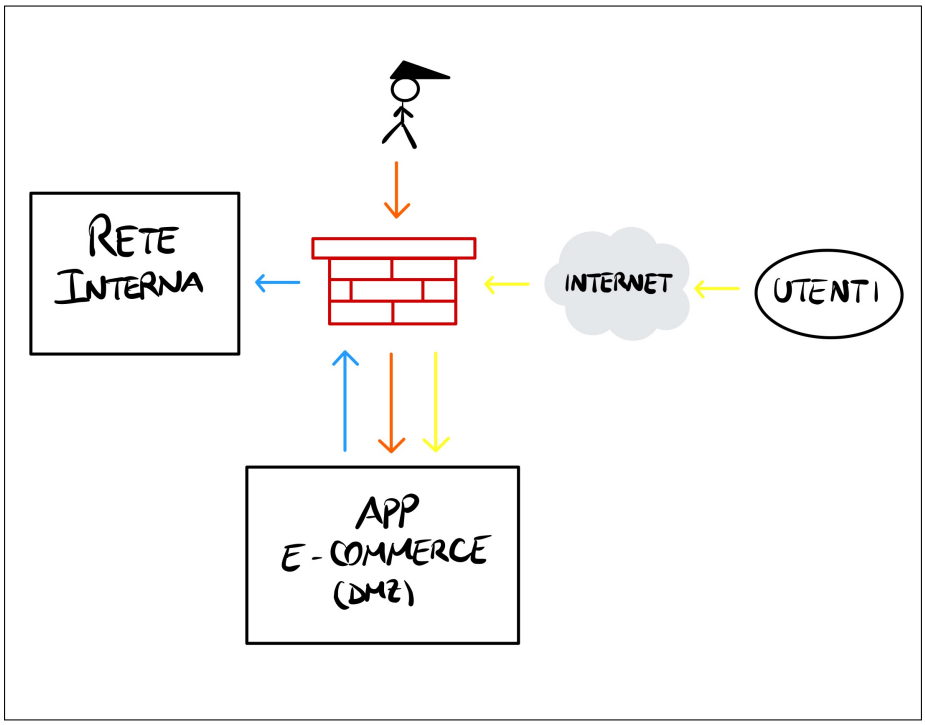
ARCHITETTURA DI RETE

XSS, SQLi, DDoS, Malware
Alessandro Morabito @ Epicode

1 Introduzione

In questo progetto ci è data una configurazione di rete come in figura:



. Ci viene quindi chiesto quali sono delle possibile azioni preventive che possono essere implementate per difendere l'applicazione e-commerce in zona *DMZ* da attacchi *SQLi* e *XSS*.
Successivamente ci viene chiesto di stimare la perdita a livello economico a seguito di un attacco *DDoS* andato a buon fine, supponendo che il server non sia raggiungibile dagli utenti per 10 minuti e quanto questi spenderebbero, al minuto, in una situazione normale.
Infine, immaginando uno scenario in cui il nostro server venga infettato da un malware, modificare l'architettura della rete facendo in modo di non permettere la diffusione di tale malware all'interno di essa, ma non rimuovendo l'accesso all'attaccante.

