**PROSEDUR PENGUATAN KEAMANAN WEB SERVER (HARDENING)**

**{{nama organisasi}}**

**{{logo organisasi}}**

|  |  |
| --- | --- |
| No Dokumen | : 0.1 (Draft) |
| Revisi | : 00 |
| Tanggal Terbit | : |

|  |  |  |
| --- | --- | --- |
| Disusun  oleh : | Diketetahui  oleh: | Disetujui / Disahkan  oleh : |
|  |  |  |
| **NN**  NN | **NN**  NN | **NN**  NN |

**Riwayat Dokumen**

|  |  |  |
| --- | --- | --- |
| **Versi** | **Tanggal** | **Perubahan** |
| 0.1 (Draft) | 28/02/2023 | Rilis draft awal dengan hanya menyalin sepenuhnya konten dari **PEDOMAN TATA KELOLA KEAMANAN APLIKASI BERBASIS WEB - BSSN** |
| … | … | … |
|  |  |  |

**HARDENING WEB SERVER NGINX**

|  |
| --- |
| **HARDENING WEB SERVER**  **NGINX** |

1. **TUJUAN**

Memberikan petunjuk/*guidance* dalam melakukan hardening pada web server NGINX

1. **RUANG LINGKUP**

Tahapan hardening pada web server NGINX

1. **REFERENSI**
   1. OWASP Secure Configuration Guide Project <https://www.owasp.org/index.php/OWASP_Secure_Configuration_Guide>
2. **LANGKAH - LANGKAH**
   1. Mengecek konfigurasi default files & port Nginx sebagai berikut:
      * **/usr/local/nginx/conf/** or **/etc/nginx/**– The nginx server configuration directory and

/usr/local/nginx/conf/nginx.conf is main configuration file.

* + - **/usr/local/nginx/html/** or **/var/www/html**– The default document location.
    - **/usr/local/nginx/logs/** or **/var/log/nginx** – The default log file location.
    - Nginx **HTTP default port**: TCP 80
    - Nginx **HTTPS default port**: TCP 443

Setelah itu kita dapat menguji perubahan konfigurasi Nginx dengan perintah sebagai berikut:

# /usr/local/nginx/sbin/nginx -t atau

# nginx -t Outputs:

the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok configuration file /usr/local/nginx/conf/nginx.conf test is successful

Untuk perubahan konfigurasi, dengan mengetik: # /usr/local/nginx/sbin/nginx -s reload

atau

# nginx -s reload

Atau untuk menghentikan layanan server Nginx, dengan mengetik:

# /usr/local/nginx/sbin/nginx -s stop atau

# nginx -s stop

* 1. Melakukan Hardening

**#1: Mengaktifkan SELinux**

Security-Enhanced Linux (SELinux) is adalah fitur kernel Linux yang menyediakan mekanisme untuk mendukung keamanan kendali akses sehingga dapat menghentikan banyak serangan kedalam sistem linux.

**Do Boolean Lockdown**

Jalankan perintah getsebool -a dan lockdown system:

|  |
| --- |
|  |
| getsebool -a **| less** getsebool -a **| grep** off getsebool -a **| grep** on |
|  |

Untuk mengamankan mesin, lihat pengaturan perintah getsebool ke ‘on’ atau ke ‘off’. Atur boolean SE Linux yang benar untuk menjaga fungsionalitas dan perlindungan. Perlu dicatat bahwa SELinux menambahkan overhead 2- 8% untuk instalasi RHEL atau CentOS.

**#2: Meminimalisasi previllege**

File halaman web ; /html /php harus melalui partisi yang terpisah. Sebagai contoh, buat partisi bernama /dev// sda5 dan mount di /nginx. Memastikan /nginx di-mount dengan izin noexec, nodev dan nosetuid. Berikut entri /etc

/fstab untuk instalasi nginx:

LABEL=/nginx/nginx ext3 defaults,nosuid,noexec,nodev 1 2

**#3: Hardening Linux /etc/sysctl.conf**

Kita dapat mengontrol dan mengkonfigurasi kernel linux di [/etc/sysctl.conf.](https://www.cyberciti.biz/faq/linux-kernel-etcsysctl-conf-security-hardening/)

|  |
| --- |
|  |
| # Avoid a smurf attack net.ipv4.icmp\_echo\_ignore\_broadcasts **=** 1  # Turn on protection for bad icmp error messages net.ipv4.icmp\_ignore\_bogus\_error\_responses **=** 1  # Turn on syncookies for SYN flood attack protection net.ipv4.tcp\_syncookies **=** 1  # Turn on and log spoofed, source routed, and redirect packets net.ipv4.conf.all.log\_martians **=** 1  net.ipv4.conf.default.log\_martians **=** 1 # No source routed packets here  net.ipv4.conf.all.accept\_source\_route **=** 0  net.ipv4.conf.default.accept\_source\_route **=** 0 # Turn on reverse path filtering net.ipv4.conf.all.rp\_filter **=** 1  net.ipv4.conf.default.rp\_filter **=** 1  # Make sure no one can alter the routing tables net.ipv4.conf.all.accept\_redirects **=** 0  net.ipv4.conf.default.accept\_redirects **=** 0  net.ipv4.conf.all.secure\_redirects **=** 0  net.ipv4.conf.default.secure\_redirects **=** 0 # Don't act as a router net.ipv4.ip\_forward **=** 0  net.ipv4.conf.all.send\_redirects **=** 0  net.ipv4.conf.default.send\_redirects **=** 0 |

|  |
| --- |
| # Turn on execshild kernel.exec-shield **=** 1  kernel.randomize\_va\_space **=** 1 # Tuen IPv6  net.ipv6.conf.default.router\_solicitations **=** 0  net.ipv6.conf.default.accept\_ra\_rtr\_pref **=** 0  net.ipv6.conf.default.accept\_ra\_pinfo **=** 0  net.ipv6.conf.default.accept\_ra\_defrtr **=** 0  net.ipv6.conf.default.autoconf **=** 0  net.ipv6.conf.default.dad\_transmits **=** 0  net.ipv6.conf.default.max\_addresses **=** 1 # Optimization for port usefor LBs  # Increase system file descriptor limit  fs.file-max **=** 65535  # Allow for more PIDs (to reduce rollover problems); may break some programs 32768 kernel.pid\_max **=** 65536  # Increase system IP port limits net.ipv4.ip\_local\_port\_range **=** 2000 65000  # Increase TCP max buffer size setable using setsockopt() net.ipv4.tcp\_rmem **=** 4096 87380 8388608  net.ipv4.tcp\_wmem **=** 4096 87380 8388608  # Increase Linux auto tuning TCP buffer limits # min, default, and max number of bytes to use  # set max to at least 4MB, or higher if you use very high BDP paths  # Tcp Windows etc net.core.rmem\_max **=** 8388608  net.core.wmem\_max **=** 8388608  net.core.netdev\_max\_backlog **=** 5000  net.ipv4.tcp\_window\_scaling **=** 1 |
|  |

**#4: Menghapus semua module Nginx yang tidak diperlukan**

Mengkonfigurasi dan menginstal nginx hanya menggunakan modul yang diperlukan. Misalnya, menonaktifkan modul SSI dan autoindex dengan mengetik:

# ./configure --without-http\_autoindex\_module --without-http\_ssi\_module # make

# make install

Type the following command to see which modules can be turn on or off while compiling nginx server: # ./configure --help | less

Disable modul Nginx yang tidak diperlukan

**(Optional) Mengubah Nginx Version Header**

Edit src/http/ngx\_http\_header\_filter\_module.c, enter:

# vi +48 src/http/ngx\_http\_header\_filter\_module.c Keluar baris – baris sebagai berikut:

|  |
| --- |
|  |
| static char ngx\_http\_server\_string[] = "Server: nginx" CRLF;  static char ngx\_http\_server\_full\_string[] = "Server: " NGINX\_VER CRLF; |
|  |

|  |
| --- |
| Mengubahnya dengan : |
|  |
| static char ngx\_http\_server\_string[] = "Server: Ninja Web Server" CRLF; static char ngx\_http\_server\_full\_string[] = "Server: Ninja Web Server" CRLF; |
|  |

Save and close file tersebut. Lakukan compile di Server. Menambahkan yang berikut ini di nginx.conf untuk mematikan nomor versi nginx yang ditampilkan di semua halaman kesalahan yang dihasilkan secara otomatis:

server\_tokens off

**#5: Use mod\_security (Hanya untuk backend Server Apache)**

Mod\_security menyediakan firewall level aplikasi untuk Apache. Instal mod\_security untuk semua server web Apache backend. Ini akan menghentikan banyak serangan injeksi.

**#6: Install SELinux Policy To Harden The Nginx Webserver**

Secara default SELinux tidak akan melindungi server web nginx. Namun, dapat menginstal dan menyusun perlindungan sebagai berikut. Pertama, instal dukungan waktu kompilasi SELinux yang diperlukan:

# yum -y install selinux-policy-targeted selinux-policy-devel

Unduh kebijakan SELinux yang ditargetkan untuk meningkatkan keamanan server web nginx di server Linux

# cd /opt

# wget '[http://downloads.sourceforge.net/project/selinuxnginx/se-ngix\_1\_0\_10.tar.gz?use\_mirror=nchc'](http://downloads.sourceforge.net/project/selinuxnginx/se-ngix_1_0_10.tar.gz?use_mirror=nchc%27) Untar the file berikut:

# tar -zxvf se-ngix\_1\_0\_10.tar.gz Compile file yang sama

# cd se-ngix\_1\_0\_10/nginx # make

Outputnya akan keluar sebagai berikut:

Compiling targeted nginx module

/usr/bin/checkmodule: loading policy configuration from tmp/nginx.tmp

/usr/bin/checkmodule: policy configuration loaded

/usr/bin/checkmodule: writing binary representation (version 6) to tmp/nginx.mod Creating targeted nginx.pp policy package

rm tmp/nginx.mod.fc tmp/nginx.mod

Instal modul SELinux nginx.pp yang dihasilkan: # /usr/sbin/semodule -i nginx.pp

**#7: Firewall Berbasis Iptables Terbatas**

Script berikut akan memblokir semua port kecuali

* Incoming HTTP (TCP port 80) requests
* Incoming ICMP ping requests
* Outgoing ntp (port 123) requests
* Outgoing smtp (TCP port 25) requests

|  |
| --- |
|  |
| *#!/bin/bash* IPT="/sbin/iptables" *#### IPS ######*  *# Get server public ip*  SERVER\_IP=$**(ifconfig** eth0 **| grep** 'inet addr:' **| awk** -F'inet addr:' '{ print $2}' **| awk** '{ print $1}'**)** LB1\_IP="204.54.1.1"  LB2\_IP="204.54.1.2"  *# Do some smart logic so that we can use damm script on LB2 too*  OTHER\_LB="" SERVER\_IP=""  **[[** "$SERVER\_IP" == "$LB1\_IP" **]] &&** OTHER\_LB="$LB2\_IP" **||** OTHER\_LB="$LB1\_IP"  **[[** "$OTHER\_LB" == "$LB2\_IP" **]] &&** OPP\_LB="$LB1\_IP" **||** OPP\_LB="$LB2\_IP"  *### IPs ###*  PUB\_SSH\_ONLY="122.xx.yy.zz/29"  *#### FILES #####*  BLOCKED\_IP\_TDB=**/**root**/**.fw**/**blocked.ip.txt  SPOOFIP="127.0.0.0/8 192.168.0.0/16 172.16.0.0/12 10.0.0.0/8 169.254.0.0/16 0.0.0.0/8 240.0.0.0/4 255.255.255.255/32 168.254.0.0/16 224.0.0.0/4 240.0.0.0/5 248.0.0.0/5 192.0.2.0/24"  BADIPS=$**( [[** -f ${BLOCKED\_IP\_TDB} **]] && egrep** -v "^#|^$" ${BLOCKED\_IP\_TDB}**)**  *### Interfaces ###* PUB\_IF="eth0" *# public interface* LO\_IF="lo" *# loopback* VPN\_IF="eth1" *# vpn / private net*  *### start firewall ###* |

|  |
| --- |
| **echo** "Setting LB1 $(hostname) Firewall..."  *# DROP and close everything*  $IPT -P INPUT DROP  $IPT -P OUTPUT DROP  $IPT -P FORWARD DROP  *# Unlimited lo access*  $IPT -A INPUT -i ${LO\_IF} -j ACCEPT  $IPT -A OUTPUT -o ${LO\_IF} -j ACCEPT  *# Unlimited vpn / pnet access*  $IPT -A INPUT -i ${VPN\_IF} -j ACCEPT  $IPT -A OUTPUT -o ${VPN\_IF} -j ACCEPT  *# Drop sync*  $IPT -A INPUT -i ${PUB\_IF} -p tcp **!** --syn -m state --state NEW -j DROP  *# Drop Fragments*  $IPT -A INPUT -i ${PUB\_IF} -f -j DROP  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags ALL ALL -j DROP  *# Drop NULL packets*  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags ALL NONE -m limit --limit 5**/**m --limit-burst 7 -j LOG --log-prefix " NULL Packets "  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags ALL NONE -j DROP  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags SYN,RST SYN,RST -j DROP  *# Drop XMAS*  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags SYN,FIN SYN,FIN -m limit --limit 5**/**m --limit-burst 7 -j LOG --log- prefix " XMAS Packets "  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP |

|  |
| --- |
| *# Drop FIN packet scans*  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags FIN,ACK FIN -m limit --limit 5**/**m --limit-burst 7 -j LOG --log- prefix " Fin Packets Scan "  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags FIN,ACK FIN -j DROP  $IPT -A INPUT -i ${PUB\_IF} -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP  *# Log and get rid of broadcast / multicast and invalid*  $IPT -A INPUT -i ${PUB\_IF} -m pkttype --pkt-type broadcast -j LOG --log-prefix " Broadcast "  $IPT -A INPUT -i ${PUB\_IF} -m pkttype --pkt-type broadcast -j DROP  $IPT -A INPUT -i ${PUB\_IF} -m pkttype --pkt-type multicast -j LOG --log-prefix " Multicast "  $IPT -A INPUT -i ${PUB\_IF} -m pkttype --pkt-type multicast -j DROP  $IPT -A INPUT -i ${PUB\_IF} -m state --state INVALID -j LOG --log-prefix " Invalid "  $IPT -A INPUT -i ${PUB\_IF} -m state --state INVALID -j DROP  *# Log and block spoofed ips*  $IPT -N spooflist  **for** ipblock **in** $SPOOFIP  **do**  $IPT -A spooflist -i ${PUB\_IF} -s $ipblock -j LOG --log-prefix " SPOOF List Block "  $IPT -A spooflist -i ${PUB\_IF} -s $ipblock -j DROP  **done**  $IPT -I INPUT -j spooflist  $IPT -I OUTPUT -j spooflist  $IPT -I FORWARD -j spooflist  *# Allow ssh only from selected public ips*  **for ip in** ${PUB\_SSH\_ONLY} |

|  |
| --- |
| **do**  $IPT -A INPUT -i ${PUB\_IF} -s ${ip} -p tcp -d ${SERVER\_IP} --destination-port 22 -j ACCEPT  $IPT -A OUTPUT -o ${PUB\_IF} -d ${ip} -p tcp -s ${SERVER\_IP} --sport 22 -j ACCEPT  **done**  *# allow incoming ICMP ping pong stuff*  $IPT -A INPUT -i ${PUB\_IF} -p icmp --icmp-type 8 -s 0**/**0 -m state --state NEW,ESTABLISHED,RELATED -m limit --limit 30**/**sec -j ACCEPT  $IPT -A OUTPUT -o ${PUB\_IF} -p icmp --icmp-type 0 -d 0**/**0 -m state --state ESTABLISHED,RELATED -j ACCEPT  *# allow incoming HTTP port 80*  $IPT -A INPUT -i ${PUB\_IF} -p tcp -s 0**/**0 --sport 1024:65535 --dport 80 -m state --state NEW,ESTABLISHED  -j ACCEPT  $IPT -A OUTPUT -o ${PUB\_IF} -p tcp --sport 80 -d 0**/**0 --dport 1024:65535 -m state --state ESTABLISHED -j ACCEPT  *# allow outgoing ntp*  $IPT -A OUTPUT -o ${PUB\_IF} -p udp --dport 123 -m state --state NEW,ESTABLISHED -j ACCEPT  $IPT -A INPUT -i ${PUB\_IF} -p udp --sport 123 -m state --state ESTABLISHED -j ACCEPT  *# allow outgoing smtp*  $IPT -A OUTPUT -o ${PUB\_IF} -p tcp --dport 25 -m state --state NEW,ESTABLISHED -j ACCEPT  $IPT -A INPUT -i ${PUB\_IF} -p tcp --sport 25 -m state --state ESTABLISHED -j ACCEPT  *### add your other rules here #### #######################*  *# drop and log everything else*  $IPT -A INPUT -m limit --limit 5**/**m --limit-burst 7 -j LOG --log-prefix " DEFAULT DROP "  $IPT -A INPUT -j DROP  **exit** 0 |
|  |

