Protocollo TCP e sicurezza

Il **protocollo TCP** è bidirezionale, quando viene infatti stabilita una connessione client/server, quest'ultimo ha la possibilità di rispondere al client sulla stessa connessione, questa caratteristica rende TCP ideale per applicazioni che richiedono una comunicazione affidabile e ordinata, come il trasferimento di file, la navigazione web e l'invio di email.

Per bloccare una connessione TCP è sufficiente bloccare il primo pacchetto di tale connessione, cioè quello che contiene il campo SYN=1. Senza il primo pacchetto, qualunque altro pacchetto successivo al primo non potrà essere riassemblato sul lato ricevente. Questo approccio è comunemente utilizzato nei firewall e nei sistemi di filtraggio dei pacchetti per gestire il traffico TCP.

Ecco una spiegazione dettagliata:

1. Funzionamento della Connessione TCP

La connessione TCP utilizza un processo chiamato **three-way handshake (**prevede tre messaggi scambiati tra i dispositivi per stabilire una connessione, sincronizzare il trasferimento dei dati e garantire che entrambe le parti siano pronte a ricevere e inviare informazioni. Quasi tutte le nostre attività online prevedono questo tipo di handshake) per stabilire una comunicazione affidabile tra due host. Questo processo avviene in tre fasi:

1. **SYN (Synchronize)**: Il client invia un pacchetto con il flag **SYN=1** per iniziare la connessione.
2. **SYN-ACK**: Il server risponde con un pacchetto con **SYN=1** e **ACK=1** per riconoscere la richiesta.
3. **ACK**: Il client invia un ultimo pacchetto con **ACK=1** per completare l'handshake.

Solo dopo questo scambio, la connessione viene considerata stabilita.

2. Blocco del Pacchetto SYN

Bloccando il pacchetto SYN iniziale, il server non riceverà mai la richiesta di connessione, quindi l'handshake non può avvenire e la connessione TCP non verrà stabilita.

Metodi di Blocco

- **Firewall di tipo stateless (**è una tipologia di firewall che esamina i pacchetti di rete **senza tenere traccia dello stato delle connessioni** a cui appartengono. Questi firewall applicano le regole di filtraggio in base alle informazioni contenute nei singoli pacchetti, indipendentemente dal contesto della comunicazione.**)**: può bloccare i pacchetti basandosi

sulle informazioni dell'intestazione TCP (ad esempio, bloccando tutti i pacchetti con SYN=1 per una determinata porta).
- **Firewall di tipo stateful**: Tiene traccia dello stato delle connessioni e può applicare regole più complesse.

Applicazioni Pratiche

Protezione da attacchi SYN flood: Alcuni firewall implementano misure specifiche per mitigare gli attacchi SYN flood, che sfruttano la gestione dei pacchetti SYN per sovraccaricare un server. Gli **attacchi SYN flood** sono una forma di **attacco di tipo Denial of Service (DoS)** in cui un attaccante sfrutta il processo di **handshake a tre vie del protocollo TCP** per consumare risorse del server e renderlo incapace di rispondere a richieste legittime. Vediamo come funziona in dettaglio.

Come funziona il processo normale di handshake TCP

1. Il client invia un pacchetto iniziale con **SYN = 1** per richiedere la connessione.
2. Il server risponde con un pacchetto con **SYN = 1 e ACK = 1** per accettare la connessione.
3. Il client invia un pacchetto finale con **ACK = 1** per completare l'handshake e stabilire la connessione.

Come funziona un attacco SYN flood

1. L'attaccante invia un grande numero di pacchetti con **SYN = 1** al server, richiedendo connessioni.
2. Il server risponde a ciascuna richiesta con un pacchetto **SYN + ACK**, mettendo la connessione in uno stato "mezzo aperto" e in attesa del terzo pacchetto di conferma (**ACK**) dal client.
3. Il client attaccante **non invia mai il pacchetto finale ACK**, lasciando il server in attesa.

Effetto sull'obiettivo

- Ogni connessione incompleta occupa risorse del server, come memoria e capacità di gestione delle connessioni.
- Poiché il server ha un limite sul numero di connessioni simultanee, un attacco SYN flood può rapidamente saturare questa capacità.
- Le richieste legittime dei veri utenti non possono essere servite, provocando un'interruzione del servizio.

Tecniche avanzate utilizzate dagli attaccanti

- **Sorgenti IP falsificate (IP spoofing)**: L’attaccante utilizza indirizzi IP fasulli per rendere difficile tracciare l’origine degli attacchi.
- **Volume elevato di richieste SYN**: Un gran numero di pacchetti viene inviato in rapida successione per sovraccaricare il server.