2 Scenario 1

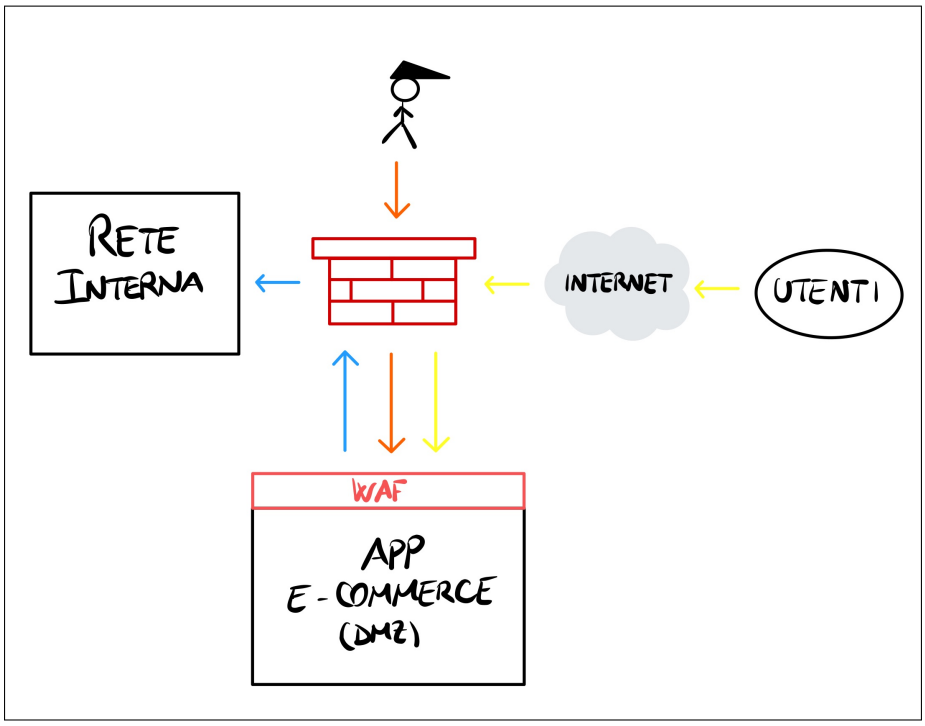
2.1 XSS

Il *Cross-Site Scripting* è un attacco che permette l'esecuzione di uno script malevolo a livello di client. A seconda del metodo di implementazione dell'attacco, questo può assumere nomi diversi. Ricordiamo il *XSS Stored*, dove il codice viene salvato direttamente in una pagina del server, e il *XSS Reflected*, dove il codice viene diffuso all'interno di un parametro *GET* nell'URL, che successivamente, malevolo, viene diffuso.
Le conseguenze di questo tipo di attacco sono molto varie, a seconda del codice malevolo che può essere eseguito, ma coinvolgono generalmente le persone che visitano direttamente la pagina.

2.2 SQLi

La *SQL Injection* è un attacco che, come il *XSS*, è causato da un inefficace filtro e sanificazione dell'input dell'utente. Con questo attacco, però, il codice iniettato sono stringhe del linguaggio di query *SQL*, che permette la comunicazione con un database. Un attaccante, in particolare, può potenzialmente enumerare gli elementi all'interno di un database e ottenere accesso a dati confidenziali, oppure effettuare login aggirando il sistema di autenticazione. Dal momento che l'attaccante interagisce direttamente con il database, non è necessaria la condivisione di link o l'accesso a una determinata pagina da parte delle vittime.
Una *SQLi* si dice *Blind* quando il server non fornisce risposte circa errori nella query inserita. In questo caso un attaccante, per verificare se è presente la vulnerabilità (e talvolta anche per l'attacco in sé), utilizza un approccio "indiretto" come espressioni booleane o temporali per controllare le variazioni del comportamento del server.

Dal punto di vista dell'architettura della rete, per difendersi da entrambi questi tipi di attacchi si può implementare un *Web Application Firewall*, che analizza le richieste verso il server e le confronta con un database per verificare la presenza di codice malevolo.



La soluzione appena proposta non richiede modifiche all'applicazione in sé e fornisce un buon livello di protezione ma può essere costosa. Modificando il codice a livello dell'applicazione, si possono utilizzare funzioni per sanare l'input utente.

3 Scenario 2

3.1 DDoS

Il *Distributed Denial of Service* è un tipo di attacco che coinvolge un alto numero di dispositivi, tendenzialmente appartenenti a una *botnet* malevola. Questi dispositivi sono anch'essi vittime e eseguono l'attacco dopo essere stati compromessi. In particolare, essi inviano un elevato numero

di richieste con l'obbiettivo di sovraccaricare il server. Se un attacco *DDoS* va a buon fine, il server non riesce a gestire tutte queste richieste e diventa non più raggiungibile. In alternativa, l'attacco potrebbe non voler mirare necessariamente a provocare una disconnessione del server, ma solamente a sovraccaricarlo per rendere più efficaci altri attacchi lanciati in successione (*attacco a due tempi*). Per difendersi da un attacco *DDoS* la migliore opzione è la distribuzione del carico su un *Content Delivery Network*, ovvero la distribuzione delle richieste su server diversi.

Seguendo i dati fornitici dall'esercizio, sappiamo che gli utenti spendono in media €1500 in un minuto. Supponendo che il server non sia raggiungibile per 10 minuti e che i soldi spesi dagli utenti corrispondono allo stesso guadagno da parte nostra, allora in quei 10 minuti non possiamo guadagnare €15000.

4 Scenario 3

Supponiamo che il nostro server sia stato infettato da un malware, di voler impedire che la nostra rete possa essere infettato proprio da quel malware, e di voler lasciare il collegamento aperto tra il black hat e il server.
Si procede con la pratica di *isolamento*, per impedire qualunque tipo di connessione tra la rete interna e l'applicazione e-commerce mettendo queste due sottoreti in condizione di *air gap*, ad esempio tramite *Network Access Control* sfruttando la segmentazione.
Facendo in questo modo il black hat continuerebbe ad avere accesso al server, così come gli utenti. Questo potrebbe permettere la diffusione del malware verso gli utenti. Per quanto si potrebbe essere arrivati a questa scelta considerando sia i danni economoci sia di reputazione, non è qualcosa che ritengo moralmente accettabile.