Hacker Highschool
SECURITY AWARENESS FOR TEENS

LES 10
INTERNETBEVEILIGING EN PRIVACY

Hacker Highschool
SECURITY AWARENESS FOR TEENS

ISECOM
INSTITUTE FOR SECURITY AND OPEN METHODOLOGIES

Informatie over de “Gebruiksvoorwaarden”

De lessen en werkboeken van het Hacker Highschool (HHS) project zijn beschikbaar onder de volgende door ISECOM gestelde voorwaarden:

Alle informatie uit het HHS-project mag, niet-commercieel, gebruikt worden voor en door basisschool-leerlingen en studenten van middelbaar en hoger onderwijs. Dit materiaal mag niet worden gereproduceerd voor (door-)verkoop in welke vorm dan ook. Gebruik van dit materiaal in een klas, cursus, training, kamp of andere georganiseerde vorm van kennisoverdracht waarvoor geld wordt gevraagd is expliciet verboden zonder een licentie. Om een licentie te regelen kunt u het onderdeel LICENSE bezoeken op de website van de Hacker Highschool, www.hackerhighschool.org/license.

Het HHS-project is een leermiddel en, zoals met elk leermiddel, de docent/trainer bepaalt in grote mate het effect van het leermiddel. ISECOM kan geen aansprakelijkheid aanvaarden voor de positieve of negatieve gevolgen van het gebruik van dit materiaal en de daarin opgenomen informatie.

Het HHS-project is een initiatief van de open community, en wanneer u de resultaten van onze inspanning waardevol genoeg vindt om het te gebruiken, vragen we u uw steun te betuigen door:

- de aankoop van een licentie;
- een donatie
- ons te sponsoren.

Inhoudsopgave

Contributors

Simon Biles

Pete Herzog, ISECOM

Bill Matthews

Hernán Marcelo Racciatti

Chris Ramirez

P. Shreekanth

Kim Truett , ISECOM

Marta Barceló, ISECOM

Dario Riquelme Zornow

Translation by Raoul Teeuwen

10.1 Fundamenten van Internetbeveiliging

Wat jij uitspookt op het Internet (het WWW) is jouw zaak. In ieder geval: dat denk je. Maar het is niet waar. Wat je op het web doet is net zo geheim als wanneer je je huis verlaat en ergens heen gaat. Nogmaals, jij vindt waarschijnlijk dat het alleen jou aangaat wat je op het web doet, en velen, waaronder Isecom/HackerHighSchool zijn het daarmee eens. Maar stel je een detective voor die je overal volgt waar je gaat of staat in je dorp/stad, en die alles noteert wat je ziet en met wie je sprak.
Deze les leert je hoe je jezelf moet beschermen op het web, en om dat te kunnen, moet je leren waar de gevaren zich bevinden.
Het World Wide Web werkt op een vrij directe manier. Zodra je via je ISP bent verbonden met het Internet open je een webbrowser, geeft op welke website je wil zien, en je krijgt die website op je sdcherm. Maar de waarheid zit hem in de details. Hoe werk dat mooie web nu echt?
Een snelle reis naar het World Wide Web Consortium (W3C), die aardige mensen die bepalen volgens welke standaarden het web werkt, leert je alles over de werking van het web. http://www.w3.org. Zelfs de geschiedenis van het web: http://www.w3.org/History.html . Maar leren definities en standaards je om iets veilig te gebruiken? Kennelijk niet. De mensen die kwaad in de zin hebben, volgen niet persé de standaards.

10.1.1 Hoe het web echt werkt

Om een verbinding op te bouwen met het web, moeten verrassend veel stappen worden doorlopen, ook al merk je daar als gebruiker meestal weinig van.
Wat gebeurt er nu echt als je de HackerHighSchool-site wil bezoeken? We gaan er van uit dat je al een verbinding met het internet hebt. Dan worden de volgende stappen doorlopen:
1. Je opent je browser.
2. Je tikt de URL (website naam) in.
3. De website naam wordt opgeslagen in de History Cache (geschiedenis van de browser) op je harde schijf.
4. Je computer neemt contact op met de ingestelde DNS-server om het ip-adres op te zoeken dat bij de ingevoerde website-naam hoort.
5. Je computer maakt een verbinding met de server, op TCP-poort 80 van het gevonden IP-adres (als je als adres "HTTP://" hebt gebruikt, TCP-poort 443 wordt benaderd als je "HTTPS://" voor de websitenaam tikt) (overigens worden er andere stappen genomen als je HTTPS gebruikt, met ondermeer server certificaten, maar dat werken we niet in dit voorbeeld uit).
6. Je computer vraagt de pagina of directory (map) op die je hebt opgegeven, en als je geen pagina op hebt gegeven wordt standaard meestal "index.htm" opgehaald. Maar de server bepaalt de standaard, niet jouw browser.
7. De pagina('s) worden opgeslagen in een cache op je harde schijf. Zelfs als je instelt dat de pagina in het interne geheugen (RAM) moet worden opgeslagen, is de kans groot dat hij op je harde schijf geraakt, bijvoorbeeld in de PAGEFILE of in de SWAPFILE (wisselbestand).
8. De browser laat je bijna op hetzelfde moment zien wat er is opgeslagen. Er is een verschil in de "waargenomen snelheid" en de "daadwerkelijke snelheid" van het surfen op internet, en dat verschil zit hem er in hoe snel de pagina gedownload is van de webserver (daadwerkelijke), en hoe snel de pagina daarna door je brwoser en grafische kaart op het beeld wordt vertoond (waargenomen). Het feit dat jij het niet gezien hebt, wil nog niet zeggen dat het niet in de cache van je browser zit.

De geschiedenis van het World Wide Web (of het "web" zoals we het kortweg nu noemen) start in CERN[1] in 1989. Het werd bedacht door Tim Berners-Lee en Robert Cailliau die een simpel hypertekst-gebaseerde systeem bouwde om informatie te delen. In de jaren erna ontwikkelde Tim Berners-Lee verder aan dat systeem, totdat CERN in 1993 aankondigde dat iedereen het web gratis mocht gebruiken, en dat zorgde ervoor dat het web als het ware 'explodeerde' (enorm snel groeide).
Het Web is een "client/server" gebaseerd systeem, met clients zoals Internet Explorer, Firefox, Safari, Opera en anderen die een verbinding leggen met web-servers zoals IIS (de webserver van Microsoft) en Apache (een populaire open-source webserver) die de clients voorzien van inhoud in de vorm van HTML[2] pagina's. Veel organisaties en personen bieden informatie aan in de vorm van verzamelingen pagina's die op servers staan, en daarmee beschikt de hele wereld over een gigantische berg aan informatie.
Maar waarom maken we ons druk om internetbeveiliging? Web servers zijn te vergelijken met de etalage van een winkel. Stel dat het jouw winkel is. Dan kun je in je etalage reclame zetten, en voorbeelden van produkten of informatie over dingen die je aanbiedt, maar jij wil kunnen bepalen wat er in je etalage te zien is. Je wil niet dat de etalage open/onbeschermd is, want dan zou elke voorbijganger er gratis uit kunnen halen wat ze willen. En je wil ook weten als iemand een baksteen door de ruit probeert te gooien, dat het glas dan niet snel breekt! Helaas zijn webservers complexe programma's, en de kans is groot dat ze een aantal fouten (bugs) bevatten, en die worden misbruikt door de mensen met minder goede bedoelingen om bij gegevens te komen waar ze eigenlijk niet bij mogen.
En het omgekeerde is ook waar. Er zitten ook risico's aan de client klant, zoals aan je browser. Er zijn een aantal kwetsbaarheden ontdekt in het afgelopen jaar waarmee een bezoek aan een website van een kwaadwillende, de computer van de bezoeker schade op kan lopen.