**#8: Mengendalikan Serangan Buffer Overflow**

Mengedit nginx.conf dan menset batas ukuran buffer untuk semua klien. # vi /usr/local/nginx/conf/nginx.conf

|  |
| --- |
|  |
| ## Start: Size Limits & Buffer Overflows ## client\_body\_buffer\_size 1K; client\_header\_buffer\_size 1k; client\_max\_body\_size 1k; large\_client\_header\_buffers 2 1k;  ## END: Size Limits & Buffer Overflows ## |
|  |

dimana,

1. **client\_body\_buffer\_size 1k** – (default is 8k or 16k): menentukan ukuran tubuh (body) buffer
2. **client\_header\_buffer\_size 1k** – menentukan ukuran header buffer untuk permintaan dari klien. Untuk sebagian besar permintaan header, ukuran buffer 1K sudah mencukupi. Dapat tingkatkan jika ada tren peningkatan dari cookies (mis., klien WAP).
3. **client\_max\_body\_size 1k**– menentukan ukuran tubuh maksimum yang diterima dari permintaan klien, ditunjukkan oleh baris-Panjang Konten di header permintaan. Jika ukuran lebih besar dari yang diberikan, maka klien mendapatkan notifikasi error " Request Entity Too Large" (413). Dapat ditingkatkan ketika mendapatkan unggahan file melalui metoda POST.
4. **large\_client\_header\_buffers 2 1k** – menentukan jumlah dan ukuran buffer maksimum untuk header besar dari permintaan klien. Secara default ukuran satu buffer sama dengan ukuran halaman, tergantung pada platform ini baik 4K atau 8K, jika pada akhir koneksi permintaan koneksi dikonversikan ke status tetap- hidup, maka buffer ini dibebaskan. 2x1k akan menerima URI data 2kB. Ini juga akan membantu memerangi bot buruk dan serangan DoS.

Kita juga dapat mengendalikan timeout untuk meningkatkan performa server dan memotong klien dengan mengedit sebagai berikut:

|  |
| --- |
|  |
| ## Start: Timeouts ## client\_body\_timeout 10;  client\_header\_timeout 10;  keepalive\_timeout 5 5;  send\_timeout 10; ## End: Timeouts ## |
|  |

1. **client\_body\_timeout 10;** – menentukan batas waktu baca untuk tubuh terhadap permintaan dari klien. Batas waktu ditetapkan hanya jika tubuh tidak masuk dalam satu langkah baca. Jika setelah waktu ini klien mengirim sesuatu, nginx menginformasikan error " Request time out" (408).
2. **client\_header\_timeout 10;** – memberikan batas waktu dengan membaca judul permintaan klien. Batas waktu ditetapkan hanya jika header tidak masuk dalam satu langkah baca. Jika setelah waktu ini klien mengirim sesuatu, nginx menginformasikan error " Request time out" (408).
3. **keepalive\_timeout 5 5;** – Parameter pertama menetapkan batas waktu untuk koneksi tetap-hidup dengan klien. Server akan menutup koneksi setelah waktu ini. Parameter kedua opsional menetapkan nilai waktu di header Keep-Alive: timeout = waktu respons. Header ini dapat meyakinkan beberapa browser untuk menutup koneksi, sehingga server tidak perlu melakukannya. Tanpa parameter ini, nginx tidak mengirim header Keep- Alive (meskipun ini bukan yang membuat koneksi “tetap hidup”). Instruksi menetapkan batas waktu dengan membaca judul permintaan klien. Batas waktu ditetapkan hanya jika header tidak masuk dalam satu langkah baca. Jika setelah waktu ini klien mengirim sesuatu, nginx menginformasikan error " Request time out” (408).
4. **send\_timeout 10;** – memberikan batas waktu respons kepada klien. Timeout dibuat bukan pada seluruh transfer jawaban, tetapi hanya antara dua operasi pembacaan, jika setelah waktu ini klien tidak mengambil sesuatu, maka nginx mematikan koneksi.

**#9: Mengendalikan Koneksi Secara Simultan**

Kita dapat menggunakan modul NginxHttpLimitZone untuk membatasi jumlah koneksi simultan untuk sesi yang ditugaskan dari satu alamat IP. Dengan mengedit nginx.conf sebagai berikut:

|  |
| --- |
|  |
| ### Directive describes the zone, in which the session states are stored i.e. store in slimits. ### ### 1m can handle 32000 sessions with 32 bytes/session, set to 5m x 32000 session ### limit\_zone slimits $binary\_remote\_addr 5m;  ### Control maximum number of simultaneous connections for one session i.e. ### ### restricts the amount of connections from a single ip address ###  limit\_conn slimits 5; |
|  |

Pernyataan diatas membatasi jumlah koneksi klien hanya untuk 5 klien secara bersamaan

**#10: Memperbolehkan akses hanya untuk ke domain saja**

Jika bot hanya melakukan pemindaian server acak untuk semua domain, cukup melakukan penolakan saja. Hanya mengizinkan domain virtual yang dikonfigurasi atau reverse proxy. Tidak menampilkan permintaan menggunakan alamat IP:

## Only requests to our Host are allowed i.e. nixcraft.in, images.nixcraft.in and [www.nixcraft.in](http://www.nixcraft.in/) if ($host !~ ^(nixcraft.in|[www.nixcraft.in](http://www.nixcraft.in/)|images.nixcraft.in)$ ) {

return 444;

} ##

**#11: Membatasi Metoda yang tersedia**

GET dan POST adalah metoda yang paling umum di Internet. Jika server web tidak memerlukan penerapan semua metoda yang tersedia, metoda – metoda tersebut harus dinonaktifkan. Berikut ini cara memfilter dan hanya memperbolehkan metoda GET, HEAD, dan POST:

## Only allow these request methods ##

if ($request\_method !~ ^(GET|HEAD|POST)$ ) { return 444;

}

## Do not accept DELETE, SEARCH and other methods ##

**Penjelasan:**

* Metoda GET digunakan untuk meminta dokumen seperti [https://www.bssn.go.id/index.php.](https://www.bssn.go.id/index.php)
* Metoda HEAD identik dengan GET kecuali jika server tidak harus mengembalikan tubuh pesan dalam responnya.
* Metoda POST dapat melibatkan apa saja, seperti menyimpan atau memperbarui data, atau memesan produk, atau mengirim E-mail dengan mengirimkan formulir. Ini biasanya diproses menggunakan skrip sisi server seperti PHP, PERL, Python dan sebagainya. Kita harus menggunakan ini jika akan mengunggah file dan memproses formulir di server.

**#12: Menangkal User Agent Tertentu**

Memblokir user agent seperti pemindai, bot, dan spammer yang mungkin menyalahgunakan web server kita.

## Block download agents ##

if ($http\_user\_agent ~\* LWP::Simple|BBBike|wget) { return 403;

} ##

Memblokir robot seperti msnbot dan scrapbot (contoh):

## Block some robots ##

if ($http\_user\_agent ~\* msnbot|scrapbot) { return 403;

}

**#13: Memblokir Spam Rujukan**

Spam Rujukan sangat berbahaya. Kita dapat memblokir akses ke spammer rujukan dengan garis-garis ini. ## Deny certain Referers ###

if ( $http\_referer ~\* (babes|forsale|girl|jewelry|love|nudit|organic|poker|porn|sex|teen) )

{

# return 404;

return 403;

} ##

**#13: Menghentikan Image Hotlinking**

Tautan gambar atau HTML biasa terjadi dimana seseorang membuat tautan ke situs kita atau salah satu gambar di situs kita, dengan menampilkannya di situs mereka sendiri. Dampaknya kita akan membayar tagihan bandwidth dan membuat konten terlihat seperti bagian dari situs pembajak. Ini biasanya dilakukan di forum atau di blog. Berikut cara memblokir dan menghentikan hotlinking gambar di web server:

# Stop deep linking or hot linking location /images/ {

valid\_referers none blocked [www.contoh.com](http://www.contoh.com/) contoh.com; if ($invalid\_referer) {

return 403;

}

}

**Contoh lain: Menulis dan menampilkan kembali gambar**

Another example with link to banned image:

valid\_referers blocked [www.contoh.com](http://www.contoh.com/) contoh.com; if ($invalid\_referer) {

rewrite ^/images/uploads.\*\.(gif|jpg|jpeg|png)$ <http://www.examples.com/banned.jpg> last

}

**#14: Membatasi Akses Direktori**

Kita dapat mengatur kontrol akses direktori yang sudah ditentukan. Semua direktori web harus dikonfigurasi berdasarkan kebutuhan/kepentingan tertentu.

**Membatasi Akses berdasarkan IP Address**

Kita dapat membatasi akses terhadap suatu direktori berbasis IP Address untuk direktori /docs/ directory:

location /docs/ {

## block one workstation deny 192.168.1.1;

## allow anyone in 192.168.1.0/24 allow 192.168.1.0/24;

## drop rest of the world deny all;

}

**Penggunaan Password Untuk Memproteksi Direktori**

Membuat file password dan menambah user: dani:

# mkdir /usr/local/nginx/conf/.htpasswd/

# htpasswd -c /usr/local/nginx/conf/.htpasswd/passwd dani

Mengedit file nginx.conf dan proteksi direktori yang dibutuhkan sebagai berikut:

### Password Protect /personal-images/ and /delta/ directories ### location ~ /(personal-images/.\*|delta/.\*) {

auth\_basic "Restricted";

auth\_basic\_user\_file /usr/local/nginx/conf/.htpasswd/passwd;

}

Ketika file password digenerate user dapat ditambahkan didalam direktori usr/local/nginx/conf/.htpasswd/passwd dengan perintah sebagai berikut:

# htpasswd -s /usr/local/nginx/conf/.htpasswd/passwd userName

**#15: Mengkonfigurasi SSL Nginx Konfigurasi SSL**

Langkah dasar dan pertama dalam keamanan web adalah menerapkan SSL agar dalam mengakses aplikasi web sudah menggunakan https dan menambahkan lapisan enkripsi dalam komunikasi.

* Menggunakan [Open SSL](https://geekflare.com/openssl-commands-certificates/) untuk mengenerate CSR (Certificate Signing Request) dengan 2048 bit dan sha- 2 melalui perintah sebagai berikut:

OpenSSL req -nodes -new -sha256 -newkey rsa:2048 -keyout bestflare.key -out bestflare.csr

* Dari perintah diatas akan mengenerate CSR dan file key dan langsung aktif. Setelahnya untuk tidak lupa mengganti nama file .csr dan key tersebut .
* Setelah mendapatkan CSR yang ditKitatangani oleh otoritas sertifikat dan memiliki sertifikat yang ditKitatangani, kita dapat menerapkannya di Nginx seperti di bawah ini.
  + Masuk ke server nginx
  + Buka folder conf di mana Kita memiliki file ssl.conf.

Note: dan default instalasi di Linux akan kita dapatkan di **/etc/nginx/conf.d.**

* Mengedit file dan menambahkan ssl certificate, yang akan mengaktifkan Nginx pada port 443

server {

listen 443 ssl;

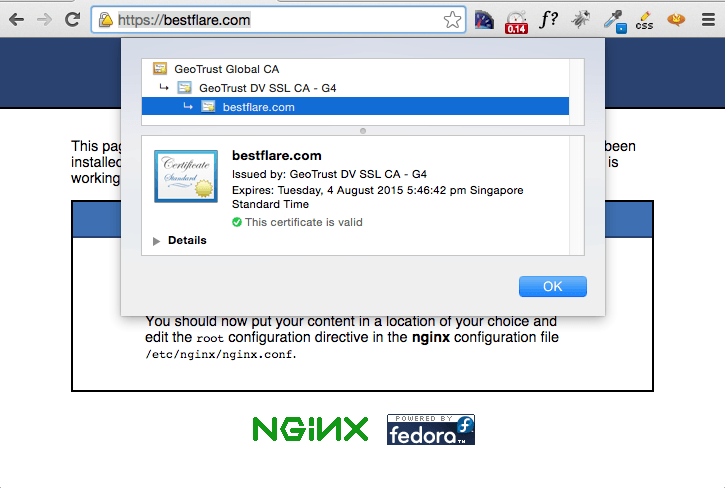
server\_name bestflare.com; ssl on;

ssl\_certificate /opt/cert/bestflare.pem; ssl\_certificate\_key /opt/cert/bestflare.key;

}

Note: Tidak lupa untuk mengubah path sertifikat dan file key.

* Save konfigurasi dan restart Nginx. Maka SSL cert is berhasil diimplementasikan.



**Optimalisasi SSL/TLS**

Setelah mendapatkan SSL bukan berarti web secara penuh aman, tetapi perlu penerapan konfigurasi untuk mengamankan web server.

**Menonaktifkan SSL 3 dan tetap mempertahankan TLS**

Secara umum SSL 3 rentan, dan disarankan menggunakan protokol TLS yang kuat.

* Mengedit file ssl.conf dan tambahkan di bawah ini dalam blok server

ssl\_protocols TLSv1.2;

* Save file ssl.conf dan restart Nginx

**Menonaktifkan weak cipher suites**

* Tambahkan statement berikut pada server block didalam file ssl.conf:

ssl\_ciphers "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+ECDSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH+aRSA+RC4 EECDH EDH+aRSA HIGH !RC4 !aNULL !eNULL !LOW !3DES !MD5 !EXP !PSK !SRP !DSS";

* Save file ssl.conf dan restart Nginx

**Install Chain Certificate**

Kita perlu mendapatkan sertifikat berantai dari lembaga otoritas. Sebagian besar Anda akan menemukan di situs web mereka.

