IP Spoofing in Scioltezza

Autore: ORK
Data Pubblicazione: Aprile 2001
Versione Documento: 1.0

Originariamente questo articolo è stato scritto per essere pubblicato sulla pagina del LUG di Gorizia (GOLUG).

Contatti :
e-mail : orkmail@katamail.com
web : web.tiscalinet.it/ork33

Special TNX To :
Nicola, ^DruiD, Diaul, Dr. Foobar

** Information wants to be Free !! **

Indice

Informazioni su questo testo

Questo testo è stato scritto con l'intento di far capire cosa sia l'IP Spoofing e quali siano le potenzialità degli attacchi che si possono attuare con questa tecnica. Ho cercato di dare un'impronta semplice e chiara in maniera da rendere facile la comprensione. E' richiesta comunque una conoscenza minima sul funzionamento delle reti.

Tutte le informazioni di seguito riportate sono da considerare a scopo informativo e didattico. Tutto quello che troverete scritto è reperibile in rete in numerosi altri documenti (molti dei quali in inglese).

1 Nozioni di base sui protocolli di rete

Prima di cominciare è necessario avere delle nozioni di base sul funzionamento dei protocolli di rete.

Lo standard per il software di gestione delle reti (usato per il funzionamento di internet) prevede la strutturazione di quest'ultimo in 4 livelli. Ognuno di questi livelli si occupa di svolgere una certa tipologia di funzioni. Un determinato livello di una macchina può conversare (cioè può essere capito) solo con lo stesso livello di un'altra macchina. Le regole e le convenzioni adottate in queste conversazioni sono chiamate protocolli e per ogni livello ne sono stati stabiliti uno o più.

Quando dobbiamo comunicare dei dati in rete, questi ultimi vengono inseriti nel livello più alto. A partire da lì ogni livello passa al livello immediatamente inferiore i dati ricevuti dal livello superiore con l'aggiunta di alcune informazioni di controllo. Arrivati al livello 1 il pacchetto finale, ottenuto dai dati da trasmettere più le aggiunte di tutti i dati di controllo dei 4 livelli, viene trasmesso al dispositivo di rete attraverso il quale avviene la comunicazione. Quando il pacchetto di dati arriva all'host destinazione fa la strada inversa e cioè parte dal livello 1 e sale fino al 4. Ogni livello preleva ed interpreta i dati di controllo inseriti dallo stesso livello dell'host sorgente e passa il pacchetto al livello superiore. Questo modello può essere schematizzato così:

```
     Host                         Host     Descrizione Livello
   Sorgente                   Destinazione

 -----------                  -----------
|           |                |           |
| LIVELLO 4 |                | LIVELLO 4 | Livello di Applicazione
|-----------|                |-----------|
| LIVELLO 3 |                | LIVELLO 3 | Livello di Trasporto (TCP)
|-----------|                |-----------|
| LIVELLO 2 |                | LIVELLO 2 | Livello Internet (IP)
|-----------|                |-----------|
| LIVELLO 1 | --> Rete -->   | LIVELLO 1 | Livello Host-Rete
|___________|                |___________|
```

Per ogni livello i dati sono racchiusi in pacchetti. Ogni pacchetto ha 2 aree logiche distinte: l'header e la parte dati. Quando i dati devono percorrere la pila dall'alto verso il basso ogni livello crea un pacchetto contenente nella parte dati il pacchetto ricevuto dal livello superiore e nella parte header i dati di controllo aggiunti da quel livello. Quando i dati devono risalire la pila dal livello 1 al 4 ogni livello preleva l'header e lo interpreta, poi passa al livello superiore la parte dati. Lo schema seguente illustra un pacchetto di livello 2.

```
 ---------------
|   Header IP   |
|_______________| Come si può vedere il pacchetto
|  ___________  | TCP è contenuto nella zona
| |Header TCP | | DATI dal pacchetto del IP.
| |___________| | Il pacchetto di livello 4 a sua
| | Pacchetto | | volta è formato da una parte dati
| |    di     | | e un  header ed è contenuto nella
| | Livello 4 | | parte dati del pacchetto TCP.
| |___________| |
|_______________|
```

I livelli di interesse per questo articolo sono il secondo e il terzo. Il secondo livello si occupa dell'instradamento dei pacchetti, ovvero si prende carico di portare un pacchetto dall'host sorgente a quello destinazione. Il terzo livello invece si occupa della gestione delle conversazioni tra host sorgente e destinazione.

In internet il protocollo di livello 2 usato per l'instradamento dei pacchetti è il protocollo IP. Per il livello 3 invece sono usati diversi protocolli tra cui il TCP, l'UDP e l'ICMP.

1.1 Nozioni di base sul protocollo IP

L'header di un pacchetto IP è formato da (almeno) 20 byte in cui sono contenuti vari campi. I campi che ci interessano per capire questo articolo sono i seguenti:

- Indirizzo IP sorgente (32 bit): questo è l'indirizzo che verrà Spoofato
- Indirizzo IP destinazione (32 bit)

1.2 Nozioni di base sul protocollo TCP

1.2.1 Formato dell'Header di un pacchetto TCP

L'header di un pacchetto TCP è formato da (almeno) 20 byte in cui sono contenuti vari campi. I campi che ci interessano sono i seguenti:

- Porta Sorgente (16 bit)
- Porta Destinazione (16 bit)
- Sequence number (32 bit)
- Acknowledgement number (32 bit)
- Window Size (16 bit): grandezza della finestra di comunicazione
- Flag SYN (1 bit): usato per creare una connessione
- Flag ACK (1 bit): indica che il campo Acknowledgement è valido
- Flag FIN (1 bit): usato per chiudere una connessione
- Flag RST (1 bit): usato per far reinizializzare una connessione

