Università degli Studi di Camerino

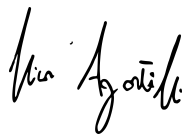
Scuola di Scienze e Tecnologie

Corso di Laurea in Informatica (Classe L-31)

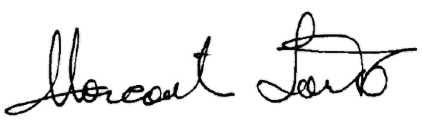
DDoS: una forma nascosta di contagio dietro la pandemia

Laureando
Nico Agostinelli

Matricola 101221



Relatore
Fausto Marcantoni



Indice

1. Attacchi DDoS e pandemia

L'attacco DDoS (Distribuited Denial of Service) è un attacco informatico che punta a far esaurire le risorse di un sistema informatico, bloccandone il funzionamento del servizio.
Nell'anno 2020, nel periodo della pandemia, si è verificato un aumento sostanziale degli attacchi DDoS. L'ATLAS Security Engineering and Response Team (ASERT) di NETSCOUT (Gruppo di ricercatori e ingegneri che si occupano di analizzare e raccogliere dati sui vari malwere e si focalizzano principalmente sugli attacchi DDoS), ha osservato che ci sono stati più di 10 milioni di attacchi DDoS in un solo anno. Più precisamente si sono verificati 10.089.687 attacchi nel corso dell'anno 2020. Sono quasi 1,6 milioni di attacchi in più rispetto al conteggio di 8,5 milioni del 2019.[Rep]
Il numero degli attacchi DDoS sono costantemente in aumento. Ma il contesto è importante quando si esaminano le statistiche DDoS del 2020.
Da marzo fino alla fine dell'anno, gli aggressori DDoS hanno operato durante la pandemia di COVID-19.

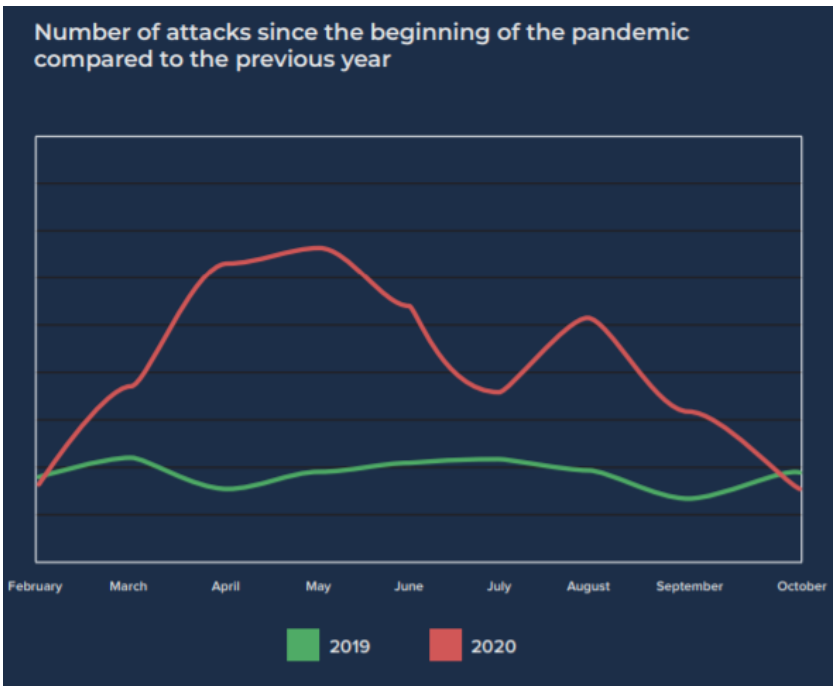


Figura 1.1: Confronto degli attacchi DDoS del 2020 rispetto al 2019.[Gra]

Mentre la maggior parte del mondo ha assistito a una crisi sanitaria globale senza precedenti, gli attori malintenzionati hanno visto nuove vulnerabilità e

opportunità.
È raro che l'attività annuale sia così profondamente influenzata da un evento, ma è il caso dell'attività e delle tendenze degli attacchi DDoS del 2020.
Non è un caso che questo numero fondamentale di attacchi globali arrivi in un momento in cui le aziende hanno fatto così tanto affidamento sui servizi online per sopravvivere. Non solo le aziende hanno sfruttato i servizi online, ma anche le persone hanno dovuto riadattarsi alla situazione che si stava creando, con la nascita di nuovi siti web, lo smart-working e l'aumento dello shopping online, che hanno influenzato notevolmente l'incremento del traffico internet.

I paesi principalmente colpiti sono stati America, Corea del Sud, Regno Unito, Brasile e Cina.
Gli attacchi sono principalmente con scopo di estorsione, in quanto gli hacker spesso minacciano di intensificare gli attacchi se non viene pagato loro un certo ammontare di denaro in Bitcoin come riscatto.
Come risultato il sistema informatico colpito smette di funzionare per diverso tempo, stando nelle mani dei malfattori.

Un gruppo di hacker denominato come **Lazarus Bear Armada**, denominato così in quanto durante un attacco informatico hanno dichiarato di far parte di uno dei gruppi di hacker degli Advanced Persistent Threat Groups (APT) - Lazarus Group, Fancy Bear e Armanda Collective, è stato molto attivo nel 2020, causando numerosi attacchi colpendo diversi settori specifici dalle istituzioni finanziarie alle organizzazioni sanitarie.
Il loro modus oprandi è quello di inviare un attacco DDoS di dimensioni ridotte, andando a chiedere un riscatto al loro bersaglio facendogli capire che, se non pagano, amplificheranno l'attacco con uno più grande e sofisticato. Hanno agito in diverse parti del mondo tra cui Europa, Medio Oriente, Africa, Stati Uniti e America Latina.

2. Introduzione agli attacchi DoS

DoS è l'acronimo di *Denial of Service*, in italiano interruzione del servizio, è un attacco informatico che consiste nel far esaurire deliberatamente le risorse di un sistema informatico che fornisce un servizio ai client, come un sito web su un web server, facendo in modo che non possa più erogarne il servizio.[Dos]
Esistono diversi tipologie di attacchi DoS che sfruttano un singolo host per effettuare un attacco e sono potenzialmente rintracciabili.
Andremo a vederne successivamente alcune tipologie, ma non prima di aver parlato di qual'è stato il primo attacco DoS conosciuto.

2.1 Primo attacco Dos della storia

David Dennis, studente adolescente, nel 1974 ha scoperto un comando da poter eseguire sui terminali PLATO (PLATO è una delle prime piattaforme multiutente) nel CERL(Computer-based Education Research Laboratory presso l'Università dell'Illinois Urbana-Champaign).

Figura 2.1: Computer PLATO.[Com]

Il comando scoperto da Dennis, chiamato ''external" o ''ext", consentiva l'interazione con dispositivi esterni collegati ai terminali, ma se il terminale non

prevedeva dispositivi collegati, il risultato era il blocco del terminale che doveva essere spento e riacceso per ripristinarne la funzionalità.
PLATO utilizza come linguaggio di programmazione "TUTOR" e i livelli di privilegi di sistema sono impostati su "Modalità autore" che tutti gli autori/sviluppatori avevano.

In una e-mail di Dennis, inviata ad un suo amico, spiega come sono andate le cose:

''*...Quello che ho fatto è stato aver sentito parlare di un nuovo comando chiamato comando "external" in TUTOR, o "ext". In particolare, uno dei ragazzi stava dicendo che se non avessi un dispositivo collegato, un comando ext causerebbe il blocco del terminale e lo spegnimento. Ricorda che lo spegnimento è stato sconsigliato, a causa della costante preoccupazione per l'alimentazione instabile dei pannelli al plasma.*

...ogni account utente su PLATO era impostato su default, "può accettare comandi ext" e attivato per impostazione predefinita.

...ho scritto un programmino che ha inviato gli ext a tutti entro un intervallo di numeri di sito, ho aspettato di essere fuori dal CERL una mattina e ho lasciato fare.

...31 utenti si sono spenti tutti in una volta, c'è stato un grande caos in classe, i monitor del sito sono stati avvisati."[Dea]

Il comando -ext- era 'nuovo' nel 1974. E' stato concepito in modo da poter collegare il terminale di PLATO a un dispositivo periferico esterno e controllare tale dispositivo utilizzando una connessione seriale.
Nel tempo PLATO è diventato più sicuro e robusto.

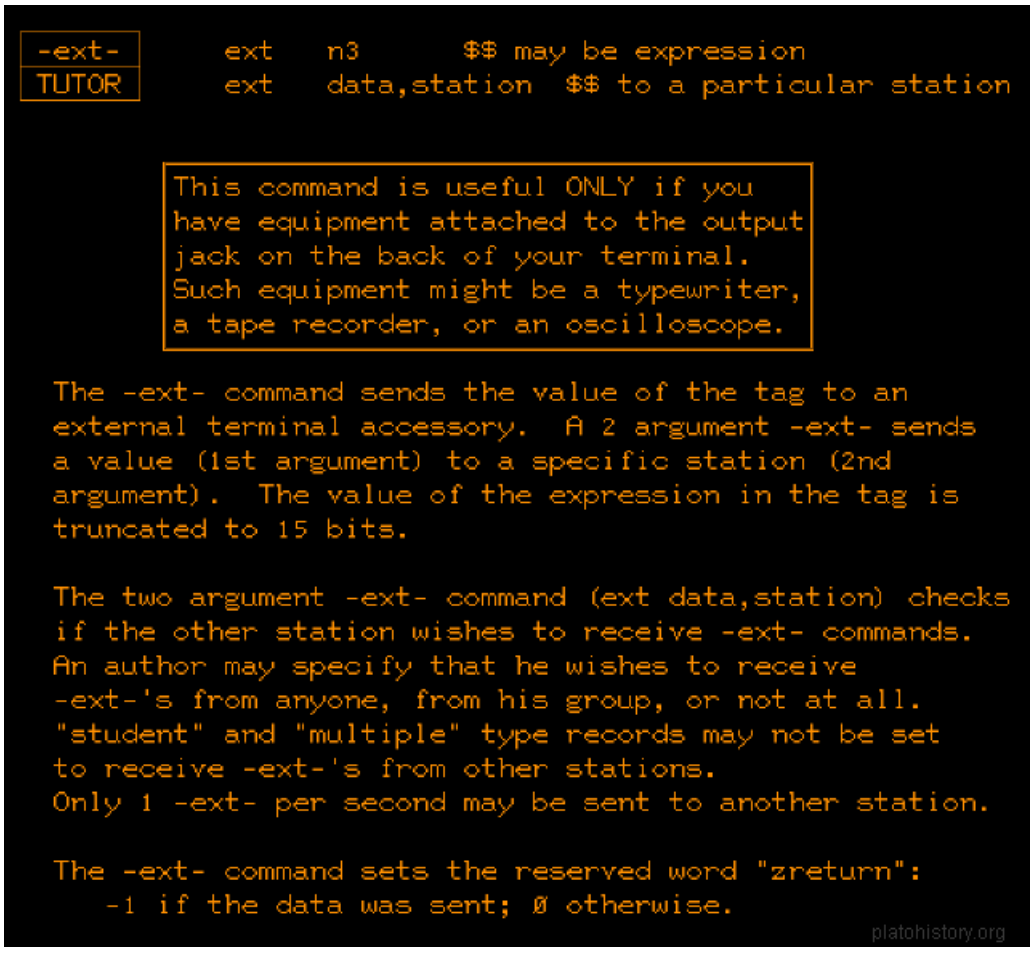


Figura 2.2: Schermata del funzionamento del comando -ext-

3. Tipologie di attacchi DoS eseguiti da un singolo host

3.1 SYN-Flood

Il *SYN Flood* rappresenta il capostipite degli attacchi DoS e conosciuto come attacco "semi-aperto", dove un utente malevolo invia una serie di richieste **SYN-TCP** verso il sistema oggetto dell'attacco.
Quando un client cerca di iniziare una connessione **TCP** verso un server scambiano una serie di messaggi.

1. Il client richiede una connessione inviando un messaggio SYN al server e così facendo richiedere di sincronizzare la Sequence Number (Segmento di 32-bit del TCP che serve per il riconoscimento del pacchetto inviato).

2. Il server *acknowledges*, risponde a tale richiesta inviando un messaggio SYN-ACK al client e inviando la Sequence Number ricevuta incrementata di una unità.
   Il server inoltre alloca dello spazio in memoria salvando la request fatta dal client.

3. Il client risponde con un **ACK** e il server dealloca la request fatta al primo punto dal client consentendo di fatto la riuscita della connessione in maniera corretta.

Questo processo è chiamato *three-way handshake* e costituisce il fondamento per ogni connessione stabilita utilizzando i protocolli TCP/IP.

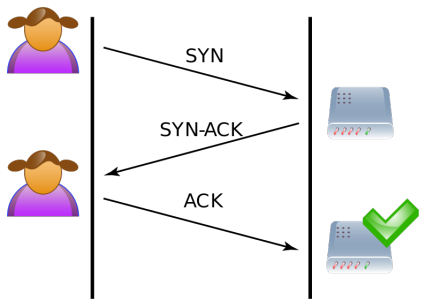


Figura 3.1: Three-way handshake.[Imme]

Nel SYN flood la connessione segue il principio del three-way-handshake modificato in modo da mettere fuori uso i servizi internet destinati agli utenti.
Nello specifico l'attaccante può ricorrere a due tecniche distinte per mettere a segno il proprio attacco.

Tecnica 1

1. L'attaccante richiede una connessione al server utilizzando lo **spoofing**(è un tipo di attacco informatico che impiega in varie maniere la falsificazione dell'identità (spoof). Lo spoofing può avvenire a qualunque livello della pila ISO/OSI e può riguardare anche la falsificazione delle informazioni applicative), camuffando il proprio indirizzo IP con un indirizzo IP diverso.
2. Il server risponde alla richiesta inviata dal client malevolo con un SYN-ACK all'indirizzo IP non corretto.

A questo punto la connessione rimarrà aperta(ovvero si troverà nello stato **semi-aperta**) perchè non riceverà mai il messaggio di ACK da parte del client.

Tecnica 2

1. L'attaccante richiede una connessione al server inviando una richiesta SYN.
2. Il server risponde alla richiesta inviata dal client con un SYN-ACK.

In questo caso il client decide intenzionalmente di non rispondere, e dunque di instaurare una connessione con il server, lasciando di fatto aperta la connessione.
Non avendo chiuso le connessioni, il server si trova in tutte e due i casi in uno stato vulnerabile ad ogni possibile attacco malevolo.
Molte volte il SYN flood viene utilizzato solo come fase iniziale di altri attacchi molto più sofisticati.[Syn]

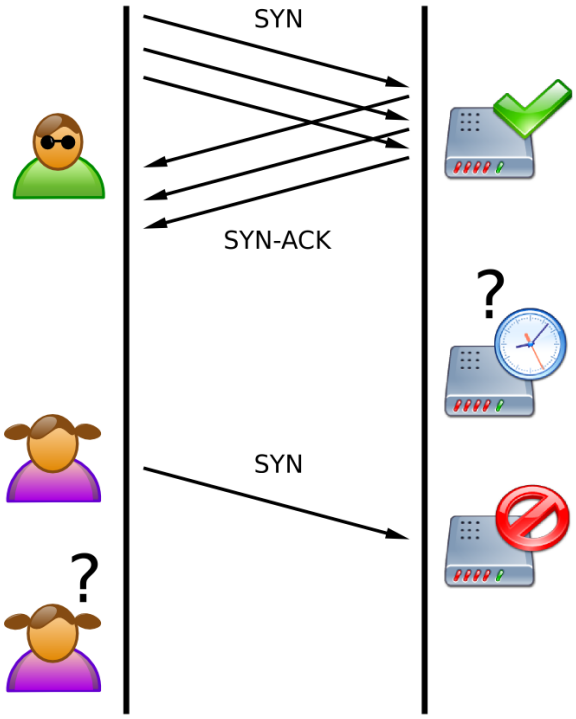


Figura 3.2: TCP SYN flood.[Syn]

3.1.1 Contrastare il SYN Flood

Ci sono diversi modi per contrastare un attacco SYN Flood:

- **Aumentare la capacità del backlog del SYN**, il backlog è come una tabella dove ogni singola casella contiene le informazioni di una connessione TCP, gestita dal sistema operativo. Dopo aver stabilito la connessione three-way handshake, viene inoltrato alle applicazioni in ascolto sulle porte e poi rimosso dal backlog. Per contrastare un SYN flood bisogna aumentare la capacità del backlog in quanto ogni casella occupa uno spazio in memoria che ne limita la quantità di dati che può contenere.

- **Riciclare le connessioni TCP semi-aperte più vecchie**, in questo modo si libera uno spazio per una nuova connessione semi-aperta e in combinazione con l'aumento dello spazio sul backlog, permette di raggiungere un sistema in caso di attacco SYN flood. Non funziona in caso di attacco volumetrico.

- **SYN cache**, invece di salvare i dati completi del TBC(Transmission control block) della connessione semi-aperta nel backlog, ne viene mantenuta solo una piccola parte. Utilizza una funzione crittografica di hash per impedire all'aggressore di indovinare le informazioni crittografiche della connessione.

- **SYN coockes**, sviluppa il concetto di SYN cache rinunciando al TBC. Si codificano i parametri rilevanti di connessione nella sequenza numerica del pacchetto SYN/ACK. La funzione crittografica di hash garantisce che l'aggressore non riesca facilmente a indovinare la sequenza numerica.