* Text

  Description automatically generatedTambahkan konten sertifikasi dalam sertifikat situs web tersut seperti dibawah ini. Dalam contoh ini adalah /opt/cert/bestflare.pem
* Save file dan restart Nginx

220

**Secure Diffie-Hellman untuk TLS**

Salah satu praktik terbaik yang akhir-akhir ini ditambahkan dalam daftar adalah untuk mengamankan diffie- hellman.

Membuat DH Group unik dan menambahkan ssl\_dhparam di file ssl.conf

* Generate Unique DH Group dengan menggunakan OpenSSL

OpenSSL dhparam -out dhparams.pem 4096

Ini akan memakan waktu beberapa menit dan akan menghasilkan file dhparams.pem pada direktori kerja saat ini

* Salin dhparams.pem ke folder cert
* Ubah ssl.conf dan tambahkan berikut dalam blok server

ssl\_dhparam /opt/cert/dhparams.pem;

* Save file ssl.conf dan restart Nginx Hasil lengkap dari optimasi ssl.conf

# HTTPS server configuration server {

listen 443 ssl;

server\_name bestflare.com; ssl on;

ssl\_certificate /opt/cert/bestflare.pem; ssl\_certificate\_key /opt/cert/bestflare.key; ssl\_protocols TLSv1 TLSv1.1 TLSv1.2; ssl\_prefer\_server\_ciphers on;

ssl\_ciphers "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+ECDSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH+aRSA+RC4 EECDH EDH+aRSA HIGH !RC4 !aNULL !eNULL !LOW !3DES !MD5 !EXP !PSK !SRP !DSS";

221

ssl\_dhparam /opt/cert/dhparams.pem;

}

**Menonaktifkan Metoda Http yang tidak diperlukan**

Mengizinkan modul GET, HEAD & POST Http tetapi tidak untuk TRACE atau DELETE karena berisiko mendapat serangan Pelacakan Lintas Situs (Cross-Site Tracking) dan berpotensi peretas mencuri informasi cookie.

* Salin dhparams.pem ke folder cert
* Ubah ssl.conf dan tambahkan berikut ini dalam blok server

if ($request\_method !~ ^(GET|HEAD|POST)$ )

{

return 405;

}

Save file tersebut and restart Nginx. Disini kita akan menampilkan notifikasi 405 (Not Allowed) jika seseorang mencoba menggunakan TRACE, DELETE, PUT, OPTIONS.

ChKitans-iMac:~ chKitan$ telnet bestflare.com 80 Trying 128.199.100.162...

Connected to bestflare.com.

Escape character is '^]'.

TRACE / HTTP/1.1

Host: testing

HTTP/1.1 405 Not Allowed Server: nginx

Date: Sat, 11 Jul 2015 06:04:34 GMT

Content-Type: text/html Content-Length: 166

Connection: close

**Memproteksi dari serangan Clickjacking**

Kita dapat memasukkan X-FRAME-OPTIONS dalam HTTP Header untuk mencegah serangan clickjacking. Hal ini dapat dilakukan dengan menambahkan statement tersebut dalam file Nginx.conf

add\_header X-Frame-Options "SAMEORIGIN";

Header di atas akan menginstruksikan browser hanya untuk memuat sumber daya dari asal yang sama.

**Proteksi dengan X-XSS**

Memasukkan proteksi X-XSS HTTP Header untuk memitigasi serangan Cross-Site Scripting

* Ubah file default.conf atau ssl.conf dengan menambahkan statement sebagai berikut:

add\_header X-XSS-Protection "1; mode=block";

* Save konfigurasi file diatas dan restart Nginx.

**#16: Pengamanan Nginx dan PHP**

PHP adalah salah satu bahasa scripting sisi server yang populer. Untuk pengamanannya dapat mengedit

/etc/php.ini sebagai berikut:

|  |
| --- |
|  |
| # Disallow dangerous functions disable\_functions **=** phpinfo, system, mail, exec ## Try to limit resources ##  # Maximum execution time of each script, in seconds max\_execution\_time **=** 30  # Maximum amount of time each script may spend parsing request data |

|  |
| --- |
| max\_input\_time **=** 60  # Maximum amount of memory a script may consume (8MB) memory\_limit **=** 8M  # Maximum size of POST data that PHP will accept. post\_max\_size **=** 8M  # Whether to allow HTTP file uploads. file\_uploads **=** Off  # Maximum allowed size for uploaded files. upload\_max\_filesize **=** 2M  # Do not expose PHP error messages to external users display\_errors **=** Off  # Turn on safe mode safe\_mode **=** On  # Only allow access to executables in isolated directory safe\_mode\_exec\_dir **=** php-required-executables-path # Limit external access to PHP environment safe\_mode\_allowed\_env\_vars **=** PHP\_  # Restrict PHP information leakage expose\_php **=** Off  # Log all errors log\_errors **=** On  # Do not register globals for input data register\_globals **=** Off  # Minimize allowable PHP post size post\_max\_size **=** 1K  # Ensure PHP redirects appropriately  cgi.force\_redirect **=** 0 |

|  |
| --- |
| # Disallow uploading unless necessary file\_uploads **=** Off  # Enable SQL safe mode sql.safe\_mode **=** On  # Avoid Opening remote files allow\_url\_fopen **=** Off |
|  |

**#17: Menjalankan Nginx dalam suatu Chroot Jail (Containers)**

Menempatkan nginx di chroot jail akan meminimalkan kerusakan karena potensi pembobolan dengan mengisolasi server web ke bagian kecil dari sistem file. Anda dapat menggunakan jenis pengaturan chroot tradisional dengan nginx. Jika dimungkin menggunakan Chroot Jail FreeBSD, XEN, Debian / Ubuntu, LXD pada Fedora, atau virtualisasi OpenVZ.

**#18: Membatasi Koneksi Input Per IP Address pada level firewall**

Server web harus mengawasi koneksi dan membatasi koneksi per detik. PF Firewall dan Iptables firewall dapat membatasi pengguna akhir sebelum mengakses server nginx.

**Linux Iptables: Membatasi Koneksi Input Nginx Per Detik**

Contoh berikut adalah upaya mendrop koneksi yang masuk jika adanya upaya melakukan koneksi lebih dari 15 kali dan melebihi waktu 60 detik ke port 80:

**/**sbin**/**iptables -A INPUT -p tcp --dport 80 -i eth0 -m state --state NEW -m recent --set

**/**sbin**/**iptables -A INPUT -p tcp --dport 80 -i eth0 -m state --state NEW -m recent --update --seconds 60 -- hitcount 15 -j DROP

service iptables save

**BSD PF Firewall: Membatasi Koneksi Input Nginx Per Detik**

* Mengedit file konfigurasi: /etc/pf.conf dan memperbarunya sebagai berikut.

webserver\_ip="202.54.1.1" table <abusive\_ips> persist

block in quick from <abusive\_ips>

pass in on $ext\_if proto tcp to $webserver\_ip port www flags S/SA keep state (max-src-conn 100, max-src-conn- rate 15/5, overload <abusive\_ips> flush)

* Statement diatas akan membatasi jumlah maksimum koneksi per sumber hingga 100. 15/5 menentukan jumlah koneksi per detik atau rentang detik yaitu tingkat membatasi jumlah koneksi hingga 15 dalam rentang 5 detik. Jika ada yang melanggar aturan maka akan diblokir.

**#19: Mengkonfigurasi Operating System untuk memproteksi Web Server**

File dalam DocumentRoot (/ nginx atau / usr / local / nginx / html) tidak boleh dimiliki atau ditulis oleh pengguna tertentu (contoh: nginx)

# find /nginx -user nginx

# find /usr/local/nginx/html -user nginx

Mengubah kepemilikan file menjadi root atau pengguna lain. Seperangkat izin khusus / usr / local / nginx / html /

# ls -l /usr/local/nginx/html/ Contoh outputs:

-rw-r--r-- 1 root root 925 Jan 3 00:50 error4xx.html

-rw-r--r-- 1 root root 52 Jan 3 10:00 error5xx.html

-rw-r--r-- 1 root root 134 Jan 3 00:52 index.html

Kita harus menghapus file cadangan yang tidak diperlukan yang dibuat oleh vi atau editor teks lainnya: # find /nginx -name '.?\*' -not -name .ht\* -or -name '\*~' -or -name '\*.bak\*' -or -name '\*.old\*'

# find /usr/local/nginx/html/ -name '.?\*' -not -name .ht\* -or -name '\*~' -or -name '\*.bak\*' -or -name '\*.old\*'

**#20: Membatasi Koneksi Keluar Nginx**

Hacker akan mengunduh file secara lokal di server kita dengan menggunakan alat seperti wget. Menggunakan iptables untuk memblokir koneksi keluar dari pengguna Nginx. Modul ipt\_owner berupaya untuk mencocokkan berbagai karakteristik pembuat paket, untuk paket yang dibuat secara lokal. Ini hanya valid dalam rantai OUTPUT. Dalam contoh ini, mengizinkan pengguna yang bernama dani untuk terhubung ke luar menggunakan port 80:

|  |
| --- |
|  |
| **/**sbin**/**iptables -A OUTPUT -o eth0 -m owner --uid-owner dani -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT  **/**sbin**/**iptables -A OUTPUT -o eth0 -m owner --uid-owner dani -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j DENY |
|  |

Dapat juga menambahkan aturan di atas ke skrip shell berbasis iptables untuk tidak mengizinkan pengguna server web nginx terhubung di luar.

**#21: Memelihara Software agar tetap up to date**

Secara regular harus selalu memperbarui perangkat lunak dan kernel Linux kita. Menerapkan patches sesuai versi atau distro Linux yang digunakan. Jika menggunakan Linux Debian / Ubuntu dapat menggunakan perintah apt-get / perintah apt untuk menerapkan patches:

$ sudo apt-get update

$ sudo apt-get upgrade

Jika menggunakan Linux RHEL/CentOS/Oracle/Scientific Linux, dapat menggunakan [yum command:](https://www.cyberciti.biz/faq/rhel-centos-fedora-linux-yum-command-howto/)

$ sudo yum update

Jika menggunakan Linux Linux dapat menggunakan [apk command:](https://www.cyberciti.biz/faq/10-alpine-linux-apk-command-examples/)

# apk update # apk upgrade

**#22: Proteksi dari clickjacking**

Menambahkan statement berikut didalam file nginx.conf atau virtual domain untuk menghindari clickjacking: add\_header X-Frame-Options SAMEORIGIN;

**#23: Menonaktifikan content-type sniffing pada beberapa web browser**

Menambahkan statement berikut didalam file nginx.conf atau virtual domain: add\_header X-Content-Type-Options nosniff;

**#24: Mengaktifikan filter terhadap Cross-site scripting (XSS)**

Menambahkan statement berikut didalam file nginx.conf atau virtual domain: add\_header X-XSS-Protection "1; mode=block";

**#25: Memonitor Nginx**

Memeriksa secara rutin file Log. Log ini akan memberi informasi tentang serangan terhadap server dan memastikan tingkat keamanan yang diperlukan.

# grep "/login.php??" /usr/local/nginx/logs/access\_log # grep "...etc/passwd" /usr/local/nginx/logs/access\_log

# egrep -i "denied|error|warn" /usr/local/nginx/logs/error\_log

**HARDENING WEB SERVER APACHE**

|  |
| --- |
| **HARDENING WEB SERVER APACHE** |

1. **TUJUAN**

Memberikan petunjuk/*guidance* dalam melakukan hardening pada web server Apache

1. **RUANG LINGKUP**

Tahapan hardening pada web server Apache

1. **REFERENSI**
   1. Security Tips - Apache HTTP Server Version 2.4
2. **LANGKAH - LANGKAH**
   1. Sebelum melakukan rekonfigurasi di web server Apache ini, sebaiknya perlu diketahui konfigurasi default dari Apache tersebut, yaitu:
      1. Direktori root untuk web server ada di: /var/www/html or /var/www
      2. File untuk konfigurasi utamanya ada di: /etc/httpd/conf/httpd.conf (RHEL/CentOS/Fedora) and

/etc/apache2/apache2.conf (Debian/Ubuntu).

* + 1. Default HTTP Port: 80 TCP
    2. Default HTTPS Port: 443 TCP
    3. Menguji hasil setting file konfigurasi menggunakan sintaks: httpd -t
    4. Akses ke log file web server ada di: /var/log/httpd/access\_log
    5. Akses ke file error log web server ada di: /var/log/httpd/error\_log
  1. Menyembunyikan identitas versi OS dan Apache ketika terjadi error
     1. Default sebelum dilakukan rekonfigurasi:

Salah satu ancaman keamanan yang besar bagi suatu web server, jika identitas versi OS dan Apache ini ditampilkan



* + 1. Rekonfigurasi untuk menyembunyikan identitas versi OS dan Apache

Buka file konfigurasi dengan editor vim dan cari "ServerSignature", yang secara default “On” menjadi “Off”, dan mematikan tanda tangan server ini pada baris kedua yaitu "ServerTokens Prod".

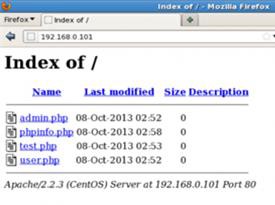
# vim /etc/httpd/conf/httpd.conf (RHEL/CentOS/Fedora) # vim /etc/apache2/apache2.conf (Debian/Ubuntu) ServerSignature Off

ServerTokens Prod

# service httpd restart (RHEL/CentOS/Fedora) # service apache2 restart (Debian/Ubuntu)

* 1. Menonaktifkan Directory Listing

Secara default web server Apache akan memperlihatkan semua daftar konten dari direktori root, seperti gambar berikut ini:



Kita dapat mematikan daftar direktori dengan menggunakan direktif options dalam file konfigurasi terhadap direktori tertentu. Untuk itu kita perlu membuat entri di file httpd.conf atau apache2.conf.

<Directory /var/www/html> Options - Indexes

</Directory>

[Graphical user interface, text, application, email

Description automatically generated](https://www.tecmint.com/wp-content/uploads/2013/10/Hide-Apache-Directory-Listing.png)

* 1. Mengupdate Apache secara rutin

Komunitas pengembang Apache terus berupaya mengatasi masalah keamanan dan merilis versi terbarunya dengan opsi keamanan baru. Sangat disarankan untuk menggunakan versi terbaru dari Apache sebagai server web.

Untuk memeriksa versi Apache: Kita dapat memeriksa versi Apache saat ini dengan perintah httpd -v.

# httpd -v

Server version: Apache/2.2.15 (Unix) Server built: Aug 13 2013 17:29:28

Kita dapat memperbarui versi Apache dengan perintah berikut.

# yum update httpd

# apt-get install apache2

disarankan juga untuk terus memperbarui Kernel OS Linux ke rilis yang lebih stabil dan terbaru.

* 1. Menonaktifkan Modul – modul yang tidak dibutuhkan

Berikut adalah modul – modul default yang ada didalam OS Linux. Untuk menonaktifkan modul tinggal memasukkan tanda # disetiap baris pernyataan modul. Beberapa modul yang umum sering dinonaktifkan yaitu: mod\_imap, mod\_include, mod\_info, mod\_userdir, dan mod\_autoindex.