1.2.2 Creazione di una connessione

In TCP, per creare una connessione viene usato il protocollo three-way handshake che funziona, come dice il nome, in tre passi:

1. Il client spedisce un pacchetto TCP avente il flag SYN impostato, il flag ACK non impostato e con un valore X del Sequence number
2. Il server risponde con entrambi i flag SYN e ACK impostati, con il Sequence number con un valore Y e Acknowledgement number con valore X+1
3. Il client risponde con il flag ACK impostato, con il Sequence number con valore X+1 e Acknowledgement number con valore Y+1

```
                 SYN=1 ACK=0 SEQ-NUM=X
CLIENT  ---------------------------------------> SERVER

           SYN=1 ACK=1 SEQ-NUM=Y ACK-NUM=X+1
CLIENT <---------------------------------------  SERVER

             ACK=1 SEQ-NUM=X+1 ACK-NUM=Y+1
CLIENT  ---------------------------------------> SERVER
```

Dopo che la connessione è stata inizializzata si può procedere al trasferimento dei dati. Nel seguito verrà indicato con pacchetto SYN il primo dei tre, con SYN/ACK il secondo e con ACK il terzo.

1.2.3 Gestione Sequence number e Acknowledgement number in una connessione

Il TCP realizza una connessione affidabile, è in grado cioè di recuperare pacchetti persi, duplicati e fuori ordine. Tutto questo è possibile grazie all'assegnamento di un numero di sequenza (Sequence number) ad ogni byte trasmesso e alla richiesta di conferma dei dati ricevuti dal destinatario (Acknowledgement number). All'interno di un pacchetto il Sequence number indica il numero di sequenza del primo byte di dati contenuto, mentre l'Acknowledgement number indica il numero del prossimo byte atteso e conferma la ricezione fino al byte indicato meno 1. Prendiamo in esame il seguente esempio per capire meglio.

```
     SEQ-NUM=1000 ACK-NUM=5000 DATI=100 byte
1) A -----------------------------------------> B

      SEQ-NUM=5000 ACK-NUM=1100 DATI=250 byte
2) A <----------------------------------------- B

      SEQ-NUM=1100 ACK-NUM=5250 DATI=150 byte
3) A -----------------------------------------> B

      SEQ-NUM=5250 ACK-NUM=1250 DATI=0 byte
4) A <----------------------------------------- B
```

Nel caso 1) A spedisce a B 100 byte il cui primo byte ha come numero di sequenza 1000. Inoltre conferma la ricezione dei byte ricevuti in precedenza fino al 4999 ed indica che si aspetta che il prossimo byte trasmesso sia quello con numero di sequenza 5000.

Nel caso 2) B spedisce ad A 250 byte il cui primo byte ha come numero di sequenza 5000. Inoltre conferma la ricezione dei byte ricevuti in precedenza fino al 1099 ed indica che si aspetta che il prossimo byte trasmesso sia quello con numero di sequenza 1100.

Nel caso 3) A spedisce a B 150 byte il cui primo byte ha come numero di sequenza 1100. Inoltre conferma la ricezione dei byte ricevuti in precedenza fino al 5249 ed indica che si aspetta che il prossimo byte trasmesso sia quello con numero di sequenza 5250.

Nel caso 4) B non ha dati da spedire, si limita solo a confermare i dati ricevuti. Questo pacchetto non verrà confermato in quanto non contiene dati.

1.2.4 Chiusura di una connessione

Sebbene le connessioni TCP siano full duplex, per comprendere la chiusura di una connessione è meglio pensarle come una coppia di connessioni simplex. Ogni connessione simplex viene chiusa indipendentemente dall'altra. Per chiudere una connessione entrambe le parti possono spedire un pacchetto TCP contenente il flag FIN impostato a indicare che non ci sono più dati da spedire. Quando viene confermata la ricezione del flag FIN quella connessione viene spenta. Quando entrambe le direzioni vengono chiuse la connessione viene rilasciata. Quindi normalmente occorrono 4 passaggi per chiudere una connessione.

1.2.5 Il Flag RST

Il flag RST viene usato per far reinizializzare una connessione che è diventata instabile per qualche motivo, per rifiutare un pacchetto non valido o per rifiutare l'apertura di una connessione. In generale se si riceve un pacchetto con il flag RST impostato significa che c'è un problema.

1.3 Nozioni di base sul protocollo ICMP

L'ICMP è un protocollo di test e diagnostica della rete. Quando avviene qualcosa di inatteso, l'evento viene segnalato tramite un pacchetto ICMP. Alcuni dei messaggi ICMP più importanti sono:

- DESTINATION UNREACHABLE, che sta ad indicare che l'host non è raggiungibile;
- TIME EXCEEDED, che sta ad indicare che il pacchetto è stato scartato a causa del suo contatore che ha raggiunto lo zero.
- ECHO REQUEST e ECHO REPLY, usati per controllare se una destinazione è raggiungibile e attiva. Alla ricezione di un Echo Request l'host è vincolato a rispondere con un Echo Reply.
- TIMESTAMP REQUEST e TIMESTAMP REPLY, differiscono dai precedenti solo dal fatto che il tempo di arrivo della richiesta e il tempo di partenza della risposta vengono registrati nella risposta stessa. Questo servizio viene usato per misurare le prestazioni della rete.