10.1.2 Aan de deur rammelen

Standaard HTML pagina's worden verzonden via HTTP[3]; dit op TCP gebaseerde protocol maakt gebruik van platte tekst, zodat we via tools als "telnet" of "netcat" makkelijk verbinding kunnen maken met de server. We kunnen op die manier inzicht krijgen in de software die op de server draait. Een voorbeeld:

```
simon@exceat:~> netcat www.computersecurityonline.com 80
HEAD / HTTP/1.0
HTTP/1.1 200 OK
Date: Fri, 07 Jan 2005 10:24:30 GMT
Server: Apache/1.3.27 Ben-SSL/1.48 (Unix) PHP/4.2.3
Last-Modified: Mon, 27 Sep 2004 13:17:54 GMT
ETag: "1f81d-32a-41581302"
Accept-Ranges: bytes
Content-Length: 810
Connection: close
Content-Type: text/html
```

Door het intikken van "HEAD / HTTP/1.0" gevolgd door twee keer drukken op de "Return" (Enter) toets, krijg ik alle bovenstaande informatie over de HTTP Server. Elke versie en 'merk' HTTP Server geeft een eigen set aan informatie terug na dit commando – een IIS server zal bijvoorbeeld het volgende melden:

---

1 *Centre Européen pour la Recherche Nucléaire* (Europees Centrum voor Nucleair Onderzoek)
2 Hyper Text Markup Language

3 Hyper Text Transfer Protocol

```
simon@exceat:~> netcat www.microsoft.com 80
HEAD / HTTP/1.0
HTTP/1.1 200 OK
Connection: close
Date: Fri, 07 Jan 2005 11:00:45 GMT
Server: Microsoft-IIS/6.0
P3P: CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA PSD TAI TELo OUR
SAMo CNT COM INT NAV ONL PHY PRE PUR UNI"
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: public, max-age=9057
Expires: Fri, 07 Jan 2005 13:31:43 GMT
Last-Modified: Fri, 07 Jan 2005 10:45:03 GMT
Content-Type: text/html
Content-Length: 12934
```

Je kunt nog doorgaan en nog meer informatie verzamelen door gebruik te maken van het "OPTIONS"-mogelijkheid in het HTTP verzoek, dus:

```
simon@exceat:~> netcat www.computersecurityonline.com 80
OPTIONS / HTTP/1.0
HTTP/1.1 200 OK
Date: Fri, 07 Jan 2005 10:32:38 GMT
Server: Apache/1.3.27 Ben-SSL/1.48 (Unix) PHP/4.2.3
Content-Length: 0
Allow: GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, PATCH, PROPFIND,
PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK, TRACE
Connection: close
```

Daarmee weet je welke HTTP commando's je op die server kunt gebruiken. Als je dit elke keer met de hand moet doen, kost dat nogal wat moeite, en als je dan ook handmatig op moet zoeken welke reactie welke betekenis heeft en welke kwetsbaarheden welke server heeft, daar wordt niemand blij van. Gelukkig voor ons hebben een aantal ondernemende mensen een geautomatiseerde oplossing hiervoor bedacht, en die heet "nikto". "Nikto" is een Perl script die een aantal tests automagisch uitvoert ! De mogelijke opties zijn:

```
-Cgidirs+ Scan these CGI dirs: 'none', 'all', or a value like '/cgi/'
-cookies print cookies found
-evasion+ ids evasion technique (1-9, see below)
-findonly find http(s) ports only, don't perform a full scan
-Format save file (-o) Format: htm, csv or txt (assumed)
-generic force full (generic) scan
-host+ target host
-id+ host authentication to use, format is userid:password
-mutate+ mutate checks (see below)
-nolookup skip name lookup
-output+ write output to this file
-port+ port to use (default 80)
-root+ prepend root value to all requests, format is /directory
-ssl force ssl mode on port
-timeout timeout (default 10 seconds)
-useproxy use the proxy defined in config.txt

-Version print plugin and database versions
```

-vhost+ virtual host (for Host header)
(+ means it requires a value)
These options cannot be abbreviated:
-debug debug mode
-dbcheck syntax check scan_database.db and user_scan_database.db
-update update databases and plugins from cirt.net
-verbose verbose mode
IDS Evasion Techniques:
1 Random URI encoding (non-UTF8)
2 Directory self-reference (/./)
3 Premature URL ending
4 Prepend long random string
5 Fake parameter
6 TAB as request spacer
7 Random case sensitivity
8 Use Windows directory separator (\)
9 Session splicing
Mutation Techniques:
1 Test all files with all root directories
2 Guess for password file names
3 Enumerate user names via Apache (/~user type requests)
4 Enumerate user names via cgiwrap (/cgi-bin/cgiwrap/~user type requests)

"Nikto" is behoorlijke allesomvattend in zijn rapportage, zoals je hieronder kunt zien:

exceat:/# ./nikto.pl -host www.computersecurityonline.com

- Nikto 1.34/1.29 - www.cirt.net
+ Target IP: 217.30.114.2
+ Target Hostname: www.computersecurityonline.com
+ Target Port: 80
+ Start Time: Fri Jan 7 12:23:56 2005

- Scan is dependent on "Server" string which can be faked, use -g to override
+ Server: Apache/1.3.27 Ben-SSL/1.48 (Unix) PHP/4.2.3
- Server did not understand HTTP 1.1, switching to HTTP 1.0
+ Server does not respond with '404' for error messages (uses '400').
+ This may increase false-positives.
+ Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, PATCH, PROPFIND, PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK, TRACE
+ HTTP method 'PUT' method may allow clients to save files on the web server.
+ HTTP method 'CONNECT' may allow server to proxy client requests.
+ HTTP method 'DELETE' may allow clients to remove files on the web server.
+ HTTP method 'PROPFIND' may indicate DAV/WebDAV is installed. This may be used to get directory listings if indexing is allowed but a default page exists.
+ HTTP method 'PROPPATCH' may indicate DAV/WebDAV is installed.
+ HTTP method 'TRACE' is typically only used for debugging. It should be disabled.
+ Apache/1.3.27 appears to be outdated (current is at least Apache/2.0.50). Apache 1.3.31 is still maintained and considered secure.
+ Ben-SSL/1.48 appears to be outdated (current is at least 1.55)
+ PHP/4.2.3 appears to be outdated (current is at least 5.0.1)
+ PHP/4.2.3 - PHP below 4.3.3 may allow local attackers to safe mode and gain access to unauthorized files. BID-8203.
+ Apache/1.3.27 - Windows and OS/2 version vulnerable to remote exploit. CAN-2003-0460
+ Apache/1.3.27 - Apache 1.3 below 1.3.29 are vulnerable to overflows in mod_rewrite and mod_cgi. CAN-2003-0542.
+ /~root - Enumeration of users is possible by requesting ~username (responds with Forbidden for real users, not found for non-existent users) (GET).
+ /icons/ - Directory indexing is enabled, it should only be enabled for specific directories (if required). If indexing is not used all, the /icons directory should be removed. (GET)
+ / - TRACE option appears to allow XSS or credential theft. See http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details (TRACE)
+ / - TRACK option ('TRACE' alias) appears to allow XSS or credential theft. See http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details (TRACK)

+ /CVS/Entries - CVS Entries file may contain directory listing information. (GET)

+ /images/ - index of image directory available (GET)
+ /manual/ - Web server manual? tsk tsk. (GET)
+ /cgi-bin/cgiwrap - Some versions of cgiwrap allow anyone to execute commands remotely. (GET)
+ /cgi-bin/cgiwrap/~adm - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~bin - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~daemon - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~lp - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~root - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~xxxxx - Based on error message, cgiwrap can likely be used to find valid user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /cgi-bin/cgiwrap/~root - cgiwrap can be used to enumerate user accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop user enumeration. (GET)
+ /css - Redirects to http://www.computer-security-online.com/css/ , This might be interesting...
+ 2449 items checked - 15 item(s) found on remote host(s)
+ End Time: Fri Jan 7 12:25:36 2005 (100 seconds)

• 1 host(s) tested

Met de diverse opties van Nikto kun je de tool precies laten doen wat je wil bereiken, waaronder stealth, mutatie en cookie detectie.

