten het netwerk** komen, maar een **intern IP** adres hebben **als source adres**.
- Egress en ingress filtering is tegenwoordig in alle routers voorzien en is de **default beveiliging tegen spoofing**

DNS-Spoofing

- Elke pc heeft een **hosts file**
- Deze file was de voorloper van DNS
- In deze file staat een URL en het IP die bij deze URL hoort
- Is tegenwoordig leeg ➔ legacy
- Als je naar een website surft wordt er **eerst in de hosts file gekeken**
- Als er niks in deze file gevonden wordt dan wordt er pas een request naar de DNS-server gestuurd

19. DDos

Inleiding

- Denial of service
- Aanval die tot gevolg heeft dat een bepaalde **service verloren gaat** of niet meer kan functioneren
- DoS-tools worden vooral ontwikkeld **om aan te tonen** dat bekende OS **lekken in hun beveiliging** hebben
    - Drukt de ontwikkelaars met hun neus op het feit dat er meer aandacht aan beveiliging moet worden besteed
- **Niet altijd het gevolg van een aanval** ➔ veel mensen die op een bepaald moment iets online willen bekijken of doen.

Werking

- Alleen mogelijk omdat software **programmeerfouten** bevat
- Aanval beoogt:
    - Zoveel mogelijk **bandbreedte** in beslag nemen
        - Aanval op netwerkbronnen
    - Zoveel mogelijk **bronnen verbruiken**
        - Aanval op systeembronnen
    - Programma's en systemen laten **crashen**

Ping of death – vastlopen van programma's en systemen

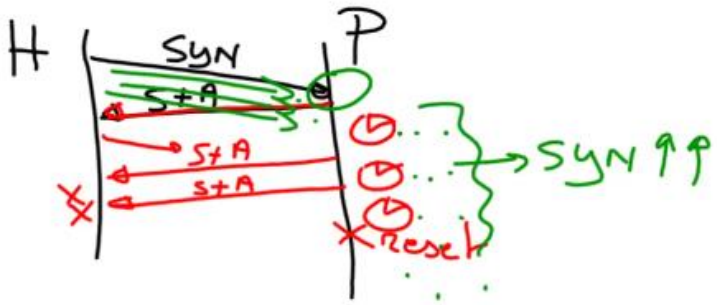
- Werkt meestal niet meer
- Grote ICMP-echo verzoeken (ping) verzenden (**ICMP-protocol**)
- Ping is normaal iets van een **64 bytes**
- **Men stuurt 1 grote ping, in stukjes, gekapt, over het netwerk**
- Als hij aankomt bij de bestemming, **worden de delen terug tot 1 grote ping gemaakt**
- **Bufferoverflow** bij bestemming want er was niet zo een grote ping verwacht
- Het systeem **crasht**.

E-mailbommen – vastlopen van programma's en systemen

- Zeer geniepige, simpele aanval, met soms grote gevolgen
- **Heer veel berichten** naar uw postvak **sturen**
    - **Vult** dus de **harde schijf** van de mailserver
- **Postvak vol**
    - Er kunnen geen nieuwe berichten meer naar toe gestuurd worden
- **Harde schijf mailserver vol**
    - Niemand kan nog e-mails naar die mailserver sturen
- Kan leiden tot serieus gegevensverlies, verhoogde bandbreedte verbruik, hogere netwerkkosten (verbindingskosten)

SYN-flood – zoveel mogelijk bronnen verbruiken

- Een computersysteem heeft maar een **beperkte aantal bronnen beschikbaar** (geheugen, opslagcapaciteit, processor time, …)
- Heeft te maken met de **3-way handschake**
- **Heel veel aanvragen tot connectie naar de server sturen**
- **Server stuurt antwoord terug**
- **De server wacht op jouw antwoord**
- **Er komt geen antwoord**
- **Reset duurt even** (**dit maakt de aanval zo zwaar**)
- **Veel aanvragen**
    - men kan de **inkomende berichten niet verwerken** want hij is continue bezig met de andere verzoeken.



HTTP-aanval op webserver – zoveel mogelijk bronnen verbruiken

Tegelijkertijd HTTP-verbindingen tot stand brengen met webserver.

De server kan het **aantal aanvragen niet aan**, wordt trager en valt uiteindelijk uit.

