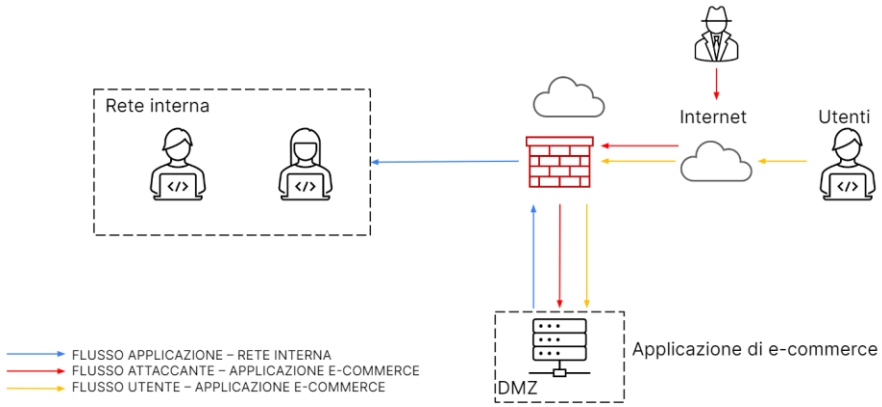


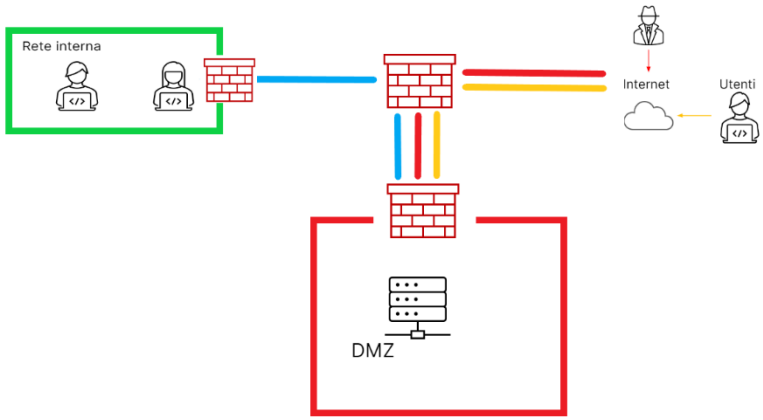
1. **Azioni preventive**: quali azioni preventive si potrebbero implementare per difendere l'applicazione Web da attacchi di tipo SQLi oppure XSS da parte di un utente malintenzionato?
   Modificate la figura in modo da evidenziare le implementazioni
2. **Impatti sul business**: l'applicazione Web subisce un attacco di tipo DDoS dall'esterno che rende l'applicazione non raggiungibile per **10 minuti**.
   Calcolare l'impatto sul business dovuto alla non raggiungibilità del servizio, considerando che in media **ogni minuto gli utenti spendono 1.500 €** sulla piattaforma di e-commerce. **Fare eventuali valutazioni di azioni preventive che si possono applicare in questa problematica**
3. **Response**: l'applicazione Web viene infettata da un malware.
   La vostra priorità è che il malware non si propaghi sulla vostre rete, mentre non siete interessati a rimuovere l'accesso da parte dell'attaccante alla macchina infettata.
   Modificate la figura in slide 2 con la soluzione proposta.
4. **Soluzione completa**: unire i disegni dell'azione preventiva e della response (unire soluzione 1 e 3)
5. **Modifica «più aggressiva» dell'infrastruttura (se necessario/facoltativo magari integrando la soluzione al punto 2)**

1- Difesa contro SQL Injection e Cross-Site Scripting (XSS)