10.1.3 Door mat glas heen kijken - SSL

Het duurde niet lang voordat mensen het er over eens waren dat HTTP in platte tekst geen goede basis was voor veilige communicatie. Dus werd er gewerkt aan een versie waarbij gebruik werd gemaakt van encryptie. Die komt in de vorm van SSL[4], en dat is een redelijk veilige 40 of 128 bit publieke sleutel encryptie-methode. De 40 bit sleutel is een stuk minder veilig dan de 128 bit-versie en is, met speciale hardware, binnen enkele minuten te kraken via een brute force aanval, terwijl het via die methode kraken van de 128 bit sleutel langer duurt dan de leeftijd van het universum. Er zijn echter meer complexe technische aanvallen waarbij gebruik wordt gemaakt van wat een "known cyphertext" aanval wordt genoemd – daarbij wordt de encryptie-sleutel achterhaald door een groot aantal berichten (> 1 miljoen) te analyseren. In ieder geval is de kans klein dat je nu even snel een 128-bit-encryptie gaat proberen te kraken – de vraag is: wat kunnen we te weten komen over SSL HTTP Servers? Eigenlijk best veel. SSL doet niet meer dan het encrypten van het standaard HTTP-verkeer. Als we een SSL-tunnel opzetten, dan kunnen we vervolgens op de in sectie 1.1. behandelde manier de server bevragen. Het opzetten van een SSL-tunnel is redelijk simpel, en er is zelf een speciaal stuk gereedschap om dat te doen, namelijk "stunnel". Voer het volgende in in een bestand met de naam stunnel.conf, (waarbij je "ssl.enabled.host" verandert in de naam van de SSL server waar je een verbinding mee wil opzetten):

```
client=yes
verify=0
[psuedo-https]
accept = 80
connect = ssl.enabled.host:443
TIMEOUTclose = 0
```

Stunnel zal dan een verbinding opzetten tussen poort 80 op je computer, met remote SSL Poort 443 en over die tunnel zal vervolgens platte tekst worden uitgewisseld, dus je kunt een verbinding met de server opzetten op dezelfde manier als we hiervoor hebben behandeld :

---

4 Secure Sockets Layer

simon@exceat:~> netcat 127.0.0.1 80
HEAD / HTTP/1.0
HTTP/1.1 200 OK
Server: Netscape-Enterprise/4.1
Date: Fri, 07 Jan 2005 10:32:38 GMT
Content-type: text/html
Last-modified: Fri, 07 Jan 2005 05:32:38 GMT
Content-length: 5437
Accept-ranges: bytes
Connection: close

10.1.4 Iemand anders in jouw plaats iets laten doen – Proxies

Proxies zijn tussenpersonen in de het HTTP afhandelingsproces. De client stelt een vraag aan de proxy, de proxy
stelt de vraag aan de server, de server reageert richting de proxy en tot slot geeft de proxy het resultaat terug aan de client. Proxy servers zijn zelf kwetsbaar voor aanvallen en vormen ook een lanceerplatform voor aanvallen op andere webservers. Ze kunnen echter ook bijdragen aan de beveiliging door verbindingen te filteren, zowel naar als vanaf servers.

10.2 Web Kwetsbaarheden

De simpele handeling om iemand iets te geven waar ze om vragen is een stuk ingewikkelder als je in de schoenen van een verkoper staat. Websites die je iets proberen te verkopen, bedrijven die produkten verkopen, bloggers die ideeen en persoonlijkheid verkopen of kranten die nieuws verkopen, moeten meer doen dan simpelweg wat HTML-teksten en plaatjes op een website zetten. Dat soort aanbieders maken dynamische webpagina's die je helpen te beslissen wat je moet vragen/te vinden wat je zoekt, laten je alternatieven zien, bevelen andere opties aan, proberen je extra accessoires te laten kopen, zorgen dat je veilig kunt betalen etc. En daarvoor is complexe software nodig. Als we overstappen van simpele websites naar complexe web applicaties krijgen we ook te maken met een heel andere wereld op gebied van uitdagingen rond beveiliging.

10.2.1 Script Talen

Er zijn veel script talen gebruikt om applicaties te ontwikkelen waarmee bedrijven hun produkten en services kunnen aanbieden op het web. Dat extra verkoopkanaal biedt allerlei mogelijkheden om extra te verkopen, maar geeft aanvallers ook een extra toegang tot het bedrijf. Het merendeel van de kwetsbaarheden in web applicaties vind je niet bij de bugs in de ontwikkel-taal, maar in de routines en procedures waarmee de web applicatie is ontwikkeld, evenals de manier waarop de webserver is geconfigureerd. Een voorbeeld: als

een webformulier vraagt om een postcode en de gebruiker voert "abcde" in, kan de webapplicatie onvoorspelbaar reageren als de programmeur niet heeft geregeld dat foute invoer netjes wordt afgehandeld. Er zijn veel talen die gebruikt worden om web applicaties te maken, waaronder CGI, PHP en ASP.

Common Gateway Interface (CGI): Whatis.com definieert een CGI als "A standaard manier voor een webserver om een verzoek van de gebruiker door te geven aan een programma en gegevens terug te ontvangen om door te sturen naar de gebruiker." CGI maakt deel uit van het Hypertext Transfer Protocol (HTTP).
Er kunnen meer talen gebruikt worden om te zorgen dat een programma gegevens van een gebruiker kan ontvangen en verwerken. De populairste CGI applicaties zijn: C, C++, Java en PERL.

PHP – Hypertext Preprocessor (PHP): PHP is een open-source server-side script taal waar het script is opgenomen (embedded) binnen een web pagina, samen met de HTML. Voordat een pagina wordt verstuurd naar een gebruiker roep de webserver PHP aan om te kijken welke PHP-acties er nodig zijn en die uit te voeren. Waar HTML statische informatie toont, is het met PHP voor ontwikkelaars mogelijk pagina's te bouwen die gebruikers voorzien van dynamische informatie, die zonodig is gepersonaliseerd op basis van invoer van die gebruiker. HTML pagina's die PHP scripting bevatten kun je doorgaans herkennen omdat ze als extensie ".php" hebben (I. p.v. .htm of .html).

Active Server Pages (ASP): Webpagina's die Active server pages (ASP) gebaseerd zijn, zijn database gedreven dynamische pagina's die je herkent aan hun .ASP extensie. Ze gebruiken ActiveX
scripting – meestal VB Script of Jscript code. Als een browser een ASP-pagina opvraagt, genereert de Webserver een pagina met HTML-code en stuurt die vervolgens naar de browser – op die manier krijgt de gebruiker real time data te zien, maar ze zijn kwetsbaarder voor beveiligingsproblemen.

10.2.2 Veel voorkomende Web Applicatie Problemen

Web applicaties hebben niet persé hun eigen soort problemen, maar het feit dat het webapplicaties zijn zorgt wel voor enkele bijzonder problemen die samenhangen met het web. Nu het testen van webapplicaties volwassener is, is er ook meer aandacht voor de beveiliging van web-applicaties en is er een aparte categorie samengesteld van web-kwetsbaarheden. Veel voorkomende problemen met webapplicaties zijn hieronder opgesomd, conform het OSSTMM Risk Assessment Values (http://www.isecom.org/securitymetrics.shtml) overzicht, een specifieke manier om beveiliging te meten op basis van hoe het andere dingen beinvloedt.