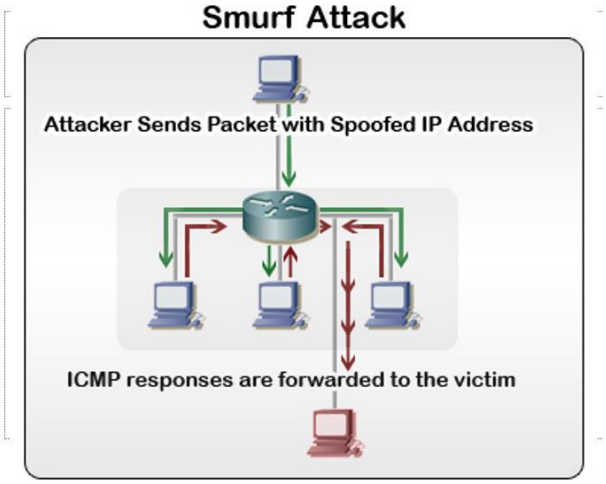
- **CPU verbruik**
    - Script voor **filteren**, van hoog naar laag, van laag naar hoog, van hoog naar laag enz.
    - **Complexe berekeningen**
    - **Processor time voor 1 aanvraag laten toenemen**
- **SQL CPU**
    - **Sql querry flooding**
    - Server heeft extra info nodig om aanvraag af te werken, moet die gaan opzoeken, aanvragen, …
    - **Tijd dat het duurt om 1 aanvraag af te werken verlengen**
- **Storage**
    - Comments blijven posten door middel van een script
    - PHP-scripts
    - Kleinste bestand is 1 kb ➔ wordt al snel 10 GB
    - **De server gaat gegevens opslaan**
    - Heel veel gegevens ➔ harde schrijf van server is vol ➔ problemen!

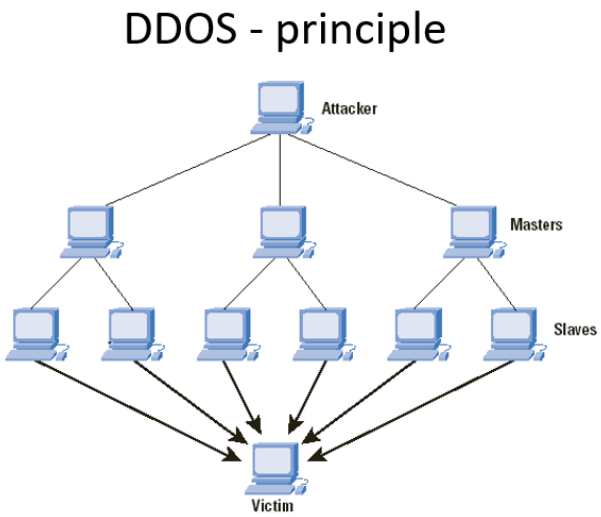
Bovenste drie hebben allemaal te maken met **scripting**

Oplossing: beperken wat een gebruiker op je website kan doen.

Smurf-aanval – inbeslagname bandbreedte

- **Netwerk kan maar een bepaalde hoeveelheid netwerk-trafiek aan**
  - Snelheid van het onderliggende netwerk is hier een belangrijke factor
- Als de **bandbreedte** wordt opgesoupeerd kunnen er geen nieuwe gegevens meer verzonden worden
- Is een aanval van de **actieve soort.**
  - **Na** de DoS aanval komt de bandbreedte weer **ter beschikking**
- Meestal wordt er gebruik gemaakt van **fouten in het protocol**
  - Bandbreedte wordt opgebruikt door speciaal bewerkte netwerk-gegevens te verzenden ➔ router loopt dan bv vast
  - Andere methode:
    - **Smurf-aanval**
    - **Veel of alle computers een broadcast laten doen**
    - Ping request sturen naar verschillende broadcast IPs, maar de source IP van deze request is **gespoofed**
    - **1 berichtje, naar broadcast (vele berichtjes), die allemaal tegelijkertijd een antwoord naar de target sturen**





DDoS

- **Distributed Dos-aanval**
- Netwerkverkeer bleek afkomstig te zijn van een heleboel **verschillende systemen tegelijkertijd**
  - Eerst dacht men dat het een georganiseerde aanval was door vele crackers
- Functioneert op basis van het **master-slave (zombie) principe**

Master-Slave principe

- De **master** is de controlerende station **(command of control)** ➔ aanvaller definieert daarop zijn **doel en methode**
- **Slaves** zijn **externe systemen** waarop het **aanvalsprogramma op geïnstalleerd is (aanvalslog, worm, backdoor)**
- **Master zegt tegen slaves** ➔ **“voer aanval uit”, “stop aanval”**
  - **Moeilijker tegen te houden** omdat het van zoveel verschillende systemen komt
  - Duurt dus redelijk lang om dat allemaal na te gaan + veel processing time
  - Server kan het hoofd niet meer boven water houden ➔ **crash**
  - Master is redelijk moeilijk te vinden omdat hij de **voorbereiding weken voor de aanval** doet ➔ **logs zijn nu verdwenen**

Men kan tegenwoordig een **botnet** huren om command of control te krijgen. Je moet wel **opletten met backtracking** ➔ connectie is nog niet lang genoeg geleden gemaakt dus de **logs bestaan nog.**

20. PortScanners

Soorten

Willekeurig

- Meestal werkt een aanvaller met een **poortscanner**, zoals **nmap**
- Geeft het programma **een blok IP-adressen om te controleren**
- Het programma geeft terug **welke computers met die IP-adressen verbonden zijn**, welke **poorten** daar **open staan** en **welke OS de computer gebruikt**
- De aanvaller gebruikt die info om meestal al gekende **kwetsbaarheden** van het OS of bepaalde programma's **te misbruiken**
- Hiervoor wordt gewoon **google** gebruikt om met die info methoden, scripts, programma's te gebruiken die daar omschreven staan.

