

# [TOOL] WebRoot – Web Server Brute Forcer

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-04/0137.html>

---

**From:** SecuriTeam ([support\\_at\\_securiteam.com](mailto:support_at_securiteam.com))

**Date:** 04/25/05

To: [list@securiteam.com](mailto:list@securiteam.com)

Date: 25 Apr 2005 15:14:51 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

-----

WebRoot – Web Server Brute Forcer

---

## SUMMARY

## DETAILS

Have you ever been auditing a system where files are stored on a web server and accessed without authentication directly by an application that knows each file URL.

Have you tried a number of spider tools but they are based on links so they don't pull up anything.

CIRT.DK WebRoot is a Webserver auditing tools, that tries each and every combination (incremental) or a list of words from a file, against the Webserver.

In short:

A Brute Forcing tool to discover hidden directories, files or parameters in the URL of a webserver.

Tool source:

```
#!/usr/bin/perl
```

```
#oO
```

```
#
```

Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
# ***** !!! WARNING !!! *****
# ***** DO NOT DISTRIBUTE *****
# * FOR SECURITY TESTING ONLY! *
# *****
# * By using this code you agree that I makes no warranties or
representations, express or implied, about the *
# * accuracy, timeliness or completeness of this, including without
limitations the implied warranties of *
# * merchantability and fitness for a particular purpose. *
# * I makes NO Warranty of non–infringement. This code may contain
technical inaccuracies or typographical errors. *
# * This code can never be copyrighted or owned by any commercial company,
under no circumstances what so ever. *
# * but can be use for as long the developer, are giving explicit approval
of the usage, and the user understand *
# * and approve of all the parts written in this notice. *
# * This program may NOT be used by any Danish company, unless explicit
written permission from the developer . *
# * Neither myself nor any of my Affiliates shall be liable for any
direct, incidental, consequential, indirect *
# * or punitive damages arising out of access to, inability to access, or
any use of the content of this code, *
# * including without limitation any PC, other equipment or other
property, even if I am Expressly advised of *
# * the possibility of such damages. I DO NOT encourage criminal
activities. If you use this code or commit *
# * criminal acts with it, then you are solely responsible for your own
actions and by use, downloading,transferring, *
# * and/or reading anything from this code you are considered to have
accepted the terms and conditions and have read *
# * this disclaimer. Once again this code is for penetration testing
purposes only. And once again, DO NOT DISTRIBUTE! *
# *****
#
#oO
#
# Author/Developer: Dennis Rand – CIRT.DK
# Website: http://www.cirt.dk
# Copyright: (c)2005 by Dennis Rand
# Remember: This program may NOT be used, published or downloaded by any
Danish company, unless explicit written permission.
# This would be violation of the law on intellectual property rights, and
legal actions will be taken.
# What this tool does:
# Have you ever been auditing a system where files are stored on a web
server and accessed without authentication directly
# by an application that knows each file URL.
#
# Have you tried a number of spider tools but they are based on links so
they don't pull up anything.
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
#
# CIRT.DK WebRoot is a Webserver auditing tools, that tries each and every
# combination (incremental) or a list of words from
# a file, against the Webserver.
#
# In short:
# A Brute Forcing tool to discover hidden directories, files or parameters
# in the URL of a webserver.
#
#oO
#
# Usage:
# Scan localhost port 80 search for 200 OK response at the url
# http://127.0.0.1:80/admin/> incremental lowercase 1 to 3
# characters.
# WebRoot.pl -host 127.0.0.1 -port 80 -match "200 OK" -url
# "/admin/<BRUTE>" -incremental lowercase -minimum 1 -maximum 3
#
#oO
#
# Installation notes:
# perl -MCPAN -e shell
# cpan > install Bundle::LWP
# cpan > install IO::Socket
# cpan > install Getopt::Long
# cpan > install Algorithm::GenerateSequence
# cpan > Net::SSLeay::Handle
# cpan > quit
#
#oO
#
# TODO:
# Make results go into an HTML Report – COMPLETED
# Make scanner use multi threats for speed. – MISSING
# Make support for SSL – COMPLETED
# Make support for POST bruteforcing. – MISSING
# Make support for Cookies – COMPLETED
# Make some sort of false positive check – MISSING
# Make support for recursive scan. – MISSING
#
# Ideas, Features, and code updates, or error corrections please send to
# contact@cirt.dk
#
#oO
#
# Version descriptions
# Version 1.0
# I'm back from scratch, this time I'm going to make it a bit better, but
# have patience.
# For now results are only written to screen.
#
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
# Version 1.1
# We now have support for saving the scanning into an HTML file
# Decide how many lines of output from the server goes into the report.
#
# Version 1.2
# More information added into the report start
# Now WebRoot also supports scanning of a HTTPS connection.
# The response in the report now shows the HTML
#
# Version 1.3
# Fixed a bug in the -diff and -match options.
#oO
```