Contromisure per mitigare un attacco SYN flood

1. **SYN Cookies**: Tecnica in cui il server non riserva risorse finché non riceve il pacchetto ACK finale. Invece di memorizzare lo stato della connessione, genera un cookie che sarà verificato quando arriverà l'ACK.
2. **Riduzione del timeout SYN**: Impostare un tempo limite più breve per le connessioni incomplete.
3. **Limitazione delle connessioni SYN**: Limitare il numero di connessioni incomplete per indirizzo IP.
4. **Firewall e sistemi di prevenzione delle intrusioni (IPS)**: Monitorare e bloccare il traffico sospetto.

Conclusione

L'attacco SYN flood sfrutta una vulnerabilità nel processo di handshake TCP per esaurire le risorse di un server, causando un'interruzione del servizio. Le tecniche di difesa, come i SYN cookies e i limiti di connessione, sono essenziali per mitigare questi attacchi.

Limiti

Bloccare solo il pacchetto SYN potrebbe non essere sufficiente in caso di:

- **Connessioni già stabilite**: Pacchetti successivi non vengono bloccati se la connessione è già attiva.
- **Tecniche di evasione**: Alcuni attaccanti potrebbero tentare di camuffare i pacchetti per bypassare i filtri.

Conclusione

Bloccando i pacchetti con **SYN=1**, è possibile impedire l'inizio di nuove connessioni TCP. Tuttavia, per una sicurezza efficace, è consigliabile combinare questo approccio con altre misure come il monitoraggio dello stato delle connessioni e il filtraggio basato su applicazioni e indirizzi IP.

- Quando si stabilisce una connessione TCP tra due dispositivi, il processo inizia con un meccanismo chiamato **handshake a tre vie**:

1. Il client invia un pacchetto con il flag **SYN** impostato a 1 per richiedere una connessione.
2. Il server risponde con un pacchetto **SYN-ACK**.
3. Il client completa il processo inviando un pacchetto **ACK**.

- **Riconoscere i pacchetti SYN** significa identificare quei pacchetti che iniziano una connessione.

Per le politiche di sicurezza è importante riconoscere i pacchetti di apertura della connessione, in quanto consente ai client interni di connettersi ai server esterni ma può vietare a client esterni di connettersi ai server interni. I campi presenti in un pacchetto TCP sono:

– **URG** (**URGent**);

– **PSH** (**PuSH**);

– **ACK** (**ACKnowledgement**);

– **SYN** (**SYNchronize**);

– **FIN** (**FINish**);

– **RST** (**ReSeT**)

I campi **URG** e **PSH** vengono utilizzati per identificare dati particolarmente critici e permettono di gestire la priorità e l'urgenza dei dati trasmessi, in particolare **PSH** comunica al ricevente di interrompere il **buffering** (processo di **memorizzazione temporanea dei dati** in una memoria intermedia (chiamata **buffer**) prima che vengano elaborati o trasmessi a un livello successivo. Questo meccanismo viene utilizzato per gestire eventuali discrepanze tra la velocità con cui i dati vengono ricevuti e la velocità con cui possono essere processati o consegnati al destinatario) e consegnare i dati allo strato applicativo, mentre **URG** identifica i dati che il mittente considera genericamente importanti. In pratica entrambi possono essere trascurati dai firewall, infatti, pur presenti nell'intestazione TCP dei pacchetti, **non influenzano direttamente le decisioni di sicurezza** prese dalla maggior parte dei firewall. Analizziamo perché:

---

1. Cosa sono i firewall e come funzionano

Un firewall è un dispositivo o software di sicurezza che controlla il traffico di rete in base a **regole predefinite**. Può esaminare vari aspetti dei pacchetti di dati, tra cui:

- **Indirizzi IP di origine e destinazione**
- **Numeri di porta**
- **Protocolli utilizzati (es. TCP, UDP)**
- **Flag TCP come SYN e ACK**, che indicano lo stato della connessione

I firewall analizzano i pacchetti per permettere o bloccare connessioni in base a criteri specifici.

---

2. Perché URG e PSH possono essere trascurati

- **URG** e **PSH** non determinano direttamente l'apertura o il mantenimento di una connessione.
- Questi flag sono utilizzati per gestire **come** i dati vengono trattati a livello applicativo o all'interno del buffer TCP, ma non rappresentano segnali critici per la sicurezza della connessione o l'autenticità dei dati.
- A differenza del flag **SYN**, che è fondamentale per stabilire una nuova connessione e viene esaminato dai firewall per impedire accessi non autorizzati, i flag **URG** e **PSH** non presentano una funzione di controllo della sessione che possa essere utilizzata per attacchi comuni.

I campi **FIN** e **RST** vengono utilizzati per chiudere le connessioni. Il campo **RST** viene utilizzato per una chiusura brutale, mentre **FIN** viene utilizzato per una chiusura concordata tra client e server.