# grep LoadModule /etc/httpd/conf/httpd.conf

# have to place corresponding `LoadModule' lines at this location so the # LoadModule foo\_module modules/mod\_foo.so

LoadModule auth\_basic\_module modules/mod\_auth\_basic.so LoadModule auth\_digest\_module modules/mod\_auth\_digest.so LoadModule authn\_file\_module modules/mod\_authn\_file.so LoadModule authn\_alias\_module modules/mod\_authn\_alias.so LoadModule authn\_anon\_module modules/mod\_authn\_anon.so LoadModule authn\_dbm\_module modules/mod\_authn\_dbm.so LoadModule authn\_default\_module modules/mod\_authn\_default.so LoadModule authz\_host\_module modules/mod\_authz\_host.so LoadModule authz\_user\_module modules/mod\_authz\_user.so LoadModule authz\_owner\_module modules/mod\_authz\_owner.so LoadModule authz\_groupfile\_module modules/mod\_authz\_groupfile.so LoadModule authz\_dbm\_module modules/mod\_authz\_dbm.so LoadModule authz\_default\_module modules/mod\_authz\_default.so LoadModule ldap\_module modules/mod\_ldap.so

LoadModule authnz\_ldap\_module modules/mod\_authnz\_ldap.so LoadModule include\_module modules/mod\_include.so LoadModule log\_config\_module modules/mod\_log\_config.so LoadModule logio\_module modules/mod\_logio.so

LoadModule env\_module modules/mod\_env.so LoadModule ext\_filter\_module modules/mod\_ext\_filter.so

....

* 1. Menjalankan Apache dengan User & Group Yang terpisah

Dalam instalasi default, Apache menjalankan prosesnya dengan tidak menggunakan user atau daemon. Maka untuk alasan keamanan, sangat disarankan untuk menjalankan Apache di akun yang tidak istimewa. Jadi dibuat akun yang tidak mudah ditebak. Contoh: http-web.

**Membuat User & Group Apache**

# groupadd http-web

# useradd -d /var/www/ -g http-web -s /bin/nologin http-web

Kemudian menginformasikan kepada Apache untuk menjalankan dengan user baru tersebut. Dan untuk melakukannya, kita perlu mengentri di /etc/httpd/conf/httpd.conf dan restart layanan yaitu:

Buka /etc/httpd/conf/httpd.conf dengan editor vim dan cari kata kunci "Pengguna" dan "Grup" dan selanjutnya menentukan nama pengguna dan nama grup yang akan digunakan (contoh: http-web).

User http-web Group http-web

* 1. Penggunaan Allow and Deny untuk membatasi akses direktori

Kita dapat membatasi akses ke direktori dengan opsi "Allow" dan "Deny" dalam file httpd.conf. Dalam contoh ini, kita akan mengamankan direktori root, dengan mengentri opsi "Allow" dan "Deny" berikut ini di file httpd.conf.

<Directory /> Options None Order deny,allow Deny from all

</Directory>

* + 1. Opsi Options “None” – Opsi ini tidak mengizinkan pengguna untuk mengaktifkan fitur opsi apapun.
    2. Order deny, allow – Proses diurutkan mulai "Deny" setelah itu "Allow".
    3. Deny from all – Ini akan menolak permintaan dari semua orang ke direktori root, tidak ada yang akan dapat mengakses direktori root.
  1. Penggunaan Modul mod\_security dan mod\_evasive untuk mengamankan Apache

Dua modul ini “mod\_security” dan “mod\_evasive” adalah modul Apache terkait keamanan yang sangat populer.

**Mod\_Security**

Mod\_security berfungsi sebagai firewall untuk aplikasi web yang memungkinkan untuk memantau lalu lintas secara real time. Selain itu, mampu melindungi situs web atau server web dari serangan brute force. Cukup dengan menginstal mod\_security di server.

Instalasi mod\_security pada LinuxUbuntu/Debian

$ sudo apt-get install libapache2-modsecurity

$ sudo a2enmod mod-security

$ sudo /etc/init.d/apache2 force-reload

Instalasi mod\_security pada Linux RHEL/CentOS/Fedora/ # yum install mod\_security

# /etc/init.d/httpd restart

**Mod\_Evasive**

Mod\_evasive bekerja sangat efisien dalam mencegah serangan DoS, DDoS, dan brute force HTTP. Modul ini mendeteksi serangan dengan tiga metoda, yaitu:

* + 1. Jika begitu banyak permintaan datang ke halaman yang sama dalam beberapa kali per detik.
    2. Jika ada proses anak mencoba untuk membuat lebih dari 50 permintaan bersamaan.
    3. Jika ada IP yang masih mencoba untuk membuat permintaan baru ketika itu sementara daftar hitam. Mod\_evasive dapat diinstal langsung dari sumbernya.
  1. Menonaktifikan Simbolic Link Apache

Secara default Apache mengikuti symlinks dan dapat mematikan fitur ini dengan FollowSymLinks dengan direktif Opsi. Dan untuk melakukannya kita perlu membuat entri berikut di file konfigurasi utama.

Options -FollowSymLinks

Dan, jika ada pengguna atau situs web tertentu yang perlu mengaktifkan FollowSymLinks, kita cukup menulis aturan dalam file “.htaccess” dari situs web itu.

# Enable symbolic links

Options +FollowSymLinks

Catatan: Untuk mengaktifkan aturan penulisan ulang di dalam file ".htaccess" "AllowOverride All" harus ada dalam konfigurasi utama secara global.

* 1. Mematikan Server Side Includes dan CGI Execution

Kita dapat mematikan server side includes (mod\_include) dan CGI execution jika tidaj diperlukan dengan memodifikasi file konfigurasi sebagai berikut:

Options -Includes Options -ExecCGI

Kita dapat juga melakukan ini untuk direktori tertentu dengan tag direktori. Dalam contoh ini, kita

mematikan eksekusi file Includes dan Cgi pada direktori “/ var / www / html / web1”.

<Directory "/var/www/html/web1"> Options -Includes -ExecCGI

</Directory>

Berikut adalah beberapa nilai lain yang dapat “on” atau “off” dengan direktif opsi yaitu:

* + 1. Opsi All – opsi ini berarti mengaktifkan semua opsi sekaligus dan ini adalah nilai default. Jika tidak ingin menentukan nilai apa pun secara eksplisit dalam file conf Apache atau .htaccess.
    2. Opsi IncludeNOEXEC - Opsi ini memungkinkan sisi server termasuk tanpa izin eksekusi ke file perintah atau cgi.
    3. Opsi MultiViews – Opsi ini memungkinkan konten yang dinegosiasikan multiview dengan modul mod\_negotiation.
    4. Opsi SymLinksIfOwnerMatch - Mirip dengan FollowSymLinks. Tapi hanya akan mengikuti ketika pemiliknya sama antara link dan direktori asli yang dilinkan.
  1. Membatasi Ukuran Request

Secara default, Apache tidak memiliki batasan terhadap ukuran total permintaan HTTP tetapi ketika mengizinkan permintaan besar pada server web, ada kemungkinan menjadi korban serangan Denial of Service. Kita dapat membatasi ukuran permintaan arahan Apache "LimitRequestBody" dengan tag direktori.

Anda dapat mengatur nilai dalam byte dari 0 (tidak terbatas) hingga 2147483647 (2GB) yang diizinkan. Kita dapat menetapkan batas ini sesuai dengan kebutuhan situs. Misalkan kita mengizinkan unggahan dan Anda ingin membatasi ukuran unggahan untuk direktori tertentu.

Dalam contoh ini, user\_uploads adalah direktori yang berisi file yang diunggah oleh pengguna, dan dibatasi sampai 500 ribu:

<Directory "/var/www/myweb1/user\_uploads"> LimitRequestBody 512000

</Directory>

* 1. Memproteksi Serangan DDOS dan Hardening

Berikut adalah beberapa opsi direktif yang dapat membantu untuk mengendalikannya.

* + 1. TimeOut: opsi direktif ini memungkinkan untuk mengatur waktu server akan menunggu suatu event tertentu selesai sebelum gagal. Nilai standar/defaultnya adalah 300 detik. Menjaga nilai ini tetap rendah di situs-situs yang terkena serangan DDOS.
    2. MaxClients: opsi direktif ini memungkinkan kita untuk menetapkan batas koneksi yang akan dilayani secara bersamaan. Setiap koneksi baru akan diantrikan setelah batas ini.. Nilai standarnya adalah 256.
    3. KeepAliveTimeout: Jumlah waktu server akan menunggu permintaan berikutnya sebelum menutup koneksi. Nilai default adalah 5 detik.
    4. LimitRequestFields: Ini membantu kita untuk menetapkan batas pada jumlah header permintaan http yang akan diterima dari klien. Nilai default-nya adalah 100. Disarankan untuk menurunkan nilai ini jika serangan DDos terjadi sebagai akibat dari begitu banyak header permintaan http.
    5. LimitRequestFieldSize: Ini membantu kita untuk menetapkan batas ukuran pada header permintaan http.
  1. Mengaktifkan Logging Apache

Apache memungkinkan kita untuk logging secara independen terpisah dengan logging OS. Logging Apache menyediakan lebih banyak informasi, seperti perintah yang dimasukkan oleh pengguna yang telah berinteraksi dengan server web. Untuk melakukannya, perlu memasukkan modul mod\_log\_config.

Ada tiga arahan terkait logging yang tersedia pada Apache.

* + 1. TransferLog: Membuat file log.
    2. LogFormat: Menentukan format khusus.
    3. CustomLog: Membuat dan memformat file log.

Logging Apache ini juga dapat menggunakannya untuk hosting virtual. Sebagai contoh, di sini adalah konfigurasi host virtual situs web dengan logging diaktifkan.

<VirtualHost \*:80>

DocumentRoot /var/www/html/contoh.com/ ServerName [www.contoh.com](http://www.contoh.com/)

DirectoryIndex index.htm index.html index.php ServerAlias contoh.com

ErrorDocument 404 /story.php

ErrorLog /var/log/httpd/contoh.com\_error\_log

CustomLog /var/log/httpd/contoh.com\_access\_log combined

</VirtualHost>

* 1. Pengamanan Apache dengan Sertifikat SSL

Apache menggunakan modul mod\_ssl untuk mendukung sertifikat SSL. # OpenSSL genrsa -des3 -out contoh.com.key 1024

# OpenSSL req -new -key contoh.com.key -out exmaple.csr

# OpenSSL x509 -req -days 365 -in contoh.com.com.csr -signkey contoh.com.com.key -out contoh.com.com.crt Setelah sertifikat Anda dibuat dan ditandatangani. Sekarang Anda perlu menambahkan ini dalam konfigurasi Apache. Buka file konfigurasi utama dengan editor vim dan tambahkan baris berikut dan restart layanan.

<VirtualHost 172.16.25.125:443> SSLEngine on

SSLCertificateFile /etc/pki/tls/certs/contoh.com.crt SSLCertificateKeyFile /etc/pki/tls/certs/contoh.com.key SSLCertificateChainFile /etc/pki/tls/certs/sf\_bundle.crt ServerAdmin [ravi.saive@contoh.com](mailto:ravi.saive@contoh.com)

ServerName contoh.com DocumentRoot /var/www/html/example/

ErrorLog /var/log/httpd/contoh.com-error\_log

CustomLog /var/log/httpd/contoh.com-access\_log common

</VirtualHost>

Dan jika buka browser dan ketik https://contoh.com, maka akan melihat sertifikat yang ditandatangani.

**HARDENING WEB SERVER MICROSOFT IIS 8**

|  |
| --- |
| **HARDENING WEB SERVER MICROSOFT IIS 8** |

1. **TUJUAN**

Memberikan petunjuk/*guidance* dalam melakukan hardening pada web server Microsoft IIS 8

1. **RUANG LINGKUP**

Tahapan hardening pada web server Microsoft IIS 8

1. **REFERENSI**
   1. [Hardening IIS - OWASP](https://www.google.com/url?sa=t&rct=j&q&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwid2L3vt8XmAhXNTN8KHdL3BZYQFjAAegQIAhAB&url=https%3A%2F%2Fwww.owasp.org%2Findex.php%2FHardening_IIS&usg=AOvVaw1Euw2Zz8QIWuHC-FUM9YKz)
2. **LANGKAH - LANGKAH**
   1. **Menonaktifkan Konfigurasi Default Web Server IIS**

Cara terbaik untuk menghindari lubang keamanan potensial ini adalah dengan menempatkan semua konten di folder \ wwwroot baru di luar \ inetpub.

* + 1. Membuka IIS Manager
    2. Pada the Connections pane expand the Sites node dan pilih Default Web Site
    3. Pada Actions pane klik stop pada Manage Aplikasi Berbasis Web
    4. Kemudian klik Application Pool list dan pilih “DefaultAppPool”
    5. Pada Actions pane klik stop pada Application Pool Tasks
    6. Restart IIS.
  1. **Menonakatifkan Akses Direktori Web Content and Script**

Izin berlebihan untuk akun pengguna web yang anonim berkontribusi pada kompromi server web.

* + 1. Browse ke web content pada C:\inetpub\wwwroot\
    2. Copy atau cut content ke folder web khusus dan terbatas pada drive non-sistem seperti D:\webroot\
    3. Mengubah pemetaan aplikasi atau Direktori Virtual apapun untuk mencerminkan lokasi baru
  1. **Host Headers**

Serangan rebinding DNS dapat membahayakan atau menyalahgunakan data atau fungsi situs.

* + 1. Membuka IIS Manager
    2. Pada Connections pane expand the Sites node dan pilih Default Web Site
    3. Pada the Actions pane klik Bindings
    4. Pada Dialog Box Site Bindings, pilih binding yang header hostnya akan dikonfigurasi, dalam contoh ini port 80
    5. Klik Edit
    6. Dibawah host name, masukan FQDN Aplikasi Berbasis Web, sebagai contoh <[www.examplesite.com](http://www.examplesite.com/)>
    7. Klik OK, kemudian Close
  1. **Directory Browsing**

Penyerang dapat mengeksploitasi fitur Directory Browsing untuk mengakses file yang tidak sah melalui traversal direktori.

* + 1. Buka IIS Manager
    2. In the Connections pane expand the Sites node and select Web Site
    3. Click Directory Browsing icon in IIS (Feature View)
    4. In the Actions pane click Disable to disable Directory Browsing
  1. **Application Pool Identity**

Membuat identitas khusus untuk setiap kumpulan aplikasi sehingga akan lebih baik melacak masalah yang terjadi dalam setiap situs web.

* + 1. Buka IIS Manager
    2. Dalam the Connections pane, expand the Server node dan klik Application Pools
    3. Highlight an Application Pool to review and in the Connection pane click Advanced Setting
    4. Scroll down to the Process Model section and set the value for Identity to ApplicationPoolIdentity, Network Service or a custom identity with rights and privileges equal to or less than the built-in-security principal.
    5. Restart IIS.
  1. **Application Pools**

Menetapkan aplikasi intensif sumber daya ke kumpulan aplikasi mereka sendiri meningkatkan kinerja server dan aplikasi.

* + 1. Buka IIS Manager
    2. Buka the Sites node underneath the machine node
    3. Pilih the Site to be changed
    4. Pada Actions pane, pilih Advanced Settings
    5. Klik Select… box next to the Application Pool text box
    6. Pilih Application Pool yang diinginkan
    7. Kemudian pilih, dan klik OK
  1. **Memastikan Application Pools di Bawah Identitas yang Unik**

Mengatur Application Pools dengan identitas yang unik akan mengurangi potensi kerusakan identitas sehingga aplikasi tidak akan mudah dikompromikan.