3.2 Smurf

L'attacco Smurf è il capostipite degli attacchi amplificati. Il suo scopo è sovraccaricare il computer della vittima con messaggi provenienti da diversi nodi nella rete in risposta a false richieste che vengono spacciate per richieste della vittima. Si basa su una tecnica di amplificazione dovuta alla natura del protocollo IP che può essere facilmente ovviata con una corretta configurazione dei dispositivi di rete.
L'amplificazione si ottiene inviando un pacchetto ICMP ECHO ad un indirizzo di broadcast di una rete, tramite il comando Ping.

La struttura del protocollo IP non impedisce lo *spoofing* degli indirizzi IP, questo permette a un qualsiasi nodo di generare un pacchetto con indirizzo IP mittente arbitrario. Di conseguenza è possibile l'invio di un pacchetto IP dal proprio nodo, usando l'indirizzo IP di una seconda macchina presente in rete. Una tipologia di richiesta d'informazione del protocollo di diagnostica ICMP (Internet Control Message Protocol) consente l'invio di richieste di eco (ECHO REQUEST) al fine di:

- Stabilire se è possibile la comunicazione a livello IP tra due nodi.
- Verificare l'esistenza di un nodo in una rete.
- Calcolare in termini di tempo la distanza che intercorre tra due nodi.

Il prossimo passo sarà quello di scegliere una rete che faccia poi da amplificatore per l'attacco e cominciare a spedire quanti più pacchetti possibili verso l'indirizzo di broadcast della rete prescelta. Un attacco Smurf risulta efficace se vengono adoperati molti gateway, generalmente chi vuole effettuare un attacco Smurf si prepara una lista di broadcast su un file di testo corrispondenti ai gateway mal configurati presenti nella rete.
Questo attacco, a causa del fatto che "amplifica" la quantità di dati usati dall'attaccante, fa parte dei cosiddetti **amplification attack**. Tale tecnica consente di generare molto traffico, soprattutto considerando che i messaggi ICMP sono incapsulati in un datagram IP, che consente l'inserimento di dati opzionali, e nel caso di richieste eco (ECHO REQUEST), tali dati saranno replicati nei messaggi di risposta. Il nodo vittima (target) riceverà un flood di pacchetti ICMP da numerosi indirizzi IP. Gli effetti di questo attacco si verificano in:

- Saturazione della banda disponibile e quindi disconnessione delle connessioni TCP.
- Disconnessione dalla rete in connessioni analogiche.
- Crash dei firewall software.
- Crash di sistemi operativi obsoleti.

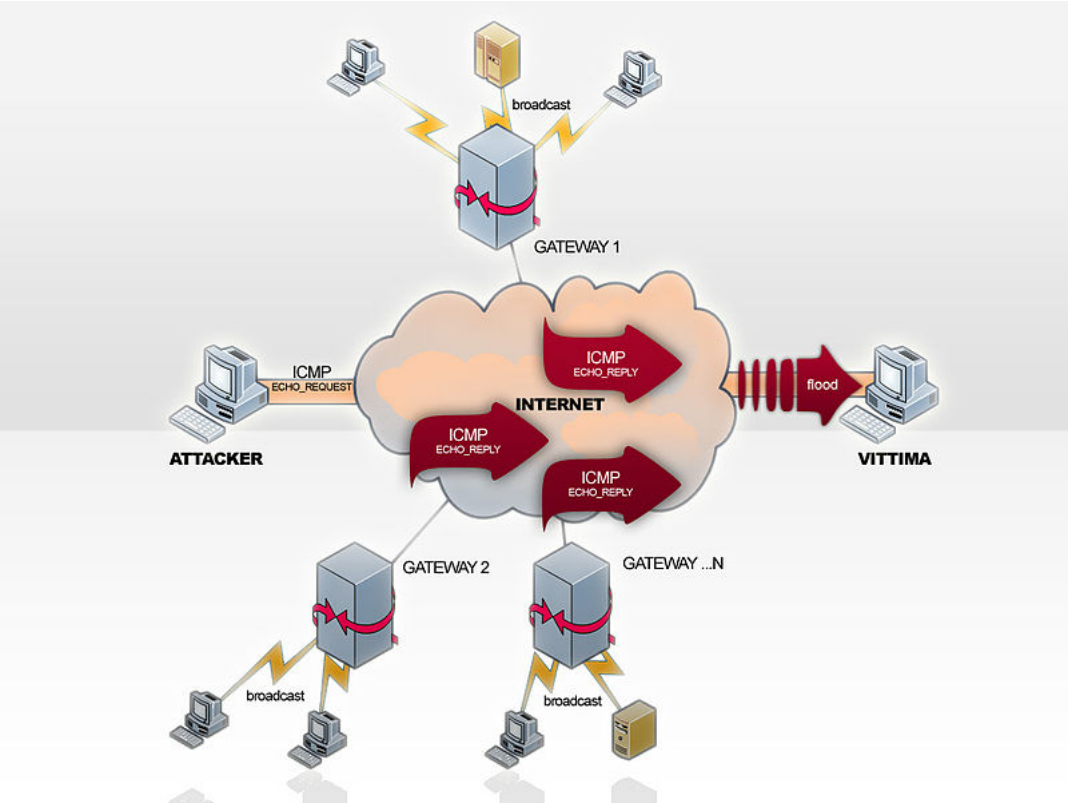


Figura 3.3: Attacco Smurf.[Smu]

3.2.1 Contrastare un attacco Smurf

Un attacco Smurf può essere combattuto su due fronti, lato client e lato router. A lato client possiamo avere singoli computer appartenenti ad una rete, che possono essere protetti dal diventare amplificatori Smurf, impostando il blocking dei pacchetti ICMP in ingresso, di conseguenza non risponderanno più a nessun ping. Numerosi firewall incorporano già questa funzionalità.
Lato router, configurarlo in modo tale da non ridirigere i pacchetti via broadcast, in quanto è possible limitare la banda da destinare al servizio ICMP tale da diminuire l'amplificazione attivando la funzione di CAR(Committed Access Rate) presente su IOS (In informatica Cisco IOS è un sistema operativo per dispositivi di rete instradatori (router) e commutatori (switch) dell'azienda Cisco Systems, installato attualmente su tutti gli switch e sulla maggior parte del router della azienda stessa, che costituiscono circa l'80% dei router della rete mondiale).

3.3 RUDY (RU-DEAD YET)

Questo attacco DoS attacca utilizzando il livello 7, cioè il livello applicativo.
Il suo scopo non è quello di inondare un server web, ma di esaurire il numero di connessioni rendendo impossibilitati gli utenti a stabilirne. Come bersaglio viene scelto un URL di un sito Web, a cui viene effettuata una scansione finchè non vengono trovati i campi del modulo Web incorporati.
Lo strumento crea successivamente una richiesta HTTP POST con l'intestazione HTTP della lunghezza del contenuto impostata su un valore molto grande.
Poi viene avviato l'invio del modulo ad una velocità molto bassa. Divide i dati in numerosi piccoli pacchetti e li invia pochi secondi uno dopo l'altro. Ciò consente allo strumento di mantenere la connessione aperta per un lungo periodo di tempo. Eseguendo le stesse richieste HTTP di seguito a velocità bassa, la tabella di connessione del server HTTP o altre risorse del server vengono esaurite. Di conseguenza, il server non può più gestire il traffico legittimo.
L'attacco può essere eseguito da un indirizzo IP o, per rendere più difficile la protezione, da diversi indirizzi IP come attacco di negazione del servizio distribuito.

3.3.1 Difendersi da un attacco RUDY

Il metodo di mitigazione più efficace per gli attacchi di connessione lenta consiste nell'eliminare tutte le connessioni lente configurando attentamente il server Web e il sistema operativo per limitare i valori di timeout. Tuttavia, l'effetto collaterale di un tale approccio è che gli utenti legittimi con connessioni Internet lente potrebbero non essere in grado di utilizzare il sito Web o l'applicazione Web.

4. Attacchi DDoS

4.1 Come funziona un attacco DDoS

Gli attacchi DDoS(Distribuited Denial of Service), al contrario, utilizzano più fonti per effettuare un attacco sfruttando le **botnet**, rendendo difficile il rintracciamento dell'attaccante. L'esempio in analogia è quello di un gruppo di persone che affollano la porta d'ingresso di un negozio, e non consentendo alle parti legittime di entrarci, interrompendo così le normali operazioni. Ciò rende effettivamente impossibile fermare l'attacco semplicemente bloccando una singola fonte.

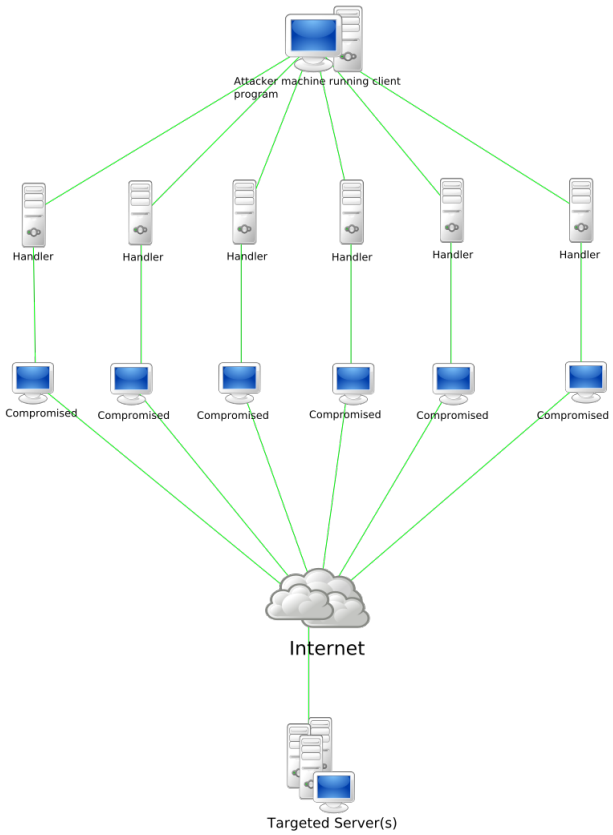


Figura 4.1: Attacco DDoS. [Scha]

Per rendere efficace un attacco DDoS vengono utilizzati molti computer inconsapevoli, detti **zombie**, sui quali precedentemente è stato inoculato un programma appositamente creato per attacchi DoS che si attiva ad un comando proveniente dall'attaccante. I computer infettati vanno a formare una botnet, controllata da un **botmaster**. Sfruttando il protocollo **TCP/IP** riescono a mascherare la vera provenienza dell'attacco, in quanto il protocollo non garantisce una particolare sicurezza sull'identificazione dei mittenti dell'attacco ma ne protegge l'anonimato.

Un attacco DDoS è caratterizzato da una asimmetria che si verifica tra la richiesta e le risposte correlate in una sessione DNS, portando ad un ingente danno. Il flusso enorme di risposte generato provocherà nel sistema una "inondazione" di traffico, rendendo il server inadeguato alla gestione delle abituali funzioni on-line.

4.2 DRDoS

Il **Distribuited Reflect Denial of Service**(**DRDoS**) è un particolare attacco DDoS. Il computer attaccante produce delle richieste di connessione verso server con connessioni di rete molto veloci utilizzando come indirizzo IP quello del bersaglio dell'attacco. Rispetto al DDoS, il DRDoS è molto più sofisticato, in quanto si basa su uno dei protocolli, spesso ignorato per via della sua scarsa affidabilità, cioè il protocollo **UDP**, applicando lo sproofing sull'indirizzo IP. Il protocollo richiede che il destinatario restituisca una risposta all'indirizzo IP di origine, quindi se l'indirizzo è stato modificato nei pacchetti UDP, i pacchetti di risposta raggiungeranno l'indirizzo IP fornito dall'attaccante. Molte di queste risposte sovraccaricano la vittima.

5. Software per effettuare attacchi DDoS

Andiamo ad analizzare alcuni softwere utilizzati per effettuare un attacco Dos o DDoS. In particolare andremo a parlare del funzionamento di UFONet, TOR'S HAMMER, Slowloris, H.U.L.K, GoldenEye e THC-SSL-DOS, per poi citare Loic e Zeus che sono stati due softwere importanti e attualmente non più in utilizzo.

5.1 UfoNet

Figura 5.1: Logo UFONet.[Loga]

UFONet è un software opensource, P2P e crittografico che permette di eseguire attacchi DoS e DDoS, sul Layer 7 (APP/HTTP) tramite lo sfruttamento di vettori **Open Redirect** su siti web di terze parti per fungere da botnet e sul Layer3 (Network) abusando del protocollo.
Funziona anche come DarkNET crittografato per pubblicare e ricevere contenuti creando una rete client/server globale.
UFONet è un programma multi-piattaforma che richiede Python 3 ed alcune sue librerie.

5.1.1 Come funziona l'Open Redirect

L'Open Redirect è un'altra tipologia di attacco DoS.
Sfrutta una vulnerabilità di alcuni siti che permette di reindirizzare un utente che sta navigando nel sito stesso, a un sito diverso che può essere anche malevolo.

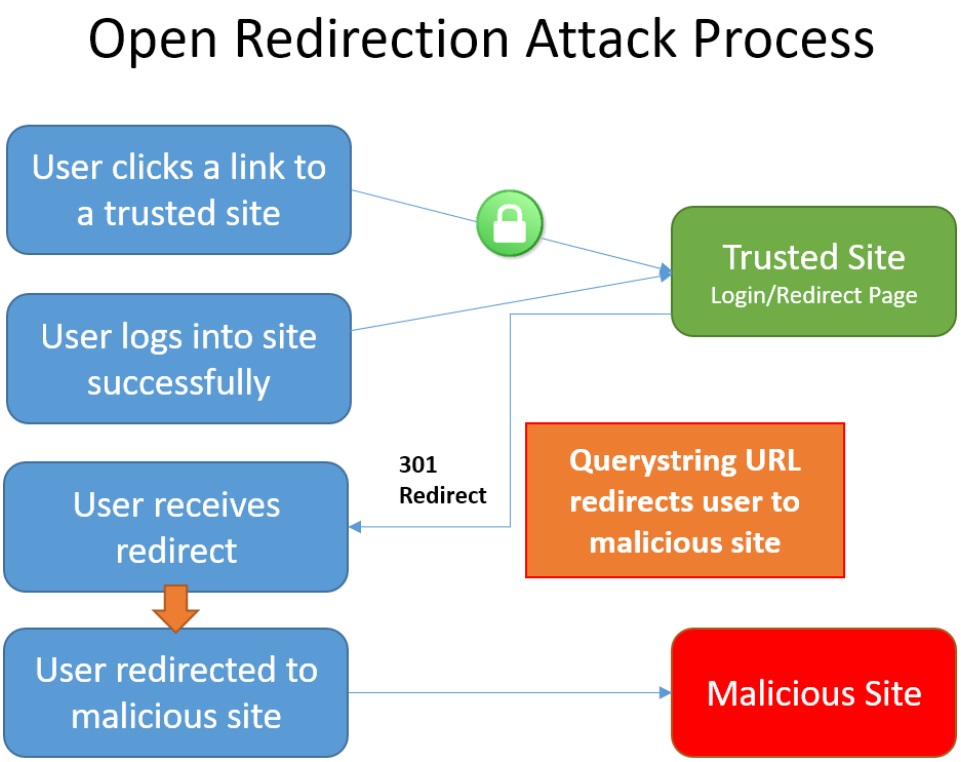


Figura 5.2: Funzionamento dell'Open Redirect.[Ope]

Una vulnerabilità dell'Open Redirect permette all'attaccante di manipolare l'utente e reindirizzarlo da un sito ad un altro, che potrebbe risultare malevolo.
La comunità di cyber-security non enfatizza particolarmente questo tipo di vulnerabilità in quanto considerata di basso livello e comunemente connessa a tecniche di fishing e ingegneria sociale.
Tuttavia le vulnerabilità Open Redirect possono permettere all'attacante di andare oltre. La vera pericolosità di questa vulnerabilità è che può essere utilizzata in combinazione con altri attacchi tipo Server Side Request Forgery, XSS-Auditor bypass.
Ecco come si presenta un esempio di codice php che ottiene un Url da una stinga di query:

```
$redirect_url = $_GET['url'];
header("Location:" . $redirect_url);
```

Questo codice ottiene un Url da input utente e poi reindirizza la vittima ad un altro Url.
Il codice php dopo la funzione **header** continua la sua esecuzione.
Se un utente configura il browser per ignorare l'Open Redirect e comunque possibile accedere al resto della pagina.
Questo codice è vulnerabile agli attacchi finchè non vengono implementate validazioni che verifichino l'autenticità dell'Url, per esempio: degli attaccanti

potrebbero usare questa vulnerabilità per indirizzare gli utenti ad un sito malevolo facente parte di una campagna di fishing, e, finchè non c'è validazione, gli attaccanti possono creare un iperlink per il medesimo scopo.
Ecco come si presenta un iperlink che sfrutta una vulnerabilità Open Redirect:

```
http://sitovulnerabile.com/file.php?url=http://sitomalevolo.com
```

Time	Protocol	Source	Destination	Info
04:59:33.007336	TCP	192.168.1.114	50.63.202.1	57450→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.048829	TCP	50.63.202.1	192.168.1.114	80→57450 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460
04:59:33.049002	TCP	192.168.1.114	50.63.202.1	57450→80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
04:59:33.068289	TCP	192.168.1.114	69.163.163.134	57451→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.068599	TCP	192.168.1.114	69.163.163.134	57452→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.068876	TCP	192.168.1.114	69.163.163.134	57453→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.069119	TCP	192.168.1.114	69.163.163.134	57454→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.073311	TCP	192.168.1.114	69.163.163.134	57455→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.073672	TCP	192.168.1.114	69.163.163.134	57456→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
04:59:33.142299	TCP	69.163.163.134	192.168.1.114	80→57452 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM=1 WS=1024
04:59:33.142453	TCP	192.168.1.114	69.163.163.134	57452→80 [ACK] Seq=1 Ack=1 Win=65536 Len=0
04:59:33.142788	HTTP	192.168.1.114	69.163.163.134	GET / HTTP/1.1

Figura 5.3: Pacchetti scambiati in una Redirect.[Pac]

In questo esempio possiamo vedere uno scambio di informazioni tra un client e un server dove l'utente, accedendo al sito in cui voleva navigare, viene successivamente mandato in un altro sito attraverso una Redirect.