Authenticatie
Dit zijn de identificatie- en authorisatie-mechanismen die worden gebruikt om zeker te stellen dat een persoon of computer die webservices wil gebruiken, dat ook mag.

Elke keer als je inlogt op een webpagina die je persoonlijke gegevens heeft, ben je aan het authenticeren. Authenticatie betekent vaak het geven van een inlognaam en wachtwoord. Soms moet je een identificatie-nummer geven, of moet je simpelweg van een geaccepteerd IP-adres komen (white-listing).

Non-Repudiation (Niet-weerlegbaarheid)

Een record waarmee je kunt bewijzen dat de data die naar of vanuit een webapplicatie is verstuurd, ook daadwerkelijk is verstuurd, en wanneer. De afzender kan het niet ontkennen/weerleggen.

Alhoewel je dat misschien niet kunt zien, houden de meeste webapplicaties bij welke aankopen je doet vanaf een bepaald IP-adres en een bepaalde browser op een bepaald besturingssysteem als een record, dat het waarschijnlijk was dat jij die aankoop deed. Zonder specifieke "authenticatie" kunnen ze niet zeker weten dat jij het was of iemand anders.

Vertrouwelijkheid
Een manier om zeker te stellen dat communicatie met de webapplicatie niet afgeluisterd kan worden door een ander persoon.

Het HTTPS-deel van een interactie met een web-applicatie zorgt vrij goed voor vertrouwelijkheid. Het zorgt ervoor dat de informatie die over het internet reist, niet door anderen ingezien kan worden.

Privacy
Een manier om te zorgen dat de manier waarop je verbinding legt en communicatie onderhoudt met een webapplicatie voor een ander niet voorspelbaar is.

Alhoewel het zeldzaam is, is het voor te stellen dat een webapplicatie die zeer privacygevoelige informatie bevat, niet eens laat zien dat hij bestaat tenzij je van de juiste plaats (ip-adres) verbinding legt en weet op welke deur je moet kloppen en hoe je binnenkomt. Een manier is dat je op 5 plaatsen moet klikken in een plaatje, in de juiste volgorde, voordat je een inlogscherm te zien krijgt. Een andere manier noemt men port-knocking (aan de poort kloppen), en het betekent dat de webserver een bepaalde volgorde van acties verwacht voordat hij een poort, b.v. de HTTP-poort, opent.

Schadeloos-stelling (Indemnification)
Een manier om vast te stellen dat de webapplicatie juridische bescherming heeft, dat geregeld is dat je schadeloosstelling krijgt in het geval er iets mis gaat.

Sommige websites vermelden bij het inloggen duidelijk dat dat alleen bedoeld is voor daarvoor geautoriseerde personen. Als iemand een inlognaam en wachtwoord steelt of een wachtwoord weet te kraken (brute-force) dan kan de inbreker, als hij gepakt wordt, niet zeggen dat hij niet wist dat hij daar eigenlijk niet binnen mocht.

Integriteit
Dit is een bewijs van de geldigheid van de communicatie met de webapplicatie om zeker te stellen dat wat verstuurd is en wat ontvangen werd, identiek is, en als er wijzigingen worden doorgevoerd, dat zowel de website als de gebruiker daar een logbestand van hebben.

Sommige webapplicaties leveren een "HASH" samen met bestanden die je kunt downloaden. Die hash is een controlegetal, berekend op basis van het te downloaden bestand. Als je het bestand downloadt, kun je de hash ook zelf berekenen; als de getallen overeenkomen, is er niet (onderweg of anders) gerommeld met het bestand.

Veiligheid (Safety)
Dit is hoe we de webapplicatie beschermen tegen zijn eigen beveiligingsmaatregelen. Als de beveiliging uitvalt, moeten we zorgen dat daarmee niet ook de webapplicatie (deels) uitvalt.

Het is bijvoorbeeld goed mogelijk te zorgen dat de applicatie een daemon gebruikt om zichzelf opnieuw te initialiseren of om te voorkomen dat een applicatie crashed na een aanval, door de applicatie zichzelf alleen virtueel te laten presenteren. Er zijn ook scenario's mogelijk waar een webapplicatie een inbraak-detectiesysteem (intrusion detection system, IDS) gebruikt om een aanval te "stoppen" door de aanvaller op zijn IP-adres te blokkeren. (het hierna volgende stukje vond ik lastig te vertalen; daarom heb ik het laten staan in het Engels) In this case, we can't say Safety exists if the security device is configured to prevent an attacker from spoofing the web app's own resources and causing this defense to block important traffic. Instead, it is considered either a misconfiguration of the defense or in some cases a weakness of design. Don't confuse a poorly made or "accidental" defense with a designed loss control.

Bruikbaarheid (Usability)
Een manier om te voorkomen dat de gebruiker keuzes moet maken op gebied van beveiliging, als hij de webapplicatie wil gebruiken. Dat betekent dat beveiliging ingebouwd moet zijn, en dat de gebruiker niet hoeft te kiezen welk beveiligingsmechanisme hij aan of uit moet zetten.

Als een webapplicatie HTTP over SSL (HTTPS) gebruikt kunnen we zeggen dat hij Bruikbaarheid toepast als onderdeel van de beveiliging. Als de applicatie je echter ook de mogelijkheid geeft om onveiliger te communiceren, b.v. om je credit-card-nummer via het onveiliger e-mail uit te wisselen in plaats van een webform met HTTPS, dan is er geen sprake van Bruikbaarheid.

Continuiteit
Dit is hoe we zorgen dat een service die gebaseerd is op een webapplicatie van storingen, onafhankelijk van de problemen die kunnen optreden en rampen die voorkomen. Vaak heeft een webapplicatie die veel verkeer moet verwerken een reverse proxy voorgebouwd, zodat het verkeer naar 1 van vele mirror-servers wordt geleid. Zodoende kan de service beschikbaar blijven, ook als er een webserver uitvalt. Een ander voorbeeld is een webapplicatie waarvan de website op meerdere websites op het internet wordt gecached, zodat als je die site bezoekt, je niet persé de 'echte' webserver bezoekt, maar een gecachte versie van de pagina. Als een cache crashed of niet meer goed is (gets corrupted) wordt her verkeer naar een andere cache of de 'echte' webserver geleid.

Alarm
Een melding, direct of met vertraging, over een probleem met één van deze mechanismes.

Een simpele vorm van een alarm is een logbestand die wordt gemaakt door een webserver. Niet zo prettig aan een alarm is dat je het kunt negeren. Dat is zeker waar als het continu af gaat. En in het geval van een logbestand gaat er misschien helemaal geen alarm af. Een alarm is zo goed als je reactie-tijd.

Oefeningen:

1. Start Google en tik "inurl:search.asp" of "inurl:search.php". Probeer bij één of enkele websites die je als zoekresultaat krijgt, het volgende in te tikken in het zoekveld **<script>alert ("hello")</script>**. Wat gebeurt er? Probeer dit bij verschillende sites.

2. In Google, tik "inurl:login.asp" en "inurl:login.php". Probeer op één van de sites die je als zoekresultaat krijgt in de velden om in te loggen special characters (@#$^&) voor de gebruikersnaam (username) en het wachtwoord. Wat gebeurt er? Probeer dit nu bij diverse sites.

3. Nu je weet welke beveiligingsmechanismes een webapplicatie kan toepassen, surf naar je favoriete interactieve website en stel vast welke beveiligingsmechanismes er worden toegepast die aan de RAV classificaties voldoen.
4. Veel besproken web-kwetsbaarheden zijn Cross Site Scripting (XSS) en SQL injection. Zoek uit wat het is/hoe ze werken en hoe een aanvaller ze gebruikt om data/informatie te stelen van een web-applicatie.