* + 1. Buka IIS Manager
    2. Buka the Application Pools node underneath the machine node
    3. Create new and then select Application Pool that have been created
    4. Klik kanan Application Pool dan pilih Advanced Settings.
    5. Di bawah bagian Process Model, temukan opsi Identity dan pastikan ApplicationPoolIdentity diatur
  1. **Identitas Pengguna Anonymous**

Mengkonfigurasi identitas pengguna anonim dalam identitas *Application Pools* akan membantu isolasi situs.

* + 1. Buka the IIS Manager GUI dan telusuri server, site, atau application yang diinginkan
    2. Pada Features View, double klik icon Authentication
    3. Pilih opsi Anonymous Authentication dan pada the Actions pane pilih Edit
    4. Pilih Application pool identity in the modal window dan tekan tombol OK
  1. **Webserver TLS Version**

Penyerang dapat mencegat informasi clear text yang sensitif melalui jaringan.

* + 1. Buka IIS Manager
    2. Buka the Sites node underneath the machine node
    3. Double click the SSL icon
    4. Klik the Require SSL and Require SSL 128-Bit check boxes. Aktifkan protokol TLS 1.2 pada R2, pastikan kunci berikut disetel ke 0.

HKLM/System/CurrentControlSet/Control\SecurityProviders\SCHANNEL\Protocols\TLS1.2\Server\Disa bledByDefault

* 1. **Mengaktifkan Pembatasan IP Address dan Domain**

Pemfilteran Alamat IP dan Pembatasan Domain memungkinkan administrator untuk mengonfigurasi server dengan memblokir akses admin dan hanya pendaftaran IP.

* + 1. Buka IIS Manager
    2. Buka the IP Address dan Domain Restrictions feature
    3. Buka Feature atau Double click IP Address dan Setting Domain Restriction
    4. Klik Edit and Feature settings in Action Pane
    5. Piliht Deny in Access untuk klien yang tidak ditentukan sehingga tidak dapat mengakses panel admin
    6. Kemudian klik Add Allow entry for memungkinkan akses untuk alamat IP atau Rentang alamat (Pengguna Internal atau Eksternal)
  1. **Menghapus entri yang tidak diperlukan dari dokumen default**

Jika tidak ada dokumen default di direktori, klien akan menerima kesalahan " file not found" atau " directory browsing denied". Termasuk jika kita menggunakan satu dokumen standar atau menggunakan dokumen pertama dalam daftar, hal ini mempercepat waktu permintaan.

* + 1. Buka IIS Manager
    2. Buka the Sites node underneath the machine node
    3. Pada Features View, find and double-click the Default Document
    4. Klik dokumen yang ingin Anda hapus, dan kemudian klik Remove in Action Pane
    5. Klik Add in Action pane and type the name of Default Document in the box that you want to add
    6. Kemudian klik OK

Note: Tidak memberikan nama umum (seperti: default.aspx, index.html, dll.) ke halaman awal / beranda situs

* 1. **Enkripsi String Koneksi DB**

Menyimpan semua informasi sensitif dalam file teks biasa atau string koneksi DB akan berdampak pada kompromi keamanan.

* + 1. Buka Command Prompt dengan Administrator privileges
    2. Pada Command Prompt, masukkan:
       - Software aspnet\_regiis.exe terletak di% systemroot% \ Microsoft.NET \ Framework \ versionNumberfolder.
       - Jika konfigurasi web Anda berada di jalur direktori "/ SampleApplication /", masukkan yang berikut untuk mengenkripsi ConnectionString. Gunakan Aspnet\_regiis.exe dengan opsi -pef dan tentukan jalur aplikasi seperti yang ditunjukkan di bawah ini.

aspnet\_regiis -pef "connectionStrings" -app "/SampleApplication/Web.config"

* + - * Cukup melakukan perintah berikut untuk mendekripsi elemen connectionStrings di file Web.config.

aspnet\_regiis -pdf "connectionStrings" -app "/SampleApplication/Web.config"

* + - * Catatan: Parameter "ConnectionStrings" case sensitive
  1. **Otentikasi Formulir SSL**

Menerapkan SSL untuk Formulir Otentikasi akan melindungi kerahasiaan kredensial selama proses login dan membantu mengurangi risiko pencurian informasi pengguna.

* + 1. Buka IIS Manager dan telusuri untuk tier yang sesuai
    2. Pada Features View, double-click Authentication
    3. Pada Authentication page, pilihForms Authentication
    4. Pada Actions pane, klik Edit
    5. Cek checkbox dari Requires SSL pada cookie settings section, klik OK
  1. **Form Authentikasi**

Penyerang dapat melakukan hi-jack sesi pengguna untuk mendapatkan akses tidak sah ke aplikasi.

* + 1. Buka IIS Manager dan arahkan ke tingkat di mana Formulir Otentikasi diaktifkan
    2. Pada Features View, double-click Authentication
    3. Pada Authentication page, pilih Forms Authentication
    4. Pada Actions pane, klik Edit
    5. Pada bagian Cookie settings, pilih Use cookies dari Mode dropdown
  1. **Mode Proteksi Cookie**

Penyerang dapat melakukan hi-jack cookie untuk mendapatkan akses tidak sah ke aplikasi.

* + 1. Buka IIS Manager dan Forms Authentication is enabled
    2. Pada Features View, double-click Authentication
    3. Pada Authentication page, pilih Forms Authentication
    4. Pada Actions pane, klik Edit
    5. Pada Cookie settings section, verify the drop-down for Protection mode is set for Encryption and validation
  1. **Memastikan Elemen Kredensial Format password Tidak Clear Text**

Penyerang dapat mencegat kredensial otentikasi yang dikirim dalam bentuk clear text.

* + 1. Temukan dan buka file konfigurasi tempat kredensial disimpan
    2. Temukan elemen <credentials>
    3. Jika ada, pastikan kata sandi tidak diposisi clear text
    4. Ubah Format password ke SHA1 atau MD5
  1. **Mengkonfigurasikan SSL untuk Otentikasi Dasar**

Kredensial yang dikirim dalam bentuk clear text dapat dengan mudah dicegat oleh kode jahat atau penyerang. Menerapkan Secure Sockets Layer akan membantu mengurangi kemungkinan kredensial yang dibajak.

* + 1. Buka IIS Manager
    2. Dalam Connections pane on the left, pilih server yang akan dikonfigurasi
    3. Pada the Connections pane, expand the server, then expand Sites dan pilih site yang kan dikonfigurasi
    4. Pada the Actions pane, click Bindings; the Site Bindings dialog appears
    5. Jika suatu HTTPS binding tersedia, klik Close dan lihat"To require SSL"
    6. Jika tidak ada HTTPS binding, lakukan langkah - langkah berikut:

1. Pada Site Bindings dialog, klik Add; maka akan muncul Add Site Binding dialog
2. Dibawah Type, pilih https
3. Dibawah SSL certificate, pilih an SSL certificate
4. klik OK, dan close
5. Ubah password Format ke SHA1 atau MD5
   1. **Mengaktifkan Logging IIS Lanjut**

Korelasi log dan penetapan waktu untuk setiap aktivitas berbahaya yang terdeteksi tidak dapat dilakukan secara akurat.

* + 1. Buka Internet Information Services (IIS) Manager
    2. Klik the server in the Connections pane
    3. Double-click the Advanced Logging icon pada the Home page
    4. Klik Enable Advanced Logging in the Actions pane

**Catatan:** Berikut field yang harus dilog:

1. date
2. time
3. s-ip
4. cs-method
5. cs-uri-stem
6. cs-uri-query
7. s-port
8. c-ip
9. cs(User-Agent)
10. cs(Referer)
11. sc-status
12. sc-bytes

**HARDENING BERBASIS OWASP**

|  |
| --- |
| **HARDENING BERBASIS OWASP** |

* + - 1. **SQL INJECTON**

Sebagian besar aplikasi web pasti menggunakan pernyataan SQL untuk akses/operasi database (perintah untuk mengoperasikan basis data) berdasarkan input pengguna. Ini berarti jika proses pembuatan pernyataan SQL tidak dijaga dengan aman, akan sangat dimungkinkan adanya serangan dan manipulasi terhadap basis data. Hal ini biasa disebut dengan " kerentanan SQL Injection "dan metode serangan yang mengeksploitasi kerentanan ini disebut dengan" serangan SQL Injection ".

**Gambar 4.15 Visualisasi Proses SQL Injection**

Timeline

Description automatically generated with medium confidence

POTENSI ANCAMAN

Ancaman ini memungkinkan penyerang untuk:

* + - * + Melihat bahkan mengeksploitasi data sensitif yang disimpan dalam database (mis. pengungkapan informasi pribadi)
        + Memalsukan dan / atau menghapus data yang disimpan dalam database (mis. pemalsuan halaman web, perubahan kata sandi, penutupan sistem)
        + Membypass otentikasi login

Semua operasi yang diizinkan di bawah akun hak istimewa ketika masuk aplikasi bisa jadi menjadi tidak sah

* + - * + Menjalankan perintah OS menggunakan prosedur tersimpan (mis: pembajakan sistem, menjadikan PC target sebagai bot (titik peluncuran) untuk menyerang sistem lain)

SOLUSI

* + - * + Membuat semua pernyataan SQL menggunakan placeholder.

Biasanya, SQL memiliki mekanisme untuk membangun pernyataan SQL menggunakan placeholder. Ini adalah mekanisme untuk meletakkan simbol (placeholder) di tempat variabel dalam template pernyataan SQL dan mengganti dengan nilai data aktual secara mekanis. Dibandingkan dengan metode di mana aplikasi web secara langsung membuat pernyataan SQL melalui gabungan (concatenation), metode yang menggunakan placeholder dapat menghilangkan kerentanan injeksi SQL.

Proses mengganti placeholder dengan nilai data aktual disebut mengikat. Ada dua metode mengikat: satu adalah metode dimana pernyataan SQL dikompilasi menjaga placeholder di dalamnya dan mesin database menggantinya dengan nilai data aktual terkait (placeholder statis) dan metode lainnya pustaka koneksi database melakukan pelepasan diri dan menggantikan placeholder dengan nilai data aktual terkait (placeholder dinamis)

* + - * + Saat membuat pernyataan SQL melalui gabungan, menggunakan API khusus yang ditawarkan oleh mesin basis data untuk melakukan pelepasan diri dan membuat literal dalam pernyataan SQL dengan benar.

Saat memasukkan nilai sebagai tipe string, kita akan mengurung nilai dalam satu tanda kutip. Dalam hal ini, kita harus melakukan pelepasan diri untuk string literal untuk membersihkan karakter khusus (mis. 'ke' 'dan \ ke

\\). Saat memasukkan nilai sebagai tipe numerik, membuatnya diproses sebagai numerik literal (misalnya memasukkannya ke dalam tipe numerik).

* + - * + Tidak menulis pernyataan SQL secara langsung pada parameter yang akan diteruskan ke aplikasi web.

Menentukan pernyataan SQL dalam parameter aplikasi web secara langsung dapat menyebabkan risiko seseorang memalsukan nilai parameter dan memanipulasi basis data.

* + - * + Memberikan hak minimum untuk akun basis data.

Jika hak istimewa akun database yang digunakan aplikasi web dalam mengakses database lebih tinggi dari yang diperlukan, maka kerusakan serangan yang ditimbulkan bisa menjadi lebih serius.

Periksa perintah dalam berinteraksi dengan database dan memberikan hak akses minimum kepada akun tersebut, cukup untuk menjalankan perintah tersebut

* + - * + Membatasi informasi yang ditampilkan dalam pesan kesalahan di web browser.

Jika pesan kesalahan berisi informasi tentang nama mesin database atau pernyataan SQL yang dimiliki menyebabkan kesalahan, maka pengguna jahat bisa mendapatkan informasi yang berguna untuk menyerang situs web.

Kesalahan pesan dapat dimanfaatkan tidak hanya untuk memberikan tips untuk menyerang tetapi juga untuk menunjukkan hasil serangan. Disarankan untuk tidak menampilkan pesan kesalahan yang terkait dengan operasi database di browser web pengguna.

CONTOH

**PHP & PostgreSQL**

## Kondisi Rentan

$ query = "SELECT \* FROM usr WHERE uid = ' **$ uid** ' AND pass = ' **$ passh** ';

$ result = pg\_query ($ conn, $ query);

Di atas adalah bagian dari kode sumber yang mengimplementasikan otentikasi pengguna. $ uid di baris pertama adalah ID pengguna yang disediakan oleh pengguna. $ passh adalah nilai hash web aplikasi menghitung berdasarkan kata sandi yang dimasukkan pengguna. Di baris pertama, aplikasi web menggunakan variabel- variabel ini untuk menyusun pernyataan SQL dan menetapkannya ke $ query. Fungsi pg\_query () di baris kedua adalah fungsi PostgreSQL yang disediakan oleh PHP dan mengeksekusi $ query, sebagai pernyataan SQL yang diatur di baris pertama. Contoh program ini tidak memiliki proses untuk melarikan diri terkait nilai $ uid, yang memungkinkan penyerang melakukan serangan injeksi SQL dengan memasukkan nilai yang dibuat khusus dengan pernyataan SQL yang berbahaya.

Seperti dalam kasus ini, jika aplikasi web tidak melakukan pelolosan untuk nilai-nilai yang dilewatkan oleh eksternal parameter, dapat menyebabkan eksekusi pernyataan SQL yang tidak terduga.

Sebagai contoh, anggaplah pengguna memasukkan "talas '-" sebagai ID pengguna, pernyataan SQL dikirim ke database akan menjadi sebagai berikut:

SELECT FROM \* FROM WHERE uid = ' **talas' --** 'AND pass =' eefd5bc2 ... '

Kutipan tunggal (') yang digunakan dalam pernyataan SQL di atas adalah karakter khusus, yang mendefinisikan string literal dengan melampirkan string data dalam sepasang tanda kutip tunggal. Demikian juga, dua tanda hubung berturut-turut () adalah karakter khusus yang memberitahu database untuk mengabaikan semua yang muncul setelahnya sebagai komentar.

Yang berarti basis data akan mengabaikan ['AND pass = eefd5bc2 ..] ketika nilai [talas' -] ditetapkan dalam $ uid. Akibatnya, pernyataan SQL yang dikirim dan dieksekusi oleh database akan menjadi seperti ini

SELECT \* FROM usr WHERE uid = '' **talas' --**

Artinya, jika akun pengguna "talas" memang ada dalam database, penyerang bisa masuk tanpa mengetahui kata sandi yang berhubungan dengan talas. Selain itu, tidak hanya melewati otentikasi pengguna tetapi juga dapat memanipulasi basis data secarar yang tidak diinginkan dengan hanya mengubah string untuk memberi input ke $ uid. Hal ini disebabkan karena tidak ada proses melarikan diri untuk nilai elemen yang menyusun pernyataan SQL.

Fungsi pg\_query () mampu mengeksekusi beberapa query SQL. Jika fungsi ini rentan untuk injeksi SQL, penyerang bisa memasukkan lebih banyak permintaan selain yang asli, yang akan meningkatkan ancaman.

Di bawah ini adalah contoh yang menggambarkan masalah ini:

// Set two SQL queries in $query

$query = "SELECT item FROM shop WHERE id = 1; SELECT item FROM shop WHERE id = 2;"

$result = pg\_query($conn, $query);

## Tindakan Korektif #1

Menggunakan fungsi pg\_prepare () atau fungsi pg\_execute () daripada menggunakan fungsi pg\_query ().