5.1.2 Sfruttare una Open Redirect per fishing

Un utente che clicca su un link mentre naviga su un sito legittimo normalmente non sospetta niente se una form di login o un altro tipo di form appare.
Un attaccante può sfruttare questa situazione per mandare alle vittime il link a un sito affidabile ma con una vulnerabiiltà di Open Redirect che ne effettui il reindirizzamneto a un Url malevolo. Questi Url sono normalmente strutturati come una pagina di fishing che copia perfettamente il sito legittimo originale.

Una volta che la vittima entra sul sito malevolo viene tipicamente portata a inserire delle credenziali in un form di login che attraverso uno script, controllato dall'attaccante ha normalmente lo scopo di rubare le credenziali inserite.
Queste credenziali sono raccolte dall'attaccante e usate per impersonare le ignare vittime sul sito legittimo. Dal momento che il dominio del sito legittimo viene mostrato all'utente quando clicca il link malevolo la probabilità che questo attacco abbia successo è molto alta.

5.1.3 Open Redirect usata per Server Side Request Forgery

La SSRF è un attacco che può compromettere un server. Sfruttare una vulnerabilità SSRF permette ad un hacker di scoprire un sistema che si nasconde dietro un firewall, un middleware di protezione o filtri, per il cui bypass la Open Redirect è estremamente utile.
In questa maniera l'attaccante può accedere al contenuto di un dominio che può reindirizzare a qualsiasi numero di altri luoghi.

5.1.4 Open Redirect usata per XSS-Auditor bypass

I browser moderni, per esempio, hanno un sistema di default chiamato XSS-Auditor, ovvero un componente che ha lo scopo di bloccare attacchi dall'andare a segno.
Tuttavia ci sono modi di bypassare questo sistema utilizzando l'Open Redirect, dal momento che l'XSS-Auditor non ha modo di bloccare l'attaccante dall'includere uno o più script che siano hostati su quello stesso dominio.
Ecco come si presenta una vulnerabiltà Open Redirect che permette di bypassare l'XSS-Auditor:

```
<script
    src="https://sito.com/path/https/hacker.com/payload.js">
</script>
```

5.1.5 Fixare le vulnerabiltà Open Redirect

Ci sono diversi modi per prevenire questo tipo di vulnerabilità. Di seguito alcuni di questi metodi raccomandati dalla Open Web Application Security Project (OWASP):

- Non usare reindirizzamenti;
- Non accettare Url come input utente per una destinazione;
- Se è assolutamente necessario accettare in input un Url dagli utenti, chiedi loro di fornire un token o un altro identificativo che fa riferimento lato server all'Url completo. Tuttavia, bisogna fare attenzione a non introdurre una vulnerabilità di enumerazione che permette agli utenti di ciclare tutti gli identificativi per trovare tutti i possibili target;
- Quando l'input utente è imperativo assicurarsi che tutti i valori inseriti siano validi, filtrati per l'applicazione e autorizzati per ogni utente;
- Creare una lista di Url sicuri per filtrare correttamente l'input. E' preferibile utilizzare una white-list piuttosto che una black-list;
- Forzare i redirect prima ad una pagina che notifica gli utenti che stanno venendo reindirizzati al di fuori del sito. Questo messaggio deve mostrare in chiaro la destinazione e chiedere agli utenti di cliccare su un link se vogliono realmente procedere alla nuova destinazione.

5.1.6 Come utilizzare UFONet

Utilizzando il prompt dei comandi possiamo avviare UFONet.
Per consultare la lista di tutti i comandi disponibili digitiamo:

```
$ ./ufonet [options]
```

Verrà visualizzata una lista di opzioni con cui l'utente può comunicare con il software.

5.1.7 Trovare host vulnerabili

Se si volesse effettuare un attacco, si dovrebbe prima di tutto ricercare degli host vulnerabili disponibili per l'utilizzo. UFONet può scavare nei risultati di diversi motori di ricerca per trovare possibili siti vulnerabili all'Open Redirect.
UFONet utilizza come motore di ricerca di default "Duck-Duck Go".

Una stringa di query comune di una url dovrebbe essere strutturata così:

```
'proxy.php?url='
'check.cgi?url='
'cecklink?url='
'validator?url='
```

Pertanto si può iniziare una ricerca con:

```
$ ./ufonet -s 'proxy.php?url=' --sa
```

Questo comando dice a UFOnet di cercare siti contenenti "proxy.php?url=", utilizzando tutti i motori di ricerca integrati. I siti che contengono "proxy.php?url=" possono essere vulnerabili ai reindirizzamenti aperti.
Si possono anche caricare stringhe di ricerca da un file di testo con il comando:

```
$ ./ufonet --sd 'botnet/dorks.txt'
```

Dove, l'argomento del comando –sd è il percorso del file contente i dork di ricerca. Questo comando utilizza dorks.txt di UFOnet come un elenco di stringhe per cercare potenziali vulnerabilità di reindirizzamento aperto. Alla fine del processo,

chiederà se si vuole controllare l'elenco recuperato per vedere se gli URL sono vulnerabili.

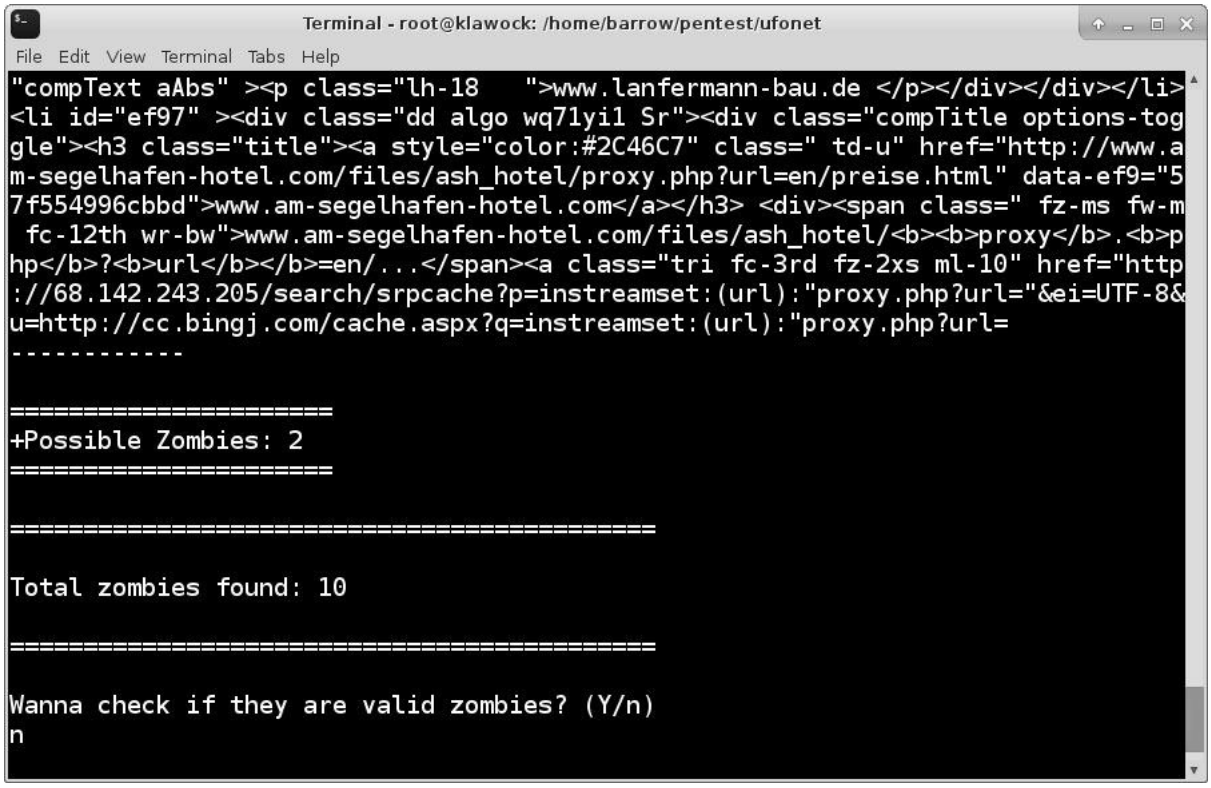


Figura 5.4: [Immf]

Se la risposta è "Y", verrà verificata la vulnerabilità o meno degli zombie individuati.

Un'altra opzione per localizzare gli obiettivi è scaricare la lista degli zombie già trovati messa a disposizione dalla comunità di UFONet.

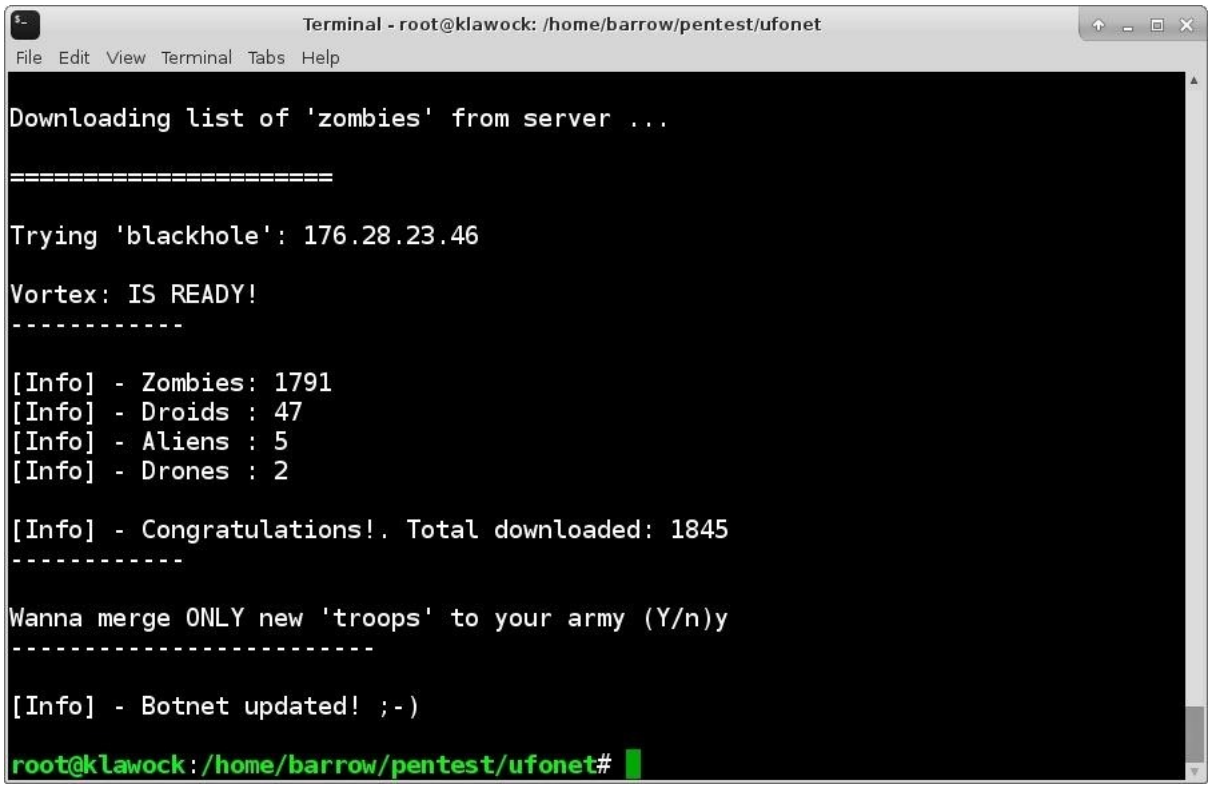


Figura 5.5: [Immd]

5.1.8 Test

Ora che si ha un elenco di zombie a disposizione, si va ad impostare una pagina vulnerabile in una VM (macchina virtuale).
Questa pagina è un semplice reindirizzamento aperto e appartiene al file **botnet/zombies.txt**.

UFOnet archivia i dati sugli host vulnerabili in formato testo nella cartella botnet.
Ogni file di testo ha un nome a tema e rappresenta una forma diversa di reindirizzamento aperto.

- **Zombie**: HTTP GET 'Open Redirect' bot.
- **Droid**: HTTP GET 'Open Redirect' bot con parametri.
- **Alien**: bot HTTP POST "Reindirizzamento aperto".
- **UCAV**: questi siti controllano se il target del DDoS è attivo.
- **Dorks**: un elenco di stringhe di ricerca potenzialmente vulnerabili.

Successivamente andiamo a vedere se la VM è vulnerabile ad un reindirizzamento aperto.

```
$ ./ufonet -t botnet/zobies.txt
```

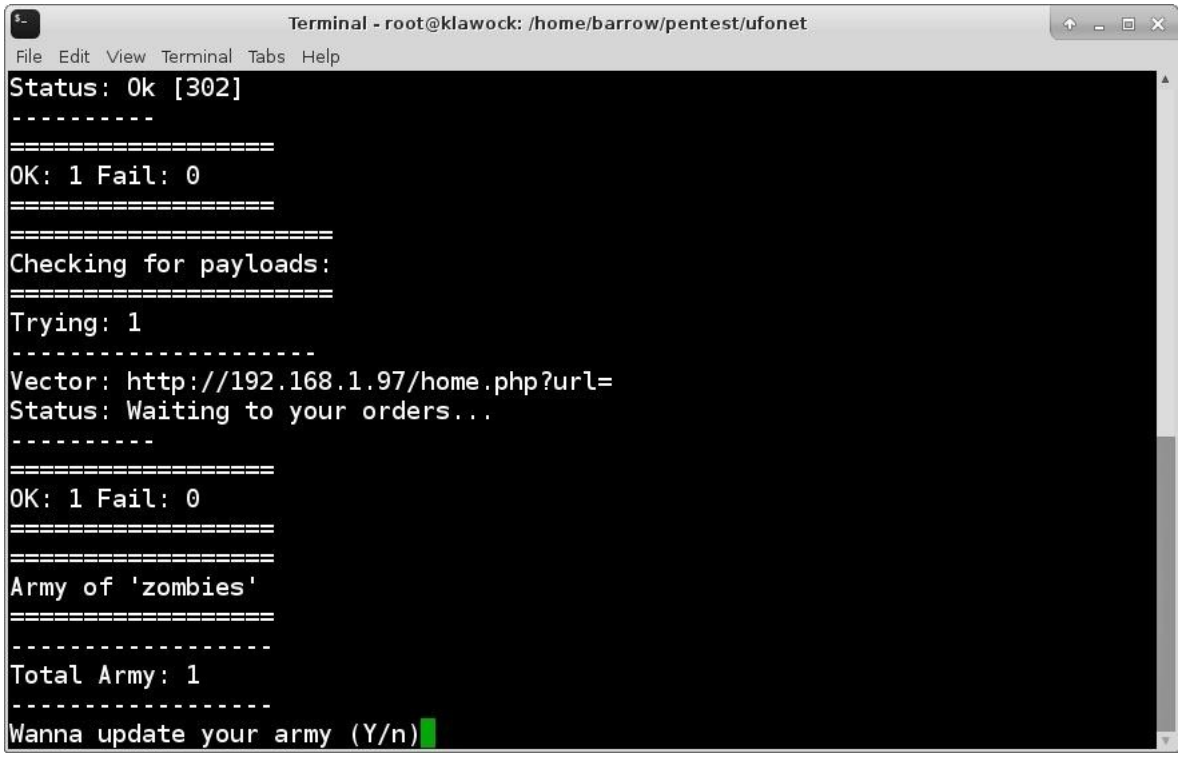


Figura 5.6: [Immb]

Il risultato ci mostra che la VM è vulnerabile.

5.1.9 Ispezione dell'obbiettivo

Ora andiamo ad ispezionare l'obiettivo per un possibile attacco.
UFONet può cercare il file più grande sull'obiettivo, ma concentrarsi su file più grandi non è un passaggio necessario, anche se potrebbe consumare più larghezza di banda dal sito di destinazione, creando un po' più di caos.
Poiché la mia VM è composta da due pagine ospitate, la pagina Apache predefinita e la pagina di reindirizzamento aperta vulnerabile, questo comando non scoprirà nulla di importante.
Tuttavia, in alcuni casi, si può scoprire un file di grandi dimensioni. eseguendo il comando:

```
$ ./ufonet -i http://target.com
```

Al posto di target adiamo ad inserire l'url del nostro obbiettivo.

Vediamo un esempio di ispezione dei file.