10.2.3 Richtlijnen voor het bouwen van veilige Web Applicaties

Alhoewel er veel meningen zijn en de meeste details van het programmeren met beveiliging in het achterhoofd afhankelijk zijn van de vaardigheden van de programmeur en de mate waarin hij vaardig is in een programmeertaal , zijn er onderstaande basis richtlijnen die afgeleid zijn van OSSTMM (http://www.osstmm.org) materiaal.
1. Zorg dat beveiliging geen beslissingen van de gebruiker vergt.
2. Zorg dat alle invoer en uitvoer met betrekking tot de applicatie ook echt nodig is (vanuit het bedrijf gezien, "business justification").
3. Stop alle invoer in quarantaine en voer invoercontroles uit, ook op de applicatie-inhoud (content).
4. Beperk rechten (voor systemen en gebruikers).
5. Versleutel data.
6. Bereken hash-es voor de onderdelen.
7. Zorg dat alle interacties aan de serverkant plaatsvinden.
8. Bouw beveiliging op in lagen.
9. Onzichtbaarheid is het best – zorg dat alleen de noodzakelijke services te zien zijn.
10. Stel grenzen en bewaking in om alarm te kunnen genereren.
11. Gebruikers en helpdesks moeten zich bewust zijn van de noodzaak van beveiliging (security awareness).
Oefeningen:
1. Geef voorbeelden voor 3 van bovenstaande richtlijnen.
2. Geef 3 soorten technologie die je kunt toepassen om alarm te slaan rond webapplicaties.

10.3 HTML Basis – A korte introductie

HTML is een set instructies die beschrijft hoe informatie kan worden gepresenteerd door een webserver (Apache, Internet Information Server) aan een browser (Firefox, Opera). Het vormt het hart van het World Wide Web.
HTML kan veel meer doen dan alleen het tonen van informatie op een web pagina. Je kunt er ook invoerformulieren mee bouwen, waarna de data can worden verwerkt door hogere programmeertalen (Perl, PHP, etc). In een bedrijfsomgeving is dit waar HTML erg nuttig voor is, maar gezien vanuit een hacker is HTML daar het meest kwetsbaar.

10.3.1 HTML lezen

HTML bestaat uit een serie tags, ook wel markups genoemd. Elke openings tag, bijvoorbeeld <h1>, moet een afsluitende tag hebben, </h1>. Daardoor weet de browser waar hij kan stoppen met (bijvoorbeeld) bepaalde opmaak. Kijk eens naar de onderstaande code:
<html>

```
<head><title>Hello World</title></head>
<body>
<h1>Hello World!</h1>
</body>


</html>
```

Figuur 1: HTML Code

We zeggen met de <html>-tag tegen de browser dat dit een HTML document is. We geven op welke titel het document heeft ('Hello World') met de <title> tag. De <body> tag vertelt de browser "hierna volgt de informatie die je op het scherm moet tonen." Tot slot geeft de <h1> tag aan dat de browser die informatie moet tonen in de stijl die hoort bij de "Heading 1" stijl. De tags die beginnen met het '/' -teken zijn simpelweg sluit-tags, zodat de browser weet waar hij kan stoppen.

Oefening 1: Kopieer de code uit figuur 1 en plak die code in een tekstbestand. Geef het tekstbestand de naam hello.html. Open dat bestand in je favoriete browser: je zou dan iets als het volgende moeten zien:

10.3.2 De HTML-broncode bekijken

Alle moderne browsers bevatten een manier waarmee je de HTML-code van een webpagina kunt bekijken. In de meeste gevallen vind je een optie “view source” (“bron bekijken”) onder de “view” (“Beeld”)-menu-keuze in je browser.
Oefening 2: Kies View --> View Source in je browser terwijl je op je favoriete webpagina staat.

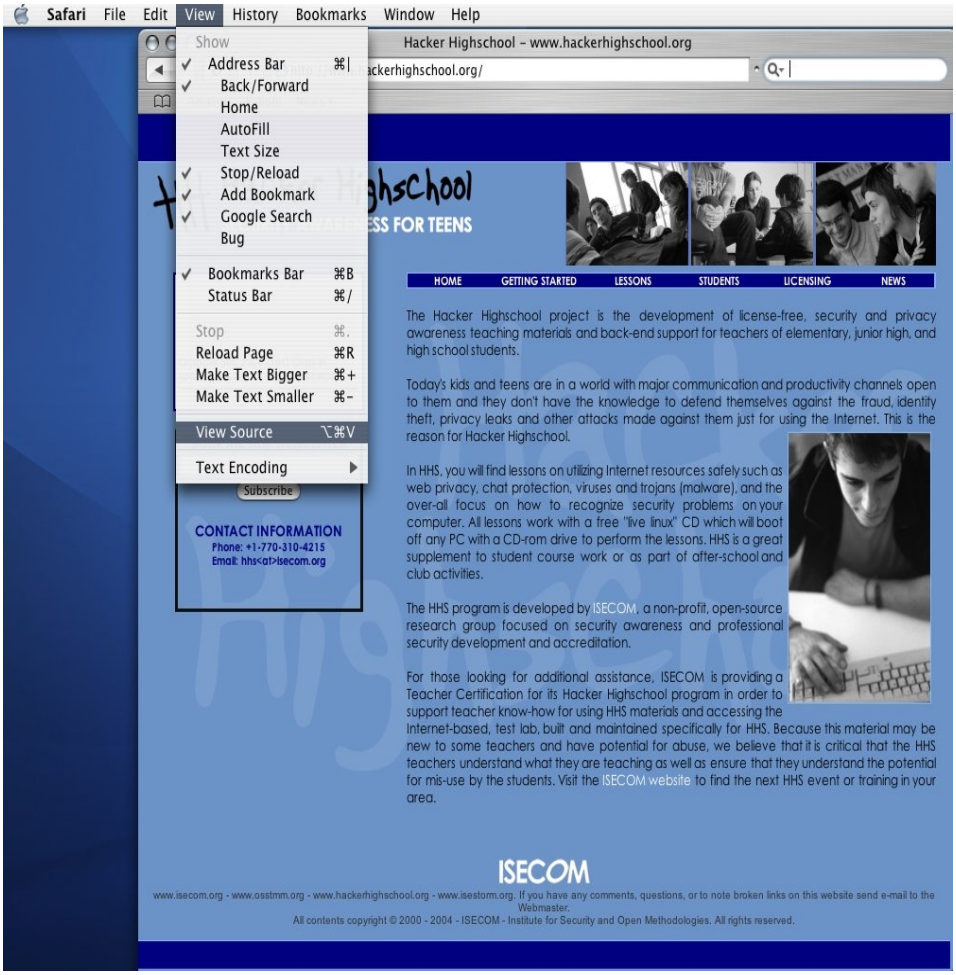

Illustrazione 1Menu Visualizza

Het resultaat zou iets als het volgende moeten zijn:

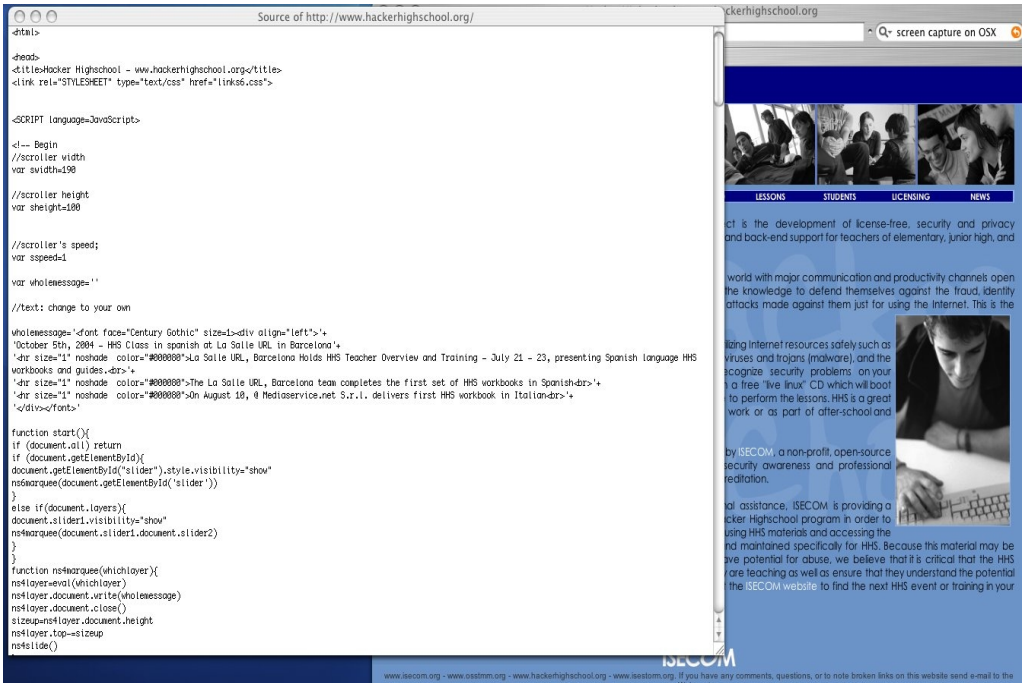

Illustrazione 2Sorgente visualizzato nell'editor di testo

HTML code kan worden bekeken door iedereen met een webbrowser. Daarom is het erg belangrijk voor webpagina-bouwers dat je geen wachtwoorden of andere belangrijke informatie in je HTML-code opneemt. Zoals je ziet is het daar namelijk niet erg veilig.

10.3.3 Links

Links (of hyper-links) zijn het echte hart van HTML pagina's: de grootste kracht van HTML is de mogelijkheid om links naar andere pagina's, documenten, plaatjes etc te maken. Een link ziet er, in HTML-code, als volgt uit www.yahoo.com. De link wordt op een webpagina getoond als www.yahoo.com . Daarmee kunnen bezoekers van de site doorklikken naar Yahoo.
Links kunnen worden gecontroleerd door zogenoemde link checker programma's. Die programma's zoeken in HTML broncode naar de tags en maken een lijstje van alle zo gevonden links. Spammers gebruiken dezelfde werkwijze om achter email-adressen of contact
formulieren te komen, zodat ze die kunnen gebruiken voor hun spampraktijken. Link checkers kunnen ook worden gebruikt om na te gaan of een website "broken" links bevat (links die niet meer werken). Dit komt regelmatig voor, zelfs bij relatief kleine sites.
Oefening 1: Maak een link
Maar een link naar www.hackerhighschool.org die op de webpagina wordt getoond met de tekst Hacker High School.

Bonus oefening: Gebruik het hulpmiddel

Illustratie 2Broncode bekeken in een tekst editor

1. Zoek en download een link check programma
2. Laat het programma www.hackerhighschool.org controleren en leg vast hoeveel broken links je vind.

10.3.4 Proxy methodes voor Web Applicatie Manipulatie

Een HTTP proxy-server speelt tussenpersoon tussen een webserver en een web client(browser). Hij vangt alle verbindingen af tussen die 2, legt ze vast en kan zelfs data aanpassen om te zien hoe de server reageert. Dit kan nuttig zijn als je applicaties wil testen voor diverse cross-site scripting aanvallen (provide reference link here?), SQL Injection aanvallen en elke andere rechtstreekse aanval. Proxy test hulpmiddelen (SpikeProxy, WebProxy, etc) kunnen je helpen bij het testen van proxies. Alhoewel sommige hulpmiddelen automatische opties hebben, zul je snel leren dat ze maar een zwakke vervanging van een echte persoon is die zulke tools bedient.
Oefening 1: Kies je software

1. Download een proxy utility
2. Installeer de software conform het README bestand
3. Wijzig je browser instellingen zodat ze naar de nieuwe proxy wijzen

• Dit is voor dit soort tools normaal gesproken poort 8080 op localhost, maar lees de instructies om er zeker van te zijn.

Zodra de proxy server is geinstalleerd en je browser wijst er naar, surf dan eens wat rond de site die je wil testen. Onthoud, doe dat alleen op een website waarvan je toestemming hebt om er op te testen. Nadat je wat rondgesurfd hebt, wijs je browser naar de admin-pagina van de proxy (voor SpikeProxy http://www.immunitysec.com/resources-freesoftware.shtml) en begin de site te testen. Van de admin pagina kun je de tool de authenticatie van de site laten brute force'n of testen op
cross-site scripting. (we adviseren de Mozilla- of Firefox-browser te gebruiken in combinatie met
http://livehttpheaders.mozdev.org/ en http://addneditcookies.mozdev.org/ om headers en cookies 'on the fly' aan te kunnen passen zonder de noodzaak van een aparte proxy-poort. Het maakt het testen niet alleen een stuk simpeler, het vormt ook een krachtiger set gereedschap dan we gebruiken in ISECOM's OSSTMM Professional Security Tester cursus (OPST). Maar ook voor andere dingen is het handig dat je weet hoe je proxies moet opzetten, zoals advertentie- en spamfilters, privacy filters, etc. We vonden dat je een echte moest installeren, en Spike is daarvoor prima geschikt.
Een proxyserver kan een krachtig gereedschap zijn om vast te stellen hoe solide een web applicatie is. Zowel voor penetratie-tests als kwetsbaarheids-analyse (vulnerability assessments) is het nodig dat je een goede proxy tool in je gereedschapskist hebt zitten. Er staan gedetailleerde handleidingen over hoe je SpikeProxy moet gebruiken op http://www.immunitysec.com/resources-papers.shtml.

10.4 Je server beschermen

Er zijn verschillende stappen die je kunt nemen om je server te beschermen. Daaronder valt ondermeer het up to date houden van je software en het toepassen van de laatste (beveiligings)patches. Denk niet alleen aan applicaties, maar ook het besturingssysteem, je webserver etc. Aanvullend kun je firewalls en Intrusion detections systemen, die we hierna behandelen, gebruiken om je webserver te beschermen.