```
use IO::Socket;
use Getopt::Long;
use Algorithm::GenerateSequence;
use Net::SSLay::Handle qw/shutdown/;

$ver = "1.3";
$copyright = "(c)2005 by Dennis Rand – CIRT.DK";
$host = "127.0.0.1";
$port = "80";
$user_agent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1;
WebRoot $ver)";
$command = "GET";
$url = "/<BRUTE>";
$http_version = "HTTP/1.1";
$timeout = "10";
$l_number = 5;
$delay = "0";
@lowerc = ('a'..'z');
@upperc = ('A'..'Z');
@integ = ('0'..'9');
@spec = (',', '%', '-', '_', '&', '?', '#', '=', '@', '/', '\\', '"');
@all = (@lowerc, @upperc, @integ, @spec);
$sucess,$total = 0;
$timestampe = localtime;
$rap_updates = 0;
$recursive_f = "WebRoot.Recursive.Scan";
```

```
GetOptions(
```

```
    "host=s" => \$host,
    "port=i" => \$port,
    "ssl" => \$ssl,
    "timeout=i" => \$timeout,
    "delay=i" => \$delay,
    "incremental=s" => \$incremental,
    "minimum=i" => \$inc_minimum,
    "maximum=i" => \$inc_maximum,
    "wordlist=s" => \$wordlist,
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
"url=s" => \$url,
"command=s" => \$command,
"different=s" => \$diff,
"match=s" => \$match,
"useragent=s" => \$user_agent,
"cookie=s" => \$cookie,
"http_version=s" => \$http_version,
"reportlines=i" => \$l_number,
"saveas=s" => \$log,
"recursive" => \$recursive,

"help|?" => sub {
print "\n" x 2;
print "\t\t#####\n";
print "\t\t# Webservice Bruteforcing $ver #\n";
print "\t\t# ***** !!! WARNING !!! ***** #\n";
print "\t\t# **** FOR PENETRATION USE ONLY **** #\n";
print "\t\t# ***** #\n";
print "\t\t# $copyright #\n";
print "\t\t#####\n\n";
print "\t\t Basic settings\r\n";
print "\t\t -host\t\t Set the host ip or hostname to scan\r\n";
print "\t\t -port\t\t Set the port where the webserver are
located\r\n";
print "\t\t -timeout\t Set a maximum timeout for each try\r\n";
print "\t\t -delay\t Set a delay between each attempt\r\n";
print "\r\n";
print "\t\t Scanning options\r\n";
print "\t\t -incremental\t Set if the scanning has to
bruteforce\r\n";
print "\t\t use with \"lowercase\", \"uppercase\", \"integer\",
\"special\" or \"all\"\r\n\r\n";
print "\t\t -minimum\t Set the min chars for the incremental
scan\r\n";
print "\t\t -maximum\t Set the max chars for the incremental
scan\r\n";
print "\t\t -wordlist\t Set if a wordlist is supplied\r\n";
print "\t\t -url\t\t Set the URL to bruteforce.\r\n";
print "\t\t Use <BRUTE> where you want the bruteforcing\r\n";
print "\r\n";
print "\t\t Advanced scanning options\r\n";
print "\t\t -diff\t\t If the result has to be different, from the
response(Default)\r\n";
print "\t\t use with \"404 File not found\" and it will find
anything NOT matching in the response\r\n\r\n";
print "\t\t -match\t\t If the result has to match the
response\r\n";
print "\t\t use with \"200 OK\" and it will find anything
matching\r\n\r\n";
print "\t\t -command\t Set the HTTP command if not GET\r\n";
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
    print "\t\t Remeber you can also use <BRUTE> in this
field\r\n\r\n";
    print "\t\t -useragent\t Enter your own useragent\r\n";
    print "\t\t -cookie\t Enter a cookie value\r\n";
    print "\t\t -http_version\t If you want to use anything other then
HTTP/1.1\r\n";
    print "\t\t -recursive\t Make WebRoot scan recursively when
scanning for directories(Missing)\r\n";
    print "\r\n";
    print "\t\t Report options\r\n";
    print "\t\t -saveas\t Save report as defines name\r\n";
    print "\t\t -reportlines\t Amount of lines output from webserver
to put into report (Def: $l_number)\r\n";