Gericht

- Heeft van te voren **iemand geselecteerd als doelwit**.
- Reden: roem, minachting, diefstal van info, financieel gewin,…
- Iemand die werkt bij Microsoft zal eerder doelwit zijn dan iemand die werkt voor PXL
- Dus het bedrijf waarvoor je werkt, je sociale status bepalen hoe aantrekkelijk je bent voor een aanvaller.
- Aanvaller doet eerst uitgebreid onderzoek naar die persoon ➔ **spear pishing attack**

Werking

Scant een netwerk device voor openstaande poorten

- **Crackers:** hoe en waar in te breken
- **Admins:** hoe is de security, zijn er lekken

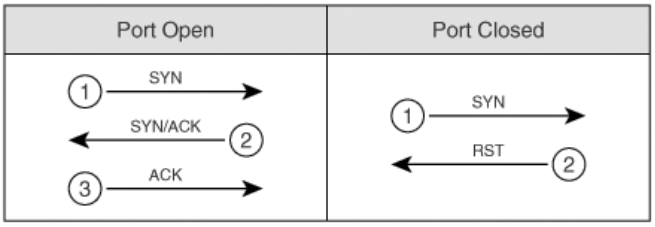
Verschil tussen portscan en portsweep

- **Portscan:** kijkt welke poorten er openstaan op 1 host
- **Portsweep:** kijkt of een specifieke poort openstaat bij verschillende hosts

Scans geven meestal 1 van de 3 mogelijke antwoorden:

- Open of accepted ➔ reply was positief
- Closed, denied, not listening ➔ reply was negatief
- Filtered, dropped, blocked ➔ geen reply ontvangen

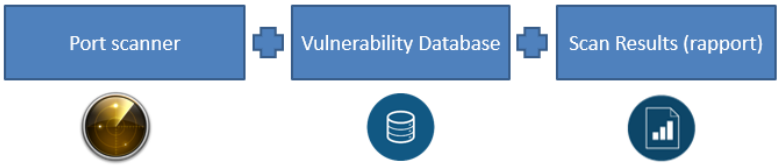
Bovenstaande is **gebaseerde op de 3-way handshake** ➔ krijg je een ACK terug van de server op die poort?



21. Vulnerability scanner

Inleiding

- = gereedschappen om **kwetsbare plekken op te sproken** (= scanner) in uw beveiliging
- Werkt met een **poortscanner + kwetsbaarheidsdatabank + rapport**



- De allereerste twee producten waren ISS en SATAN
- SATAN was een betere versie van ISS
- **Tegenwoordig:** nessus, nexpose, open VAS

Scanmechanisme (poortscanner)

- **Ongeveer hetzelfde principe als een poortscanner**
  - Nmap ➔ welke besturingssysteem heeft de pc, welke poorten staan open
- **Verschil:** het scanmechanisme kan detecteren of een **poort** openstaat (poortscanner), maar het kan ook bepalen **welke service** op welke poort draait en **welke versie** het is **(finger printing)**.
  - Bepalen welke service op welke poort draait en welke versie het is, is niet gemakkelijk want er zit veel kwaliteitsverschil op (zie false positives)
- **Banner-grabbing methode**
  - **Connecteren naar een bepaalde poort** ➔ geeft **banner info** terug ➔ **regex** gebruiken om **nuttige info** eruit te halen

Kwetsbaarheidsgegevens (kwetsbaarheidsdatabank)

- Deze scanners hebben een **interne database met info over zwakke plekken** die bepaalde versies van services kunnen hebben
- De database is meestal **gevuld met alle CVE's known to date**
- CVE = **Common Vulnerabilities and Exposures**
  - Er is ook een open source version ➔ OSVDB (Open Source Vulnerability Database)

Rapportmechanisme (rapport)

- **Helder meedelen wat ze gevonden hebben zodat er een actie ondernomen kan worden**
- Er is ook **een classificatie van problemen** hierdoor kan er een **false positive** of **false negatieve** tussen zitten
  - **False positive:** beveiligingslek is gerapporteerd maar **het is er geen**
    - Niet goed als er teveel van inzitten ➔ veel werk om alles te controleren
  - **False negative:** is erger dan false positive want er is geen beveiligingslek gerapporteerd **terwijl er wel één is.**
  - De **kwaliteit van de scanner hangt af van deze waarden**, zie **confusion matrix**