1.4 Introduzione all'IP Spoofing

Le basi principali della sicurezza in Internet sono rappresentate dai controlli sui numeri IP sorgenti delle connessioni. Ad esempio controlli di questo tipo sono fatti dai TCP Wrapper e dai Firewall. Inoltre servizi tipo rlogin o rsh permettono a certi host fidati di accedere alle risorse del sistema senza bisogno di password. L'IP Spoofing consiste nel falsificare l'indirizzo IP sorgente della connessione in modo da far credere di essere un altro host per poter superare certe difese o per portare a termine certe tipologie di attacchi. Poiché i protocolli TCP, UDP, ICMP (e altri) sono incapsulati in pacchetti IP tutti questi protocolli sono di riflesso teoricamente Spoofabili.

A seconda della tipologia di attacco che l'attaccante vuole portare a termine e della posizione sulla rete in cui egli si trova ci sono differenti tecniche basate sull'IP Spoofing. Le tre categorie di attacchi che esaminerò in questo testo sono IP Spoofing non cieco, IP Spoofing cieco e DOS.

Il concetto di Spoofing non cieco è molto semplice: l'attaccante cerca di farsi passare per un host che fa parte della sottorete in cui egli stesso si trova, quindi i pacchetti indirizzati all'host sono visibili anche a lui e perciò è a conoscenza del Sequence number e dell'Acknowledgement number aggiornati.

Per quanto riguarda lo Spoofing cieco, l'attaccante sta cercando di farsi passare per un host qualsiasi (che non è nella sua sottorete) e quindi in qualche maniera dovrà cercare di "indovinare" il Sequence number corretto per continuare la connessione.

Infine gli attacchi di tipo DOS (Denial of Service) sono attacchi diretti a bloccare un determinato host impedendogli di svolgere le sue normali attività e quindi negando agli utenti di poter accedere ai servizi a cui l'host era preposto.

2 IP Spoofing non Cieco

Solitamente i vari host di una sottorete sono collegati da apparecchi (HUB) che, per far arrivare un pacchetto ad un determinato host, lo trasmettono in broadcast a tutti i computer della sottorete. Quando il pacchetto arriva ad un determinato host, quest'ultimo esamina l'indirizzo di destinazione contenuto nel pacchetto. Se questo indirizzo coincide con quello dell'host stesso, il pacchetto viene processato, altrimenti viene scartato in quanto era destinato ad un altro host. Tuttavia esiste una modalità particolare in cui è possibile impostare la scheda di rete: la modalità promiscua. Quando la scheda di rete si trova in questa modalità, permette di processare tutti i pacchetti che arrivano.

Come già accennato sopra, nel caso dello Spoofing non cieco l'attaccante sta cercando di farsi passare per un host che fa parte della sua sottorete; quindi, impostando la scheda di rete in modo promiscuo, egli riesce a leggere tutti i pacchetti indirizzati all'host che intende impersonare e può così scoprire Sequence number e Acknowledgement number della connessione in corso e cercare di inserirvisi. Alcuni tipi di attacchi che possono essere messi a segno con questa tecnica sono la chiusura di una connessione esistente e l'Hijacking.

C'è da notare che ultimamente al posto degli HUB sono sempre più usati degli apparecchi chiamati switch che invece di trasmettere i pacchetti in broadcast li trasmettono solo al destinatario corretto. Questo permette di eliminare certe tipologie di attacchi quali sniffing e IP Spoofing Cieco aumentando allo stesso tempo le prestazioni della rete

2.1 Chiusura di una connessione esistente

Avendo a disposizione Sequence number e Acknowledgement number di una connessione in corso l'attaccante può spedire in un momento preciso un pacchetto creato accuratamente con l'intento di far cadere la connessione. Per raggiungere questo obiettivo può usare uno dei due flag RST o FIN compresi nell'header dei pacchetti TCP.

2.1.1 Chiusura di una connessione esistente usando il flag RST

Il flag RST, tra le altre cose, viene utilizzato per reinizializzare una connessione che è diventata instabile per qualche motivo. Un pacchetto TCP avente lo scopo di resettare una connessione ha solamente il Sequence number valido, l'Acknowledgement number è disabilitato. Per procedere al reset di una connessione esistente l'attaccante procede secondo alcuni passi. Immaginiamo esista una situazione di questo tipo:

```
A e C = Host della stessa sottorete
B     = Host di una rete diversa da A e C
H     = HUB
R     = Router che delimita la sottorete

A ------H---R------------ B
        |
C ------/
```

Supponiamo che tra gli host A e B esista una connessione e che l'attaccante si trovi nella postazione C. Per cercare di resettare la connessione esistente, come prima cosa l'attaccante aspetterà di ricevere un pacchetto proveniente dalla connessione A-B. Supponendo che riceva un pacchetto proveniente da B verso A, egli prima calcolerà il Sequence number a partire dall'Acknowledgement number del pacchetto ricevuto, poi costruirà e spedirà un pacchetto con le seguenti impostazioni:

Campi del pacchetto IP:
IP sorgente = A (IP Spoofato)
IP destinazione = B

Campi del Pacchetto TCP:
Porta Sorgente = Porta usata dall'host A
Porta Destinazione = Porta usata dall'host B
Sequence number contenente il valore appena calcolato
Flag RST impostato

Il risultato sarà il reset della connessione. Ci sono però dei problemi tecnici: questo metodo, infatti, funziona solo se il pacchetto dell'attaccante arriva prima della reale risposta dell'host A. L'host B riceverà due pacchetti con lo stesso Sequence number (uno mandato dall'attaccante, e l'altro mandato dall'host A), quindi prenderà per buono il primo arrivato e scarterà il secondo credendolo un duplicato.

2.1.2 Chiusura di una connessione esistente usando il flag FIN

Il flag FIN sta ad indicare "non ci sono più dati da parte del mittente". Se, dopo che un host ha spedito il flag FIN e questo è stato accettato, vengono ricevuti altri pacchetti questi vengono ignorati.