$result = ($conn, "query", 'SELECT \* FROM usr WHERE uid= $1 AND pass=$2);

$result = ($conn, "query", array($uid, $passh));

Fungsi pg\_prepare () dan fungsi pg\_execute () adalah fungsi PostgreSQL yang disediakan di PHP 5.1.0 dan yang lebih baru dan hanya didukung oleh PostgreSQL 7.4 dan yang lebih baru.

Fungsi pg\_prepare () menghasilkan pernyataan yang disiapkan. Argumen ketiga adalah pernyataan SQL dimana variabel dirujuk menggunakan placeholder (variabel terikat) $ 1, $ 2 ... tanpa nilai aktual.

Fungsi pg\_execute () mengeksekusi pernyataan yang disiapkan oleh fungsi pg\_prepare () dibuat. Ketika placeholder digunakan dalam pernyataan yang disiapkan, fungsi pg\_execute () dikonversi setiap elemen dari argumen ketiga ($ uid dan $ passh dalam kasus ini) menjadi string dan mengaturnya dalam placeholder yang sesuai (disebut "mengikat") dan mengeksekusi pernyataan SQL yang lengkap. Penggunaan placeholder akan menyelamatkan kita dalam melakukan pelepasan diri secara eksplisit.

## Tindakan Korektif #2

Menggunakan fungsi pg\_query\_params() daripada menggunakan fungsi pg\_query ().

$result = ($conn, 'SELECT \* FROM usr WHERE uid = $1 AND pass = $2', array($uid, $passh));

Fungsi pg\_query\_params () adalah fungsi untuk PostgreSQL yang disediakan oleh PHP 5.1.0 atau yang lebih baru dan hanya didukung oleh PostgreSQL versi 7.4 atau yang lebih baru. Fungsi pg\_query\_params () tidak membuat pernyataan etapi dilengkapi dengan kemampuan placeholder. Dibutuhkan pernyataan SQL di mana placeholder ($ 1, $ 2, ...) digunakan sebagai argumen kedua dan nilai aktual untuk placeholder sebagai argumen ketiga. Penggunaan placeholder akan menyelamatkan kita dalam melakukan pelepasan diri secara eksplisit.

## Tindakan Korektif #3

Menggunakan fungsi pg\_escape\_string() dan melakukan pelepasan diri (*escaping*) untuk semua elemen dalam pernyataan SQL yang dieksekusi melalui fungsi pg\_query ().

$query = "SELECT \* FROM usr WHERE uid = '".($uid)."' AND pass = '".($passh)."'";

$result = pg\_query($conn, $query);

Fungsi pg\_escape\_string () adalah fungsi PostgreSQL yang tersedia dalam PHP 4.2.0 atau yang lebih baru dan hanya didukung oleh PostgreSQL 7.2 atau yang lebih baru. Hal ini akan meloloskan diri dari karakter khusus yang ditunjuk di PostgreSQL.

Penggunaan pg\_escape\_string () akan melakukan pelepasan diri secara otomatis. Dalam kode di atas, $ passh melewati proses pelolosan. $ passh adalah nilai hash yang dihitung dari kata sandi dan tidak mungkin dieksploitasi dalam upaya injeksi SQL. Namun demikian, direkomendasikan untuk melakukan pelepasan diri untuk elemen- elemen yang diproses secara internal ini termasuk $ passh.

**PHP & MySQL**

## Kondisi Rentan

$query = "SELECT \* FROM usr WHERE uid = '' AND pass = ''";

$result = mysql\_query($query);

Ini adalah bagian dari kode sumber yang mengimplementasikan otentikasi pengguna. Program ini juga tidak memiliki proses melepaskan diri untuk nilai input $ uid, yang memungkinkan penyerang untuk memulai serangan injeksi SQL dengan memasukkan nilai yang dibuat khusus yang akan berubah menjadi pernyataan SQL yang berbahaya.

## Tindakan Korektif #1

Menggunakan fungsi mysqli, seperti mysqli\_prepare (), mysqli\_stmt\_bind\_param () dan mysqli\_stmt\_execute () daripada menggunakan mysql\_query.

Fungsi mysqli\_prepare (), mysqli\_stmt\_bind\_param () dan mysqli\_stmt\_execute () adalah fungsi MySQL yang disediakan dalam ekstensi PHP mysqli dan hanya didukung oleh MySQL 4.1.3 atau yang lebih baru.

Fungsi mysqli\_prepare () menghasilkan pernyataan yang disiapkan. Argumen kedua adalah pernyataan siap di mana pernyataan SQL diekspresikan menggunakan placeholder “?” tanpa nilai aktual.

Fungsi mysqli\_stmt\_bind\_param () mengikat nilai data aktual (nilai bind) ke placeholder dalam pernyataan yang disiapkan yang dibuat oleh fungsi mysqli\_prepare ().

Ketiga dan argumen selanjutnya ($ uid dan $ passh dalam kasus ini) adalah nilai ikat. Argumen kedua "ss" menunjukkan jenis nilai bind (untuk string).

Karena kedua elemen, $ uid dan $ passh, adalah tipe "string", dengan mengatur dua ss Fungsi mysqli\_stmt\_execute () mengeksekusi pernyataan yang telah disiapkan

// Create a prepared statement

$stmt = ($conn, "SELECT \* FROM usr WHERE uid= ? AND pass = ?");

// Bind $uid and $passh to the SQL statement (corresponding placeholders) ($stmt, "ss", $uid, $passh);

// Execute the SQL statement ($stmt);

## Tindakan Korektif #2

Menggunakan fungsi mysql\_real\_escape\_string () untuk melakukan pelepasan diri ke semua elemen yang membentuk pernyataan SQL yang akan dieksekusi oleh fungsi mysql\_query ().

Fungsi mysql\_real\_escape\_string () adalah fungsi MySQL yang disediakan oleh PHP 4.3.0 atau yang lebih baru.

$query = "SELECT \* FROM usr WHERE uid = '". ($uid)."' AND pass = '".

($passh)."'";

$result = mysql\_query($query);

$query = "SELECT \* FROM usr WHERE uid = '$uid' AND pass = '$passh'";

$sth =$dbh->prepare($query);

$sth->execute();

**PERL**

## Kondisi Rentan

$query = "SELECT \* FROM usr WHERE uid = '$uid' AND pass = '$passh'";

$sth =$dbh->prepare($query);

$sth->execute();

Ini adalah bagian dari kode sumber yang mengimplementasikan otentikasi pengguna. Contoh ini menggunakan database dengan standar modul antarmuka yang biasa disebut dengan DBI.

Program sampel ini tidak memiliki proses melepaskan diri untuk nilai input untuk $ uid dan memungkinkan penyerang untuk melakukan serangan injeksi SQL dengan memasukkan nilai yang dibuat khusus yang dapat diubah menjadi pernyataan SQL berbahaya, yang biasa digunakan pada Perl.

Sampel ini menunjukkan kesalahan pengkodean yang umum namun berbahaya saat menggunakan Perl DBI.

Metode persiapan () dalam modul DBI menghasilkan pernyataan yang disiapkan dan tidak mendukung placeholder. Demikian juga, metode execute () mengeksekusi pernyataan yang disiapkan yang dibuat oleh menyiapkan () metode dan juga mampu mengikat jika pernyataan yang disiapkan berisi placeholder.

Hal yang rentan dalam contoh program ini adalah bahwa program ini tidak menggunakan placeholder atau melakukan pelepasan diri meskipun pernyataan SQL yang dikomposisikan berisi variabel yang dapat dieksploitasi, yang membuat aplikasi ini rentan terhadap serangan injeksi SQL

## Tindakan Korektif #1

Saat membuat pernyataan SQL dalam metode prep () pada modul DBI, harus menggunakan placeholder "?" di tempat variabel. Kemudian, menentukan nilai mengikat yang akan ditetapkan ke placeholder dalam metode mengeksekusi ()

$sth =$dbh->prepare("SELECT \* FROM usr WHERE uid = AND pass = ");

$sth->execute($uid, $passh);

## Tindakan Korektif #2

Menggunakan metode quote () dalam modul DBI dan melakukan pelepasan diri untuk variabel.

Metode quote () akan mengambil string yang ditentukan dalam argumennya, keluar dari karakter khusus dalam string dan mengembalikan output setelah melampirkannya dengan tanda kutip ganda.

Apa yang dikenali sebagai karakter khusus berbeda dari mesin database ke mesin database dan itu adalah masalah yang harus ditangani ketika melakukan melarikan diri. DBI menyediakan satu set driver, yang disebut DBD (Database Drivers) untuk beradaptasi dengan berbagai mesin basis data. Metode quote () di DBI memungkinkan DBD menangani perbedaan mesin basis data dan menawarkan transparansi pengguna untuk masalah ini.

$sth = $dbh->prepare("SELECT \* FROM usr WHERE uid =".$dbh->($uid)." AND

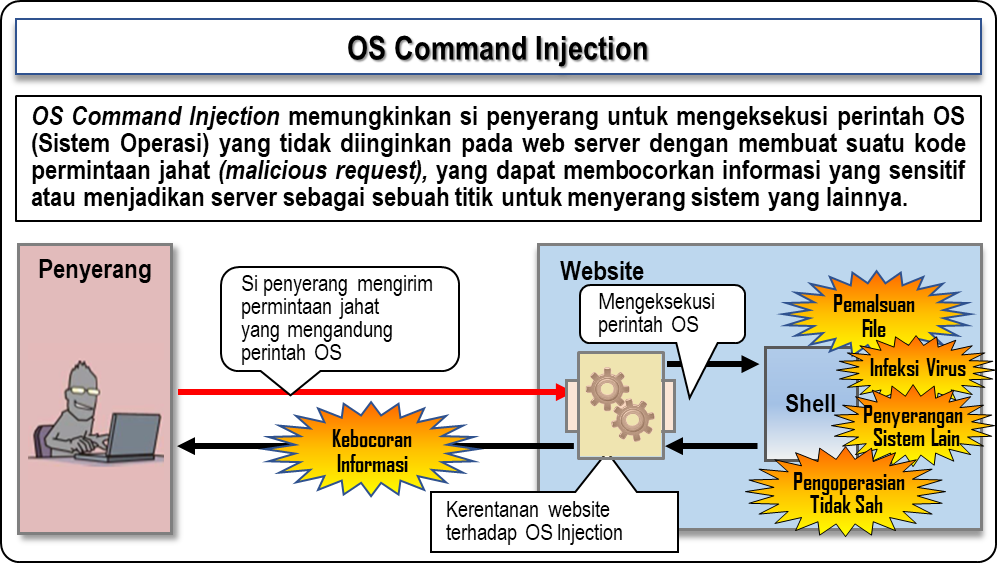
pass =".$dbh->($passh));

$sth->execute();

* + - 1. **OS COMMAND INJECTON**

Penyerang memungkinkan secara remote menjalankan perintah level OS melalui aplikasi tersebut. Masalah ini disebut "kerentanan Perintah Injeksi OS" dan metode menyerang yang mengeksploitasi kerentanan ini disebut "serangan Perintah OS". Injeksi perintah OS memungkinkan penyerang untuk mengeksekusi perintah OS yang tidak diinginkan di server web dengan permintaan yang dibuat dengan niat jahat, sehingga dapat menyebabkan bocornya informasi sensitif atau mengubah server menjadi bot (titik peluncuran) untuk menyerang orang lain.

**Gambar 4.16 Visualisasi Proses OS Command Injection**



POTENSI ANCAMAN

Ancaman ini memungkinkan penyerang untuk:

* + - * + Melihat, memalsukan, dan menghapus file yang disimpan di server (mis: pengungkapan informasi sensitif, pemalsuan file konfigurasi)
        + Memanipulasi sistem dengan tujuan jahat (mis. Mematikan OS yang tidak diinginkan, menambah / menghapus akun pengguna)
        + Mengunduh dan menjalankan program jahat (mis. virus, infeksi worm dan bot, implementasi backdoor)
        + Menjadikan sistem sebagai titik peluncuran untuk menyerang sistem lain (serangan Denial of Service, pengintaian dan spamming)

SOLUSI

* + - * + Beberapa bahasa pemrograman yang digunakan untuk menulis aplikasi web memiliki fungsi yang dapat memanggil perintah - perintah shell, seperti fungsi open () dan fungsi – fungsi lain (Perl: open (), system (), eval () PHP: exec (), passthru (), shell\_exec (), system (), popen () ). Fungsi open () misalnya akan mengambil nama file sebagai argumen dan memasukkan tanda "| (pipa) ” maka akan memanggil dan menjalankan perintah OS. Itu artinya berbahaya karena memungkinkan input eksternal digunakan sebagai argumennya.
        + Hindari menggunakan fungsi yang bisa memanggil perintah shell.
        + Saat menggunakan fungsi yang dapat memanggil perintah shell, periksa semua variabel yang membentuk parameter shell dan memastikan yang mengeksekusi hanya mereka yang diberikan untuk dieksekusi.

CONTOH

**Program Perl yang menjalankan perintah sendmail**

## Kondisi Rentan

$from =~ s/"|;|'|<|>|\|| //ig;

open(MAIL, "|/usr/sbin/sendmail -t -i -f $from");

Di atas adalah bagian dari program yang mengirim email dengan alamat email yang dimasukkan oleh formulir web pengguna sebagai pengirim.

Alamat email input disimpan dalam variabel $ from. Baris pertama menghilangkan shell khusus karakter ",;, ', <,>,

| dan spasi dari isi $. Baris kedua memanggil OS perintah sendmail untuk memulai proses pengiriman surat dan meneruskan konten $ dari ke baris perintah pilihan.

Meskipun ada sanitasi pada baris pertama, implementasi ini masih rentan terhadap perintah injeksi OS. Dalam implementasi ini, jika nilai $ from adalah [someone@example.jp,](mailto:someone@example.jp) perintah berikut adalah dieksekusi:

/usr/sbin/sendmail -t -i -f [someone@example.jp](mailto:someone@example.jp)

Namun, jika nilai $ from dibuat dengan maksud jahat dan `touch [0x09] / tmp / foo` (di mana [0x09] berarti tabulasi horisontal) dimasukkan, perintah berikut akan dieksekusi dan perintah injeksi OS bisa berhasil dilakukan.

/usr/sbin/sendmail -t -i -f `touch[0x09]/tmp/foo`

Kutipan belakang (`) adalah karakter meta shell yang mengeksekusi perintah jual yang diletakkan di antara bagian belakang mengutip dan mengembalikan output perintah ke baris perintah. Dalam program sampel, kutipan ganda dan kutipan tunggal disanitasi tetapi kutipan belakang tidak tersentuh. Kelalaian ini mengakibatkan memungkinkan penyerang untuk mengeksekusi perintah yang tidak diinginkan.

Selain itu, menghapus ruang di baris pertama dari program sampel dapat memberikan arti yang salah yang akan memberikan jaminan bahwa penyerang tidak dapat secara bebas menentukan opsi baris perintah bahkan jika dia bisa mengeksekusi perintah. Menggunakan tabulasi horizontal [0x09] seperti di atas memungkinkan penyerang untuk menentukan opsi baris perintah. Di sini, tabulasi horizontal berfungsi sebagai pemisah karakter.