1) **Web-App Firewall (WAF)**
   Possiamo configurare un firewall con regole di filtraggio ad-hoc. Il WAF può eseguire una validazione e un filtraggio avanzati sui dati di input per rilevare e bloccare i caratteri speciali o le sequenze di *escape* utilizzate negli attacchi XSS. Può anche rilevare tentativi di inserire script dannosi in campi di input come moduli, URL o parametri di query. Il WAF può analizzare i tipi di contenuto inviati dal server e rilevare se ci sono potenziali rischi di XSS. Ad esempio, può rilevare la presenza di script incorporati in pagine HTML o JavaScript non correttamente sanificato.
   Inoltre il WAF può analizzare le query SQL inviate all'applicazione web e rilevare caratteri o parole chiave sospette utilizzate negli attacchi SQLi. Può bloccare o sanificare le query per prevenire l'esecuzione di codice SQL dannoso. Un WAF può monitorare il comportamento delle query SQL e rilevare anomalie o modelli sospetti che potrebbero indicare un attacco SQLi in corso. Ad esempio, può rilevare un alto numero di richieste di query o interrogazioni che cercano di accedere a dati sensibili.
2) **Sanificazione input utente**
   Dobbiamo assicurarci che tutti i dati ricevuti dall'utente o da altre fonti esterne siano correttamente validati. Filtrare e sanificare i dati di input per rimuovere caratteri speciali e sequenze di *escape* che potrebbero essere utilizzati per inserire script dannosi.
3) **IDS e IPS** possono svolgere un ruolo importante nella prevenzione degli attacchi SQLi (SQL Injection) e XSS (Cross-Site Scripting). Vediamo come possono contribuire a mitigare questi tipi di attacchi. Un IDS/IPS può utilizzare una base di dati di firme che elenca modelli noti di attacchi SQLi e XSS. Quando rileva un'interrogazione che corrisponde a una firma conosciuta, può segnalare l'evento o intraprendere azioni preventive per bloccare l'attacco. Inoltre un IDS/IPS può monitorare il

comportamento degli script all'interno dell'applicazione web per rilevare pattern sospetti o attività anomale che potrebbero indicare un attacco XSS o SQLi in corso.

4) **“*Prepared Statemets*”**: funzionano creando una query SQL precompilata che separa i dati dall'istruzione SQL stessa. Invece di concatenare i valori direttamente nell'istruzione SQL, i "*prepared statements*" utilizzano segnaposto (*placeholder*) per i parametri. Questi segnaposto sono quindi sostituiti con i valori corrispondenti durante l'esecuzione della query.
   Poiché i "*prepared statements*" separano i dati dalle istruzioni SQL, garantiscono che i dati di input vengano trattati come dati, evitando l'interpretazione errata come codice. Ciò impedisce efficacemente l'iniezione di script dannosi che caratterizza gli attacchi XSS. Per quanto riguarda la prevenzione dal SQL, i parametri vengono passati separatamente alla query e il motore del database si occupa di eseguire la query in modo sicuro. Ciò impedisce agli attaccanti di iniettare codice SQL dannoso all'interno dei dati di input. I "*prepared statements*" offrono un meccanismo di *escape* e di sanitizzazione automatico dei parametri di input, in quanto il motore del database si assicura che siano correttamente formattati come valori di dati e non interpretati come codice SQL.
5) **Limitazione dei privilegi**: ridurre l'area di esposizione potenziale di un'applicazione o di un sistema. Ad esempio, se un'applicazione web viene compromessa, un utente con privilegi limitati avrà meno opportunità di sfruttare la vulnerabilità per scopi dannosi rispetto a un utente con privilegi elevati. Ciò aiuta a contenere l'impatto di un attacco XSS o SQLi e a limitare la potenziale compromissione del sistema.



2- Impatti sul Business

Calcolo della perdita: **1500** € x **10** m = **15000** €
Possibili azioni preventive/*remediation*

a- **Migrazione del servizio in Cloud**
   La prima cosa da fare è identificare la fonte dell’attacco, che può essere fatta tramite l’analisi del traffico della rete o l’utilizzo di strumenti specializzati.
   Una volta identificata la fonte, è possibile bloccare il traffico dannoso o redirigerlo verso una soluzione di mitigazione **DDoS**, come un servizio di protezione cloud.

b- **Implementazione di un WAF – Web App Firewall**
   Configura un firewall con regole di filtraggio per bloccare il traffico sospetto proveniente da indirizzi IP noti per essere associati ad attacchi **DDoS**. Per ulteriori riferimenti al funzionamento del firewall, si rimanda al capitolo 1.