Per procedere alla chiusura di una connessione esistente usando il flag FIN l'attaccante procede in modo simile al caso precedente, con alcune differenze. La caratteristica di questa tecnica è che l'attaccante ha la possibilità di controllare se la connessione è stata chiusa in quanto l'host che riceverà il pacchetto di FIN risponderà con un pacchetto di Acknowledgement.

A differenza del caso precedente, il pacchetto che verrà costruito deve avere sia il Sequence number che l'Acknowledgement number validi. Riprendiamo la situazione precedente:

```
A e C = Host della stessa sottorete
B     = Host di una rete diversa da A e C
H     = HUB
R     = Router che delimita la sottorete

A ------H---R------------ B
        |
C ------/
```

Supponiamo che tra gli host A e B esista una connessione e che l'attaccante si trovi nella postazione C. Per cercare di chiudere la connessione esistente, come prima cosa l'attaccante aspetterà di ricevere un pacchetto proveniente dalla connessione A-B. Supponendo che riceva un pacchetto proveniente da B verso A, egli prima calcolerà il Sequence number e l'Acknowledgement number in base al pacchetto ricevuto, poi costruirà e spedirà un pacchetto con le seguenti impostazioni:

Campi del pacchetto IP:
IP sorgente = A (IP Spoofato)
IP destinazione = B

Campi del Pacchetto TCP:
Porta Sorgente = Porta usata dall'host A
Porta Destinazione = Porta usata dall'host B
Sequence number contenente il valore appena calcolato
Acknowledgement number contenente il valore appena calcolato
Flag FIN impostato

A questo punto resterà in ascolto in attesa di un pacchetto di Acknowledgement da parte di B. Se lo riceve é sicuro che da questo momento B risponderà a tutti i pacchetti ricevuti da A con un pacchetto di reset credendo siano bugs facendo, tra l'altro, cadere anche l'altro senso di connessione.

2.2 Hijacking

L'Hijacking è una tecnica molto raffinata che permette di intromettersi in una connessione esistente e prenderne il controllo. Molte volte l'Hijacking viene snobbato in quanto trovandosi già nella sottorete dell'host che si vuole impersonare basterebbe usare uno sniffer per catturare username e password per poi collegarsi "legalmente". Tuttavia è possibile trovarsi in situazioni in cui, ad esempio, vengono usate password "usa e getta" e quindi, anche se l'attaccante riuscisse a sniffarne qualcuna, quando andrà ad usarle queste saranno già scadute.

2.2.1 Stato di Desincronizzazione

Prima di proseguire con la spiegazione sul funzionamento dell'Hijacking è necessario chiarire cosa si intende per stato di desincronizzazione di una connessione. Per semplicità da adesso in poi indicheremo con:

SVR_SEQ il Sequence number del prossimo byte che il server spedirà
SVR_ACK il prossimo byte che il server si aspetta di ricevere
SRV_WND la grandezza della finestra di ricezione del server
CLT_SEQ il Sequence number del prossimo byte che il client spedirà
CLT_ACK il prossimo byte che il client si aspetta di ricevere
CLT_WND la grandezza della finestra di ricezione del client

In una situazione di "calma" durante una connessione, cioè un momento in cui non vengono spediti dati da entrambe le parti, le seguenti equazioni sono

vere:
SVR_SEQ = CLT_ACK e CLT_SEQ = SRV_ACK
Invece mentre sono trasferiti dei dati sono vere queste altre espressioni:
$CLT\_ACK \leq SVR\_SEQ \leq CLT\_ACK + CLT\_WND$
$SRV\_ACK \leq CLT\_SEQ \leq SRV\_ACK + SRV\_WND$
da cui si può capire che un pacchetto è accettabile se il suo Sequence number appartiene all'intervallo [SRV_ACK, SRV_ACK + SRV_WIN] per il server e [CLT_ACK, CLT_ACK + CLT_WIN] per il client. Se il Sequence number supera o precede questo intervallo il pacchetto viene scartato e viene spedito un pacchetto contenente nell'Acknowledgement number il Sequence number del prossimo byte atteso.

Il termine desincronizzazione si riferisce ad una situazione in cui durante una connessione in un periodo di "calma" sono vere le seguenti equazioni SVR_SEQ $\neq$ CLT_ACK e CLT_SEQ $\neq$ SRV_ACK Se una connessione si trovasse in una situazione di questo tipo e dei dati venissero spediti da una delle parti potrebbero presentarsi due casi distinti:

- Se $CLT\_SEQ < SVR\_ACK + SVR\_WND$ e $CLT\_SEQ > SVR\_ACK$ il pacchetto è accettabile e i dati vengono memorizzati per un uso futuro, ma non vengono processati.
- Se $CLT\_SEQ > SVR\_ACK + SVR\_WND$ o $CLT\_SEQ < SVR\_ACK$ il pacchetto non è accettabile e viene scartato.

In pratica se una connessione è in questo stato i due host non possono scambiarsi dati.