Confusion matrix

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

- Hoe dikwijls zeg je **positive** en was het ook echt **positive**?
  - **True positive**
  - **Goed! Vulnerability in het rapport is ook echt een vulnerability**
- Hoe dikwijls zeg je **positive** en was het eigenlijk **negative**?
  - **False positive**
  - **Slecht! Vulnerability in het rapport blijkt na veel opzoekingswerk eigenlijk geen vulnerability te zijn**
- Hoe dikwijls zeg je **negative** en was het ook echt **negative**
  - **True negative**
  - **Goed! De non-vulnerable services in het rapport zijn ook in het echt non-vulnerable**
- Hoe dikwijls zeg je **negative**, en was het eigenlijk **positive**?
  - **False negative**
  - **Super slecht! Het rapport zegt: service is non-vulnerable dus ik denk dat ik safe zit, maar het blijkt echter wel een vulnerable service te zijn**

Via **benchmarking** kan nagegaan worden welke vulnerability scanner geschikt is voor u

- Als voorbeeld: een benchmark op 100 services, waarvan er 20 vulnerable zijn
- 4 scanners worden getest

Scanner1		Scanner2		Scanner3		Scanner4			
20	0	10	5	10	10	10	0	TP	FN
0	80	5	80	0	80	10	80	FP	TN

- **Scanner 1** ➔ dit is de ideale scanner. Haalt alle vulnerabilities eruit zonder false positives en false negatives
  - In de praktijk is er geen enkele scanner ideaal. Daarom kijken we vooral naar scanner 2, 3 en 4
- **Scanner 2, 3, 4** ➔ allen doen evengoed op vlak van TP en TN. Het zijn degelijke scanners want ze halen alle drie 10 TP uit

22. IDS - IPS

Inleiding

- Intrusion detection system (IDS)
- Intrusion prevention system (IPS)
- **Bescherming van netwerk en systemen**
  - Vijandige **indringers automatisch ontdekt**
  - Beheerders **waarschuwt**
  - De indringers mogelijk **tegenhoudt** (= IPS)

Verschil tussen IDS en IPS

- **IDS** detecteert een indringer en slaat alarm bij de administrator
- **IPS** detecteert en onderneemt acties (houdt indringers tegen)
- Bij een **IPS** kunnen er **false positives** en **false negatives** zitten ➔ goed opletten dus
- IDS is het **meest bruikbare vorm**
- Als iets **echt heel beveiligd** moet zijn dan wordt IPS gebruikt

Mogelijkheden tot detectie

- **Logboeken** bijhouden en controleren op voorvallen
- **Integriteitscontrolers** toepassen (**hashing**)
- **Eventlogs** bijhouden en speuren naar mislukte aanmeldingspogingen
- Netwerkpackets **checken op fingerprints** van known attack vectors

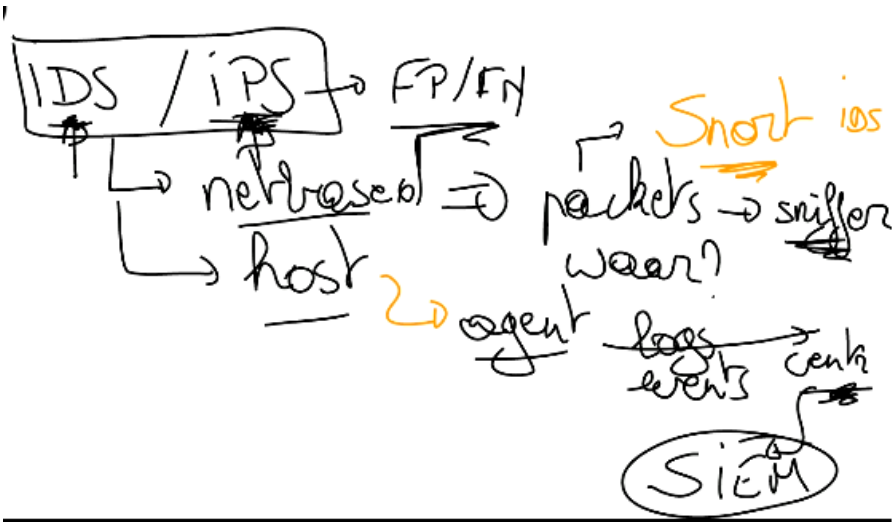
Netwerk-based IDS / Host-based IDS

Netwerk-based IDS

- **Controleert het netwerkverkeer**
- Kijkt uit naar bekende **aanvalspatronen** (signaturen, finger prints)
- Hiervoor worden **sniffers** gebruikt ➔ capturen van data
- Dezelfde **problemen** als een sniffer ➔ waar ga je de sniffer zetten? Kan niet om met encryptie
- Meest gebruikte is **Snort**, logo is een varken
- **Passieve component** ➔ andere netwerkonderdelen weten niet dat een pakket een IDS is gepasseerd
- **Beperkingen**
    - In een netwerk met veel **switchen** passeert niet al het verkeer langs de IDS
    - Mogelijkheden om IDS'en te omzeilen via **pakketversleuteling**
    - In een omgeving met **grote bandbreedte** krijgen de meeste IDS'en problemen om alles verwerkt te krijgen