c- **Bilanciamento del carico – Load Balancer**
   Utilizzando un sistema di bilanciamento del carico per distribuiremmo il traffico in modo uniforme tra più server o risorse. In modo tale da ridurre del carico su server singoli. Infatti, un attacco **DDoS** mira spesso a sovraccaricare un server o un'applicazione specifica rendendola inaccessibile. Con un bilanciamento del carico, il traffico viene distribuito tra diversi server,

riducendo la pressione su ciascun server individuale e consentendo loro di gestire meglio le richieste.

d- **Round-Robin**: Il traffico viene distribuito sequenzialmente tra i server disponibili in modo equo. Ogni richiesta viene inviata al server successivo nella lista.

e- **Least connections**: Il traffico viene inviato al server con il numero di connessioni attive più basso. In questo modo, i server meno carichi ricevono proporzionalmente più richieste rispetto a quelli più carichi.

Ridondanza

La ridondanza si riferisce alla duplicazione critica di componenti o infrastrutture di sistema in modo da avere una copia di backup disponibile nel caso in cui un componente primario fallisca o venga compromesso. Nella prevenzione degli attacchi DDoS, la ridondanza è essenziale per mantenere la continuità operativa durante un attacco.

Potremmo implementare un'architettura ridondante distribuendo le risorse su più server in diversi luoghi geografici. In caso di attacco DDoS su un determinato server, gli utenti possono essere indirizzati automaticamente verso server alternativi, garantendo la continuità del servizio.

Ridondanza di **rete**: Assicuriamoci di avere più percorsi di connettività di rete o diversi fornitori di servizi Internet (ISP) per evitare che un singolo punto di fallimento (*single point of failure*) possa interrompere l'accesso alla tua applicazione. La ridondanza di rete può aiutare a mitigare gli effetti di un attacco DDoS che mira a sovraccaricare un'infrastruttura specifica.

Ridondanza dei **dati**: Esegui copie di backup regolari dei tuoi dati critici e archiviali in sistemi separati. In caso di un attacco DDoS che potrebbe compromettere i dati, avremo una copia di backup disponibile per il ripristino.

Scalabilità

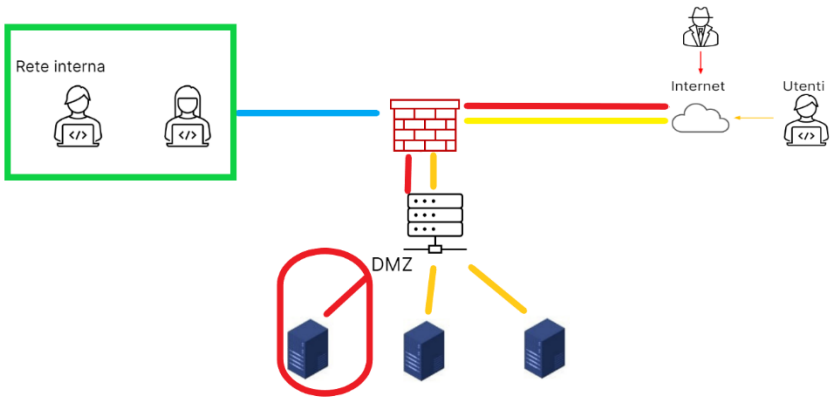
La scalabilità si riferisce alla capacità di espandere o ridurre le risorse di un sistema in modo flessibile per far fronte ai picchi di traffico o alle richieste elevate. Nel contesto della prevenzione degli attacchi DDoS, la scalabilità è importante per garantire che il sistema possa gestire un aumento improvviso del traffico durante un attacco senza compromettere la disponibilità del servizio.

Scalabilità orizzontale: La scalabilità orizzontale consiste nell'aggiungere risorse aggiuntive, come server, in modo da poter distribuire il carico tra di essi. Questo può essere fatto utilizzando tecnologie come il bilanciamento del carico, in cui il traffico viene distribuito tra più server, consentendo di gestire un maggior numero di richieste.