2.2.2 L'attacco

Riprendiamo la situazione precedente:

```
A e C = Host della stessa sottorete
B     = Host di una rete diversa da A e C
H     = HUB
R     = Router che delimita la sottorete

A ------H---R------------ B
        |
C ------/
```

Supponiamo che esista una connessione telnet da A verso B e che l'attaccante si trovi nella postazione C. Cosa succederebbe se quest'ultimo in un periodo di "calma" della connessione tra A e B mandasse un pacchetto spoofato a B in modo da far credere che provenga da A?
Semplice, B aggiornerebbe l'Acknowledgement number di A (SVR_ACK) in base al pacchetto ricevuto desincronizzandosi dal Sequence number reale di A (CLT_SEQ). A questo punto i pacchetti spediti da A verranno scartati in quanto per B hanno un Sequence number errato. Vediamo un esempio per capire meglio:

A spedisce un pacchetto contenente 10 Byte di Dati, Sequence number=100 e Acknowledgement number=500

```
  SEQ=100 ACK=500 DATI=10
A -----------------------> B
```

B si aggiorna Sequence number e Acknowledgement number:
Sequence number = 500, Acknowledgement = 100 + 10

B spedisce un pacchetto contenente 15 Byte di Dati, Sequence number=500 e Acknowledgement number=110

```
  SEQ=500 ACK=110 DATI=15
A <----------------------- B
```

Il pacchetto arriva ad A che si riaggiorna Acknowledgement number e Sequence number:
Sequence number = 110, Acknowledgement = 500 + 15

A questo punto si intromette l'attaccante con un pacchetto Spoofato, usando il Sequence number e l'Acknowledgement number corretti.

C spedisce un pacchetto spoofato contenente 20 Byte di Dati, Sequence number=110 e Acknowledgement number=515

```
     SEQ=110 ACK=515 DATI=20
A(C) -----------------------> B
```

B si aggiorna Sequence number e Acknowledgement number:
Sequence number = 515, Acknowledgement = 110 + 20

A questo punto B è desincronizzato rispetto ad A in quanto il prossimo byte che B si aspetta da A è il 130, mentre il Sequence number di A è a 110. Quindi i pacchetti che A spedirà a B da questo momento in poi verranno scartati. L'attaccante però sa cosa si aspetta B e perciò può mandare dei pacchetti creati appositamente per essere accettati. Ricordiamo che nel nostro esempio la connessione in corso era una sessione telnet, quindi adesso l'attaccante può mandare comandi di shell a B come se fosse A.

C'è da notare che nell'esempio appena descritto non c'è una desincronizzazione da entrambe le parti, infatti abbiamo che CLT_SEQ $\neq$ SRV_ACK ma SVR_SEQ = CLT_ACK. Quindi l'host A accetterà tutti i pacchetti spediti da B come risposta ai comandi dell'attaccante, e quindi vedrà tutto quello che questi sta facendo. Per evitare ciò l'attaccante deve creare una situazione di desincronizzazione anche nell'altro senso di trasmissione spedendo un pacchetto spoofato ad A come se provenisse da B. Ricordiamo che essendo l'attaccante nella stessa sottorete di A é in grado di vedere l'output dei propri comandi sniffando i pacchetti di risposta spediti da B.

Ci sono varie tecniche per ottenere la desincronizzazione di una connessione. Quella vista nell'esempio è quella usata solitamente e consiste appunto nello spedire un pacchetto spoofato contenente dei dati sia al server che al client. Più è grande il numero di byte spediti in questi pacchetti e più è grande la desincronizzazione che si andrà a creare tra i due host.

Esiste comunque un secondo e più raffinato metodo di desincronizzazione, che consiste nell'intromettersi nel protocollo three-way handshake usato per creare una connessione. Il funzionamento si può sintetizzare in 4 punti:

1. L'attaccante aspetta di ricevere il pacchetto SYN/ACK, proveniente dal server e diretto verso il client (secondo passo del three-way handshake), della connessione da desincronizzare.

2. Appena lo ha identificato spedisce un pacchetto di RST (Spoofato) verso il server e immediatamente dopo uno di SYN (sempre Spoofato) con gli stessi parametri (porta TCP e indirizzo IP) usati per la connessione da desincronizzare, ma con un differente Sequence number.

3. Il server chiuderà la prima connessione grazie al pacchetto RST, e ne aprirà una uguale ma con un Sequence number diverso, spedendo il pacchetto SYN/ACK.

4. L'attaccante non appena identifica quest'ultimo, spedisce il pacchetto ACK necessario a completare l'instaurazione della connessione.

A questo punto la connessione è aperta, ma è in uno stato di desincronizzazione in quanto per il client il Sequence number corretto è quello che era presente nel pacchetto SYN/ACK intercettato dall'attaccante al punto 1, mentre per il server quello corretto è quello introdotto dall'attaccante nel punto 2.

2.2.3 Il problema dell'ACK Storm

Come visto nel paragrafo 2.2.1 se il Sequence number di un pacchetto supera o precede l'intervallo di accettazione il pacchetto viene scartato e viene spedito un pacchetto contenente il Sequence number del prossimo byte atteso. Se ci si trova in uno stato di desincronizzazione con tutta probabilità i pacchetti che il client e server si scambieranno avranno errori di questo tipo, e quindi verranno generati questi pacchetti di errore che però a loro volta non sono validi. Questo genera un loop di pacchetti di Acknowledgement chiamato appunto ACK Storm. Se la desincronizzazione è completa, cioè da entrambe i sensi di comunicazione, viene creato un ACK Storm per ogni pacchetto spedito da entrambe gli host. Se la desincronizzazione è parziale (solitamente viene desincronizzata solo la direzione del client verso il server) verrà creato un ACK Storm solo dai pacchetti che provengono da quella direzione. Quando vengono spediti dei dati in un canale desincronizzato c'è da notare che oltre ad essere provocato un ACK Storm, il pacchetto non viene confermato e quindi dopo un pò il mittente lo ritrasmetterà credendo che sia andato perso, creando un altro ACK Storm. Sarà compito dell'attaccante spedire dei pacchetti di conferma per impedire questi ulteriori ACK Storm. Questi loop si fermeranno solo quando uno dei pacchetti dell'ACK Storm andrà perso.