Host-based IDS

- Verzamelt informatie op de host zelf
- Wordt vaak gebruikt bij servers
- **Agent wordt op de host geïnstalleerd**
- Agent **verzamelt logs en events**
- Deze agent verzameld informatie van de host zelf om nadien te gaan **doorsturen naar een centrale server**
- Deze centrale server **controleert de data** om te kijken of er een indringer bezig is of niet
- Centrale server ➔ **SIEM-server**
- Analyseren van systemlogs, integriteitscontrole van files
- Meest gebruikte is **Ossec, Tripwire**



Signature-based IDS / anomaly-based IDS

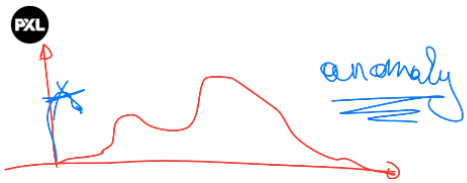
De meeste systemen vandaag zijn een combinatie van deze twee

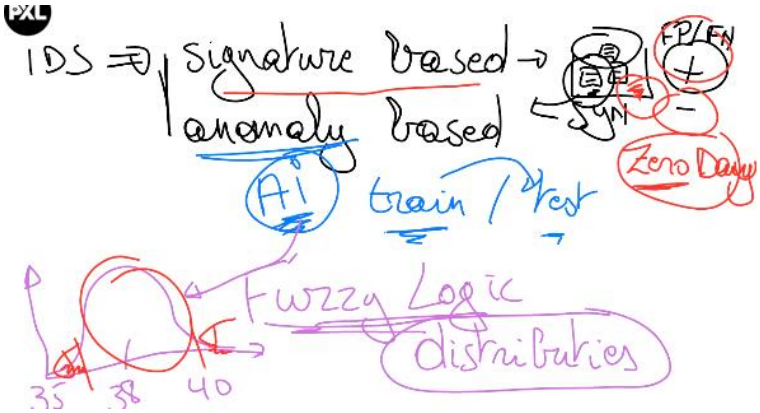
Signature-based IDS

- Gaat naar **mogelijke patronen opzoeken** in een **database**
- Zoveel SYN-pakketten in een bepaalde tijd komt overeen met deze aanval
- **Voordeel**
  - Er kan redelijk snel gekeken worden of het een aanval is of niet
  - False positive / false negative is redelijk goed
- **Nadelen**
  - Een nieuwe aanval staat nog niet in de database
  - **= Zero day attacks**

Anomaly-based IDS

- **Het systeem bepaald zelf** of er iets normaal of abnormaal aan de hand is op het netwerk
- Gebaseerd op **regelmatige gebruikerspatronen**
- **AI-detectiesysteem**
- **Training- en testmode**
- **Fuzzy logic**
  - Rekenen met **normaal en abnormaal**
  - Rekenen met **distributies**
  - Hierbij wordt statistiek gebruikt ➔ normaal distributie
- Er wordt een **hearthbeat** van het netwerk gemaakt





23. Logging & SIEM

Inleiding

Waarom logs bijhouden?

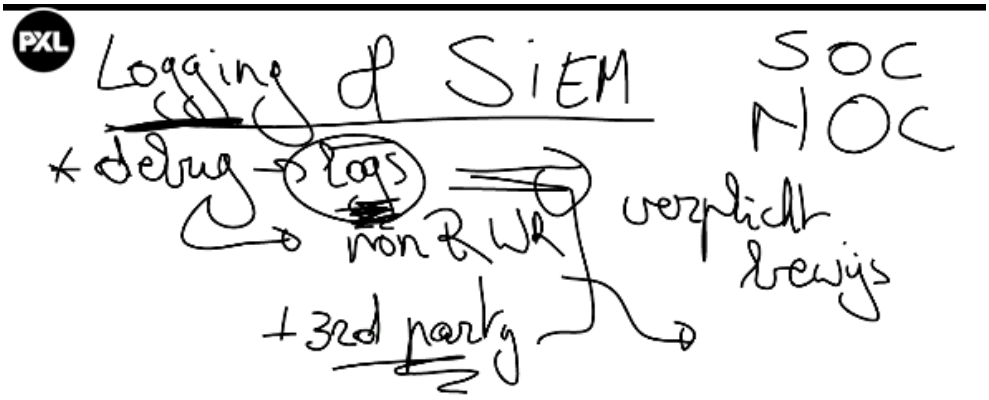
- Logboeken kunnen **problemen** helpen opsporen, **trends** weergeven, of **anomalieën** op het netwerk ontdekken

Welke stappen deed een indringer?