Scalabilità verticale: La scalabilità verticale riguarda l'aumento delle risorse su un singolo server, come l'aggiunta di CPU, memoria o capacità di storage, per gestire un carico maggiore. Questo approccio può essere utile quando un server specifico diventa un punto di congestione durante un attacco DDoS.

La scalabilità e la ridondanza lavorano insieme per garantire che la tua infrastruttura sia in grado di gestire sia i picchi di traffico normali che gli attacchi DDoS. Un'architettura scalabile e ridondante può distribuire il carico tra più risorse e garantire la disponibilità del servizio anche durante un attacco.

3- Response



Poiché non siamo interessati a rimuovere il malware dal server infetto, una possibile “scaletta” di azione potrebbe rispettare i seguenti punti:

Isolamento del server

L'isolamento di un server infetto da malware è un'azione critica per garantire che la nostra Web-App di eCommerce possa continuare normalmente con la sua attività nonostante la presenza del malware. Isolare il server infetto aiuta a limitare l'accesso e la diffusione del malware ad altri componenti del sistema, come *endpoints* o server di *Disaster Recovery* e può contribuire a mantenere la continuità operativa del business. Isolare il server ci aiuta a:

Proteggere i dati sensibili: L'isolamento del server infetto aiuta a proteggere i dati sensibili dell'applicazione, come informazioni personali degli utenti, dettagli delle transazioni o dati di pagamento. Il malware presente sul server potrebbe tentare di sottrarre o compromettere tali informazioni, ma l'isolamento messo in atto limita l'accesso del malware a tali dati.

Permette la continuità operativa: L'isolamento del server infetto consente alla Web-App di eCommerce di continuare a funzionare normalmente nonostante la presenza del malware. Isolando il server infetto, gli utenti possono continuare a utilizzare l'applicazione e accedere ai servizi offerti senza interruzioni significative o di lungo periodo.

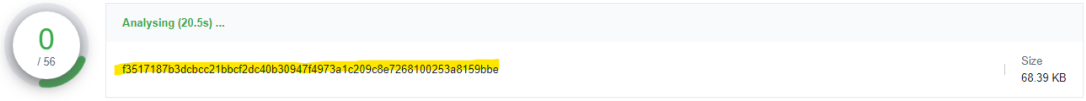
Previene della diffusione del malware: Il malware può diffondersi rapidamente a livello di rete o di sistema, compromettendo altre risorse o gli stessi utenti sulla piattaforma. L'isolamento del server infetto riduce il rischio di diffusione del malware ad altri server, database o componenti dell'infrastruttura, contribuendo a mantenere il resto dell'applicazione e dei servizi al sicuro da potenziali attacchi.

Controffensiva:

Inserimento degli IOC (indicatori di compromissione) nei nostri servizi di cybersecurity.

Abbiamo bisogno di due IOC in particolare.

1. L’hash del malware
   Per ottenere l’hash del malware possiamo utilizzare strumenti di Osint come VirusTotal.com o



   comandi come openssl

```
openssl md5 /percorso/del/file
```

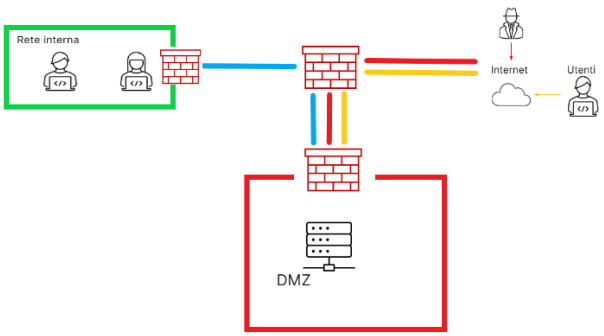
2. L’indirizzo IP dell’attaccante/i
   Analizziamo il log di acceso del server alla ricerca di richieste di caricamento di un file con metodi come HTTP POST o PUT. Cerchiamo i dettagli della richiesta di caricamento trovando l’indirizzo IP dell’attaccante.

Con i due IOC a nostra disposizione, inseriamo i dati all’interno dei nostri sistemi di sicurezza implementati, principalmente IDS/IPS e il FireWall ad-hoc.