3 IP Spoofing Cieco

Quando si parla di IP Spoofing solitamente ci si riferisce all'IP Spoofing cieco. Con questa tecnica l'attaccante cerca di farsi passare per un host qualunque di internet, non facente parte della sottorete in cui si trova. In questo caso si parla di Spoofing cieco in quanto l'attaccante non avrà nessuna possibilità di vedere i pacchetti mandati in risposta ai pacchetti (spoofati) che ha spedito. Questi ultimi infatti saranno indirizzati all'host che egli sta impersonando, impedendogli quindi di venire a conoscenza del Acknowledgement number e Sequence number corretti per continuare la connessione.

Come si può dedurre le possibilità dello Spoofing cieco sono infinitamente inferiori a quelle che si poteva avere con lo Spoofing non cieco, ma comunque qualcosa si può fare ...

Molte volte alcuni server danno un accesso privilegiato a degli host fidati attraverso servizi di rlogin senza password o servizi simili. In un caso di questo tipo l'unico controllo è fatto sul numero IP sorgente della connessione, quindi se un attaccante riuscisse ad aprire una connessione spoofando il proprio IP potrebbe lanciare dei comandi al server facendogli credere di essere l'host fidato. Se siete stati attenti fino qua avrete già intuito che questa non è un'operazione troppo facile. Andiamo a rivedere brevemente il contenuto dei 3 pacchetti TCP necessari per instaurare una connessione:

```
Pacchetto 1: Direzione: Client  --->  Server
             Flag Attivi: SYN
             Sequence number: X (numero generato dal client)

Pacchetto 2: Direzione: Server  --->  Client
             Flag Attivi: SYN, ACK
             Sequence number: Y (numero generato dal server)
             Acknowledgement number: X + 1

Pacchetto 3: Direzione: Client  --->  Server
             Flag Attivi: ACK
             Sequence number: X + 1
             Acknowledgement number: Y + 1
```

Se un attaccante tentasse di instaurare una connessione con il proprio IP Spoofato, come detto prima, non sarebbe in grado di vedere il pacchetto di risposta del server, ovvero il secondo dei 3 pacchetti. Questo è un problema, in quanto non sapendo il Sequence number del pacchetto 2 non saprà qual'è l'Acknowledgement number corretto per il pacchetto 3 e quindi l'instaurazione della connessione non potrà andare a buon fine. L'unica soluzione a questo problema è cercare di predire il Sequence number corretto.

Sebbene questa tipologia di attacco fosse stata introdotta a livello teorico molto tempo prima, la prima registrazione ufficiale risale alla vigilia di natale del 1994 e fu portata a termine dall'Hacker Kevin Mitnick.

3.1 Piccolo dettaglio tecnico

Supponiamo che un attaccante voglia mettere in pratica quanto appena descritto, cioè tentare di aprire una connessione con un server spoofando il proprio IP in maniera da farsi passare per un host fidato. Un problema che si presenta all'attaccante ancora prima della predizione del Sequence number del secondo pacchetto è che l'host che egli cerca di impersonare, non appena riceverà il secondo pacchetto, si renderà conto che non sta cercando di aprire una connessione: quindi avviserà il server mandandogli un pacchetto di reset rendendo nullo il lavoro dell'attaccante.

Per impedire che ciò accada, quest'ultimo o aspetta che l'host da impersonare sia spento, o può cercare di "buttarlo giù" con un attacco del tipo Denail of Service. L'attacco classico è il SYN flood (spiegato nella sezione 4), anche se ultimamente molti host incorporano una protezione per questo tipo di attacco. Se l'host dovesse avere questo tipo di protezione, l'attaccante potrà provare con un altro dei numerosi DOS esistenti.

3.2 Generazione del Sequence Number

E adesso veniamo al problema vero e proprio, come fa l'attaccante a predire un numero a 32 bit?

Bisogna tenere conto che il TCP è stato creato per la gestione di connessioni e non per risolvere problemi di sicurezza, quindi non dobbiamo stupirci se molti sistemi operativi relativamente vecchi risultano vulnerabili a questa tipologia di attacco. Per capire com'è possibile predire il Sequence number andiamo a vedere come i server scelgono questo numero. Fondamentalmente esistono 3 modi in cui il Sequence number può essere generato, due dei quali sono vulnerabili all'IP Spoofing. Vediamoli in dettaglio.

3.2.1 Generazione in base alla regola dei 64k

Questo sistema è molto semplice, ma nonostante ciò è molto usato. Le regole usate per generare il Sequence Number sono le seguenti:

- Incrementa ogni secondo il contatore del Sequence Number di una costante, solitamente 128000
- Se è stata aperta una connessione incrementa il contatore del Sequence Number di un'altra costante, solitamente 64000

3.2.2 Generazione in base al Tempo

Anche questa è una tecnica semplice ed abbastanza usata. Per generare il Sequence Number, dopo un'inizializzazione casuale al boot, il contatore del Sequence Number viene incrementato ogni periodo di tempo prefissato. Un esempio può essere l'incremento di 1 ogni microsecondo.

3.2.3 Generazione Random

Con questa tecnica il Sequence Number viene generato quasi casualmente, ed è pressoché impossibile predirlo. Un esempio di questa tecnica si può trovare nei nuovi kernel di Linux.

3.3 Predizione del Sequence Number

Come fa quindi l'attaccante a predire il Sequence Number per portare a termine il suo attacco?

Prima di tutto deve scoprire quale dei tre algoritmi per la generazione del Sequence Number è in uso sul server che vuole colpire. Per fare ciò manda qualche pacchetto SYN non spoofato per vedere ed analizzare le risposte del server. Questi pacchetti di risposta gli permettono di capire con una certa facilità a quale dei tre algoritmi si trova di fronte.

Per vedere se il server sta usando l'algoritmo che usa la regola dei 64k gli è sufficiente calcolare la differenza tra 2 pacchetti ricevuti e vedere se il numero ottenuto è divisibile per 64000.