    exit;
}
);

if ($port >= 0 and $port <= 65535){} else { print "Error: Port number
invalid, please use from 1-65535\r\n"; exit;}
if (($inc_minimum) > ($inc_maximum)) {print "Error: The Maximum are
larger then the Minimum\r\n"; exit;}
if (!$inc_minimum){$inc_minimum = "1"};
if (!$inc_maximum){$inc_maximum = "3"};
$sr_url = $url;
#oO
#oO
# Check for updates at www.cirt.dk
sub ChkUpdates
{
    $l = 1;
    $updates = IO::Socket::INET->new(
    Proto => "tcp",
    PeerAddr => "www.cirt.dk",
    PeerPort => "80",
    Reuse => 1,
    Timeout => 10,) || print "";
}

ChkUpdates();
$response = undef;
print $updates "GET /tools/webroot/wr_update.txt HTTP/1.0\r\nHost:
www.cirt.dk\r\nUser-Agent: Mozilla/4.0 (WebRoot Update Check)\r\n\r\n";
while(<$updates>)
{
    if(!defined($response)){ $response = $_;}
    $result .= $_;
}
if ($result =~ m/200 OK/mgsi)
{
    if($result !~ m/$ver/mgsi)
```



## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
Version $ver</TITLE>\n<BODY BGCOLOR=white>\n";
print FH "<SCRIPT LANGUAGE='JavaScript' TYPE='text/javascript'>\n";
print FH "<!-- // hide from old browsers\n\n";
print FH "// hide text from MSIE browsers\n\n";
print FH "with (document)\n";
print FH "{\n";
print FH " write('<STYLE TYPE='text/css'>');\n";
print FH " if (navigator.appName == 'Microsoft Internet Explorer')\n";
print FH "{\n";
print FH " write('.hiddentext {display:none} .outline {cursor:hand;
text-decoration:underline}');\n";
print FH "}\n";
print FH " write('</STYLE>');\n";
print FH "}\n\n";
print FH "// show text on click for MSIE browsers\n\n";
print FH "function expandIt(whichEl)\n";
print FH "{\n";
print FH " if (navigator.appName == 'Microsoft Internet Explorer')\n";
print FH "{\n";
print FH " whichEl.style.display = (whichEl.style.display == \"block\"
) ? \"none\" : \"block\";\n";
print FH "}\n";
print FH " else return;\n";
print FH "}\n";
print FH "// end hiding from old browsers -->\n";
print FH "</SCRIPT>\n";
print FH "</HEAD>\n";
print FH "<TABLE WIDTH=90% BGCOLOR=white CELLSPACING=0 CELLPADDING=2
BORDER=0>\n<TR>\n<TD>\n<CENTER>\n<FONT FACE=Tahoma COLOR=black
SIZE=+2><B>CIRT.DK WebRoot Version
$ver</B></FONT>\n</LEFT>\n</TD>\n</TABLE>\r\n";
print FH "<TABLE WIDTH=90% BGCOLOR=white CELLSPACING=0 CELLPADDING=2
BORDER=0>\n<TR>\n<TD>\n<CENTER>\n<FONT FACE=Tahoma COLOR=black
SIZE=1><B>$copyright</B></FONT>\n</LEFT>\n</TD>\n</TABLE>\n<BR>\r\n";
print FH "<TABLE WIDTH=90% BGCOLOR=black CELLSPACING=0 CELLPADDING=2
BORDER=0>\n<TR>\n<TD>\n<CENTER>\n<FONT FACE=Tahoma COLOR=white
SIZE=+1><B>Audit Rapport for $host port
$port</B>\n</FONT>\n</LEFT>\n</TD>\n</TABLE>\n<BR>\r\n";

if($rap_updates)
{
print FH " <B><CENTER>This scan was run with an outdated version of
WebRoot, get the latest version from
http://www.cirt.dk\r\n</CENTER><BR>\r\n";
print FH "<TABLE WIDTH=90% BGCOLOR=white CELLSPACING=0 CELLPADDING=0
BORDER=0><TR ALIGN=left><COLOR=black SIZE=2>\r\n";
print FH "\r\n";
print FH " <TR>\r\n";
print FH " <TD>\r\n";
```

```

print FH " <B>Host:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$host</B>\r\n";

print FH "\r\n";
print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B>Port number:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$port</B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
print FH "\r\n";

if ($incremental)
{
    print FH "\r\n";
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B><BR>Incremental:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B><BR>$incremental</B>\r\n";
    print FH " </LEFT></TD>\r\n";
    print FH " </TR>\r\n";
    print FH "\r\n";

    print FH "\r\n";
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B>Minimum:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B>$inc_minimum</B>\r\n";
    print FH " </LEFT></TD>\r\n";
    print FH " </TR>\r\n";
    print FH "\r\n";

    print FH "\r\n";
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B>Maximum:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B>$inc_maximum<BR></B>\r\n";
    print FH " </LEFT></TD>\r\n";
    print FH " </TR>\r\n";
    print FH "\r\n";
}