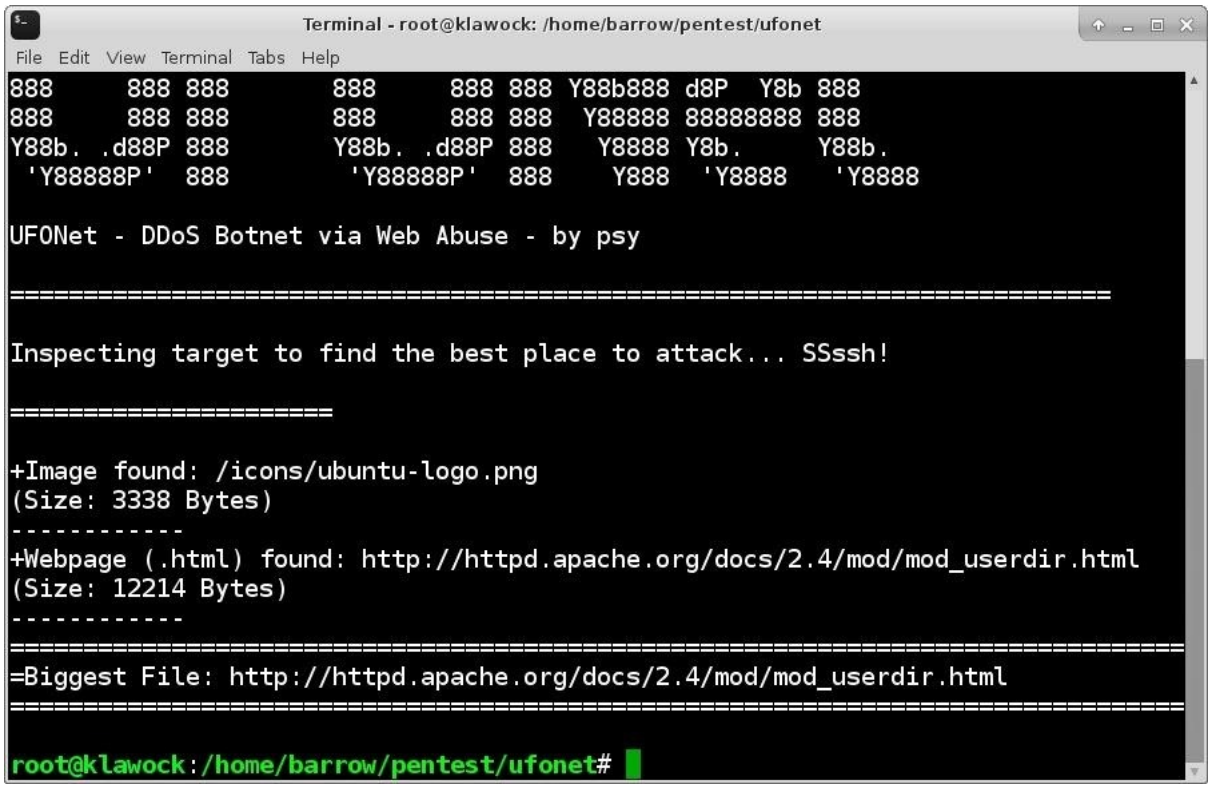


Figura 5.7: [Immc]

Ispezionando il server web vulnerabile. Sembra che il file più grande sul server web sia "ubuntu-logo.png".
Tuttavia, sembra che UFONet abbia seguito un collegamento esterno dal sito.
Se non si fossero lette le informazioni presentate, si sarebbe potuto potenzialmente attaccare il bersaglio sbagliato.
Gli strumenti possono segnalare informazioni errate, quindi è importante prestare attenzione.

5.1.10 Lanciare un attacco

Ora che è tutto pronto si può effettuare l'attacco al nostro bersaglio.
Inserendo:

```
$ ./ufonet -a http://target.com -r 10 "icons/ubuntu-logo.png"
```

In questo comando lanciamo UFONet, e il flag **-a** specifica il bersaglio da attaccare. Il flag **-r** specifica il numero di volte in cui ogni host dovrebbe attaccare. L'opzione **-b** indica il file di attacco sul server di destinazione.

Quando eseguiamo questo comando, UFONet attaccherà il bersaglio 10 volte per ogni zombi.
Se hai un elenco di 100 zombi, lancerebbe 100 zombi per 10 round per un totale di 1.000 richieste al bersaglio. Nello specifico, sta richiedendo il file più grande sul sito "ubuntu-logo.png".

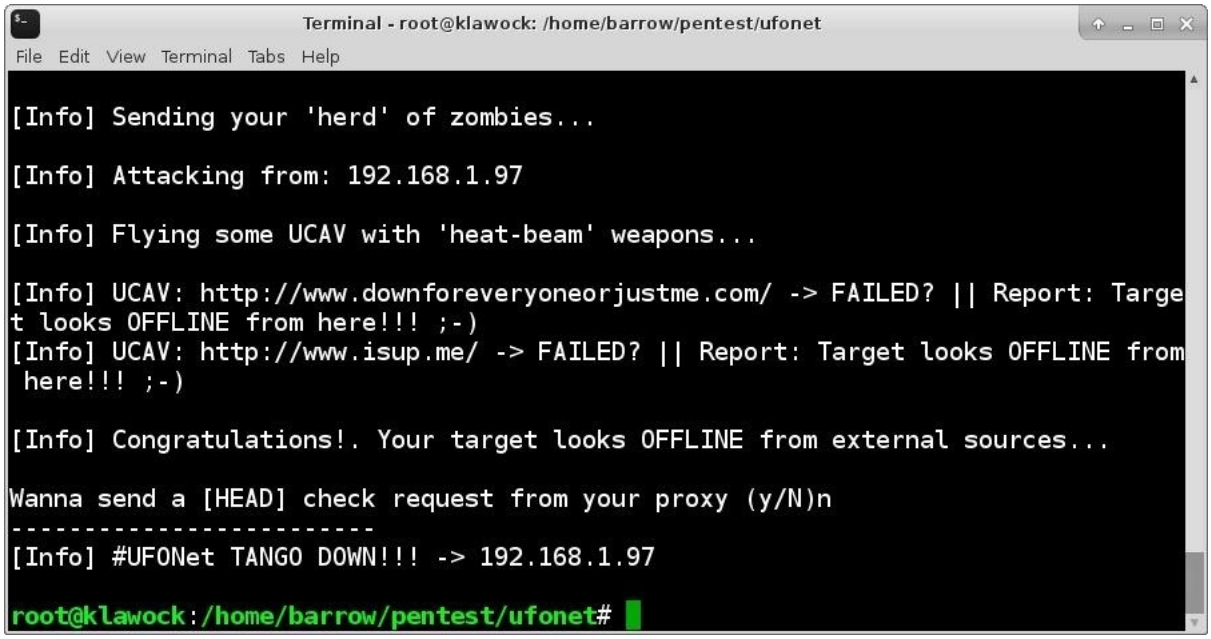


Figura 5.8: [Imma]

In questo caso, il server di destinazione risulta offline.
Ovviamente, l'obbiettivo è ancora online poichè si sta attaccando una VM in locale e siti come isup.me lo rileveranno comunque offline.

5.1.11 La GUI di UFONet

Si può gestire UFONet anche utilizzando un'interfaccia Web.
Lo strumento ha implementato un server web Python connesso al core, per offrire un'esperienza più user friendly.

Per avviare la GUI bisogna inserire il comando:

```
$ ./ufonet --gui
```

Si aprirà una schermata sul browser predefinito con tutte le funzionalità dello strumento.

Figura 5.9: Schermata iniziale di UFONet.

Questa schermata ci permette di cercare informazioni riguardo il nostro target per individuarne informazioni che ci possono aiutare a trovare eventuali vulnerabilità, dandoci anche la possibilità di modificare le impostazioni delle richieste che vengono inviate.

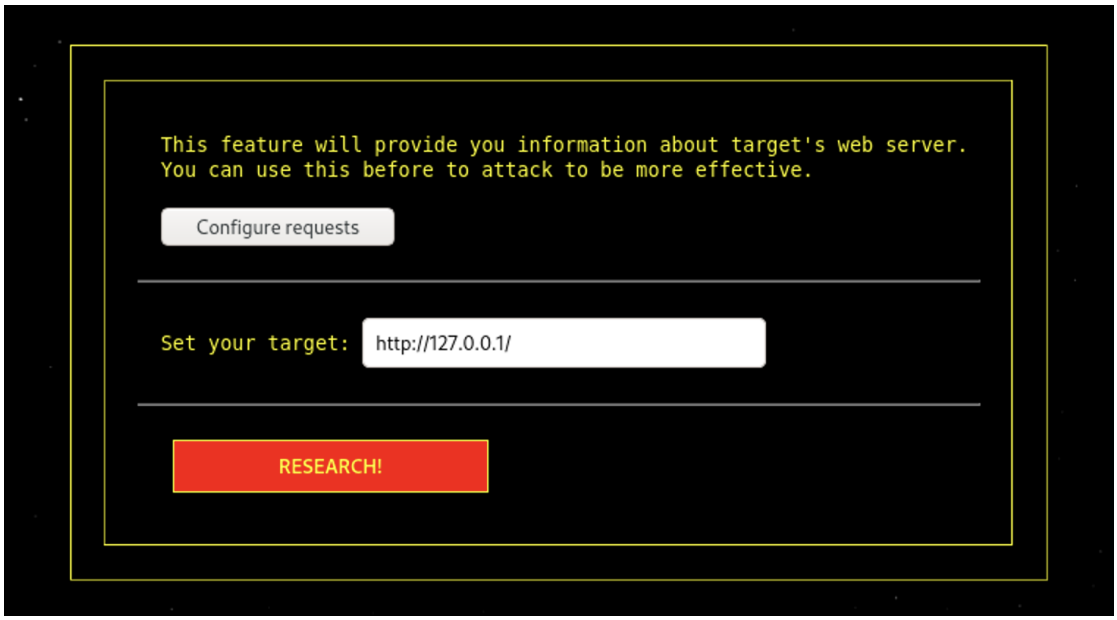


Figura 5.10: Schermata di ricerca informazioni sul target.

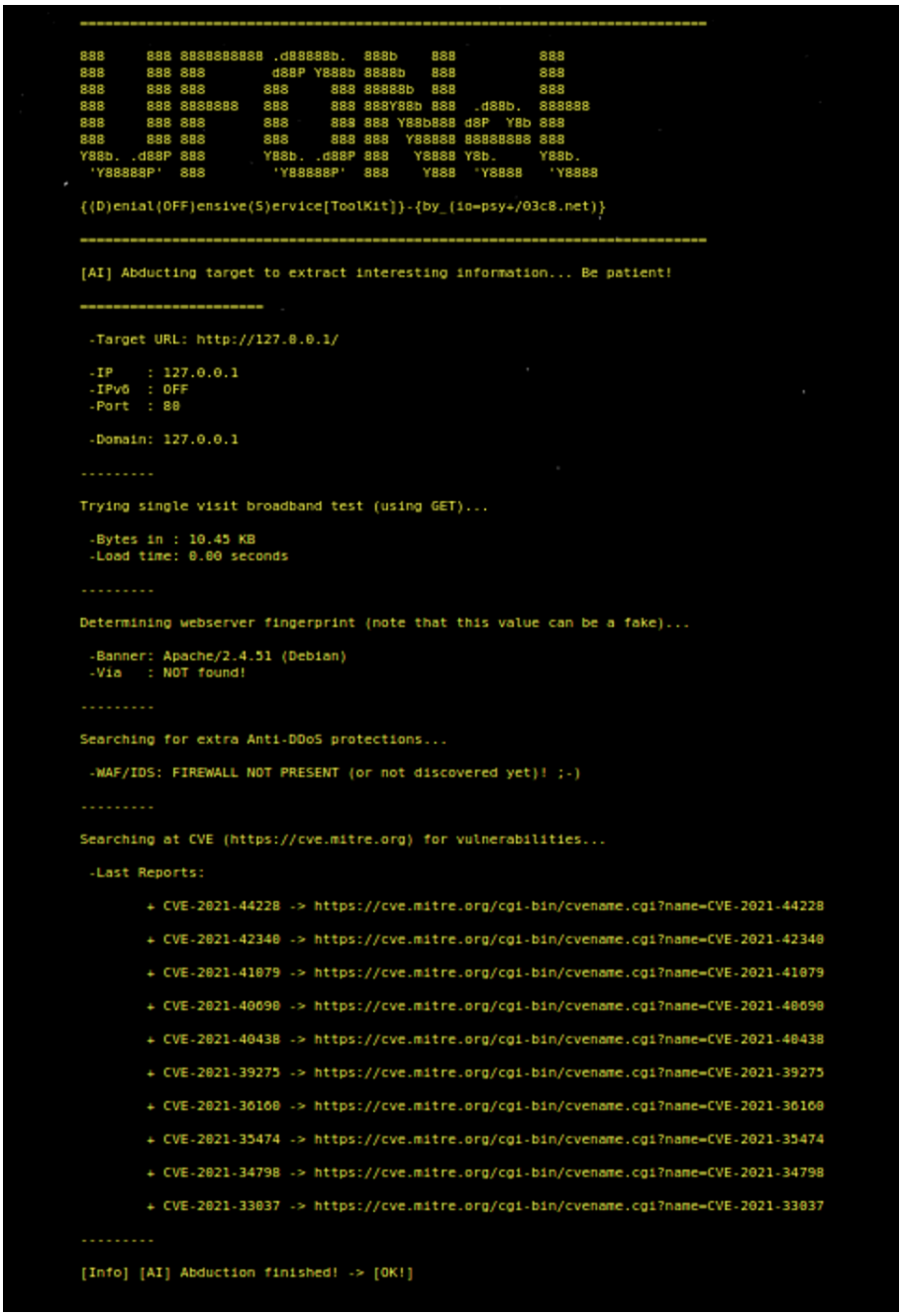


Figura 5.11: Output ricerca informazioni sul target.

Questa schermata ci permette di cercare il file più grande all'interno del server di destinazione dell'attacco, dandoci anche la possibilità di modificare le impostazioni delle richieste che vengono inviate.

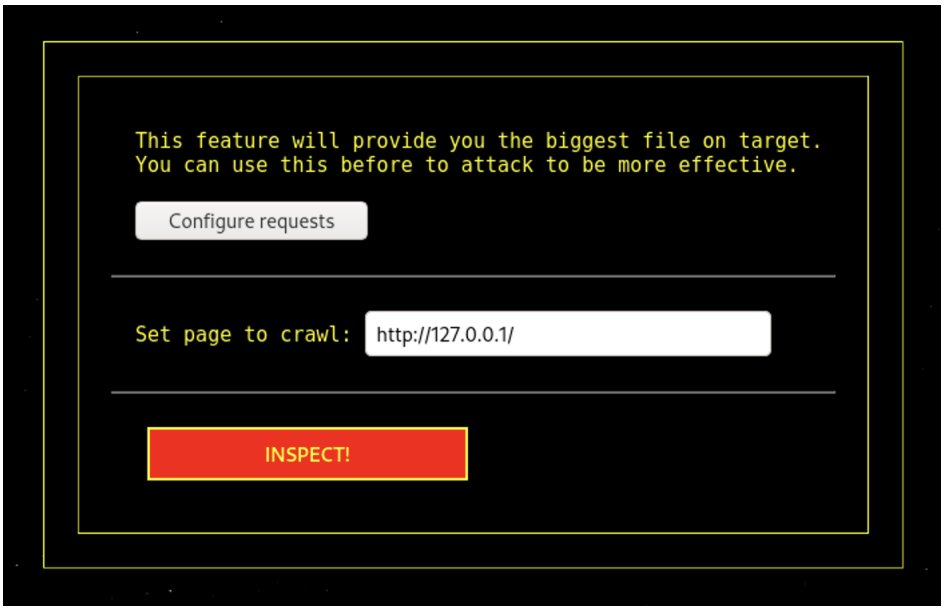


Figura 5.12: Schermata di ricerca del file dove condurre l'attacco.

Figura 5.13: Output ricerca del file.

Andiamo a vedere la schermata per la ricerca degli zombie che ci dà anche la possibilità di testare la botnet offline, consultare la lista degli zombie e generare una mappa sulla locazione degli stessi.

Figura 5.14: Schermata per la ricerca degli zombie.

Dopo aver trovato gli zombie da utilizzare, procediamo alla schermata dell'attacco, dove ci vengono richiesti:

- Il server di destinazione dell'attacco;
- Il file da attaccare;
- Il numero di round per ogni zombie.

In più ci offre la possibilità di sovrascrivere le informazioni di default dell'attacco, per esempio modificando le impostazioni delle richieste, dell'intervallo tra una richiesta e un'altra e di un dork principale per effettuare l'attacco.