Per capire se il server stà usando l'algoritmo che usa la regola in base al tempo dovrà fare dei campionamenti su una serie di pacchetti. Innanzitutto spedirà una serie di pacchetti SYN e salverà il tempo in cui gli arriva il pacchetto SYN/ACK di risposta e il relativo Sequence Number. Quindi andrà a calcolare la differenza tra i vari tempi registrati ottenendo l'intervallo di tempo trascorso tra due generazioni del Sequence Number. Infine, facendo la divisione tra il risultato ottenuto e la differenza tra i Sequence Number corrispondenti ai tempi presi in considerazione otterrà l'incremento per ogni unità di tempo. Se i risultati di tutti i campionamenti danno un valore simile si può pensare che il server usi questo algoritmo per generare i Sequence Number.

Chiaramente se questi due test non vanno a buon fine probabilmente il server usa un algoritmo di generazione random del Sequence Number e per l'attaccante non sarà possibile continuare l'attacco. Se così non è il passo successivo sarà cercare di indovinare il prossimo Sequence Number che verrà generato.

Nel caso della generazione in base alla regola dei 64k predire il Sequence Number è abbastanza semplice. Infatti all'attaccante sarà sufficiente spedire un pacchetto SYN non spoofato al server, leggere il Sequence Number del pacchetto giunto in risposta ed aggiungerci 64000. Il risultato con tutta probabilità sarà il prossimo Sequence Number per meno di un secondo, tempo dopo cui verrà incremento di 128000. Come si può intuire 1 secondo nel mondo dei computer è un tempo molto grande, quindi questo algoritmo è particolarmente vulnerabile ad un attacco di IP-Spoofing.

Nel caso della generazione in base al tempo, invece, le cose per l'attaccante si complicano un po'. Il Sequence Number in questo caso verrà calcolato in base ai risultati dei campionamenti fatti in precedenza.

3.4 L'attacco

Adesso che sappiamo come l'attaccante fa a predire il Sequence Number possiamo passare ad analizzare i passi successivi di un attacco di IP Spoofing cieco. Per prima cosa l'attaccante si assicurerà che l'host che intende impersonare sia spento, e nel caso non lo fosse, prenderà qualche iniziativa (come visto nel paragrafo 3.1).

Dopo aver capito di fronte a che tipo di algoritmo per la generazione del Sequence Number si trova procederà secondo i seguenti passi:

1. spedirà un pacchetto SYN non spoofato
2. in base al pacchetto SYN/ACK di risposta e all'algoritmo usato calcolerà il prossimo probabile Sequence Number.
3. spedirà un pacchetto SYN spoofato con l'indirizzo sorgente dell'host fidato
4. spedirà il pacchetto ACK spoofato con l'indirizzo sorgente dell'host fidato con il Sequence Number appena calcolato + 1.

In certi casi, specialmente in quelli in cui il Sequence Number viene calcolato in base al tempo, indovinare il numero corretto è abbastanza difficile. Certe implementazioni del protocollo TCP in alcuni sistemi facilitano il compito all'attaccante. In questi sistemi quando viene ricevuto un Sequence Number troppo alto rispetto a quello corretto viene generato un pacchetto di reset. Se, invece, il Sequence Number è più basso di quello corretto non viene fatto assolutamente niente. Questo fatto viene sfruttato spedendo una serie di pacchetti ACK con Sequence Number crescenti partendo da un valore leggermente inferiore a quello predetto per arrivare ad uno un pò più alto.

A questo punto l'attaccante non ha la sicurezza matematica di essere riuscito ad aprire la connessione, ma ha comunque una probabilità abbastanza elevata. Poichè stava cercando di aprire una connessione con il servizio rlogin non gli sarà chiesta nessuna password per entrare. Adesso con tutta probabilità, quindi, avrà accesso ad una shell. Il passo successivo solitamente sarà quello di inviare il comando

echo "+ +" > .rhosts

che ha l'effetto di consentire l'accesso al sistema senza password tramite servizi del tipo di rlogin a tutti i numeri IP. Comunque esistono modi diversi di procedere, tutto dipende della fantasia e dagli obiettivi dell'attaccante.

L'ultimo particolare da tenere in considerazione è che se il server spedisce dei dati, l'attaccante non saprà esattamente quando lo fa e quanti byte spedisce, quindi non potrà spedire i pacchetti di Acknowledgement per confermare la ricezione dei dati. Questo effettivamente non è un problema per lui, infatti ciò causerà solamente la ristrasmissione dei dati dopo un certo tempo limite.

4 Denial of Service (DOS)

Un attacco di tipo DOS ha come scopo finale escludere un determinato host dalla rete rendendolo irraggiungibile. Fondamentalmente esistono 4 categorie di DOS:

- Attacchi per l'esaurimento di banda, che si basano sull'inondare la rete dell'host bersaglio in maniera da consumare tutta la larghezza di banda a lui disponibile. Questo tipo di attacco si può attuare in 2 modi distinti. Il primo caso si ha quando l'attaccante ha a disposizione una connessione più veloce della vittima e quindi riesce a saturarne la banda direttamente. Il secondo metodo si ha quando l'attaccate pur non avendo una connessione veloce riesce a saturare la banda della vittima grazie all'utilizzo di altri host che hanno lo scopo di amplificare l'attacco.

- Attacchi per l'esaurimento delle risorse, che hanno come scopo colpire il sistema piuttosto che la rete in cui si trova. In genere questo si traduce con il consumo di risorse come cicli di CPU, della memoria, ecc.

- Attacchi contro difetti di programmazione, che vanno a colpire bug software o hardware. Solitamente si verificano quando vengono spediti dei dati non previsti all'elemento vulnerabile.