```

```

if($wordlist)
{
    print FH "\r\n";
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B><BR>Wordlist:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B><BR>$wordlist<BR></B>\r\n";
    print FH " </LEFT></TD>\r\n";
    print FH " </TR>\r\n";
    print FH "\r\n";
}

if($ssl)
{
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B><BR>Using SSL:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B><BR>TRUE</B>\r\n";
    print FH " </LEFT></TD>\r\n";
    print FH " </TR>\r\n";
}

print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B><BR>Bruteforce:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
if($ssl)
{
    print FH " <B><BR>https://$host:$port$r_url\r\n";
}
else
{
    print FH " <B><BR>http://$host:$port$r_url\r\n";
}
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";

if($match)
{
    print FH " <TR>\r\n";
    print FH " <TD>\r\n";
    print FH " <B>Result has to match:</B>\r\n";
    print FH " </TD>\r\n";
    print FH " <TD><LEFT>\r\n";
    print FH " <B>$match</B>\r\n";
}

```

```

print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
}

if($diff)
{
print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B>Result has to be different from:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$diff</B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
}

print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B>Scan Started:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$timestamp</B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";

print FH "</TABLE>\r\n";
print FH "<BR><TABLE WIDTH=90% BGCOLOR=black CELLSPACING=0
CELLPADDING=2 BORDER=0><TR><TD><CENTER><FONT FACE=Tahoma COLOR=white
SIZE=+1><B>Audit Results</B></FONT></LEFT></TD></TABLE>\r\n";
print FH "\r\n";
print FH "<TABLE WIDTH=70% BGCOLOR=white CELLSPACING=0 CELLPADDING=0
BORDER=0><TR ALIGN=left><BR>\r\n";
close(FH);
}

#oO
#oO
# Write results to logfile
sub write_log
{
$html_hostheader = $hostheader;
$html_hostheader =~ s/\r\n/<BR>/g;
open(FH, ">>", $log);
print FH "\r\n";
print FH " <TR>\r\n";
print FH " <TD>\r\n";
if($ssl)
{
print FH " <B><A
HREF=\"https://$host:$port$uri\">https://$host:$port$uri
COLOR=\"#FF0000\"></FONT></A></B><BR>\r\n";
}
}