Campo FIN (Finish)

- **Significato:** Il campo **FIN** viene utilizzato per segnalare che un dispositivo (client o server) ha terminato l'invio dei dati e vuole chiudere la connessione in modo **ordinato**.
- **Chiusura concordata:**
  1. Un dispositivo invia un pacchetto con **FIN = 1** per notificare che non ha più dati da inviare.
  2. L'altro dispositivo risponde con un pacchetto **ACK = 1**, confermando di aver ricevuto il messaggio.
  3. Infine, l'altro dispositivo invia a sua volta un pacchetto **FIN = 1** per indicare che anche lui ha terminato di inviare i dati.
  4. Il primo dispositivo risponde con un **ACK = 1**, completando la chiusura della connessione.
- **Esempio pratico:**
  1. Una connessione chiusa con il campo **FIN** garantisce che tutti i dati siano stati correttamente ricevuti e che la connessione si chiuda in modo coordinato.

Campo RST (Reset)

- **Significato:** Il campo **RST** viene utilizzato per segnalare che una connessione deve essere chiusa **immediatamente** in modo **brutale**, senza attendere l'invio o la ricezione di ulteriori dati.
- **Chiusura brusca:**
  - Viene utilizzato quando si verifica un errore critico o una situazione anomala, come:
    1. Un'applicazione riceve dati per una connessione che non esiste più.
    2. Un dispositivo vuole interrompere una connessione per motivi di sicurezza o errore.
  - Quando il campo **RST = 1** viene impostato, tutte le risorse associate alla connessione vengono liberate immediatamente.
- **Esempio pratico:**
  - Se un server non riconosce una richiesta di connessione o riceve dati su una porta inattesa, può inviare un pacchetto con **RST = 1** per interrompere la connessione.

Differenza tra FIN e RST

Campo	Tipo di Chiusura	Quando viene utilizzato
FIN	Ordinata (concordata)	Quando entrambe le parti decidono di chiudere la connessione dopo aver terminato lo scambio di dati.
RST	Brusca (immediata)	Quando si verifica un errore, una situazione anomala o si vuole terminare una connessione in modo rapido.

Un esempio pratico per chiarire

Immagina un server che gestisce un sito web:

1. **Chiusura con FIN:**
   - Dopo che il client ha ricevuto tutta la pagina web, decide di chiudere la connessione in modo ordinato. Entrambi i dispositivi si scambiano pacchetti con il flag **FIN**.
2. **Chiusura con RST:**
   - Se il server riceve una richiesta da un client su una porta chiusa o che non è in uso, invia un pacchetto con **RST** per rifiutare la connessione.

Conclusione

Il campo **FIN** è sinonimo di una chiusura "amichevole" e ordinata della connessione TCP, mentre il campo **RST** rappresenta un'interruzione improvvisa e forzata, solitamente utilizzata in caso di errori o situazioni anomale.

Da questo si deduce che, per garantire la sicurezza e il corretto funzionamento delle comunicazioni di rete, i campi **ACK** e **RST** devono essere verificati da un firewall in quanto:

– **ACK** consente di rilevare in maniera affidabile il primo pacchetto della connessione. Un firewall deve **verificare il campo ACK** per distinguere i pacchetti che fanno parte di una connessione esistente da quelli che cercano di aprire una nuova connessione, allo scopo di bloccare pacchetti falsificati o non autorizzati che tentano di inserirsi in una connessione esistente.

– **RST** fornisce un modo utile per chiudere una connessione senza dover spedire messaggi di errore, viene utilizzato per chiudere una connessione TCP in modo immediato e forzato. Quando questo campo è impostato a **1** un dispositivo desidera terminare una connessione a causa di errori o comportamenti anomali. Ruolo del firewall è bloccare pacchetti RST non autorizzati o sospetti che potrebbero essere utilizzati per interrompere connessioni essenziali o proteggere le connessioni legittime da interruzioni indesiderate.

Per poter ricostruire correttamente i pacchetti ricevuti, il protocollo **TCP** identifica i pacchetti attraverso un numero, chiamato **numero di sequenza**.

(elemento fondamentale del protocollo TCP per garantire la corretta ricostruzione dei dati ricevuti, anche quando i pacchetti arrivano fuori ordine o sono frammentati). Durante una connessione tra due host, ciascun host seleziona il numero di sequenza da cui iniziare lo scambio di pacchetti secondo il protocollo **Threeway Handshake**.

L'attacco SYN consiste nell'inviare un gran numero di richieste SYN a un host con un indirizzo IP sorgente inesistente o non valido e sfrutta una vulnerabilità nel processo di handshake a tre vie del protocollo TCP per sovraccaricare un host e renderlo incapace di rispondere a nuove richieste legittime. (Gli **attacchi SYN** e gli **attacchi SYN flood** si riferiscono allo stesso tipo di attacco, ma con una sfumatura: l'espressione **SYN flood** sottolinea il volume eccessivo di richieste SYN inviate per sopraffare il sistema bersaglio.).