- Attacchi DOS generici.

Molti attacchi di tipo DOS per il loro funzionamento fanno uso dell'IP Spoofing rendendo quasi impossibile capirne la provenienza. Andiamo adesso ad analizzare due esempi abbastanza popolari di DOS che fanno uso dell'IP Spoofing.

4.1 Smurf

Lo Smurf è un attacco DOS che ha lo scopo di esaurire l'intera banda dell'host vittima, sfruttando intere sottoreti che fungono da amplificatori.

Solitamente ad ogni sottorete è associato un indirizzo IP di broadcast che ha per lo più funzioni diagnostiche. Se un pacchetto è indirizzato ad un indirizzo di broadcast significa che deve essere elaborato da tutti gli host della sottorete. Questo può essere utile in certi casi, ad esempio per verificare quanti host sono attivi in una sottorete é sufficiente fare un ping all'indirizzo di broadcast, anziché farne uno ad ogni indirizzo.

Lo smurf combina questo meccanismo di risposta multipla, con l'IP Spoofing per creare un attacco pressoché impossibile da fermare. Il funzionamento dello smurf è molto semplice: l'attaccante spedisce una serie di pacchetti ICMP "ECHO REQUEST" Spoofati con l'indirizzo dell'host vittima a degli indirizzi di broadcast. Il risultato è che tutti gli host della sottorete elaborano il pacchetto e generano come risposta un pacchetto ICMP "ECHO REPLY" indirizzato all'host vittima. Per capire meglio prendiamo in esame questo esempio:

```
A = Host Attaccante
V = Host Vittima
S = Sottorete contenente 100 Host
B = Indirizzo dei broadcast della sottorete S
```

A Spedisce un pacchetto ICMP "ECHO REQUEST" Spoofato con l'indirizzo di V all'indirizzo di broadcast della sottorete.

```
V(A) -----ECHO REQUEST------> B
```

Tutti i 100 host della sottorete rispondono con un pacchetto ICMP "ECHO REPLY" a V credendo che sia lui ad aver spedito il pacchetto precedente.

```
     ----------------------->
     -----100 Pacchetti----->
S    ---------ICMP----------> V
     ------ECHO REPLY------->
     ----------------------->
```

Come si può capire con questa tecnica anche un host che ha a disposizione una banda limitata, conoscendo l'indirizzo di broadcast di qualche rete che ha un fattore di amplificazione elevato, può mettere in ginocchio connessioni molto veloci. Supponiamo infatti che l'attaccante abbia a disposizione una comune connessione a 56 kbps e che riesca a inviare con continuità ad una velocità di 5 k/s pacchetti ICMP "ECHO REQUEST" a degli indirizzi di broadcast che abbiano un fattore di amplificazione di 100. Facendo due semplici conti ci si accorge che l'attaccante riesce a generare un traffico di 500 k/s e cioè 4 Mbit al secondo.

4.2 SYN Flood

Il SYN Flood è un attacco DOS che ha come scopo ultimo l'esaurire le risorse del sistema vittima. Fino a qualche anno fa il SYN Flood era molto temuto, ma adesso molti sistemi incorporano delle protezioni che permettono loro di essere immuni a questo tipo di attacco.

Come abbiamo visto più volte in questo articolo, quando un client vuole instaurare una connessione, spedisce al server un pacchetto SYN. Quando il server riceve questo pacchetto alloca una certa quantità di risorse per una futura connessione e spedisce il pacchetto SYN/ACK aspettandosi il pacchetto ACK necessario per completare la connessione. Se il terzo e ultimo pacchetto non arriva entro un certo tempo (da qualche decina diseondi a qualche minuto a seconda del sistema) il server libera le risorse allocate in precedenza. Il problema sfruttato da SYN Flood è che i server, anche se molto potenti, esauriscono velocemente le risorse usate per la realizzazione dei collegamenti. Quindi se in un piccolo lasso di tempo (inferiore al tempo necessario al server per liberare le risorse) vengono ricevute molte richieste parziali di connessioni il server esaurisce le risorse. Ad esempio a sistemi, che sono in grado di gestire migliaia di connessioni contemporaneamente, possono bastare poche decine di richieste parziali di connessioni pendenti per esaurire questo tipo di risorsa. L'idea dell'attacco è spedire una serie di pacchetti SYN in un breve lasso di tempo in modo da consumare tutte le risorse del server. C'è solo un particolare da tenere in considerazione, e cioè che i pacchetti che l'attaccante spedisce devono avere l'indirizzo sorgente Spoofato con un indirizzo di un host inesistente o spento. Questo si spiega con il fatto che se l'host esiste, non appena riceverà il pacchetto SYN/ACK spedito dal server, risponderà con un pacchetto di reset ed il server libererà le risorse rendendo vani gli sforzi dell'attaccante. Ad un attaccante basterà spedire qualche pacchetto SYN ogni 10 secondi per tenere il server in una situazione di sovraccarico delle risorse. In una situazione di questo tipo infatti il server non riuscirà mai a liberare le risorse in quanto non appena sarà passato il tempo necessario per deallocare un pò di risorse arriverà subito un pacchetto SYN che le riallocherà.

Come si può capire questo attacco è molto pericoloso in quanto si può attuare anche con una connessione lenta, inoltre spoofando gli indirizzi IP sorgenti diventa quasi impossibile risalire all'attaccante.

Riferimenti bibliografici

[1] "A short overview of IP spoofing: PART I" reperibile in rete

[2] "A short overview of IP spoofing: PART II" reperibile in rete

[3] "IP-spoofing Demystified" tratto da "Phrack Magazine Issue 48"

[4] "A Simple Active Attack Aganinst TCP" reperibile in rete