- **Logboek**

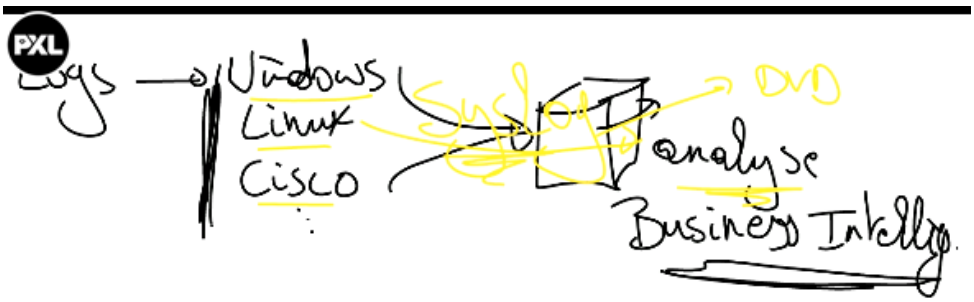
Logs vroeger

- Om te **debuggen** of als er iets **mis** was moest men naar **de logs gaan kijken**
- Werden **bijgehouden op non rewritable media** bv. Cd-rom
- Was **verplicht** op de logs bij te houden
    - **Bewijs** voor de **rechtbank**
    - Bewijs **tegen hackers**
    - Hierdoor werden er ook logs bijgehouden van een **3rd party systeem**
- Er werd dus meestal **2 logs bijgehouden**; van het besturingssysteem en 3rd party systeem
- Een 3rd party systeem werd meestal gebruikt omdat de hackers als eerste aanleerde hoe dat men de **logs van het besturingssysteem** kan **manipuleren** om zijn eigen te verbergen
    - Werd meestal niet gedaan met de 3rd party systemen
- Doel: logs voor een lange tijd bijhouden



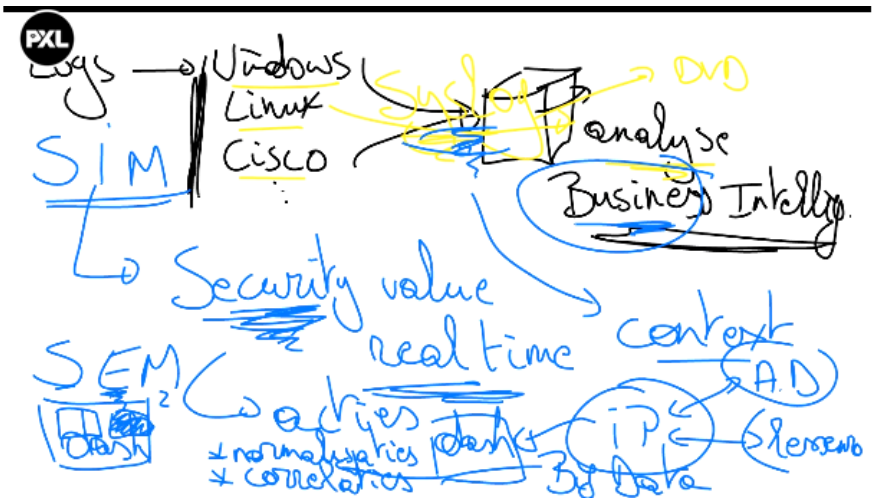
Daarna kwam SIM

- Men wou **alle logs** van **allerlei soorten systemen** op **1 centraal systeem** (server) bewaren
- Standaardformaat / protocol : **syslog**
    - Standaard voor het forwarden van logbestanden naar een centrale server via IP netwerk
- Op deze centrale server kan men **analyse op de gegevens** doen en **business intelligence** uithalen
- Ook hierbij kon men de logs op een **non-rewritable media wegschrijven** om in een brandkast te steken
- SIM ➔ security information management
- **Long-term storage, analyse, en rapportering van logs voor security doeleinden** (**analyse na een attack** bv.)



Verdere evolutie: SEM

- SEM ➔ Security Event Management
- De logs meer securityvalue geven
- **Real time monitoring** (dashboards) **van events**
- **Logs worden gebruikt in real time om attacks te detecteren** (**tijdens een attack** bv.)
- **Correlatie van alle data (context) is nodig om real time analyse te kunnen doen**
    - **Niet alleen meer van syslogs**, maar ook user accounts, SNMP data, Firewall events, IDS/IPS data, AD data…
      Allerlei soorten **machine generated code**
    - Al die data moet nu wel worden **genormaliseerd**, zodat analyse mogelijk is want alle verschillende data hebben allemaal verschillende formaten
- Correlaties worden weergegeven in **dashboards**



Tegenwoordig: SIEM

- Security information and event management
- **SIM en SEM in 1**
- **Meest gebruikte** ➔ Plunk, AlienVault, ELK (elastic)