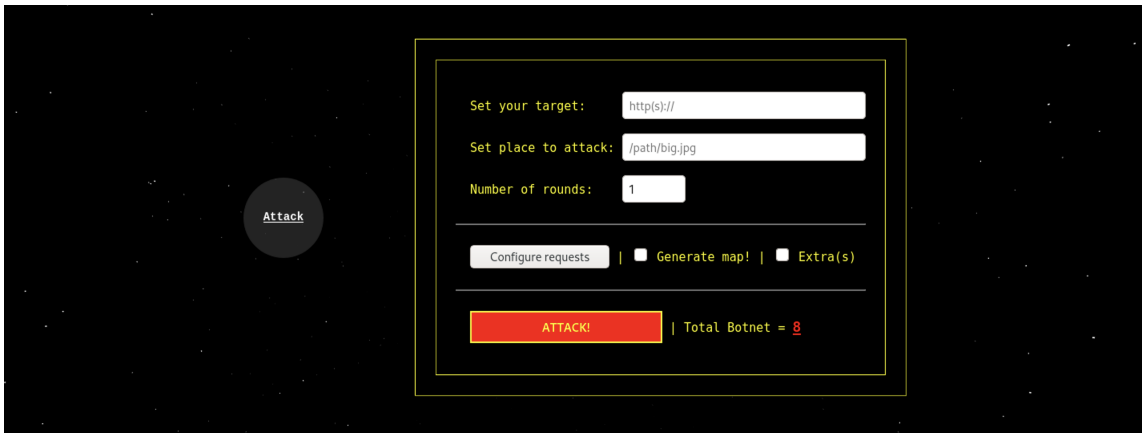
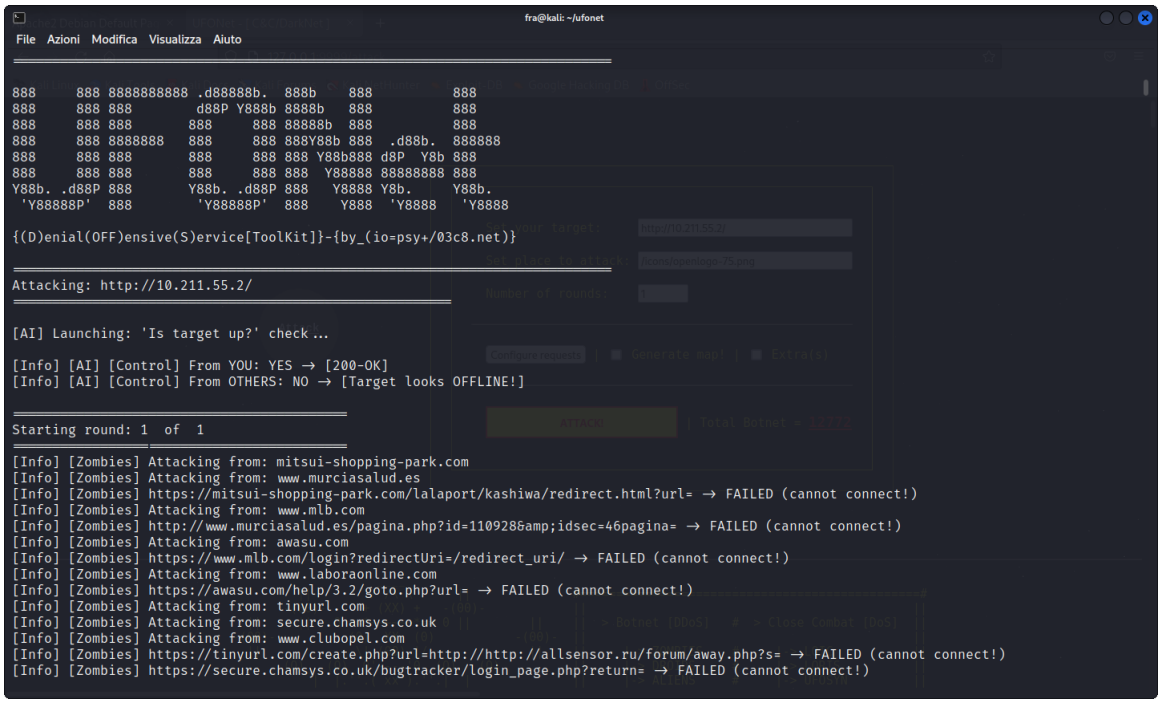


Figura 5.15: Schermata per la gestione dell'attacco.

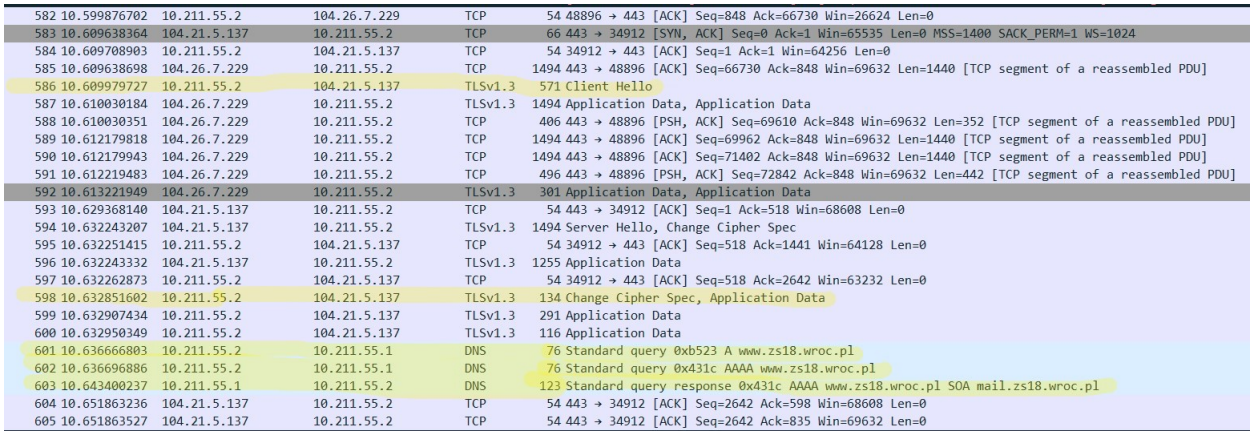
5.1.12 Analisi di un attacco con Ufonet

Vediamo cosa succede durante un attacco che facciamo partire verso il nostro server locale.



Dopo che abbiamo trovato i nostri zombie disponibili per l'attacco, andiamo a lanciarli verso il nostro server e vediamo Ufonet che ci mostra l'elenco degli zombie che stanno attaccando, mostrandoci sia quelli che hanno successo sia quelli che falliscono.

Andiamo a vedere una piccola parte dei pacchetti che vengono inviati.



582	10.599876702	10.211.55.2	104.26.7.229	TCP	54	48896 → 443 [ACK] Seq=848 Ack=66730 Win=26624 Len=0
583	10.609638364	104.21.5.137	10.211.55.2	TCP	66	443 → 34912 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM=1 WS=1024
584	10.609708903	10.211.55.2	104.21.5.137	TCP	54	34912 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0
585	10.609638698	104.26.7.229	10.211.55.2	TCP	1494	443 → 48896 [ACK] Seq=66730 Ack=848 Win=69632 Len=1440 [TCP segment of a reassembled PDU]
586	10.609979727	10.211.55.2	104.21.5.137	TLSv1.3	571	Client Hello
587	10.610030184	104.26.7.229	10.211.55.2	TLSv1.3	1494	Application Data, Application Data
588	10.610030351	104.26.7.229	10.211.55.2	TCP	406	443 → 48896 [PSH, ACK] Seq=69610 Ack=848 Win=69632 Len=352 [TCP segment of a reassembled PDU]
589	10.612179818	104.26.7.229	10.211.55.2	TCP	1494	443 → 48896 [ACK] Seq=69962 Ack=848 Win=69632 Len=1440 [TCP segment of a reassembled PDU]
590	10.612179943	104.26.7.229	10.211.55.2	TCP	1494	443 → 48896 [ACK] Seq=71402 Ack=848 Win=69632 Len=1440 [TCP segment of a reassembled PDU]
591	10.612219483	104.26.7.229	10.211.55.2	TCP	496	443 → 48896 [PSH, ACK] Seq=72842 Ack=848 Win=69632 Len=442 [TCP segment of a reassembled PDU]
592	10.613221949	104.26.7.229	10.211.55.2	TLSv1.3	301	Application Data, Application Data
593	10.629368140	104.21.5.137	10.211.55.2	TCP	54	443 → 34912 [ACK] Seq=1 Ack=518 Win=68608 Len=0
594	10.632243207	104.21.5.137	10.211.55.2	TLSv1.3	1494	Server Hello, Change Cipher Spec
595	10.632251415	10.211.55.2	104.21.5.137	TCP	54	34912 → 443 [ACK] Seq=518 Ack=1441 Win=64128 Len=0
596	10.632243332	104.21.5.137	10.211.55.2	TLSv1.3	1255	Application Data
597	10.632262873	10.211.55.2	104.21.5.137	TCP	54	34912 → 443 [ACK] Seq=518 Ack=2642 Win=63232 Len=0
598	10.632851602	10.211.55.2	104.21.5.137	TLSv1.3	134	Change Cipher Spec, Application Data
599	10.632907434	10.211.55.2	104.21.5.137	TLSv1.3	291	Application Data
600	10.632950349	10.211.55.2	104.21.5.137	TLSv1.3	116	Application Data
601	10.636666803	10.211.55.2	10.211.55.1	DNS	76	Standard query 0xb523 A www.zs18.wroc.pl
602	10.636696886	10.211.55.2	10.211.55.1	DNS	76	Standard query 0x431c AAAA www.zs18.wroc.pl
603	10.643400237	10.211.55.1	10.211.55.2	DNS	123	Standard query response 0x431c AAAA www.zs18.wroc.pl SOA mail.zs18.wroc.pl
604	10.651863236	104.21.5.137	10.211.55.2	TCP	54	443 → 34912 [ACK] Seq=2642 Ack=598 Win=68608 Len=0
605	10.651863527	104.21.5.137	10.211.55.2	TCP	54	443 → 34912 [ACK] Seq=2642 Ack=835 Win=69632 Len=0

Possiamo vedere che vengono fatte delle richieste al server da parte di Ufonet("Client Hello!"), per poi effettuare una serie di query con diversi indirizzi web, cercando di esegure una Open Redirect.

5.2 TOR'S HAMMER

TOR'S HAMMER è un programma scritto in Python per testare attacchi DoS Http di tipo POST lenti.
Originariamente creato da phiral.net, vede il suo rilascio pubblico all'inizio del 2011. La sua particolarità è che il traffico generato può essere mascherato attraverso la rete TOR per essere anonimizzato, assumendo che l'attaccante abbia il servizio di Tor in esecuzione sul local host alla porta 9150.
Riesce a mandare offline la maggior parte dei server apache e IIS non protetti con una sola istanza.
TOR'S HAMMER esegue un attacco DoS usando un classico attacco POST dove i campi Post Html sono trasmessi lentamente nella stessa sessione(i ratei sono scelti randomicamente in un range di 0.5-3 secondi).
Simile al suo predecessore R.U.D.Y, questo attacco fa in modo che il server attenda la fine di incalcolabili tread prima di processarli, questo causa la saturazione delle risorse del server e lo fa entrare in uno stato di fuori servizio per ogni richiesta legittima.

5.2.1 Come utlizzare TOR'S HAMMER

Attraverso il seguente comando si potrà vedere l'help message del programma.

```
$ python2 torshammer.py
```

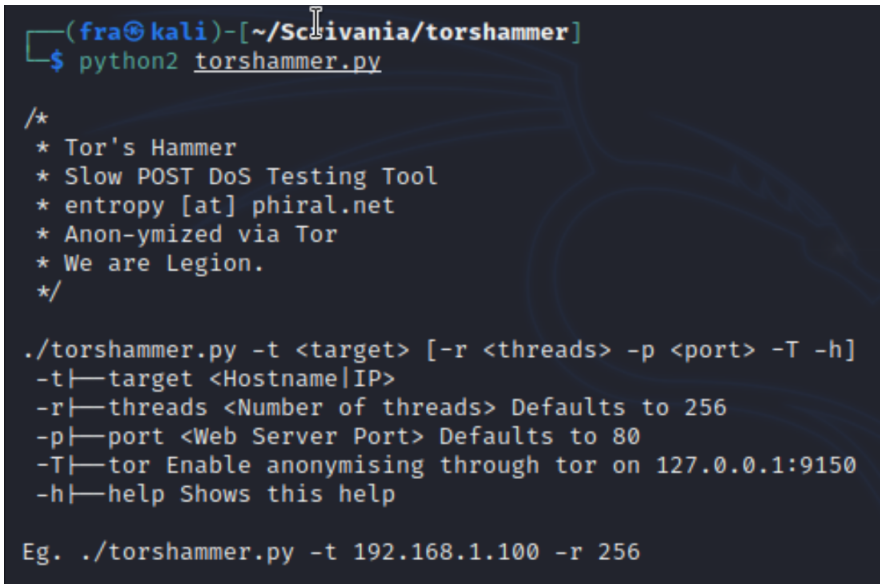


Figura 5.16: Schermata dell'help message di TOR'S HAMMER.

Di seguito vediamo come effettuare un attacco ad un target che, in questo caso di test, sarà un server apache in esecuzione sul localhost, su porta 80 con un numero di tread di 10, non effettuando il routing del traffico sotto la rete tor:

```
$ python2 ./torshammer.py -t 127.0.0.1 -p 80 -r 10
```

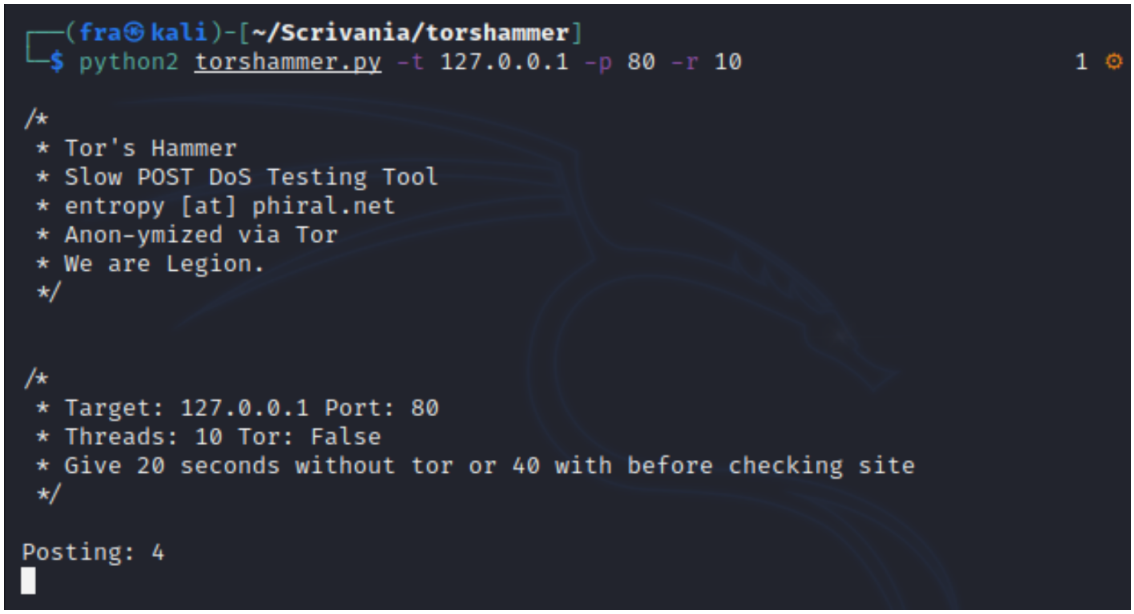


Figura 5.17: Schermata di esecuzione dell'attacco senza routing sotto rete TOR.

Ora replichiamo lo stesso attacco al localhost con lo stesso numero di tread, ma effettuando il routing sotto la rete tor (nota bene che il servizio di tor deve essere in esecuzione sulla macchina attaccante):

```
$ python2 ./torshammer.py -t 127.0.0.1 -p 80 -r 10 -T
```

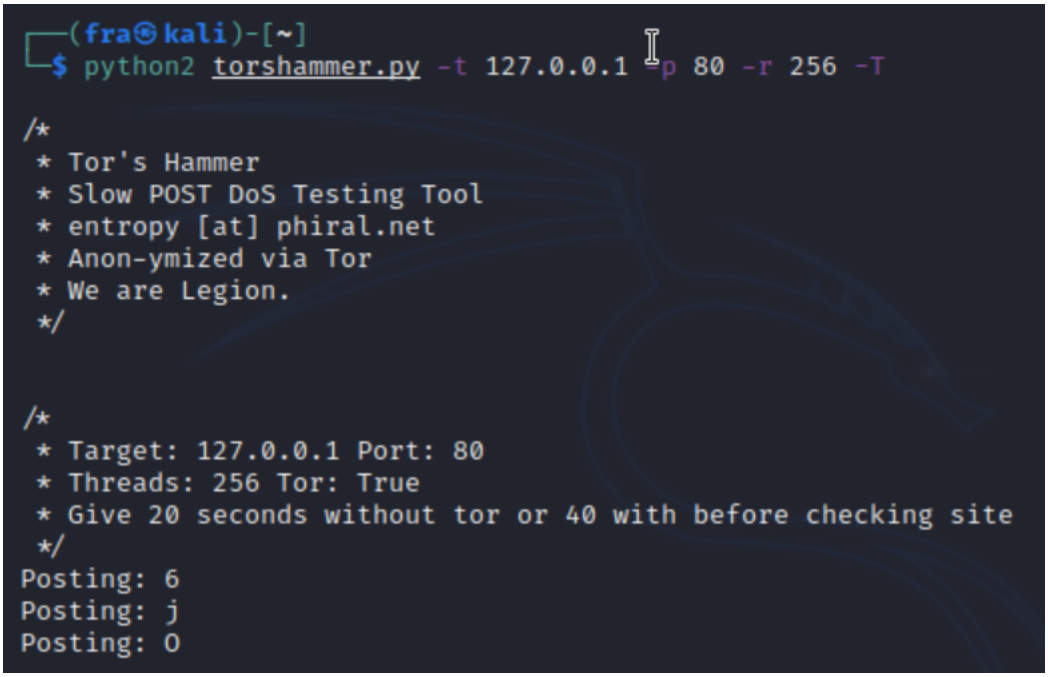


Figura 5.18: Schermata di esecuzione dell'attacco con routing sotto rete TOR.

5.3 SLOWLORIS

SLOWLORIS è un software DoS scritto in linguaggio Perl da Robert Hansen, detto RSnake, che vede la sua prima versione rilasciata il 17 giugno 2009.
Slowloris permette a una singola macchina di occupare le risorse di un server con un'ampiezza di banda minima e limitati effetti collaterali a servizi e porte non coinvolte. Il programma prova a mantenere le connessioni aperte il più a lungo possibile al server web target e lo fa connettendosi al server target e inviandogli richieste parziali. Periodicamente, poi, invierà le successive intestazioni HTTP, senza mai completarne la richiesta. I server attaccati terranno così le connessioni aperte, raggiungendo il loro numero di connessioni disponibili, negando infine ulteriori tentativi di connessione da altri client.

