Robots

Table des matières

1 Outils **1**
- 1.1 Installation . . . 1
- 1.2 Documentation . . . 1

2 Espionner **1**
- 2.1 Sniffer . . . 1
- 2.2 Via Proxy (port 8080) . . . 1

3 Enregister via proxy **3**

4 Rejouer **4**
- 4.1 Tenter sa chance . . . 4
- 4.2 Pas à pas . . . 5
- 4.3 Script . . . 5

1 Outils

1.1 Installation

```
# apt-get install ngrep

# apt-get install libwww-mechanize-shell-perl
# apt-get install libwww-mechanize-perl
# apt-get install libhttp-proxy-perl
# apt-get install libhttp-recorder-perl
```

1.2 Documentation

```
$ man WWW::Mechanize::Shell
$ dpkg -L libwww-mechanize-perl | grep man.*gz
```

2 Espionner

2.1 Sniffer

Afin d'avoir un premier aperçu des échanges, on pourra avoir recours à cette version orienté HTTP de TcpDump.

```
# ngrep -d lo
```

2.2 Via Proxy (port 8080)

Il n'empêche que le plus simple reste de se placer en intermédiaire. Le proxy suivant est un script glané via le Linux Magazine 75.

```
#!/usr/bin/perl -w
use strict;
use HTTP::Proxy;
use HTTP::Proxy::HeaderFilter::simple;
use HTTP::Proxy::BodyFilter::simple;
use CGI::Util qw( unescape );

# récupération des paramètres de configuration
# spécifiques au script et pas à HTTP::Proxy
my %args = (
    peek   => ["planet-gis"],
    header => [],
);
{
    my $args = '(' . join( '|', keys %args ) . ')';
    for ( my $i = 0 ; $i < @ARGV ; $i += 2 ) {
        if ( $ARGV[$i] =~ /$args/o ) {
            push @{ $args{$1} }, $ARGV[ $i + 1 ];
            splice( @ARGV, $i, 2 );
            redo if $i < @ARGV;
        }
    }
}

# les en-têtes à afficher
# envoyées par le serveur
my @srv_hdr = (
    qw( Content-Type Set-Cookie Set-Cookie2 WWW-Authenticate Location ),
    @{ $args{header} }
);

# envoyées par le client
my @clt_hdr =
  ( qw( Cookie Cookie2 Referer Referrer Authorization ), @{ $args{header} } );

# NOTE: dans le cas d'une méthode POST, les filtres de requête
#       recoivent toujours le corps de la requête en une seule fois
my $post_filter = HTTP::Proxy::BodyFilter::simple->new(
    sub {
        my ( $self, $dataref, $message, $protocol, $buffer ) = @_;
        print STDOUT "\n", $message->method, " ", $message->uri, "\n";
        print_headers( $message, @clt_hdr );

        # ceci est copié de CGI.pm, méthode parse_params
        my (@pairs) = split( /[&;]/, $$dataref );
        for (@pairs) {
            my ( $param, $value ) = split( '=', $_, 2 );
            $param = unescape($param);
            $value = unescape($value);
            printf STDOUT "    %-20s => %s\n", $param, $value;
        }
    }
);
```

```
my $get_filter = HTTP::Proxy::HeaderFilter::simple->new(
    sub {
        my ( $self, $headers, $message ) = @_;
        my $req = $message->request;
        if ( $req->method ne 'POST' ) {
            print STDOUT "\n", $req->method, " ", $req->uri, "\n";
            print_headers( $req, @clt_hdr );
        }
        print STDOUT $message->status_line, "\n";
        print_headers( $message, @srv_hdr );
    }
);

sub print_headers {
    my $message = shift;
    for my $h (@_) {
        if ( $message->header($h) ) {
            print STDOUT "    $h: $_\n" for ( $message->header($h) );
        }
    }
}

# création du proxy avec les paramètres restants
my $proxy = HTTP::Proxy->new(@ARGV);

# pour espionner CERTAINS sites
if ( @{ $args{peek} } ) {
    for ( @{ $args{peek} } ) {
        $proxy->push_filter(
            host    => $_,
            method  => 'POST',
            request => $post_filter
        );
        $proxy->push_filter( host => $_, response => $get_filter );
    }
}

# sinon, on espionne TOUS les sites
else {
    $proxy->push_filter(
        method  => 'POST',
        request => $post_filter
    );
    $proxy->push_filter( response => $get_filter );
}
$proxy->start;
```

3 Enregister via proxy

Quite à être au milieu de l'échange, autant en profiter pour enregistrer les requêtes.