24. Privacy

Soorten aanvallers

Script-kiddies

- Grootste groep, meestal onervaren, jong
- **Surfen op het web naar** utilities, scripts, **die ervaren aanvallers** daar gepost hebben
- Meestal **mislukt** hun aanval
- Zullen eerder **gegevens op kwaadaardige manier beschadigen** dan welk ander type cracker ook
- **Weinig sympathie voor hun**, zelf binnen de crackers community

Black hats

- **The dark side**
- **Zeer veel ervaring** met computers
- **Zelden gepakt**, doen **grondig onderzoek**, zeer **gerichte aanval**
- Je hoort weinig over hun vaardigheden of capaciteiten (ninja's van het internet)

White hats

- **The good guys**
- **Beveiligingsprofessionals die gaten opsporen en dichten** (pentesting)
- Bedrijven huren hen in om **beveiliging te testen**
- Meestal zijn ze **begonnen als black hat**

Red team <-> Blue team

- Red ➔ mensen die op een server van een bedrijf proberen **binnen te geraken**
- Blue ➔ defenders, mensen die het **netwerk monitoren**

Bespionering

2 soorten bespionering:

- **Collectieve informatievergaring**
  - Alleen info verzamelen over uw persoon, bezigheden, … **zonder direct contact** te maken
- **Penetrerende informatievergaring**
  - **Contact maken**, vertrouwen winnen, meer info verkrijgen

❖ Inlichtingsdiensten kunnen zonder huiszoekingsbevel uw internet activiteiten nagaan
  - Gebruiken zoekmachines

❖ 8 op 10 websites volgen uw bewegingen

Browserbeveiliging

3 belangrijke methoden via de browser

- Sniffen van IP-adres en in de cache
- Cookies
- Banneradvertenties en webbugs

Snuffelen naar IP-adressen in de cache

- **Meeste webservers registeren uw gegevens als je hen passeert**
  - Browser, operating system, versie, plug-ins, …
  - Door al deze gegevens creëer je eigenlijk een **uniek id,** want niet iedereen heeft dezelfde plug-ins en versies
- Al die gegevens zorgen ervoor dat een cracker **enorm veel info over jouw systeem** te weten komt
- Met deze info kan een cracker gaan zoeken naar **vulnerabilities ➔ exploit**
- **Marketingbedrijven** kunnen deze gegevens ook bijhouden om **een profiel** van iedereen aan te maken

25. Cookies

Inleiding

- **De server van een website krijg een klein stukje van de harde schijf van de persoon die naar de website surft**
- Dit klein stukje is **1kb groot** en wordt een cookie genoemd
- Hierin wordt taalsettings, winkelmandje, search items, … bijgehouden
- Waarom? Settings worden automatisch ingeladen wanneer je de website opnieuw bezoekt; usability
- Hoe hoger de usability, hoe lager de security
- Er wordt **zoveel mogelijk informatie bijgehouden** in een cookie
- Wordt ook veel gebruikt als **login-systeem** zodat je niet iedere keer opnieuw u moet authentiseren

Bescherming van cookies

Same Origin Principle

- **Scripts** die van een bepaald **domein** komen mogen alleen maar aan **de cookie van dat** bepaald **domein** komen

Evolutie van cookies

- **Uniek id opslaan in cookie die gelinkt wordt aan de database van de website**
- Hierdoor kunnen ze **zoveel data als ze willen opslaan over de gebruiker**
- **Reden:**
  - 1kb **cookie is veel te klein** om heel veel data in op te slaan
  - Gebruiker **verwijderd de cookie** soms ➔ gegevens gaan verloren

3rd-party cookies

- De website waarnaar je surft heeft **een deel van een andere website in zijn site** staan
- Hierdoor worden **de cookies van de originele website en de andere websites geladen**
- Nu kunnen de **externe websites ook data bijhouden** van wat je aan het doen bent op de originele website
- Hierdoor creëer je een **compleet profiel**; er wordt heel veel data van je bijgehouden