## Tindakan Korektif #1

Menggunakan Library

Dengan menghentikan menjalankan perintah OS. Fungsionalitas yang saat ini diaktifkan dengan menjalankan perintah OS dapat dilakukan melalui penggunaan library yang ada.

use Mail::Sendmail;

%mail = (From => $from, …); sendmail(%mail);

Program sampel ini adalah program untuk mengirim email, dengan menggunakan pengirim surat MAIL ：: Sendmail.

## Tindakan Korektif #2

**Tidak menempatkan nilai parameter di baris perintah**

Jika pustaka yang dapat diganti tidak tersedia dan Anda tidak bisa berhenti menggunakan perintah, masih ada kemungkinan Anda dapat menghapus kerentanan perintah injeksi OS dengan mengubah cara melakukan perintah.

Dalam contoh program ini, alamat email pengirim menggunakan opsi baris perintah yang menyebabkan kerentanan.

Dengan cara ini, nilai $ from tidak digunakan dalam baris perintah dan karenanya akan menghilangkan kerentanan karena perintah injeksi OS.

$from =~ s/\r|\n//ig;

open(MAIL, '|/usr/sbin/sendmail -t -i');

…

print MAIL "From: $from\n";

## Tindakan Korektif #3

Jika pustaka yang dapat diganti tidak tersedia dan tidak bisa berhenti menggunakan perintah, masih ada kemungkinan dapat menghapus kerentanan perintah injeksi OS dengan menjalankan perintah tanpa mengakses shell

open(MAIL, '|-') || exec '/usr/sbin/sendmail', '-t', '-i', '-f', '$from';

* + - 1. **UNCHECKED PATH PARAMETER/DIRECTORY TRAVERSAL**

Beberapa aplikasi web memungkinkan untuk menentukan nama file yang disimpan di server web secara langsung menggunakan parameter eksternal. Jika aplikasi web semacam itu tidak diprogram dengan hati-hati, penyerang dapat menentukan file yang yang tidak diinginkan dan meminta aplikasi web menjalankan operasi yang tidak diinginkan. Masalah ini disebut " Kerentanan Direktori Traversal"dan metode penyerangan yang mengeksploitasi kerentanan ini disebut" Serangan Direktori Traversal".

POTENSI ANCAMAN

Ancaman ini memungkinkan penyerang untuk:

* + - * + Melihat, memalsukan, dan menghapus file yang disimpan di server berupa:

Pengungkapan informasi sensitif

Pemalsuan dan penghapusan file konfigurasi, file data, dan kode sumber

**Gambar 4.17 Visualisasi Proses Unauthorized Access to File**



SOLUSI

* + - * + Tidak menggunakan nama file yang disimpan di server web secara langsung menggunakan parameter eksternal

Ketika aplikasi web dengan nama filenya ditentukan secara langsung menggunakan parameter eksternal, penyerang dapat memanipulasi parameter yang menentukan file secara yang tidak diinginkan dan melihat konten file yang seharusnya tidak diungkapkan. Misalnya, dalam kasus implementasi di mana nama file disimpan di web server ditentukan dalam parameter tersembunyi dan file itu digunakan dalam templat halaman web, seorang penyerang bisa output file yang tidak diinginkan sebagai halaman web dengan memanipulasi parameter.

Dianjurkan untuk meninjau desain dan spesifikasi aplikasi, mempertimbangkan kembali apakah itu memang diperlukan untuk memungkinkan menentukan nama file yang disimpan di server web dalam parameter eksternal dan metode alternatif tersedia.

* + - * + Menggunakan direktori tetap untuk menangani nama file dan membatalkan nama direktori yang didalamnya nama file.

Misalkan kita membuka file bernama "nama file" di direktori saat ini dan jika file itu terbuka fungsi diimplementasikan seperti open (nama file), penyerang dapat mengakses file yang tidak diinginkan dengan menentukan jalur absolut ke file. Untuk mencegah penggunaan jalur absolut, Kita bisa menggunakan direktori tetap, seperti “Dirname”, dan kode seperti open (dirname + nama file).

Namun, hanya melakukan itu masih menyisakan kamar untuk serangan direktori traversal menggunakan "../". Untuk mencegahnya, kita dapat menggunakan API, seperti basename (), yang mengekstrak hanya nama file dan menghapus nama direktori dari jalur yang diberikan seperti berikut: open (dirname + basename (filename)

* + - * + Mengelola izin akses file dengan benar.

Jika izin akses ke file di server web diterapkan dan dikelola dengan benar, server web mungkin dapat mencegah upaya serangan ketika aplikasi web mencoba untuk membuka file di direktori tersebut secara yang tidak diinginkan.

* + - * + Memeriksa nama file.

Ketika nama file berisi string karakter yang digunakan untuk menentukan direktori arbitrer, seperti “/”, ”../” dan “.. \”, batalkan prosesnya. Perhatikan bahwa jika Anda menggunakan pengodean dan dekode URL, Nilai-nilai yang disandikan URL seperti “% 2F”, “..% 2F” dan “..% 5C” atau nilai ganda yang disandikan seperti “% 252F”, “..% 252F” dan “..% 255C” dapat diartikan sebagai nilai input yang valid untuk nama file.

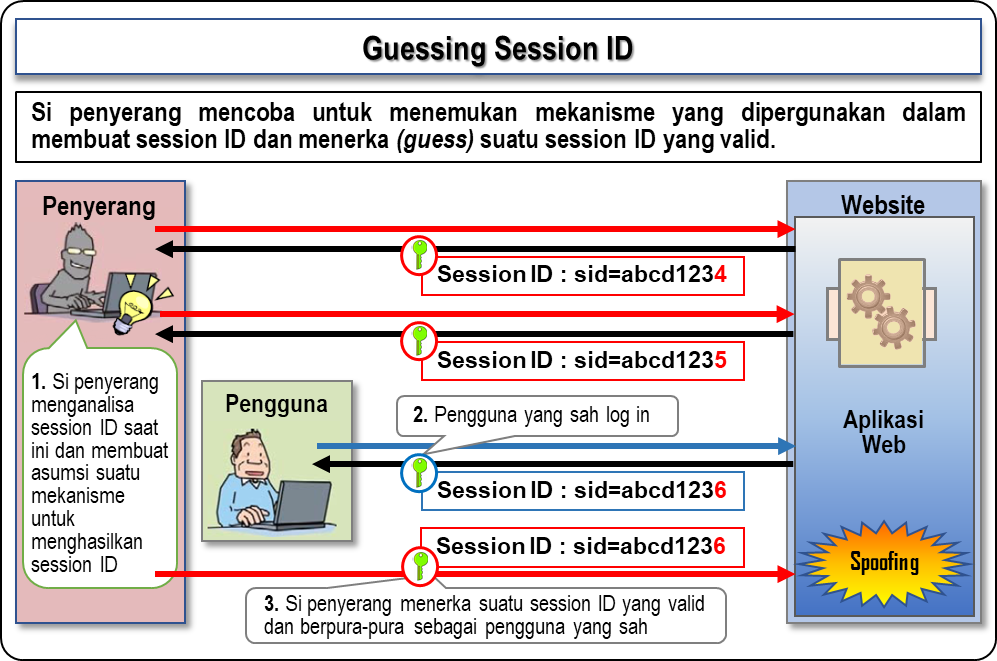
* + - 1. **MANAJEMEN SESI YANG TIDAK BENAR**

Dalam pengelolaan sesi beberapa aplikasi web mengeluarkan ID sesi, yang merupakan informasi untuk mengidentifikasi pengguna.

Jika ID sesi tidak dikelola dengan benar, penyerang dapat mencuri ID sesi dari pengguna yang sah dan mendapatkan akses tidak sah ke layanan seolah - olah menjadi pengguna yang sah.

Metode penyerangan dengan mengeksploitasi kerentanan ini dalam manajemen sesi disebut "Session Hijacking".

**Gambar 4.18 Visualisasi Proses Guessing Session ID**



Selain menebak atau mencuri ID sesi, ada serangan lain yang mengeksploitasi sesi yang tidak patut yaitu "Sesi Fixation". Itu terjadi ketika penyerang menyiapkan ID sesi dan memiliki target pengguna menggunakan ID sesi dalam beberapa cara an pengguna target yang tidak menyadarinya login ke situs web. Jika berhasil, penyerang bisa seolah - olah menjadi pengguna yang ditargetkan menggunakan ID sesi yang telah ditetapkan oleh penyerang

**Gambar 4.19 Visualisasi Proses Stealing Session ID**

Timeline

Description automatically generated

**Gambar 4.20 Visualisasi Proses Session Fixation**

Text

Description automatically generated

POTENSI ANCAMAN

Jika serangan yang mengeksploitasi manajemen sesi dimana penyerang seolah - olah menjadi pengguna yang sah dan melakukan operasi yang diizinkan untuk pengguna tersebut. Penyerang membuat pengguna yang sah menggunakan ID sesi yang telah ditentukan oleh penyerang

Hal ini dapat memungkinkan adanya:

* Akses layanan yang biasanya hanya tersedia untuk pengguna yang sah (misal: pengiriman uang tanpa izin, membeli barang yang tidak diinginkan, membatalkan keanggotaan bertentangan dengan keinginan pengguna)
* Menambah dan mengubah informasi yang biasanya diizinkan hanya untuk pengguna yang memilikinya masuk dengan benar (misal. perubahan pengaturan aplikasi yang tidak sah (kata sandi, fungsi administrator, dll.),
* Melihat informasi yang biasanya tersedia hanya untuk pengguna yang sah (misal: akses tanpa izin ke informasi pribadi, webmail, papan buletin khusus anggota)

SOLUSI

* Membuat ID sesi yang sulit ditebak

Menggunakan mekanisme yang aman, seperti generator nomor acak yang akan mengeluarkan ID rahasia dan terenkripsi.

* Tidak menggunakan parameter URL untuk menyimpan ID sesi.

Jika ID sesi diatur dalam parameter URL, browser pengguna akan meneruskan URL ID yang didalamnya tertanam sesi ke situs web berikutnya. Jika penyerang mencegatnya, ia akan memungkinkan untuk membajak sesi. Simpan ID sesi dalam cookie atau parameter tersembunyi menggunakan metode POST untuk meneruskannya.

Beberapa server aplikasi web dapat secara otomatis beralih untuk menggunakan parameter URL ketika browser pengguna diatur untuk menolak cookie. Jika demikian, ubah pengaturan server dan coba nonaktifkan fitur ini.

* Mengatur atribut aman cookie ketika menggunakan HTTPS.

Cookie memiliki atribut aman yang memungkinkan cookie dikirim hanya melalui saluran HTTPS. Pastikan untuk mengatur atribut aman. Tetapi jika menggunakan cookie dalam komunikasi HTTP buat cookie baru, terpisah dari yang digunakan dalam komunikasi HTTPS.

* Memulai sesi baru setelah berhasil melakukan login.

Beberapa aplikasi web memulai sesi mengeluarkan ID sesi sebelum pengguna masuk, mungkin ketika pengguna pertama kali mengakses situs web, dan terus menggunakan sesi yang sama. Metode ini, bagaimanapun, rentan terhadap fiksasi sesi. Anda harus menghindarinya dan lebih baik memulai sesi baru setelah pengguna berhasil login in (kelola sesi dengan ID sesi baru). Menonaktifkan ID sesi lama saat menggantinya dengan ID sesi baru.

* Menetapkan tanggal kadaluwarsa cookie saat menyimpan ID sesi didalam cookie.
  + - 1. **CROSS SITE SCRIPTING**

Beberapa aplikasi web menghasilkan halaman web berdasarkan input pengguna atau informasi header HTTP, seperti hasil pencarian, halaman konfirmasi pendaftaran pengguna, papan buletin dan laporan statistik web., penyerang dapat memasukkan konten jahat. Masalah ini disebut "Kerentanan Cross-Site Scripting " dan metode serangan yang mengeksploitasi kerentanan ini disebut "serangan Cross-Site Scripting". Tidak hanya membahayakan situs web itu sendiri tetapi juga akan mempengaruhi keamanan para pengunjung situs web.

**Gambar 4.21 Visualisasi Proses Cross Site Scripting**

Timeline

Description automatically generated

POTENSI ANCAMAN

Ancaman ini memungkinkan penyerang untuk:

* + - * + Menampilkan halaman web palsu di situs web yang sah (misal. kebingungan yang disebabkan oleh informasi yang salah, pengungkapan informasi sensitif melalui serangan phishing)
        + Mencuri cookie yang disimpan oleh browser web
* Pencurian ID sesi yang disimpan dalam cookie akan menyebabkan spoofing
* Pengungkapan informasi pribadi dan data sensitif yang disimpan dalam cookie curian.

SOLUSI

**Tindakan untuk Aplikasi Web yang Tidak Mengizinkan Input Teks HTML**

* Melakukan Escaping untuk semua yang akan dikeluarkan ke web halaman.

Untuk mencegah skrip lintas situs, melakukan pelepasan untuk semua elemen halaman web, seperti konten dan nilai atribut HTML. Salah satu cara untuk mengimplementasikan melepaskan diri adalah dengan mengganti karakter khusus digunakan untuk mengontrol tata letak halaman web, seperti "<", ">" dan "&", dengan masing- masing entitas HTML "& lt;", “& gt;” dan “& amp;”. Jika aplikasi web perlu membuat tag HTML, pastikan untuk melampirkan semua nilai atribut dalam tanda kutip ganda, kemudian lakukan pelarian dengan mengganti tanda kutip ganda yang terkandung dalam nilai atribut dengan entitas HTML "& quot;".

Dalam hal pencegahan kerentanan, data yang harus melalui proses pelolosan adalah input string karakter diteruskan ke aplikasi web oleh entitas eksternal, nilai-nilai yang diambil dari database atau file dan yang dihasilkan dari operasi aritmatika pada string karakter. Namun, kita bisa membuatnya tentu tidak akan melewatkan apapun dengan mengambil pendekatan yang lebih konsisten di mana semua elemen teks halaman web berada harus melalui proses pelolosan terlepas dari apakah perlu. Proses keluaran yang perlu menyertakan proses pelarian tidak terbatas pada itu untuk respons HTTP. Saat mengubah konten halaman web secara dinamis, misalnya menggunakan metode document.write dalam JavaScript atau properti innerHTML, proses yang sama diperlukan.

* Saat mengeluarkan URL dalam HTML, hanya diizinkan yang memulai dengan pola tertentu, seperti "http: //" dan "https: //".

URL dapat dimulai dengan tidak hanya "http: //" atau "https: //" tetapi juga dengan "javascript:". Jika sebuah URL sumber daya atau gambar yang akan dimasukkan ke dalam halaman HTML secara dinamis dibuat berdasarkan input eksternal, penyerang dapat meluncurkan serangan skrip lintas situs dengan menanamkan skrip ke dalam URL.

Misalnya, jika aplikasi web membuat halaman output HTML dengan mengatur URL yang ditentukan oleh pengguna seperti <a herf= *pengguna input* URL> , penyerang dapat menyisipkan skrip dengan string yang dimulai dengan "http: //" atau "https:: //" untuk nilai URL. Ambil pendekatan daftar putih di mana hanya string yang dimulai dengan http: // atau https: // yang diizinkan untuk nilai URL.

**Tindakan untuk Aplikasi Web yang Mengizinkan Input Teks HTML**

* Tidak membuat konten tag <script> ... </script> secara dinamis

Jika nilai untuk tag <script> ... </script> dibuat secara dinamis berdasarkan input eksternal, skrip yang tidak diinginkan dapat dimasukkan di sana. Kita dapat memeriksa dan membatalkan skrip yang berisiko.