Slowloris è cross-platform, tuttavia se eseguito su Windows, presentava un limite di soli 130 sockets simultanei, per questo ne è consigliato l'uso su sistemi UNIX-based che permettono un numero maggiore di connessioni simultanee verso il server di destinazione dell'attacco. Questo problema è stato risolto successivamente attraverso una versione GUI rielaborata chiamata PyLoris.

Durante le proteste sollevate agli inizi delle elezioni presidenziali Iraniane del 2009, Slowloris si distinse come importante strumento usato per compiere attacchi DoS contro siti appartenenti al governo Iraniano, tra questi gerdab.ir, leader.ir, e president.ir.

5.3.1 Come utlizzare SLOWLORIS

Attraverso il seguente comando si potrà vedere l'help message del programma.

```
$ python3 slowloris.py
```

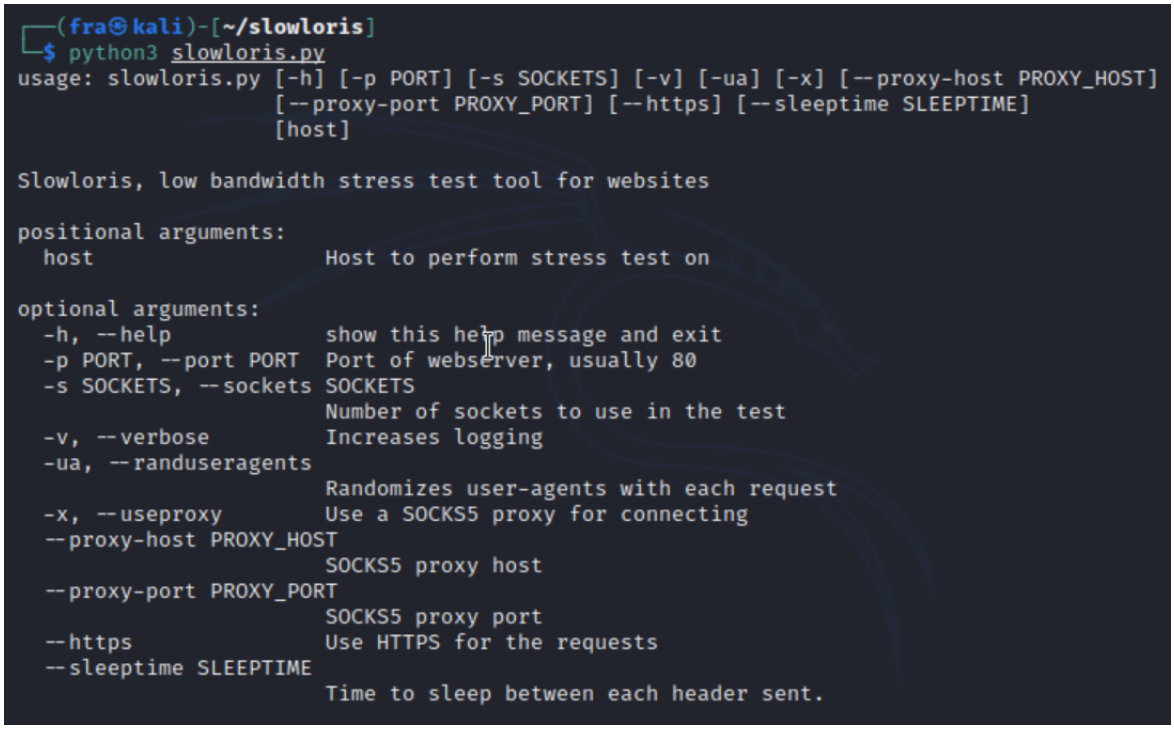


Figura 5.19: Help message di slowloris.

Come unico parametro imperativo, Slowloris richiede il link o l'indirizzo ip dell'host da attaccare, ma offre la possibilità di personalizzare l'attacco con una serie di parametri aggiuntivi opzionali, i più importanti:

- **-p / –port**: permette di impostare un numero di porta specifico per il webserver destinatario dell'attacco, il parametro di default corrisponde alla porta 80;
- **-s / –sockets**: permette di impostare il numero di sockets da utilizzare durante l'attacco;
- **-ua / –randuseragents**: permette di randomizzare l'user agent di ogni richiesta inviata;

- **-x / –useproxy**: permette di usare un proxy SOCKS come quello della rete TOR;
- **–https**: permette di usare solo il protocollo HTTPS per l'invio delle richieste;
- **–sleeptime**: permette di impostare un delay che intercorrerà tra ogni header inviato.

Di seguito vediamo come effettuare un attacco basilare al localhost utilizzando solamente il parametro imperativo, ovvero l'indirizzo ip del webserver:

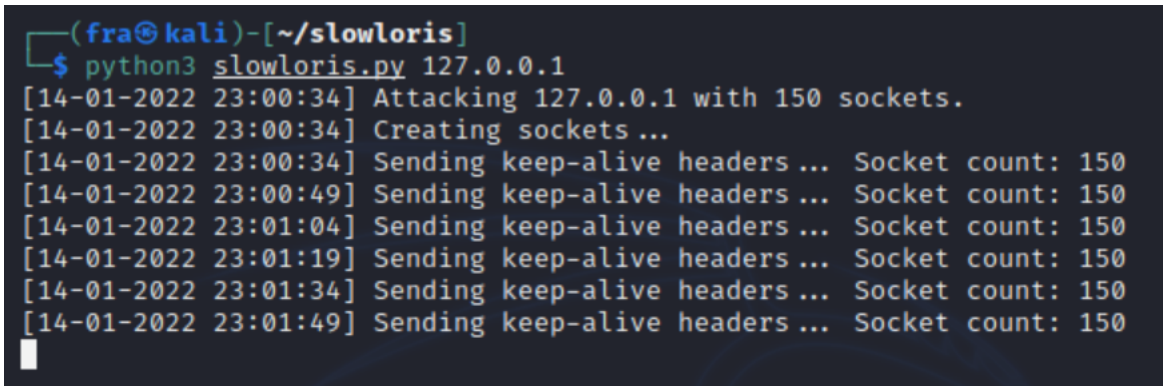


Figura 5.20: Attacco standard con Slowloris senza utilizzare parametri aggiuntivi.

Di seguito, invece, osserviamo un attacco più complesso che coinvolge un numero scelto dall'utente di sockets di 256, la randomizzazzione degli user-agents e uno sleeptime di 10 tra un thread ed un altro:

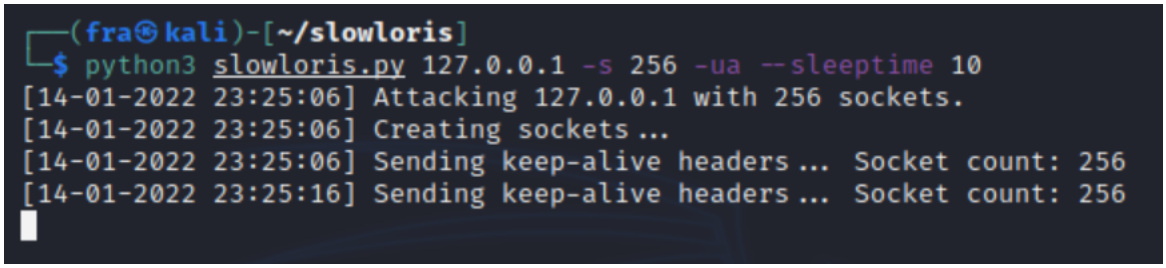


Figura 5.21: Attacco con Slowloris con l'utilizzo di parametri aggiuntivi.

In Questo caso non sarà possibile vedere a schermo direttamente l'applicazione del delay di 10 secondi ma potremmo comunque notare come il numero di sockets sia stato modificato con il numero scelto dall'utente.

5.4 H.U.L.K

H.U.L.K., acronimo di "Http Unbearable Load King", è un programma per effettuare attacchi DoS verso uno web server, ed è stato creato per scopi di ricerca.
Inizialmente sviluppato in python da Barry Shteiman (http://www.sectorix.com/2012/05/17/hulk-web-server-dos-tool/), portato poi in golang da un utente github. La primaria differenza tra la versione in python ed il porting in golang è l'architettura per la concorrenza di processi di golang, le "goroutines". La versione python di H.U.L.K. crea un nuovo thread per ogni connessione all'interno della connection pool, quindi utilizza centinaia e centinaia di threads, la versione in golang, invece, usa le più leggere goroutines che fanno uso di sole poche decine di threads. Questa architettura permette alla versione in golang di ottimizzare le risorse del sistema host e detenere una più larga connection pool rispetto alla versione in python.

5.4.1 Come utilizzare H.U.L.K.

Come dimostrazione useremo la versione originale in python, che possiamo avviare con la seguente sintassi che ci mostrerà una sorta di help message:

```
$ python2 hulk.py
```

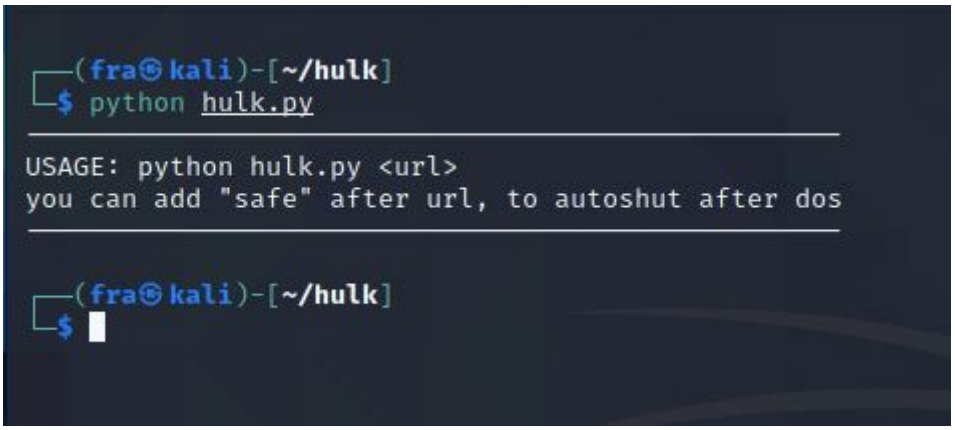


Figura 5.22: Schermata di avvio di H.U.L.K.

Per avviare un attacco, invece, la sintassi sarà la stessa con l'aggiunta del parametro "url", ovvero l'indirizzo IP o Url del web server di destinazione dell'attacco, che in questo caso per fini di test sarà il server apache in localhost:

```
$ python hulk.py 10.0.2.15
```

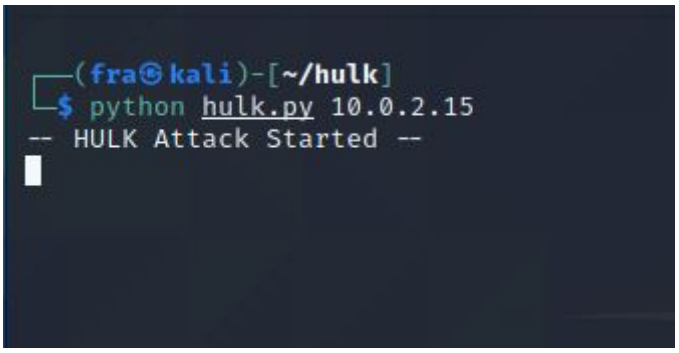


Figura 5.23: Attacco avviato con H.U.L.K.

Inoltre, come specificato nel help message precedente, aggiungendo la parola chiave "safe" alla fine della sintassi dell'attacco, H.U.L.K. si auto stopperà una volta concluso l'attacco:

```
$ python2 hulk.py 10.0.2.15 safe
```

5.5 GoldenEye

GoldenEye è un programma, sviluppato dall'utente github "jseidl" in python 3, per eseguire attacchi DoS HTTP che da la possibilità di aprire diverse connessioni parallele verso il server di destinazione dell'attacco così da testarne un'eventuale compromissione utilizzando un vettore di attacco di tipo "HTTP Keep Alive + No Cache".

5.5.1 Come utlizzare GoldenEye

Attraverso il seguente comando possiamo vedere l'help message del programma:

```
$  python3 goldeneye.py -h
```

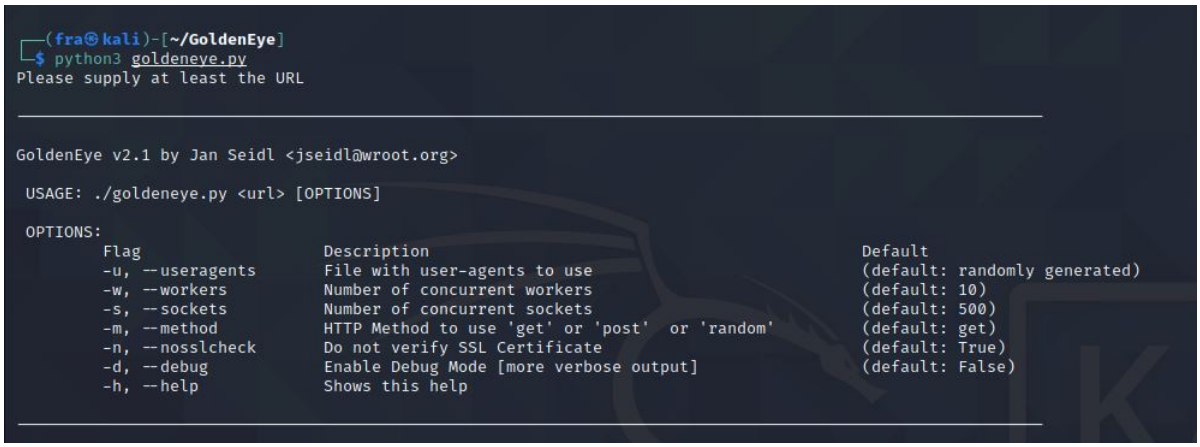


Figura 5.24: L'help message di GoldenEye.

Questa schermata ci mostra la sintassi completa per eseguire un attacco con GoldenEye, avremo infatti bisogno di un parametro imperativo, ovvero "url" e di alcuni opzionali se li desideriamo.
Di seguito la sintassi per lanciare un attacco standard con solo il parametro imperativo dell'url, che per questo esempio puramente a fine di test, sarà google.com:

```
$ python3 goldeneye.py https://www.google.com/
```

In questo esempio, GoldenEye sta attaccando il webserver specificato con le sue impostazioni di default, ovvero 10 workers (processi) per 500 connessioni ciascuno (threads). Se volessimo modificare questi parametri basterebbe cambiarli usando gli appositi flags, per esempio se volessimo lanciare un attacco usando 50 workers con 100 connessioni ciascuno, scriveremo:

```
$ python3 goldeneye.py https://www.google.com/ –w 50 –s 100
```

```
┌──(fra㉿kali)-[~/GoldenEye]
└─$ python3 goldeneye.py https://www.google.com/ -w 50 -s 100

GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

Hitting webserver in mode 'get' with 50 workers running 100 connections each. Hit CTRL+C to cancel.
```

Grazie, infine, agli altri flags addizionali proposti, siamo anche in grado, volendo, di impostare, la rotazione degli user-agent delle richieste, il metodo con cui verranno inviate, un debug più approfondito e la possibilità di saltare il check di validità del certificato ssl.

```
┌──(fra㉿kali)-[~/GoldenEye]
└─$ python3 goldeneye.py https://www.google.com/

GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

Hitting webserver in mode 'get' with 10 workers running 500 connections each. Hit CTRL+C to cancel.
0 GoldenEye strikes hit. (63 Failed)
0 GoldenEye strikes hit. (649 Failed)
0 GoldenEye strikes hit. (1552 Failed)
0 GoldenEye strikes hit. (1669 Failed)
0 GoldenEye strikes hit. (2216 Failed)
0 GoldenEye strikes hit. (2389 Failed)
0 GoldenEye strikes hit. (2804 Failed)
0 GoldenEye strikes hit. (3151 Failed)
0 GoldenEye strikes hit. (3770 Failed)
0 GoldenEye strikes hit. (4054 Failed)
0 GoldenEye strikes hit. (4518 Failed)
0 GoldenEye strikes hit. (4971 Failed)
0 GoldenEye strikes hit. (5465 Failed)
0 GoldenEye strikes hit. (5615 Failed)
0 GoldenEye strikes hit. (5935 Failed)
0 GoldenEye strikes hit. (6171 Failed)
0 GoldenEye strikes hit. (6301 Failed)
0 GoldenEye strikes hit. (6481 Failed)
0 GoldenEye strikes hit. (6655 Failed)
0 GoldenEye strikes hit. (6662 Failed)
0 GoldenEye strikes hit. (6662 Failed)
0 GoldenEye strikes hit. (6662 Failed)
0 GoldenEye strikes hit. (6662 Failed)
0 GoldenEye strikes hit. (6663 Failed)
0 GoldenEye strikes hit. (6663 Failed)
```

Figura 5.25: Attacco in corso.

Ora vediamo i pacchetti inviati da GoldenEye durante l'attacco usufruendo di Wireshark.