```

```

_}
_ else
_{
_ print FH " <B><A
HREF="http://$host:$port$uri">http://$host:$port$uri
COLOR="#FF0000"></FONT></A></B><BR>\n";
_}
_ print FH " <B><A onClick="expandIt(input$total); return false"
CLASS="outline">+ Sent to server</A></B><BR>\n";
_ print FH " <DIV ID="input$total" CLASS="hiddentext">\n";
_ print FH " $command $uri $http_version<BR>$html_hostheader<BR><BR>";
_ print FH " </DIV>\n";
_
_ print FH " <B><A onClick="expandIt(output$total); return false"
CLASS="outline">+ Response from server:</B></A><BR>\n";
_ print FH " <DIV ID="output$total" CLASS="hiddentext">\n";
_ $result =~ s/</>/g;
_ $result =~ s/>/>/g;
_ my @lines = split(/\n/, $result);
_ my $firstlines;
_ for(0 .. $! number)
_{
_ $firstlines .= $lines[$ ];
_ print FH "$lines[$ ]<BR>\r\n";
_}
_ print FH " </DIV>\n";
_ print FH " <HR>\r\n";
_ print FH " </TD>\r\n";
_ close (FH);
}

#oO
#oO
# Write end to logfile
sub end_log
{
_ $timestamp = localtime;
_ open(FH, ">>", $log);

_ print FH "\r\n";
_ print FH " </TABLE>\r\n";
_ print FH " \r\n\r\n";
_ print FH "<BR><TABLE WIDTH=90% BGCOLOR=black CELLSPACING=0
CELLPADDING=2 BORDER=0><TR><TD><CENTER><FONT FACE=Tahoma COLOR=white
SIZE=+1><B>Rapport Summary</B></FONT></LEFT></TD></TABLE><BR>\r\n";
_ print FH "<TABLE WIDTH=90% BGCOLOR=white CELLSPACING=0 CELLPADDING=0
BORDER=0><TR ALIGN=left>\r\n";
_ print FH " <TR>\r\n";
_ print FH " <TD>\r\n";
_ print FH " <B>The Scan completed:</B>\r\n";

```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$timestamp<B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
print FH "\r\n";

print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B>Possible findings:</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$sucess<B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
print FH "\r\n";

print FH " <TR>\r\n";
print FH " <TD>\r\n";
print FH " <B>Total attempts</B>\r\n";
print FH " </TD>\r\n";
print FH " <TD><LEFT>\r\n";
print FH " <B>$total<B>\r\n";
print FH " </LEFT></TD>\r\n";
print FH " </TR>\r\n";
print FH "\r\n";

print FH "</TABLE>\r\n";
print FH "\r\n";
print FH " <BR><TABLE WIDTH=90% BGCOLOR=black CELLSPACING=0
CELLPADDING=2 BORDER=0><TR><TD><CENTER><FONT FACE=Tahoma COLOR=white
SIZE=1><B>$copyright</B></FONT></LEFT></TD></TABLE><BR><BR>\r\n";
print FH " </BODY>\r\n";

print FH "<!-- \r\n";
print FH " Remember if you are a Danish company, \r\n";
print FH " and does not have explicit written permission,\r\n";
print FH " you are in violation of the law on\r\n";
print FH " intellectual property rights\r\n";
print FH "-->\r\n\r\n";
print FH "</HTML>\r\n";
close (FH);
}

#oO
#oO
# Lets create the host header
$hostheader = "Accept: */*\r\n";
$hostheader .= "Accept-Language: en\r\n";
$hostheader .= "User-Agent: $user_agent\r\n";
$hostheader .= "Host: $host\r\n";
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
if($cookie)
{
    $hostheader .= "Cookie: $cookie\r\n";
}
$hostheader .= "Connection: Close\r\n";