- Il faut préalablement corriger un bug dans le fichier */usr/share/perl5/HTTP/Recorder.pm*

```
sub unmodify {
```

```
    my $self = shift;
    my $content = shift;

    return $content unless $content;

    # get rid of the arguments we added
    my $prefix = $self->{prefix};

    #HIDE THIS
    #for my $key ($content->query_param) {
    #if ($key =~ /^$prefix-/) {
    #    $content->query_param_delete($key);
    #}
    #}

    #ADD THIS
    $content =~ s/$prefix-(.*?)\?(.*?)&//g;
    $content =~ s/$prefix-(.*?)&//g;
    $content =~ s/$prefix-(.*?)$//g;
    $content =~ s/&$//g;
    $content =~ s/\?$//g;

    return $content;
}
```

- Cf : `man HTTP::Recorder` pour le code du proxy ci-dessous.

```
#!/usr/bin/perl

use HTTP::Proxy;
use HTTP::Recorder;

my $proxy = HTTP::Proxy->new();

# create a new HTTP::Recorder object
my $agent = new HTTP::Recorder;

# set the log file (optional)
$agent->file("replay.pl");

# set HTTP::Recorder as the agent for the proxy
$proxy->agent( $agent );

# start the proxy
$proxy->start();

1;
```

Ce scripts enregistre les requettes dans le script *replay.pl*

4 Rejouer

4.1 Tenter sa chance

On doit compléter le script *replay.pl* de la manière suivante :

```
#!/usr/bin/perl -w
use strict;
use WWW::Mechanize;
my $agent = WWW::Mechanize->new();

# facultatif
$agent->proxy(['http'], 'http://127.0.0.1:8080/'); # le premier proxy espion

# copier/coler de replay.pl
...

#wget
print $agent->content, "\n";
```

4.2 Pas à pas

Nous nous inspirons du script *replay.pl*.

```
$ perl -MWWW::Mechanize::Shell -eshell
(no url)>get http://planet-gis/xoops
http://planet-gis/xoops/>forms
http://planet-gis/xoops/>form 1
http://planet-gis/xoops/>fillout
http://planet-gis/xoops/>get http://planet-gis/xoops/modules/piCal/index.php?smode=List&op=all
http://planet-gis/xoops/modules/piCal/index.php?smode=List&op=all>form 3
http://planet-gis/xoops/modules/piCal/index.php?smode=List&op=all>fillout
http://planet-gis/xoops/modules/piCal/index.php?smode=List&op=all>get http://planet-gis/xoops/module

....
>script
```

4.3 Script

Nous modifions le script *replay.pl* en conséquence.

```
#!/usr/bin/perl -w
use strict;
use WWW::Mechanize;
#-------------------
my $login="admin";
my $passwd="admin";
#-------------------
my $agent = WWW::Mechanize->new();

# Get cookie and connect
$agent->get('http://planet-gis.ias.u-psud.fr/xoops/index.php');
$agent->form_number(1);
$agent->field('uname', "$login");
$agent->field('pass', "$passwd");
$agent->click();

# Get event list
$agent->get('http://planet-gis/xoops/modules/piCal/index.php?smode=List&op=all');
my $page=$agent->content;
$agent->form_name('MainForm');
```

```
$agent->tick('dummy', 'on');

# Mark all events into checkbox
my $dummy="";
my $number="";
do {
  ($dummy, $page) = split(/<input type='checkbox' name='ids\[\]' value=/, $page, 2);
  if (defined($page))
  {
    ($dummy, $number) = split(/'/, $page, 3);
    $agent->tick('ids[]', "$number");
  }
}
while (defined($page));

# Post form
$agent->click('output_ics_confirm');

# Ask for iCalendar (Windows) format
$agent->form_number(1);
$agent->click('do_output');

# Print iCalendar file
print $agent->content;
```