10.4.1 Firewall

Firewalls waren oorspronkelijk echt brandmuren, schotten tussen 2 ruimtes die moesten voorkomen dat het vuur van de ene ruimte naar de andere oversloeg. Het woord is nu ook in gebruik voor systemen (hardware en software) die moeten voorkomen dat mensen die daar geen rechten voor hebben, bij informatie van een organisatie kunnen. Firewalls zijn een soort deurwachters die, op basis van bepaalde regels, verkeer van en naar de systemen van een organisatie toestaan of blokkeren. Ze vormen een belangrijke beveiligings-schakel die moeten voorkomen dat externen en internen een systeem van een organisatie kunnen aanvallen. Het vormt de 1ste en belangrijke lijn in de scheiding tussen interne en externe systemen.
Firewalls worden meestal geplaatst tussen het Internet en de informatiesystemen van een organisatie.
De firewall-beheerder configureert de firewall met regels die verkeer toestaan of blokkeren. De regels zijn een combinatie van Internet Protocol (IP) adressen en Poorten; die regels zijn afhankelijk van wat een organisatie wil bereiken en nodig heeft. Zo kan een school studenten toegang geven op basis van een identiteitskaart.
De regel voor een bewaker in de school kan dus zijn dat alle personen die een geldige identiteitskaart tonen, naar binnen mogen, en dat de rest wordt tegen gehouden. De bewaker moet echter een andere regel gebruiken voor mensen die de school willen verlaten; die regel kan zijn om iedereen door te laten, met uitzonder van heel kleine kinderen die niet begeleid worden door een volwassene. Op dezelfde manier moet je regels maken voor een firewall configuratie, waarbij je rekening houdt met het soort organisatie, de waarde van van de informatie, de kosten van de beveiliging, het beveiligingsbeleid en de uitkomst van de risico-analyse (welk rampen, groot en klein, kunnen optreden, wat is dan de schade, welke maatregelen kun je treffen, wat zijn de kosten, en wat beslis je dan hoe te beveiligen).
De firewall kan de inhoud van wat er langs komt niet bekijken; net zoals de bewaker in de school iedere persoon met een geldige kaart doorlaat, los van andere kenmerken of gedrag, laat een firewall verkeer voornamelijk door op basis van IP adressen en poort-nummers. Dus je kunt binnenkomen als je maar zorgt dat je een geldig IP-adres kan laten zien en/of poort-nummer gebruikt. Om dit risico aan te pakken gebruiken organisaties aanvullend Intrusion Detection Systemen (Inbraak Detectie Systemen, IDS-en), die we in de volgende paragraaf uitgebreider behandelen.
Er zijn verschillende soorten firewalls, met elk hun eigen kenmerken, zoals packet filters (baseren hun werking op IP packets), stateful firewall (baseren hun werking voornamelijk op de status van verbindingen (connection state)) en applicatie firewall (gebruiken proxy). Een voorbeeld van een firewall regel kan zijn: Houd inkomend TCP-verkeer van adres 200.224.54.253 poort 135 tegen.
(Een denkbeeldig voorbeeld); zo'n regel zorgt er voor dat een computer die verbonden is aan Internet om al het verkeer afkomstig van de computer met IP adres 200.224.54.253 die Poort 135 gebruikt, tegen te houden.

Belangrijke activiteiten op gebied van firewalls zijn het initieel configureren (ondermeer de 1ste set met regels maken), systeem beheer/onderhoud (aanvulling en wijzigingen verwerken), audit logs bekijken, reageren op alarm en het testen van de configuratie.

10.4.2 Intrusion Detectie Systemen (IDS)

Stel je nog even die school voor, met die bewaker(s): hoe stellen ze inbraakpogingen van ongeautoriseerde personen vast? Er zouden inbraakalarmen worden geinstalleerd. Dit is precies de functie van een intrusion detection systeem in de computer-wereld. Firewall (bewaker of hek) en IDS (inbraaakalarm of patrouillerende bewakers) werken samen; terwijl de firewall het in- en uitgaande verkeer regelt, slaat het IDS alarm bij inbraken.

Maar hoe helpt een IDS eigenlijk? Net als inbraakalarmen, geeft een IDS een bepaalde persoon een seintje bij onraad wanneer een ongeautoriseerd packet is gesignaleerd (inkomend of uitgaand). Daarnaast kunnen IDS-en vaak gelijk actie nemen en zorgen dat de afzender niet nog meer kwaad kan doen. Ook kan een IDS scripts uitvoeren, bijvoorbeeld om de gevolgen van een denial of service te beperken, door het netwerkverkeer van een (groep) computer(s) te blokkeren.
IDS-en kunnen host- of netwerk gebaseerd zijn; host gebaseerde IDS-en worden gebruikt op individuele computers terwijl netwerk IDS-en tussen computers staan en hun werking doen. Host gebaseerde IDSen kunnen afwijkende activiteit op belangrijke computers detecteren (zoals een afwijkende hoeveelheid verkeer vanaf een bepaalde computer), alarm slaan of aanpakken; netwerk gebaseerde IDS-en werken op een soortgelijke manier tussen computers.
IDS-en vergelijken gegevens zoals ze die meten met gegevens in een database waarin staat wat 'normaal' gebruik is. Bij afwijkingen slaan ze alarm, leggen ze vast wat ze meten, nemen ze aktie etc, afhankelijk van de gemaakte regels.
Oefeningen:
1. Zijn zowel een firewall als Intrusion Detectie Systeem in elke organisatie nodig om informatiesystemen te beveiligen? Zo ja, waarom? Zo nee, waarom niet?
2. Stel je een persoon voor die bij de ingang/receptie van een school zit. Heeft hij/zij internettoegang nodig? Welke regel zou je in een firewall kunnen zetten om te zorgen dat die functionaris geen internet kan gebruiken?
3. Kan een student bij de database met ieders raportcijfers? Hoe kun je dit controleren? Hoe stel je vast als een externe partij via internet toegang probeert te krijgen?

10.5 Veilige Communicatie

Om betrouwbaar veilig te communiceren moeten er drie dingen geregeld zijn: A) Authenticiteit b) Integriteit c) Non repudiation.
Authenticiteit: dit concept omvat dat je zeker weet dat een persoon is wie hij/zij zegt te zijn. Het is niet moeilijk email te falsificeren, of om de naam van een webpagina licht te laten afwijken, bijvoorbeeld http://www.diisney.com lijkt op het eerste oog op een site van Disney, maar het heeft 2 letters "l", en er kan dus heel iemand anders achter schuil gaan, mogelijk met slechte bedoellingen. (in dit geval komt die URL uit op een goksite).

Integriteit: Als communicatie integer is wil zeggen dat wat is verstuurd, exact in die vorm arriveert bij de beoogde ontvanger, en dus al reizend niet is gewijzigd (expres of per ongeluk).
Non repudiation: Als bovenstaande condities zijn zekergesteld, betekent non-repudiation dat de verzender de verzending niet kan ontkennen.

Een voorbeeld: als een website mij een prijs toekent, en ik kan het bewijzen, dat wil zeggen: als een website een kortingsbon stuurt, en ik kan zekerstellen dat de website authentiek is, en dat niemand het bericht heeft gewijzigd, dan kan de site niet ontkennen dat ze de bon hebben verstuurd.
Het document waarmee je deze dingen kunt controleren heet een electronisch certificaat.
Veilige communicatie is belangrijk zodat we met een gerust hart zaken kunnen doen, kunnen communiceren en privacy kunnen zekerstellen.

10.5.1 Privacy en Vertrouwelijkheid (Confidentiality)

De meeste websites ontvangen bepaalde informatie van hun bezoekers – dat kan omdat de bezoeker een webformulier invult, maar het kan ook via cookies. Dit kan redelijk zijn en je helpen – zoals op Amazon.com wordt bijgehouden welke boeken je hebt bekeken en/of gekocht, en, om de veiligheid van de browsende persoon te waarborgen, daarom hebben steeds meer websites regels opgesteld over hoe ze met persoonsgegevens omgaan, vaak als onderdeel van een disclaimer of een privacy statement.
Privacy betekent dat informatie over jou alleen voor jouw ogen is – of beperkt inzichtelijk voor je familie of vrienden, of je contacten, maar op zijn meest de personen/instanties waarvan jij zelf hebt toegestemd dat ze je informatie kunnen inzien/gebruiken. Niemand wil dat iedereen zonder toestemming maar van alles kan doen met zijn/haar persoonsgegevens , en daarom zijn sommige dingen als privé/private bestempeld, dat wil zeggen, ze mogen niet zomaar verspreid/gedeeld worden.
Aan de andere kant heb je ook nog vertrouwelijkheid (**confidentiality**), en die stelt dat informatie geheim moet blijven, maar dit keer gezien vanuit het standpunt van de ontvanger (die moet er omzichtig mee omgaan/zorgen dat geen ongeautoriseerde personen er toegang tot krijgen).
Stel dat je wel zin hebt in een prijs, maar je wil als je hem wint niet dat bekend wordt dat jij de winnaar bent, dan moet je zorgen dat de ontvangers van je organisatie snappen dat je je informatie privé wil houden, geef je aan wie toegang mag hebben tot je gegevens, en verwacht je dat ze vertrouwelijk met je informatie omgaan.
Het is (eigenlijk) nodig dat je het privacystatement bekijkt van elke website die je bezoekt en waar je, al dan niet bewust, informatie achterlaat.
Oefening:
1. Bekijk de privacy-statement eens van de grootste webmail-aanbieders: Google en Hotmail, en bekijk ook het privacystatment eens van andere organisaties, zoals Kennisnet, http://www.kennisnetictopschool.nl/footer/disclaimer . Welke overeenkomsten zijn er? Kun je websites vinden die aangeven dat ze de informatie die je achterlaat met andere organisaties delen? Wat kun je doen als organisaties toch je gegevens aan andere organisaties geven of verkopen?