#oO
#oO
# Save findings into for Recursive scanning

sub recursive_teaching
{
    if($recursive)
    {
        open(RECU_LEARN, ">>", $recursive_f);
        print RECU_LEARN "$brute^\r\n";
        close RECU_LEARN;
    }
}

#oO
#oO
#Incremental
if ($incremental)
{
    @choice = split(/./, $incremental);
    foreach $inc_input (@choice)
    {
        if ($inc_input eq "lowercase") { @do_choice = @lowerc;}
        elsif ($inc_input eq "uppercase") { @do_choice = @upperc }
        elsif ($inc_input eq "integer") { @do_choice = @integ }
        elsif ($inc_input eq "special") { @do_choice = @spec;}
        elsif ($inc_input eq "all") { @do_choice = @all;break;}
        else
        {
            print "Error: You have to specify the –incremental with
lowercase, uppercase, integer, special or all\r\n"; exit;
        }
    }
    start_log();
    $start_pos = ($inc_minimum - 1);
    do
    {
        my $len = $start_pos;
        my $gen = Algorithm::GenerateSequence->new(
            map {[@do_choice]} (0 .. $len)
        );

        local $" = "";
        while(my @c = $gen->next)
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
{
$brute = join("",@c);
  $uri = $url;
$uri =~ s/<BRUTE>/$brute/g;
$hostheader =~ s/<BRUTE>/$brute/g;
  bruteforcing();

}
$start_pos++;
} until ($start_pos >= $inc_maximum);
}

#oO
#oO
#Wordlist
if ($wordlist)
{
  foreach (split(/,/,$wordlist))
  {
    if (-f $_)
    {
      start_log();
      open(_FILE, $_);
      while (<_FILE>)
      {
        s/\r//;
        chomp;
        $brute = $_;
        $uri = $url;
        $uri =~ s/<BRUTE>/$brute/g;
        $hostheader =~ s/<BRUTE>/$brute/g;
        $found = $brute;
        bruteforcing();
      }
      close(_FILE);
    }
    else
    {
      print "\r\nThe wordfile you are trying to use: '$_' could not be
found\n";
      print "WebRoot.pl -help or -?" for more information.\n";
      exit;
    }
  }
}

#oO
#oO
# The connection are setup here.
sub connection {
  $|= 1;
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
$remote = IO::Socket::INET->new(
  Proto => "tcp",
  PeerAddr => $host,
  PeerPort => $port,
  Reuse => 1,
  Timeout => $timeout,) || die {print "\r\n Could not connect to $host on
port $port \r\n"};
}

#oO
#oO
# Let the bruteforcing begin
sub bruteforcing
{
  sleep($delay);
  $total++;
  $result = "";
  $response_code = undef;
  connection();

  if($ssl)
  {
    $ssl=1;
    eval
    {
      tie(*SSL, "Net::SSLeay::Handle", $target,$port);
    };
    print SSL "$command $uri $http_version\r\n$hostheader\r\n\r\n";
    shutdown(*SSL, 1);
    while(<SSL>)
    {
      if(!defined($response_code)){ $response_code = $_;}
      $result .= $_;
    }
  }
  else
  {
    print $remote "$command $uri $http_version\r\n$hostheader\r\n\r\n";
    while (<$remote>)
    {
      if(!defined($response_code)){ $response_code = $_;}
      $result .= $_;
    }
  }

  if($match)
  {
    if($result =~ m/$match/mgsi) {print
"X";$sucess++;write_log();recursive_teaching();} else {print "*";}
  }
}
```

## Securiteam: [TOOL] WebRoot – Web Server Brute Forcer

```
if($diff)
{
    if($result !~ m/$diff/mgsi) {print
"X";$sucess++;write_log();recursive_teaching();} else {print "*";}
    }
    close($remote);
}
end_log();
print "\r\n\r\n";
```

### ADDITIONAL INFORMATION

The information has been provided by <mailto:DennisRand@CIRT.dk> Dennis Rand.

To keep updated with the tool visit the project's homepage at:

<<http://www.cirt.dk/tools/webroot/WebRoot.txt>>

<http://www.cirt.dk/tools/webroot/WebRoot.txt>

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

=====

### DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.