Come funziona un attacco SYN flood

1. **Inizio dell'attacco:**
   - L'attaccante invia un gran numero di pacchetti **SYN** a un host bersaglio, indicando di voler iniziare una connessione TCP.
   - L'indirizzo IP sorgente del pacchetto è falsificato (**spoofing**) e spesso è un indirizzo inesistente o non valido.
2. **Risposta del server:**
   - L'host bersaglio risponde a ogni pacchetto SYN con un pacchetto **SYN-ACK**, in base al normale processo di handshake TCP.
3. **Mancanza di completamento:**
   - L'host bersaglio attende il pacchetto finale **ACK** per completare la connessione.
   - Tuttavia, poiché l'indirizzo IP sorgente è falso, l'host attaccato non riceverà mai il pacchetto ACK.
4. **Risultato dell'attacco:**
   - L'host bersaglio mantiene queste connessioni parzialmente aperte (in uno stato chiamato **half-open**) occupando risorse come memoria e processi di rete.
   - Dopo un certo numero di connessioni non completate, l'host non sarà più in grado di gestire nuove richieste, incluse quelle legittime.

Perché è efficace?

- **Consumo di risorse:**
  - L'host bersaglio mantiene uno stato per ogni connessione parzialmente aperta, utilizzando memoria e altre risorse.
- **Spoofing IP:**
  - L'attaccante può inviare richieste SYN da indirizzi IP falsi, rendendo difficile tracciare l'origine dell'attacco.
- **Volume elevato:**
  - Un attaccante può utilizzare botnet o altri strumenti per inviare milioni di pacchetti SYN in pochi secondi, sovraccaricando rapidamente l'host.

Contromisure

1. **SYN cookie:**
   - Una tecnica in cui il server non riserva risorse per una connessione finché non riceve il pacchetto ACK.
2. **Timeout più breve:**
   - Ridurre il tempo in cui una connessione **half-open** viene mantenuta.
3. **Limitazione delle connessioni per IP:**
   - Il firewall può limitare il numero di connessioni aperte da un singolo indirizzo IP.
4. **Firewall e sistemi IDS/IPS:**
   - Monitorano il traffico di rete per rilevare comportamenti anomali e bloccare le richieste SYN sospette.
5. **Protezione a livello di rete:**
   - Molti provider di servizi offrono protezioni anti-DoS a livello infrastrutturale, in grado di filtrare il traffico malevolo.

Come funziona il numero di sequenza?

1. **Definizione del numero di sequenza:**
   - Ogni pacchetto TCP contiene un **numero di sequenza** (Sequence Number) nel suo header.
   - Questo numero identifica la posizione del primo byte di dati nel flusso del segmento che il pacchetto sta trasmettendo.
2. **Inizializzazione:**
   - Quando una connessione TCP viene stabilita, il numero di sequenza iniziale (ISN - **Initial Sequence Number**) viene scelto casualmente da entrambe le parti (client e server).
   - Questo serve a proteggere la connessione e prevenire attacchi come l'**hijacking**.
3. **Tracciamento dei dati:**
   - Il numero di sequenza aumenta con ogni byte di dati inviato.
   - Se un pacchetto contiene 100 byte di dati, il numero di sequenza del pacchetto successivo sarà incrementato di 100.
4. **Conferma della ricezione (ACK):**
   - La parte ricevente utilizza il **campo ACK (Acknowledgment)** per informare il mittente di quali dati sono stati ricevuti correttamente.
   - Il valore dell'ACK corrisponde al **numero di sequenza del prossimo byte** che ci si aspetta di ricevere.

Esempio pratico:

- **Scenario:**
  - Il client vuole inviare 300 byte di dati al server.
- **Passi:**
  - Il client invia un pacchetto TCP con un numero di sequenza iniziale, ad esempio **1000**, e 100 byte di dati.
    - Numero di sequenza del pacchetto = **1000**
    - Il server legge i primi 100 byte.
  - Il server invia un ACK al client con il valore **1100**.
    - Questo significa che il server ha ricevuto i primi 100 byte e si aspetta il prossimo byte a partire da 1100.
  - Il client invia un secondo pacchetto con altri 100 byte.

- Numero di sequenza del pacchetto = **1100**

Perché è importante il numero di sequenza?

1. **Ricostruzione dei dati:**
   - I numeri di sequenza permettono al ricevente di ordinare i pacchetti, anche se arrivano fuori ordine.
2. **Gestione di pacchetti duplicati:**
   - Il numero di sequenza aiuta a identificare e scartare eventuali pacchetti duplicati.
3. **Ritrasmissioni:**
   - In caso di pacchetti persi, il ricevente può richiedere ritrasmissioni specifiche basandosi sul numero di sequenza.