10.5.2 Vaststellen of je veilig communiceert

Zelfs als je Privacy en Confidentiality hebt geregeld, dan nog kan iemand de berichten onderscheppen. Om de eerder besproken condities goed te regelen moet er een beveiligingslaag zoals de eerder besproken SSL worden toegepast, die gebruik maakt van digitale certificaten om een veilige verbinding op te bouwen (daarmee zorgt het voor de authenticiteit, integriteit en non repudiation) en een versleutelingslaag op de communicatie toepast (om informatie te beschermen in het geval iemand de informatie in handen krijgt). Deze laag heet Security Socket Layer, SSL, en kun je herkennen aan 2 dingen in de webbrowser.
De communicatie wordt geacht veilig te zijn als het web adres van de URL wijzigt van HTTP naar https, die wijziging zorgt er voor dat de communicatie ook over een andere poort gaat lopen, 443 in plaats van 80. Daarnaast verschijnt onderin de browser een gesloten slot.
Als je je muis over dat slot beweegt verschijnt er een bericht met meer informatie over de toegepaste beveiliging. Tegenwoordig wordt 128 bits aanbevolen. Dat betekent dat er een getal is gebruikt wat in 128 bits 'past'.
Er bestaat een soort aanval met de naam phishing (http://www.antiphishing.org/); daarbij doet een webpagina (of e-mail) zich voor als betrouwbaar, bijvoorbeeld hij lijkt op die/afkomstig van je bank. (de schurken bouwen gewoon een website na, inclusief alle plaatjes, zodat het er uit ziet als een kopie van de echte banksite). Als je vervolgens de website bezoekt (al dan niet via een toegemailde (nep)link, en je op die site je inloggegevens invoert, worden die naar de schurken gestuurd. Om dit soort valstrikken te doorzien moet je dus kunnen bepalen of een site 'authentiek' is, en moet je kunnen zien of er sprake is van veilige communicatie (https en het slotje, maar boeven proberen tegenwoordig met nepcertificaten zelfs het slotje in je browser na te bootsen); je moet ook in staat zijn te beoordelen of een certificaat 'te vertrouwen' is.

10.6 Manieren om te controleren

We hebben tot nu toe de basisconcepten behandeld van webbeveiliging, we hebben belangrijke aangrijpingspunten van webkwetsbaarheden behandeld, en waar je op kunt letten als je veilig wil communiceren.
Waarschijnlijk vraag je je na het lezen van dit alles af: ben ik wel echt veilig als ik alle behandelde acties onderneem? Is mijn systeem veilig? Hebben de programmeurs van de code die ik op mijn website(s) toepas wel rekening gehouden met beveiliging? Hoe kan ik dit allemaal controleren op goede veilige werking?
Het is, als het om jouw veiligheid gaat, inderdaad helaas niet voldoende om te zorgen dat je alle updates hebt toegepast en maar te vertrouwen op de goede bedoelingen van de programmeurs. Het is al eerder voorgekomen dat een patch een bepaalde kwetsbaarheid oploste, maar een nieuwe introduceerde.
Daarom moet je regelmatig controleren of je systeem nog wel veilig is. Gelukkig hebben sommige mensen, vaak in hun vrije tijd, methodes bedacht om dit te controleren, en sommige daarvan kun je gratis gebruiken. Vaak zijn ze in de loop van de tijd verbeterd omdat mensen ze gingen gebruiken, met verbetervoorstellen kwamen etc. Vind dus niet zelf het wiel uit, maar kijk even wat er al voor moois is.

Een voorbeeld van zo'n in de loop van de tijd doorontwikkelde methode is OSSTMM, en die bespreken we hier kort.

10.6.1 OSSTMM

De OSSTMM, een afkorting voor "Open Source Security Testing Manual Methodology" is één van de veelgebruikte methodes om beveiliging te testen. Zoals in de inleiding al gesteld wordt zijn de individuele tests niet revolutionair, maar de OSSTMM vormt een mooie checklist en verwijsindex als je beveiliging op een gestructureerde professionele manier wil testen. OSSTMM bevat een aantal onderdelen: Medewerkers beveiliging, Netwerk beveiliging, Telecommunicatie beveiliging, Draadloze communicatie beveiliging en fysieke beveiliging. Elk onderdeel behandelt WELKE test je moet doen, WAAROM en WANNEER.
De OSSTMM bevat niet de exacte tests, maar beschrijft wat er getest moet worden, welke vorm de testresultaten moeten hebben, de regels waar de testers zich aan moeten houden om de beste resultaten te krijgen en bevat ook de manier waarop je beveiliging kunt meten en in cijfers kunt uitdrukken via RAVs (Risk Assessment Values). De OSSTMM is bedoeld voor professionals, maar goed te lezen en het is leerzaam het door te nemen.

Oefeningen

1. Patching is tegenwoordig een belangrijke activiteit, om kwetsbaarheden zo snel mogelijk te dichten. Zoek eens naar een situatie waarbij er een probleem ontstond, juist door het uitvoeren van een patch. Discussieer eens over de dilemma's van een systeembeheerder, die weet dat een nieuwe patch nieuwe kwetsbaarheden zal veroorzaken. Moet je de patch dan toepassen? En maakt het dan nog uit of je wel of niet kunt beschikken over de sourcecode van de patch?
2. Ga naar http://cve.mitre.org en zoek naar CVEs. Voer de naam van een webserver in (bijvoorbeeld Apache). Wanneer werd de laatste kwetsbaarheid gevonden? Hoe vaak zijn er kwetsbaarheiden opgetreden (per week, per maand, etc.)? In relatie met vraag 1, hoe zinnig is patchen als oplossing voor beveiliging? Waarom wel of niet? Welke andere oplossingen zou je kunnen toepassen om niet te vervallen in het kat-en-muis-patching-spel?
3. Download de OSSTMM en kijk het eens door. Welke onderdelen zou jij benadrukken? Hoe kan OSSTMM een rol krijgen met je (misschien al bestaande) tests van de beveiliging?
4. Wat kun je met de RAVs?

Verder lezen

http://www.osstmm.org
http://nl.wikipedia.org/wiki/Privacy
http://www.mijnprivacy.nl
http://scholier.veilig.kennisnet.nl/beschermjecomputer
http://www.oreilly.com/catalog/websec2/chapter/ch08.html
http://www.w3.org/Security/Faq/
http://www.privacyalliance.org/
http://www.perl.com/pub/a/2002/02/20/css.html
http://www.oreilly.com/catalog/webprivp3p/chapter/ch01.pdf
http://www.defenselink.mil/specials/websecurity/
http://www.epic.org/
http://www.cgisecurity.com/
http://www.eff.org/privnow/