Disarankan untuk tidak membiarkan aplikasi menetapkan nilai tag <script> ... </script> secara dinamis karena akan sulit untuk menentukan skrip mana yang memang berbahaya.

* Tidak mengizinkan impor stylesheet dari situs web yang tidak diinginkan

Skrip dapat ditulis ke dalam stylesheet menggunakan fungsi seperti ekspresi (). Itu artinya skrip berbahaya dapat dimasukkan ke halaman web jika desain situs web memungkinkan untuk mengimpor stylesheet dari situs web yang tidak diinginkan.

Kita dapat memeriksa stylesheet yang diimpor dan membatalkan skrip berbahaya atau dengan tidak membiarkan aplikasi menggunakan stylesheet eksternal karena akan sulit untuk menghapusnya

* Memeriksa nilai input.

Menjadikan aplikasi web memiliki fungsi untuk memeriksa nilai input dan meminta pengguna untuk memasukkan kembali ketika mereka tidak mengikuti aturan tertentu.

* Membuat pohon parse dari input teks HTML dan ekstrak saja elemen yang diperlukan yang tidak mengandung skrip.

Parsing input teks HTML dan ekstrak hanya elemen yang diizinkan dalam daftar putih yang telah ditentukan. Langkah ini akan membutuhkan pemrograman yang kompleks dan beban pemrosesan akan tinggi.

**Tindakan untuk Aplikasi Web Secara Umum**

* Menetapkan parameter charset dari header jenis konten HTTP.

Kita dapat mengatur kode karakter di Tipe-Konten dari header HTTP seperti: “Tipe-Konten: teks / html; charset = UTF-8 ”. Ketika parameter charset tidak ada dari bagian header Jenis-Konten, browser mengasumsikan kode karakter berdasarkan aturannya sendiri dan memproses string dengan karakter asumsi yang ditetapkan untuk menampilkannya di browser web. Sebagai contoh, beberapa browser diketahui menggunakan kode karakter tertentu ketika bagian pertama dari teks HTML berisi string karakter tertentu.

Jika charset tidak ditentukan, penyerang dapat mengeksploitasi perilaku browser ini, dengan meminta browser untuk memilih set karakter tertentu dengan sengaja dengan memasukkan string karakter tertentu dan menanamkan karakter string yang akan muncul sebagai skrip ketika mereka diproses dengan set karakter itu.

Misalnya, jika karakter string "+ ADw-script + AD4-alert (+ ACI-test + ACI -) + AdsAPA- / script + AD4-" dimasukkan ke dalam teks HTML, beberapa browser akan mengenalinya sebagai string yang dikodekan oleh UTF-7. Jika string ini diproses menggunakan UTF-7, akan menjadi "<script> alert ('test') </script>" dan skrip ini akan dieksekusi.

* Menghapus string skrip dalam input teks HTML.

Identifikasi string skrip yang termasuk dalam input teks HTML dan membatalkan string tersebut dengan menggantinya dengan string yang tidak berbahaya. Misalnya kita mengganti "<script>" atau “Javascript:” dengan menambahkan karakter ke string tersebut seperti “<xscript>” atau “xjavascript:”. Alternatif lain dapat menghapus string skrip secara keseluruhan tetapi mungkin akan menghadirkan risiko baru dengan menghapus mereka yang pada gilirannya membuat string berbahaya.

Tindakan pencegahan terhadap kerentanan cross-site scripting dengan benar, karakter yang ditunjukkan di atas, seperti "+ ADw-", tidak akan luput sejak karakter tersebut diproses oleh aplikasi web yang diatur dengan kode karakter lain, seperti UTF-8, EUC-JP atau SHIFT\_JIS, yang tidak dikenali sebagai sesuatu yang harus diloloskan.

Untuk mencegah masalah ini, dapat melakukan pelolosan lain untuk teks HTML dengan asumsi itu dikodekan oleh UTF-7, tetapi dengan asumsi hanya UFT-7 tidak cukup. Jadi, untuk mengatasi masalah ini, sangat efektif untuk menentukan parameter charset tanpa menghilangkan itu. Setel kode karakter yang akan digunakan aplikasi web dengan penanganan string karakter saat mengeluarkan halaman HTML dalam tipe-konten header HTTP yang menyertainya.

* Atur atribut HttpOnly dari cookie dan nonaktifkan metode TRACE untuk mencegah pengungkapan informasi cookie.

"HttpOnly" adalah atribut yang dapat kita atur pada cookie dan akan menolak skrip dalam teks HTML akses ke cookie. Ini akan mencegah cookie dari dicuri bahkan jika situs web memiliki kerentanan cross – site scripting

Mengatur atribut HttpOnly di header HTTP Set-Cookie saat membuat cookie seperti: “Set-Cookie: [snip];

**HttpOnly**

Ada beberapa hal yang harus diketahui saat mengadopsi tindakan pencegahan ini

Pertama, Kita perlu menonaktifkan metode TRACE di server web. Saat metode TRACE diaktifkan, jika situs web memiliki kerentanan skrip lintas situs, penyerang dapat memperoleh seluruh permintaan header HTTP yang dikirimkan browser menggunakan metode serangan yang disebut "Cross-Site Tracing". Permintaan HTTP header berisi informasi cookie

Kedua, atribut HttpOnly tidak didukung oleh semua browser, jadi itu bukan solusi yang bisa bermanfaat dan melindungi semua pengunjung situs web. dengan demikian cookie akan 'dicuri' bahkan jika atribut HttpOnly adalah set.

* + - 1. **CROSS SITE REQUEST FORGERY**

Beberapa situs web mengharuskan pengguna untuk masuk untuk menawarkan layanan mereka. Di sini, jika situs web tidak memiliki mekanisme untuk memverifikasi apakah permintaan yang dibuat oleh pengguna yang masuk adalah memang permintaan yang dimaksudkan oleh pengguna. Situs web dapat menerima permintaan jahat yang dibuat oleh pihak eksternal lainnya. Jika situs web memiliki ini kerentanan, penggunanya dapat menderita karena melakukan hal-hal yang tidak diinginkan di situs web melalui perangkap yang ditetapkan oleh penyerang. Hal ini disebut "kerentanan Pemalsuan Permintaan Lintas Situs" dan serangannya metode yang mengeksploitasi kerentanan ini disebut "serangan Pemalsuan Permintaan Situs-Lintas".

**Gambar 4.22 Visualisasi Proses Site Request Forgery**

Timeline

Description automatically generated

POTENSI ANCAMAN

Ancaman ini dapat memungkinkan penyerang untuk:

* + - * + Mengkkses layanan yang biasa tersedia untuk pengguna yang memiliki akses masuk dengan bena (misal. mentransfer uang, membeli barang, atau membatalkan keanggotaan yang tidak diinginkan oleh pengguna)
        + Menambahkan dan mengubah informasi yang biasanya diizinkan hanya untuk pengguna yang memilikinya (misal. pengaturan aplikasi (kata sandi, fungsi administrator, dll.), menulis entri yang tidak benar)

SOLUSI

* + - * + Akses halaman web, di mana operasi tertentu akan dieksekusi, melalui metode POST rahasia yang memiliki halaman web sebelumnya masukkan ke dalam file tersembunyi, dan menjalankan operasi yang diminta hanya ketika rahasianya benar.
        + Meminta kata sandi tepat sebelum menjalankan operasi yang diminta dan dilanjutkan ketika kata sandi itu benar.
        + Memeriksa pengarah (referrer) URL yang diharapkan dan dilanjutkan hanya untuk URL yang benar.
        + Beri tahu alamat email yang ditentukan sebelumnya secara otomatis ketika operasi penting telah dilakukan.
        + Email dikirim ketika pasca-insiden dan karenanya tidak dapat mencegah serangan CSRF, tetapi bisa memunculkan warna merah yang ditandai bahwa ada sesuatu yang salah ketika serangan itu benar-benar terjadi. Hati-hati untuk tidak memasukkan informasi sensitif terkait privasi di badan email.
      1. **HTTP HEADER INJECTION**

Beberapa aplikasi web secara dinamis menetapkan nilai bagian header respons HTTP berdasarkan nilai yang disahkan oleh parameter eksternal. Misalnya, pengalihan HTTP diterapkan dengan mengatur pengalihan ke URL yang ditentukan dalam parameter ke bagian header lokasi, atau aplikasi web dapat menetapkan nama dimasukkan dalam papan buletin ke header Set-Cookie yang diajukan.

Jika proses membangun respons header HTTP dalam aplikasi web tersebut memiliki kerentanan, penyerang dapat menambahkan bagian header, memanipulasi badan respons dan minta aplikasi web menghasilkan beberapa respons. Masalah ini disebut “Kerentanan Injeksi Header HTTP ”dan metode serangan yang mengeksploitasi kerentanan ini disebut “Serangan Injeksi Header HTTP “. Secara khusus, serangan yang mengarahkan aplikasi web untuk menghasilkan beberapa respons biasa disebut "HTTP Response Splitting Attack".

**Gambar 4.23 Visualisasi Proses HTTP Header Injection**

Timeline

Description automatically generated

POTENSI ANCAMAN

Kerentanan ini dapat memungkinkan penyerang untuk:

* + - * + Menghadirkan ancaman yang sama dengan yang ditimbulkan oleh kerentanan site cross scripting

Jika arbitrary response body dimasukkan, akn mengakibatkan browser pengguna menampilkan informasi yang salah atau dipaksa untuk mengeksekusi skrip yang tidak diinginkan tersebut.

* + - * + Membuat cookie yang tidak diinginkan

Ketika header HTTP Set-Cookie dimasukkan, cookie yang tidak diinginkan dibuat dan disimpan di browser pengguna.

* + - * + Poison web cache

Pemisahan respons HTTP memaksa server web untuk menghasilkan beberapa respons HTTP dan bisa menyebabkan poison cache, yang mengakibatkan pemalsuan halaman web, dan memiliki server proxy cache sebagai tanggapan HTTP yang tidak dinginkan dan mengganti halaman web cache yang asli. Pengguna yang mengunjungi situs web korban akhirnya lebih pada melihat halaman web palsu yang diganti. Ancaman poison web cache akan memengaruhi lebih banyak pengguna dan cenderung bertahan lama.

**Gambar 4.24 Visualisasi Proses HTTP Response Splitting and Cache Poisoning**

Timeline

Description automatically generated

SOLUSI

* + - * + Tidak mencetak header HTTP secara langsung dan melakukannya melalui HTTP API header yang disediakan oleh lingkungan eksekusi atau bahasa pemrograman.

Di beberapa lingkungan eksekusi, aplikasi web dapat langsung mencetak HTTP header respons yang menentukan bagian Content-Type. Dalam kasus ini, jika aplikasi mencetak nilai input dilewatkan oleh parameter eksternal secara langsung, karakter umpan baris dapat diatur sepanjang. Karakter umpan baris digunakan untuk memisahkan header HTTP sehingga memungkinkan penyisipan umpan baris yang dapat menjadi penyebab injeksi / respons header / badan yang tidak diinginkan Karena struktur header cukup kompleks seperti yang dilihat pada opsi non-breaking space dan sulit untuk menangani semuanya secara manual, maka sebaiknya menggunakan API header HTTP yang ditawarkan oleh bahasa pemrograman atau lingkungan eksekusi

* + - * + Jika HTTP header API yang menawarkan netralisasi umpan baris tidak tersedia maka dapat diimplementasikan secara manual.

Dapat menambahkan ruang tanpa putus setelah umpan baris yang tidak terduga, dengan menghapus string itu mengikuti umpan garis yang tidak terduga

* + - * + Menghapus semua karakter umpan baris yang muncul di input teks eksternal

Menghapus semua karakter umpan baris yang muncul dalam teks input yang dilewatkan oleh parameter eksternal. Bahkan mungkin ingin menghapus semua karakter kontrol seolah – olah hanya karakter umpan baris. Perhatikan bahwa jika web aplikasi perlu menerima string karakter yang mungkin berisi umpan baris, seperti input data di <textarea> ... </textarea> tag, secara sistematis menghapus setiap umpan baris dari semua input data yang dapat menghalangi operasi aplikasi web.

* + - 1. **MAIL HEADER INJECTION**

Beberapa aplikasi web menyediakan fungsi yang mengirim email ke alamat email tertentu tentang, untuk misalnya, barang dagangan yang dibeli pengguna atau balasan survei. Secara umum, alamat email ini adalah ditentukan sebelumnya dan hanya administrator web yang dapat mengubah. Penyerang mungkin dapat mengatur dan mengubahnya ke alamat email yang tidak diinginkan. Kerentanan ini disebut “Kerentanan Mail Header Injection ”dan metode serangan yang mengeksploitasi kerentanan ini disebut“ Serangan Mail Header Injection".

**Gambar 4.25 Visualisasi Proses Mail Header Injection**

Timeline

Description automatically generated

POTENSI ANCAMAN

Kerentanan ini dapat memungkinkan penyerang untuk:

* + - * + Relai email pihak ketiga (digunakan sebagai landasan untuk distribusi spam)

SOLUSI

* + - * + Menggunakan nilai tetap untuk elemen header dan menghasilkan semua input eksternal ke badan email.

Dalam kasus dimana nilai elemen header email, seperti "To", "Cc", "Bcc" dan "Subject", diatur berdasarkan input eksternal, atau proses output data ke fungsi pengiriman email yang rentan, jika input eksternal langsung digunakan sebagai nilai output, karakter umpan baris termasuk dalam eksternal input akan dimasukkan sebagai jeda baris yang tidak perlu. Jika ini dibolehkan, penyerang akan mengeksploitasi memasukkan header email yang tidak diinginkan, mengubah badan email atau mengirim email ke alamat tidak diinginkan. Disarankan tidak menggunakan parameter eksternal untuk mengatur nilai elemen header email.

* + - * + Jika nilai tetap tidak dapat digunakan untuk header, dapat menggunakan API pengiriman email yang ditawarkan oleh lingkungan eksekusi aplikasi web atau bahasa pemrograman.

Contoh di mana Anda tidak bisa menggunakan nilai tetap untuk elemen header adalah case yang Anda inginkan untuk mengubah topik pembicaraan.

Jika perlu menggunakan input eksternal sebagai nilai header, disarankan untuk menggunakan API pengiriman email ditawarkan oleh lingkungan eksekusi aplikasi web atau bahasa pemrograman yang digunakan. Namun, beberapa API tidak dapat menangani karakter umpan baris dengan tepat atau dapat menyisipkan beberapa header. Dalam kasus tersebut, dapat menerapkan tambalan keamanan atau mengimplementasikan modifikasi yang diperlukan untuk tidak membiarkan garis pemisah sendiri. Misalnya, untuk mencegah jeda baris, dapat memasukkan spasi atau tab horizontal setelah umpan garis karakter agar program memproses garis sebagai satu baris berkelanjutan. Menghapus karakter setelah baris karakter atau menghentikan pembuatan halaman web jika jeda baris terdeteksi.

* + - * + Tidak mencantumkan alamat email dalam HTML.

Tidak mencantukan alamat email penerima secara langsung di parameter tersembunyi. Implementasi seperti ini akan diteruskan aplikasi web sehingga dapat dieksploitasi oleh serangan relai email pihak ketiga dengan mengubah parameter nilai.