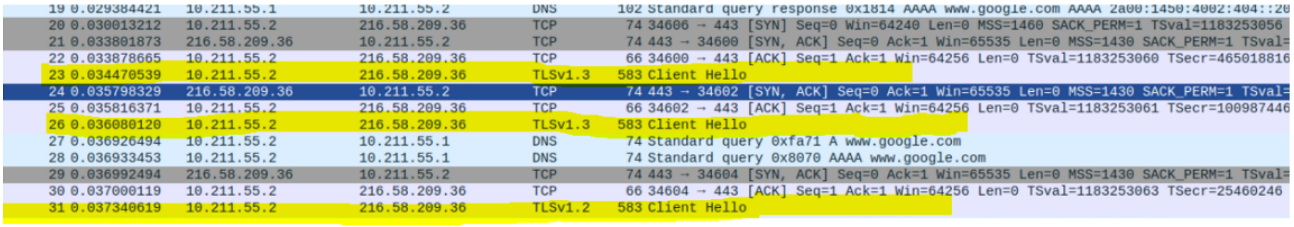


Figura 5.26: Pacchetti durante l'attacco di GoldenEye.

Attaccando un server di Google, possiamo notare che il software ne effettua inizialmente un collegamento TCP, per poi mandare i pacchetti contenenti l'attacco al server vittima, con l'intestazione "Client Hello".
Solamente che essendo il server Google molto protetto, contrasta l'attacco rendendolo inefficace.

5.6 THC-SSL-DOS

THC-SSL-DOS è un programma per eseguire attacchi dos sfruttando una vulnerabilità ad alto rischio del protocollo SSL, sviluppato dal collettivo dal nome "The Hacker's Choice" o "THC". THC-SSL-DOS, così come altri attacchi di tipo "low and slow", richiede solo un quantitativo basso di pacchetti per causare il "denial of service" di uno webserver.
L'attacco inizia stabilendo un regolare handshake SSL per poi richiedere continuamente la ri-negoziazione della chiave di crittografia, questo permette di far esaurire velocemente le risorse del server destinatario dell'attacco.

Come utilizzare THC-SSL-DOS

Con il seguente comando possiamo vedere l'help message del programma:

```
$ thc-ssl-dos
```

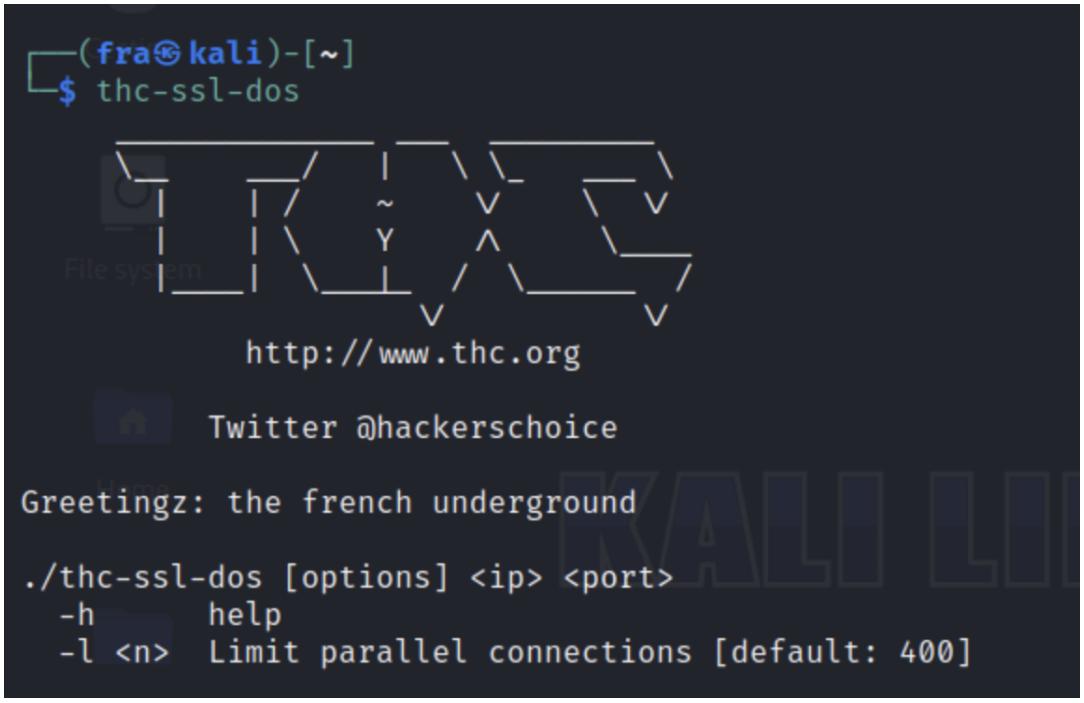


Figura 5.27: Schermata delle opzioni.

La schermata dell'help message si presenta quindi molto semplice, mostrando i parametri imperativi, ovvero indirizzo IP e porta del server di destinazione dell'attacco, e quelli opzionali, come il flag "-l" che ci permette eventualmente di impostare un numero di connessioni parallele a nostra scelta.
Se volessimo quindi lanciare un attacco contro il nostro server apache in localhost sulla porta 443 (SSL), quello che dovremo fare sarà:

```
$ thc-ssl-dos -100 192.168.1.208 443 accept
```

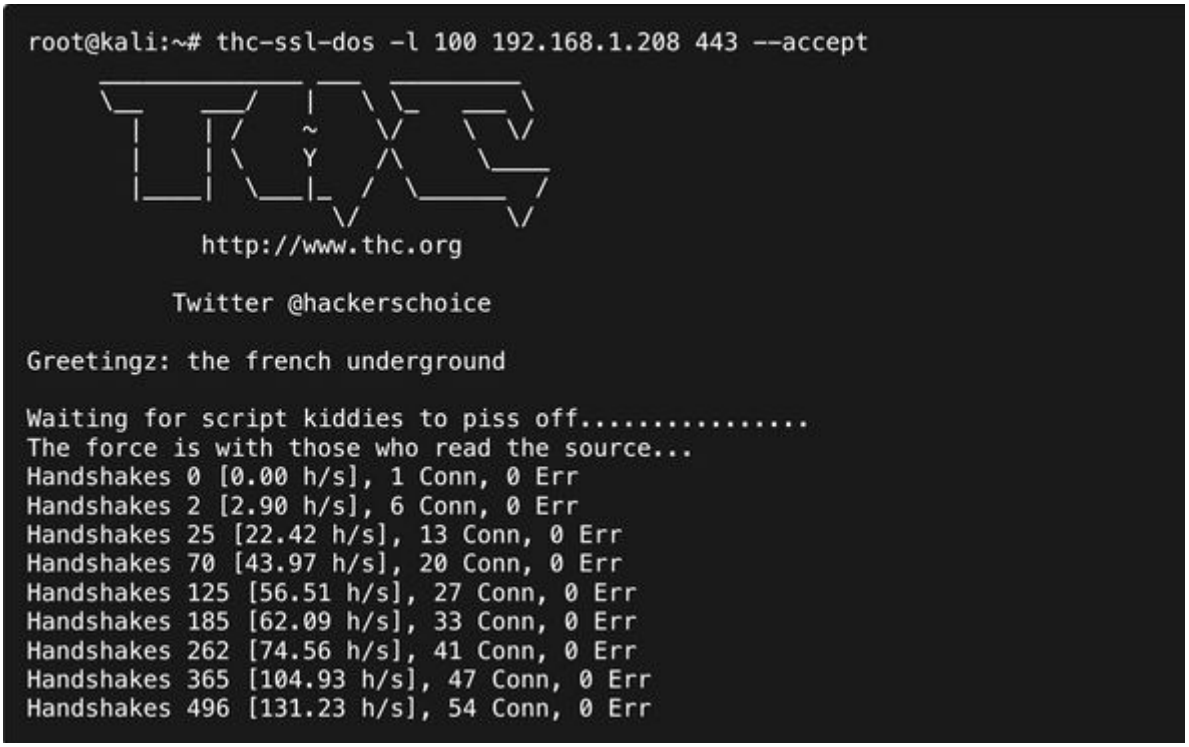


Figura 5.28: THC-SSL-DOS preparazione attacco.

(N.B.: Si include "–accept" alla fine dei parametri imperativi per confermare che abbiamo il permesso di performare quel tipo di attacco sul server di destinazione, in quanto THC-SSL-DOS viene sviluppato per scopi di test.) Così facendo, il programma effettuerà l'handshake con il target e darà inizio alla loop di rinegoziazione della chiave di crittografia, mandandolo offline, il tutto limitando le connessioni parallele ad un massimo di 100.

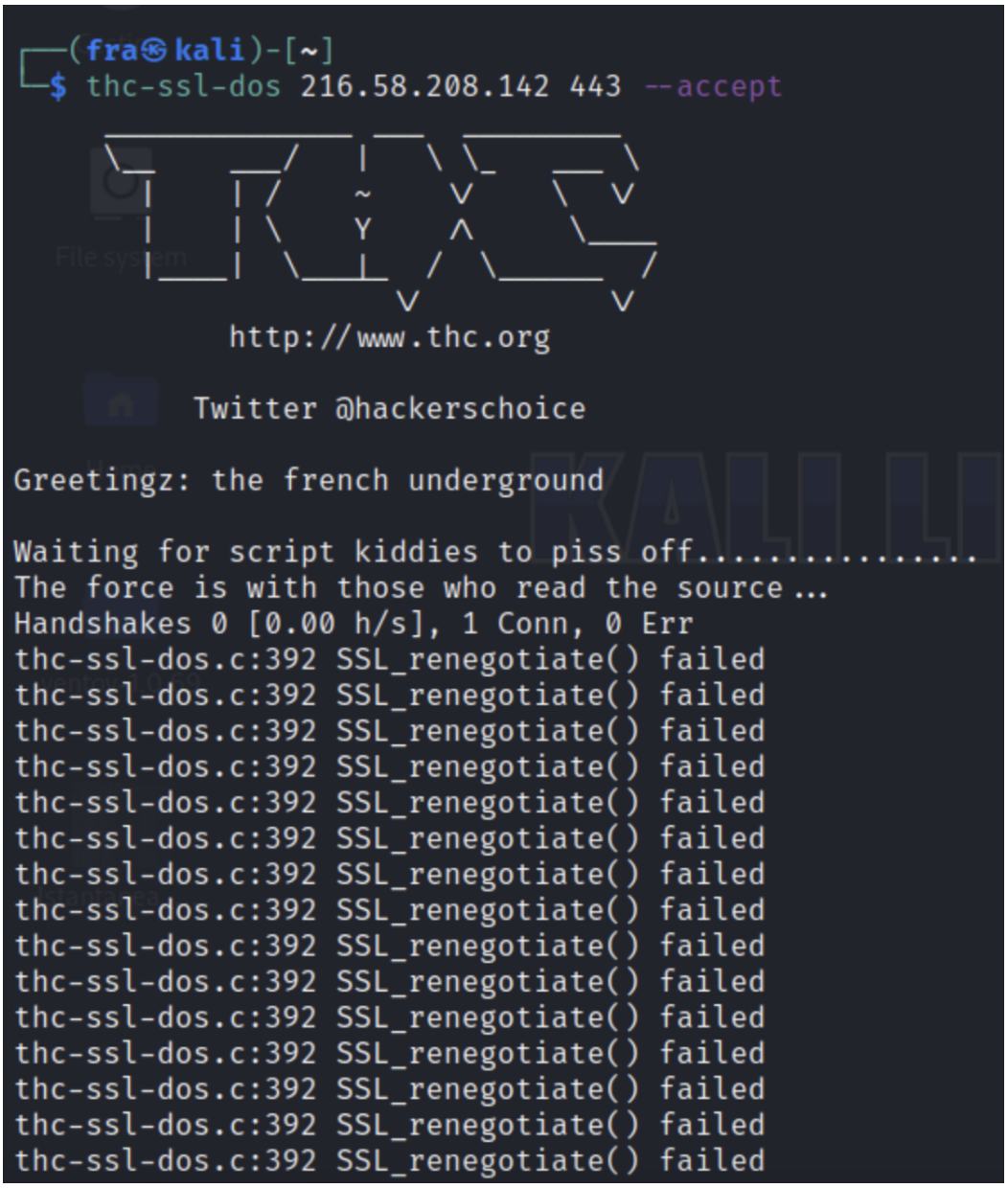


Figura 5.29: THC-SSL-DOS attacco in corso.

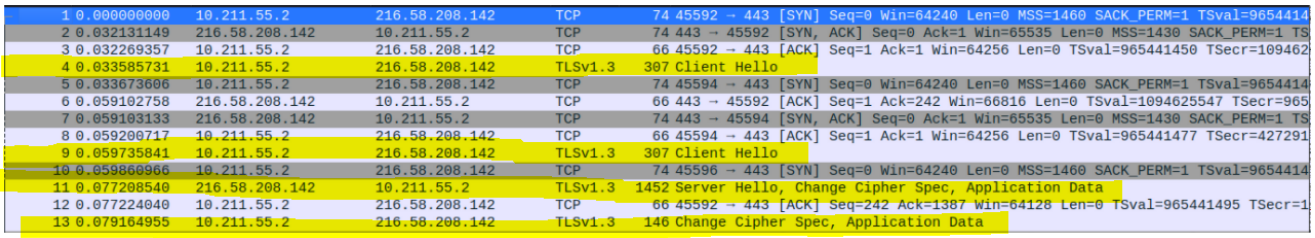


Figura 5.30: Pacchetti inviati da THC-SSL-DOS

Attaccando un server di Google possiamo vedere i pacchetti che vengono inviati anche se l'attacco non va a buon fine in quanto Google ne blocca l'attacco. Vediamo il software che cerca di mandare pacchetti attacco cercando di rinegoziare la chiave di crittografia, provando diverse volte.

5.7 LOIC

LOIC (un acronimo per Low Orbit Ion Cannon, in italiano: cannone a ioni in orbita bassa) è un software open source, scritto in C, successivaente rilasciato anche in versione web(Low Orbit Web Cannon) e javascrip(JSLoic) per generare grandi quantità di traffico di rete (richieste) verso un sistema target e testare la sua risposta sotto carico.
LOIC è stato sviluppato inizialmente da Praetox Technologies, ma successivamente è stato distribuito come software di pubblico dominio.

LOIC esegue un attacco TCP, UDP o HTTP DoS abbastanza semplice e, se utilizzato da più persone come avviene normalmente, un attacco DDoS. La sua popolarità è cresciuta in quanto una versione modificata fù utilizzata dal collettivo Anonymous con un canale di controllo basato su IRC che consente alle persone di unirsi a botnet volontarie e attaccare singoli obiettivi.
Difatti una singola utenza LOIC non sarebbe in grado di generare un impatto significativo di traffico sul target, quindi perchè un attacco abbia successo migliaia di utenti si dovrebbero coordinare simultaneamente verso lo stesso target.

5.7.1 La GUI di LOIC

Per usare LOIC, un attaccante come prima cosa avvia l'applicazione, inserisce l'Url o l'indirizzo IP del target e dispone il tipo di attacco tra TCP, UDP o HTTP flood.
Le modalità TCP e UDP mandano messaggi e pacchetti alle porte selezionate del target mentre l'HTTP flood manda infinito numero di richieste GET.

Una volta lanciato LOIC effettua richieste di connessione multiple al server target, per poi mandare una serie continua di messaggi finché il server stesso non si sovraccarica e non può più rispondere alle richieste legittime.

LOIC è facilmente reperibile da chiunque e ciò rende possibile l'organizzazione di attacchi coordinati, in più è di semplice utilizzo per chiunque a dispetto delle competenze o dell'esperienza.

C'è da dire che gli utilizzatori di LOIC non hanno modo di mascherare il loro traffico sotto proxy, e questo ha come risultato che i loro indirizzi IP sono completamente visibili al target tale da rendere semplice il loro tracciamento.

Oggi con il quasi nullo verificarsi di attacchi DoS a favore dei più potenti DDoS, software come LOIC sono in disuso.