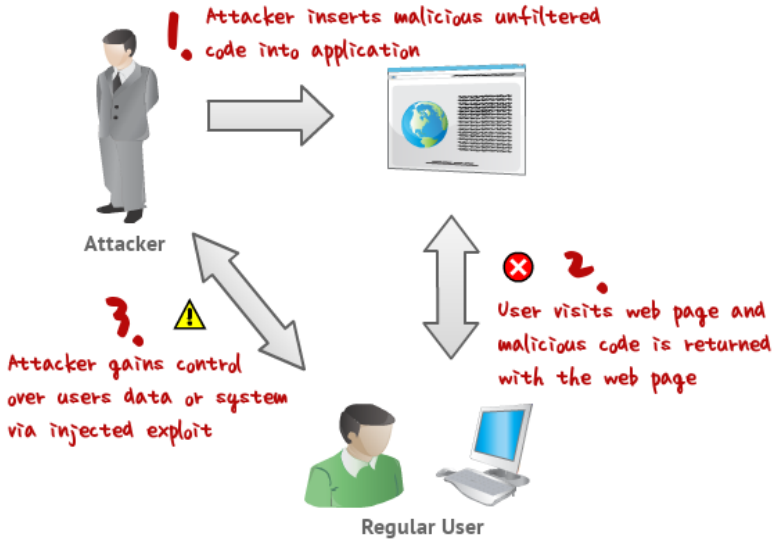
26. XSS

Inleiding

- **Cookies worden ook dikwijls gebruikt om in te loggen**
- **Een hacker wilt de authenticatie cookie bemachtigen om zich als u voor te doen**
- De hacker maakt een **script** waarin hij zegt dat **de cookie** van een bepaalde website naar zijn **ip adres** gestuurd moet worden
- Maar er worden **alleen maar cookies gestuurd naar de scripts van het juiste domein**
- **Er zijn 2 grote manieren om XSS te doen**
  - Stored (persistent) XSS
  - Reflected (non-persistent) XSS

Stored (persistent) XSS

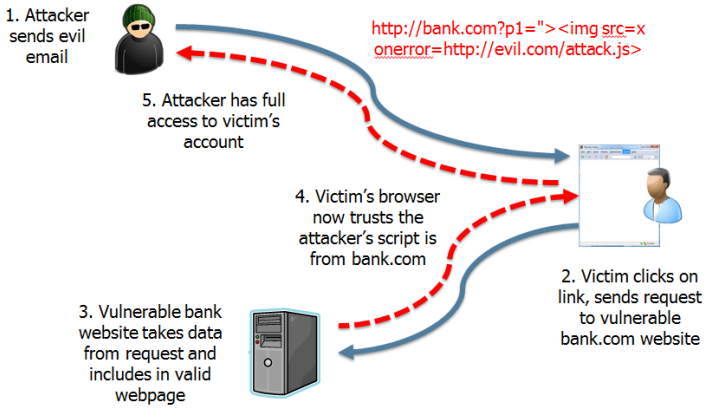
- Kwam vroeger veel voor, nu niet meer
- **De gemaakte script in een forum posten**
- **Iedereen die naar het forum gaat voert het script uit**
- **Het script** komt nu wel **van het juiste domein** en hierdoor zal **de cookie naar de hacker** gestuurd worden
- Wanneer wordt dit vandaag nog gebruikt?
    - Wanneer een server gehackt wordt en men de script in de sourcecode van de server zou steken
    - Hiervoor bestaan IDS

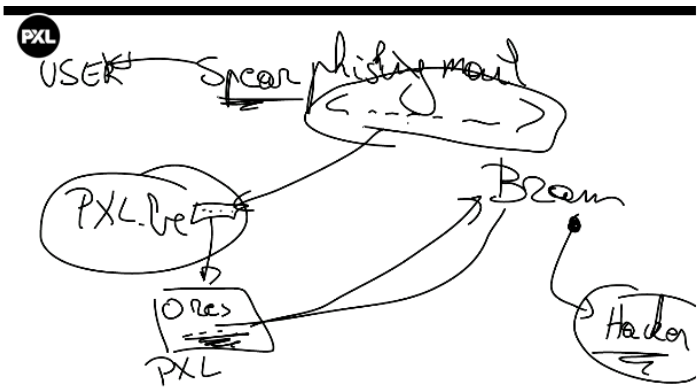


Reflected XSS

- De victim klikt op **een link in een spear phishing mail** (= zeer gepersonaliseerde mail)
- Deze link stuurt **data naar de search-bar van website X via de victim zijn browser**
    - Data = special formatted **'give_hacker_cookie_X-script'**
- Deze data is zo opgesteld dat de search-bar van site X deze niet goed begrijpt
- Website X stuurt dus een **antwoord terug naar de client**
    - **ik weet niet wat 'give_hacker_cookie_X-script' is**?
- **Dit antwoord komt in de victim zijn browser en de script wordt uitgevoerd**
- De script komt nu wel van het juiste domein en zal de cookie naar de hacker sturen
-

How Does Reflected XSS Work?





27. Banneradvertentie

Wanneer je een pagina met zo'n banner bezoekt:

- **Banner wordt geladen vanuit de webserver die beheerd wordt** door reclame-bureau
- Die gaat dan **info opslaan over de gebruiker**, eventueel in combi met een cookie, javascript, …
    - dit is om een profiel van de gebruiker te maken
    - **= social engineering**

28. Webbug

- **Klein transparant gif-bestand**
- **1x1 pixel groot**
- **Werkt op dezelfde manier als een banner**
- **Niet echt zichtbaar** als dit gebeurt, behalve als je de **source html code** bekijkt
- Wordt meestal van een **andere server geladen** dan diegene die je momenteel aan het bekijken bent
- Al die info die over u verzameld wordt, dient voor marketing! **Social engineering!**