Dopo esserci assicurati che i dati degli utenti e della nostra infrastruttura sono al sicuro, ovvero che la *business continuity* sia stata rispettata, possiamo procedere ad una analisi e, in seguito, della rimozione del malware. L'isolamento del server infetto,infatti, fornisce un ambiente controllato per l'analisi e la rimozione del malware. Gli amministratori di sistema o il SOC aziendale possono lavorare in modo sicuro nel server isolato per identificare la natura del malware, analizzarne il comportamento e sviluppare una strategia efficace per eliminarlo in modo completo.

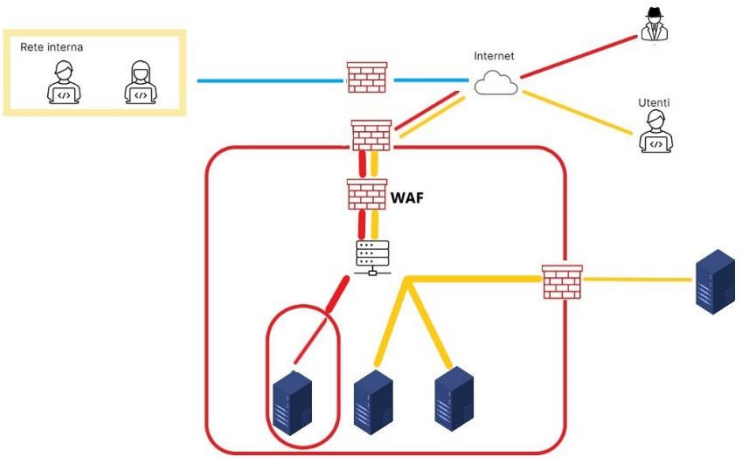
Dopo aver rimosso il malware dal server infetto, è possibile ripristinare la sicurezza dell'ambiente e delle risorse associate. Ciò può includere la scansione e la pulizia (possiamo assicurarci una pulizia completa del server dopo 7 pulizie totali https://www.ccleaner.com/knowledge/the-ultimate-guide-how-to-wipe-your-drive-and-destroy-your-data?cc-noredirect=) dei file e delle applicazioni, l'aggiornamento dei sistemi operativi e delle applicazioni con patch di sicurezza.

4- Soluzione completa



Utilizziamo come da immagine proposta, un servizio di sicurezza a perimetro della nostra DMZ.

5- Soluzione “aggressiva” (o paranoica)



A completamento delle soluzioni di sicurezza proposte nel presente report, possiamo individuarne una che assicuri un elevato livello di sicurezza portando come compromesso il solo rallentamento delle “fasi critiche”.

L’infrastruttura si compone di due firewall perimetrali, uno per la nostra web-app ed uno per un server esterno che funge da database, all’interno del quale sono conservati tutti i dati sensibili degli utenti (dati personali ed anagrafici, numero di carta di credito e così via). Il server esterno è raggiungibile solo ed esclusivamente così.

La rete interna aziendale è isolata da un terzo firewall, che ci garantisce l’accesso ad internet.

Vediamo in figura l’isolamento del server infetto, mentre sono presenti ulteriori due server di *disaster recovery* che ci garantiscono sia un traffico meno invasivo anche durante picchi di servizio, sia la possibilità di far rimanere attivo il servizio ance in caso di problematiche ad uno dei server.

L’accesso dell’utente *triggera* una richiesta da parte di uno dei server “normali” al server di *database*, che ci fornisce la corrispondenza dei dati o autorizza il pagamento per un prodotto. Il leggero ritardo che il *login* o il pagamento innescano è un *trade-off* accettabile per garantire la sicurezza dei dati degli utenti.

Il *web-app firewall* e gli ulteriori servizi di sicurezza implementati (IPS e IDS su tutti), invece, ci garantiscono una forte risposta preventiva ad eventuali comportamenti sospetti degli utenti malintenzionati o intrusioni non autorizzate.

Possiamo in seguito, considerare di applicare una politica di *zero-trust* o migrare l’intera infrastruttura in *cloud,* dove “subappaltiamo” la protezione del nostro eCommerce a servizi esterni all’azienda.