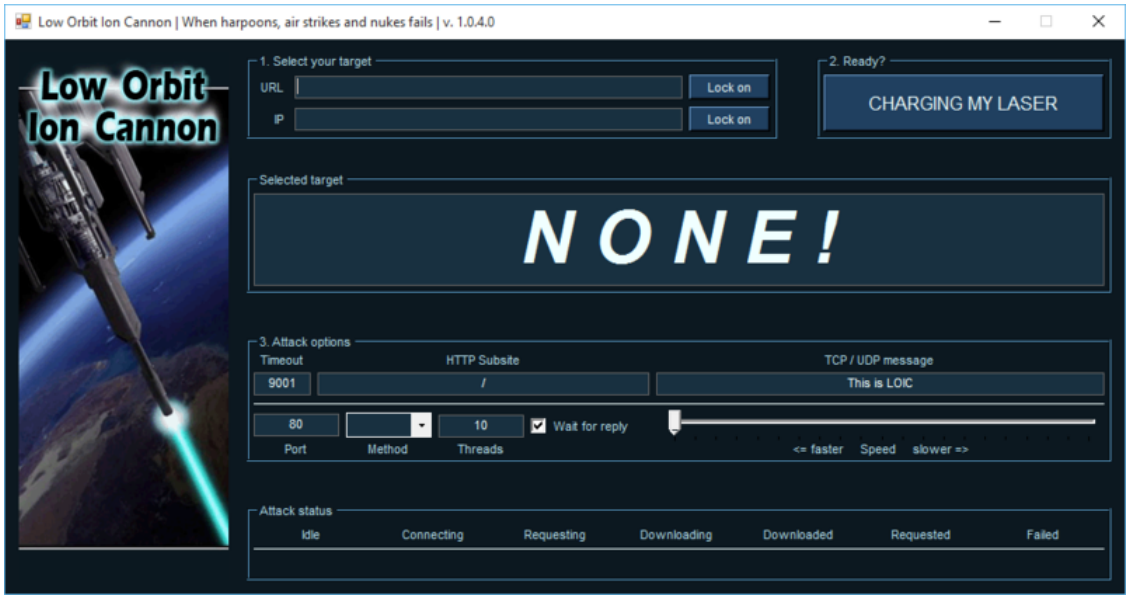


Figura 5.31: [Loi]

5.7.2 Attacchi storici effettuati attraverso LOIC

LOIC è stato utilizzato da **Progetto Chanology**, un gruppo derivato dagli Anonymous (è un movimento di protesta contro le pratiche della Chiesa di Scientology), per attaccare il sito web di Scientology, e dagli stessi Anonymous per attaccare con successo il sito web della **Recording Industry Association of America** nell'ottobre del 2010, e di nuovo durante l'**operazione Payback** nel dicembre 2010 per attaccare i siti web di società e organizzazioni che hanno osteggiato **WikiLeaks**(è un'organizzazione internazionale senza scopo di lucro che riceve in modo anonimo, grazie a un contenitore protetto da un potente sistema di cifratura, documenti coperti da segreto come quelli di Stato, militare, industriale e bancario, successivamente li carica sul proprio sito web).
La versione di LOIC usata in questi attacchi conteneva la cosiddetta modalità HIVEMIND, usava server IRC(Internet Relay Chat) per dirottare il traffico fittizio generato dagli utenti tale da creare una botnet e controllare gli attacchi senza una precedente organizzazione.
Di seguito un esempio di come ci si collegava ad un server IRC per programmare un attacco HIVEMIND.

```
LOIC.exe /hivemind irc.server.address 1234 #secret
```

In data 21 aprile 2011 un attacco LOIC è stato lanciato contro la **Sony** inizialmente creduto essere attribuibile al gruppo Anonymous a causa delle vicende giudiziali portate avanti da Sony contro **GeoHot**(è un informatico statunitense, e famoso hacker che ha violato i sistemi di sicurezza Apple e di Sony) ed altri hacker coinvolti nella scoperta del jailbreak della PlayStation 3 di proprietà Sony

ma poi smentito dallo stesso gruppo.

In data 19 gennaio 2012 il gruppo di hacker Anonymous ha lanciato un attacco LOIC che ha colpito i siti web delle organizzazioni che negli USA difendendo i diritti d'autore per la cinematografia e la musica.
L'attacco è stato effettuato in seguito alla chiusura del sito Megaupload da parte dell'FBI.

5.8 ZEUS

Figura 5.32: [Logb]

ZeuS è stato un software botnet sviluppato nel 2007, e funziona sia per effettuare un attacco DDoS sia come Trojan.
Ad oggi il software è reperibile per mezzo di alcune repository su GitHub ma non viene più aggiornato da circa una decina di anni quindi, pressoché inutile allo scopo, ma è importante parlarne poichè è stato storicamente un software molto utilizzato per questo tipo di attacchi.

E' stato utilizzato per attaccare il Dipartimento dei trasporti degli Stati Uniti, tuttavia, non si è diffuso in modo significativo fino a marzo 2009.
Gli attacchi che hanno comportato l'uso di Zeus si sono verificati per tutto il 2010, incluso un attacco dell'ottobre 2010 da parte di una grande organizzazione facente parte della criminalità organizzata che tentò di rubare oltre 70 milioni di dollari a diversi individui negli Stati Uniti con computer infettati da Zeus.
Nel maggio 2011 il codice sorgente della versione utilizzata allora di Zeus (v2) è trapelato, portando alla creazione di vari bot personalizzati basati su Zeus.

Alla fine del 2010, un certo numero di fornitori di sicurezza Internet, tra cui **McAfee**(è una società americana di softwere per la sicurezza informatica) e **Internet Identity**(era una società che si occupava di sicurezza e che ora raccoglie dati sugli attacchi informatici), ha affermato che il creatore di Zeus aveva detto che si sarebbe ritirato e aveva dato il codice sorgente e i diritti per vendere Zeus al suo più grande concorrente, il creatore del trojan SpyEye.

L'FBI nel periodo degli attacchi di Zeus, ha rilasciato un manifesto che spiegava lo schema e il funzionamento dell'attacco.

5.8.1 Cosa fà Zeus ai computer che colpisce

Zeus può fare diverse cose ma in realtà a due funzionalità principali quando và ad infettare un computer.
Innanzitutto, crea una botnet.
Una botnet consente al proprietario di raccogliere enormi quantità di informazioni o eseguire attacchi su larga scala.
Zeus funge anche da Trojan per servizi finanziari progettato per rubare credenziali bancarie dalle macchine che infetta. Lo fa attraverso il monitoraggio e il keylogging del sito Web, in cui il malware riconosce quando l'utente si trova su un sito Web bancario e registra i tasti premuti utilizzati per accedere. Ciò significa che il Trojan può aggirare la sicurezza in atto su questi siti Web.

In origine, il Trojan colpiva solo i computer che eseguivano versioni del sistema operativo Microsoft Windows, ma alcune versioni più recenti del malware sono state trovate su dispositivi mobili Symbian, BlackBerry e Android.

5.8.2 Come Zeus infetta i computer

Il virus Zeus ha due principali metodi di infezione:

- Messaggi di spam.
- Drive-by Download.

I messaggi di spam spesso arrivano sotto forma di e-mail, ma ci sono state campagne di social media progettate per diffondere il malware attraverso messaggi e post sui siti di social media. Una volta che gli utenti fanno clic su un collegamento nell'e-mail o nel messaggio, vengono indirizzati a un sito Web che installa automaticamente il malware. Poiché il malware è abile nel rubare le credenziali di accesso, a volte può essere configurato per rubare le credenziali di posta elettronica e dei social media, consentendo alla botnet di inviare messaggi di spam da fonti attendibili e ampliare notevolmente la sua portata.

I download drive-by si verificano quando gli hacker sono in grado di corrompere siti Web legittimi, inserendo il loro codice dannoso in un sito Web di cui l'utente si fida. Il malware si installa quindi quando l'utente visita il sito Web o quando l'utente scarica e installa un programma innocuo.

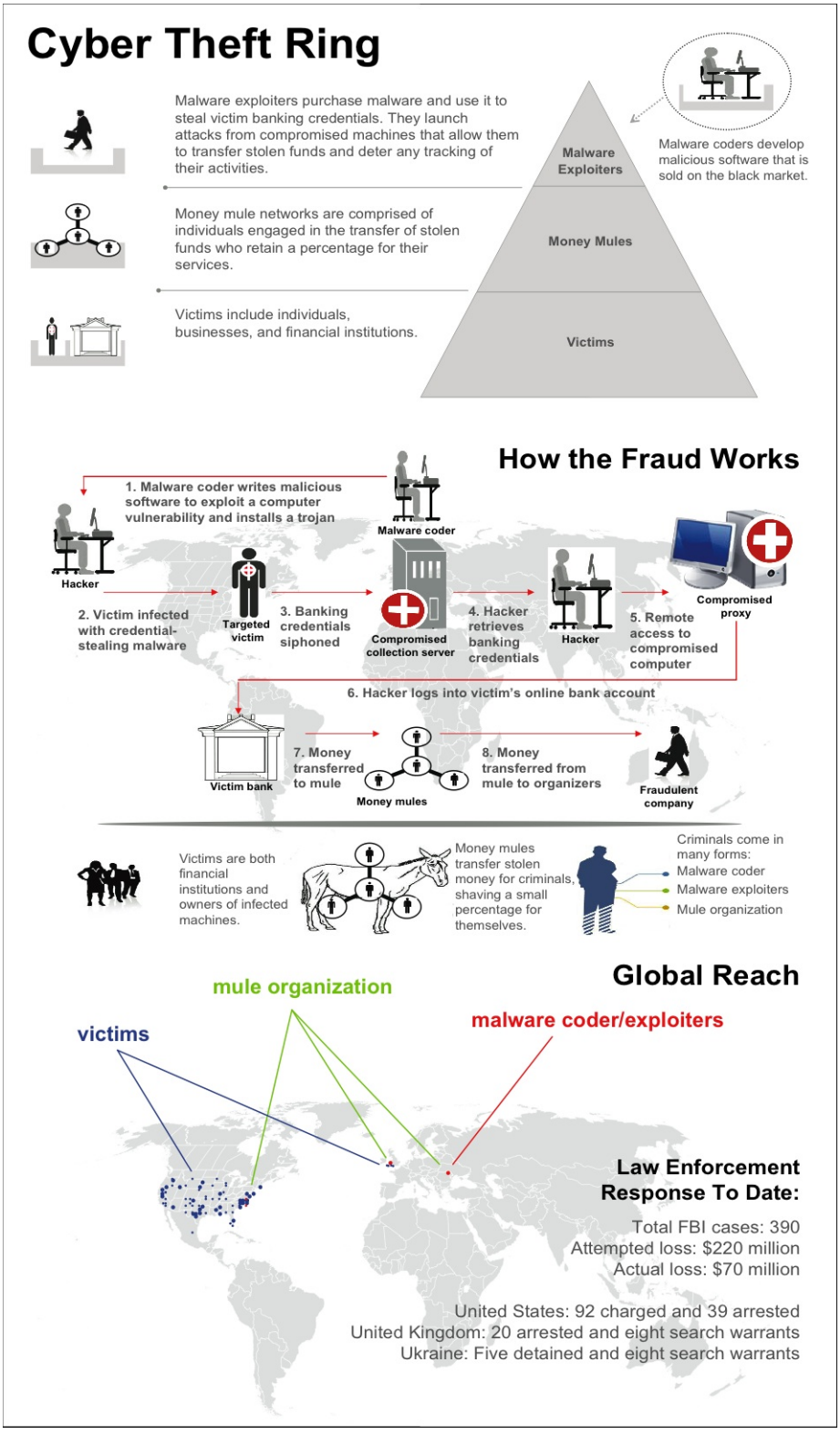


Figura 5.33: Volantino FBI per spiegare la nascita e la distribuzione di un malware[Schb]

5.9 Tabella di confronto dei software

Andiamo a confrontare i vari software per stabilire quale tra essi ‘e il migliore per eseguire un attacco.

	Piattaforme supportate	**Linguaggio**	**Tipo di attacco**	**Stato sviluppo**	**Reperibilità pacchetti**
Ufonet	Windows Linux Mac	python	DDoS vulnerabilità open-redirect	attivo	si
Tor's Hammer	Windows Linux Mac	python	DoS sotto rete tor	attivo	no
Slowloris	Windows Linux Mac	python	DoS	attivo	no
LOIC	Windows	C#	DoS	inattivo	no
ZEUS	Windows	C++	DDoS Trojan botnet	inattivo	no
HULK	Windows Linux Mac	python	DoS	attivo	no
GoldenEye	Windows Linux Mac	python	DoS	inattivo	si
THC-SSL-DOS	Windows Linux Mac	C	DoS vulnerabilità SSl	inattivo	si

	Voto	Motivazione
Ufonet	6/10	E' open source ed offre un ottimo spunto per imparare il funzionamento di una botnet.
Tor's Hammer	5/10	Concettualmente buono per via del routing del traffico sotto rete tor ma praticamente inefficace.
Slowloris	7/10	Attualmente inutilizzabile ma con una storia alle spalle di protesta contro le elezioni presidenziali iraniane del 2009.
LOIC	8/10	Attualmente fruibile sul piano teorico ha le sue radici di utilizzo da parte di Anonimus in più occasioni anche verso target importanti come Sony.
ZEUS	9/10	Seppur attualmente inutilizzabile, ZEUS si è fatto artefice di numerosi attacchi e di un gran numero di macchine infettate dal suo trojan.
HULK	6/10	Nonostante la sua prima versione in python non avesse una buona gestione della connection pull, la nuova versione di Golang risolve questo problema con una nuova gestione dei treads. Tuttavia rimane inutilizzabile.
GoldenEye	6/10	Attualmente fruibile nonostante non di grande efficacia, al contrario della maggior parte degli strumenti DoS, utilizza un vettore di attacco di tipo HTTP Keep Alive + NoCache.
THC-SSL-DOS	7/10	Attualmente fruibile, seppur indirizzato principalmente a web servers privi di protezioni, THC-SSL-DOS ha la sua forza nello sfruttamento di una vulnerabilità SSL per la rinegoziazione dell chiavi di crittografia in handshake, che lo rende ancora potenzialmente pericoloso.

Molti di questi software, al giorno d'oggi sono obsoleti e quelli che funzionano non hanno un grande impatto a livello di prestazioni, in quanto erano efficaci nel periodo in cui sono stati creati e ad oggi ci sono le difese per queste tipologie di attacchi.

Se dovessimo analizzare questi software, il più performante sarebbe stato ZEUS, in quanto ha un sistema di attacco molto potente che ha causato molti danni in passato, ma al momento non è più reperibile come software utilizzabile.

Per avere un attacco DDoS funzionante ed efficace dovremmo inoltrarci nel DarkWeb dove ci sono software che effettuano attacchi più sofisticati e dannosi.

6. Conclusioni

In un epoca sempre più caratterizzata dalla costante presenza dei sistemi informatici sia in ambito personale che aziendale, siamo continuamente esposti a rischi riguardanti la nostra privacy, con tutti i nostri dati personali esposti in rete e le aziende che per mezzo dell'IoT si aprono al mondo, anche i vettori di attacco aumentano, per questo è sempre più necessaria una consapevolezza digitale ed una cultura sull'utilizzo della rete e dei sistemi informatici stessi. Nonostante l'anzianità di servizio, gli attacchi DoS, nel tempo, sono stati mitigati dal progresso, a tal punto da risultare nello sviluppo di sempre più complesse botnet, che non si limitano più al solo scopo di generare una rete di bot per eseguire un attacco DDoS ma che oggi possono essere utilizzate anche per eseguire mining di cryptovalute utilizzando la potenza di calcolo delle macchine infettate. Ogni giorno viene segnalata la presenza di nuove botnet scoperte da aziende di sicurezza informatica, mascherate da files comunemente scaricabili dagli utenti e facenti utilizzo di exploit particolari per insinuarsi nei dispositivi delle vittime e prenderne completamente il controllo a discapito delle ignare vittime.

Ringrazio il professor Marcantoni per la sua disponibilità per la realizzazione della Tesi.
Ringrazio la mia famiglia, mia madre, mia nonna, mia sorella e in particolar modo mio fratello che senza di lui non sarei qua, e per il sostegno che mi hanno dato fin dall'inizio di questo percorso universitario, senza tralasciare il supporto del nostro cane Kiro.
Non posso non ringraziare i miei compagni di avventura che sono diventati parte integrante della mia vita, Veronica e Leonardo con cui ho stretto un forte legame nonostante la distanza, Riccardo detto gnegnè e Simone la nostra A.I., e che mi hanno supportato e sopportato nel bene e nel male.
Ringrazio Francesco Caputo per l'aiuto che mi ha dato in un momento di difficoltà.

Bibliografia

[Com] *Computer PLATO*. URL: https://www.wikiwand.com/it/PLATO.

[Dea] Braian Dear. *PLATO History*. URL: http://www.platohistory.org/blog/2010/02/perhaps-the-first-denial-of-service-attack.html.

[Dos] *Malware Protection Center - Glossary*. URL: https://web.archive.org/web/20130305114911/http://www.microsoft.com/security/portal/threat/encyclopedia/glossary.aspx#d#d.

[Gra] *Grafico del confronto degli attacchi DDoS tra il 2019 e 2020*. URL: https://www.internetpost.it/attacchi-ddos-report-2020/.

[Imma] *Attacco in corso*. URL: https://null-byte.wonderhowto.com/how-to/use-ufonet-0174158/.

[Immb] *Controllo della VM*. URL: https://null-byte.wonderhowto.com/how-to/use-ufonet-0174158/.

[Immc] *Ispezione dei file di UFONet*. URL: https://null-byte.wonderhowto.com/how-to/use-ufonet-0174158/.

[Immd] *Ricerca di più zombie*. URL: https://null-byte.wonderhowto.com/how-to/use-ufonet-0174158/.

[Imme] *Schema del funzionamento della Three-way handshake*. URL: https://it.wikipedia.org/wiki/SYN_flood.

[Immf] *Validazione degli zombie*. URL: https://null-byte.wonderhowto.com/how-to/use-ufonet-0174158/.

[Loga] *Logo di UFONet*. URL: https://ufonet.03c8.net/#intro.

[Logb] *Logo Zeus*. URL: https://rimozione-malware.com/malwares/zeus-trojan/.

[Loi] *La GUI di LOIC*. URL: https://www.darknet.org.uk/2017/10/loic-download-low-orbit-ion-cannon-ddos-booter/.

[Ope] *Open Redirect*. URL: https://hacktips.it/vulnerabilita-open-redirect/.

[Pac] *Pacchetti scambiati ina una Url Redirect*. URL: https://security.stackexchange.com/questions/165213/trying-to-examine-url-redirection-in-wireshark.

[Rep] *Report degli attacci del 2020*. URL: https://govcyberhub.com/2021/01/28/lazarus-bear-armada-extorts-government-organizations-with-sophisticated-ddos-threat/.

[Scha] *Schema di un DDoS*. URL: https://it.wikipedia.org/wiki/Denial_of_service.

[Schb] *Schema FBI per spiegare il funzionamento di Zeus*. URL: https://rimozione-malware.com/malwares/zeus-trojan/.

[Smu] *Schema del funzionamento dello Smurf*. URL: https://it.wikipedia.org/wiki/Smurf.

[Syn] *Attacco DoS SYN flood*. URL: https://it.wikipedia.org/wiki/SYN_flood.