

# Introduzione

## Normative

In azienda la sicurezza informatica è strutturata a piramide, si parte dal livello più in basso, il numero 4. Molto spesso le aziende preferiscono partire dal primo livello che contiene le normative e le regole senza però sapere qual'è il livello di conoscenza aziendale sull'argomento informatico.



Esistono diverse caratteristiche che bisogna seguire per creare le nuove normative, e sono:

- Proporzionalità [+ IMPORTANTE]: la sicurezza non deve mai essere non sostenibile da una azienda;
- Prevenzione: le norme devono prevenire gli incidenti e cercare di arrivare preparati in caso di attacco sapendo già le azioni da fare;
- Condivisione: cioè riuscire a condividere con tutti le metodologie di attacchi e come proteggersi;
- Uniformità: far sì che le normative siano applicabili fra loro e che tutte siano consone fra loro.

## GDPR

Regolamentazione europea nata per tutelare gli utenti finali e i loro dati, tutte le aziende che operano in territorio europeo (anche con sede all'estero) devono sottostare al GDPR.

Il GDPR si applica ai dati personali di cittadini europei che vengono trattati in maniera interamente o parzialmente automatizzata.

I dati coinvolti sono:

Categoria	Tipologia dato
<b>Dati personali</b>	<p><b>Dati che identificano una persona:</b></p> <ul style="list-style-type: none"> <li>• Nome</li> <li>• Numero di identificazione (codice fiscale)</li> <li>• Dati relativi all'ubicazione</li> <li>• Identificativo online</li> <li>• ...</li> </ul>
<b>Dati particolari</b>	<p><b>Dati che rivelano:</b></p> <ul style="list-style-type: none"> <li>• Origine etnica</li> <li>• Opinioni politiche</li> <li>• Religione</li> <li>• Appartenenza sindacale</li> <li>• Dati genetici</li> <li>• Dati medici</li> <li>• ...</li> </ul>

Nel GDPR si evidenziano anche alcuni attori:

Ruolo	Descrizione
Interessato	Persona fisica proprietaria dei dati trattati.
Titolare del trattamento	La persona fisica o giuridica, l'autorità pubblica, il servizio o altro organismo che singolarmente (o insieme ad altri) determina finalità e i mezzi del trattamento dei dati personali.
Responsabile del trattamento	La persona fisica o giuridica, l'autorità pubblica, il servizio o l'organismo, che tratta i dati personali per conto del Titolare del trattamento.
DPO ( <i>Data Protection Officer</i> )	Può essere un dipendente del titolare oppure un esterno. Ha il compito di Sorvegliare che il GDPR venga rispettato e fornire consulenza a titolare e responsabile in merito ai loro obblighi e misure da adottare. Inoltre, funge da contatto con le autorità di controllo.
Destinatario	La persona fisica o giuridica, l'autorità pubblica, il servizio o un altro organismo che riceve comunicazione di dati personali, che si tratti o meno di terzi
Terzo	La persona fisica o giuridica, l'autorità pubblica, il servizio o altro organismo che non sia l'interessato, il titolare del trattamento, il responsabile del trattamento e le persone autorizzate al trattamento dei dati personali sotto l'autorità diretta del titolare o del responsabile.
Rappresentante	Colui il quale agisce per conto del titolare o del responsabile del trattamento e li rappresenta per quanto riguarda i loro obblighi ai sensi del GDPR.
Impresa	La persona fisica o giuridica, indipendentemente dalla forma giuridica rivestita, che esercita un'attività economica, comprendente le società di persone o le associazioni che esercitano regolarmente un'attività economica.

I diritti dell'interessato:

Diritto	Descrizione
Diritto di opposizione	Ha il diritto di opporsi in qualsiasi momento, per motivi connessi alla sua situazione particolare, al trattamento dei dati personali che lo riguardano connessi a ragioni di interesse pubblico o all'esercizio di pubblici poteri.
Diritto di limitazione	Il diritto di limitazione (art. 18 del regolamento) consente all'interessato di ottenere il blocco del trattamento.
Diritto alla portabilità dei dati	Si applica solo ai trattamenti automatizzati basati sul consenso o sulla necessità contrattuale, e sono previste specifiche condizioni per il suo esercizio.
Diritto all'oblio	Consente di ottenere la cancellazione dei propri dati personali in casi particolari. Può essere esercitato anche dopo la revoca del consenso.
Diritto di rettifica	L'interessato (art. 16 GDPR) può rivolgersi al titolare del trattamento per ottenere la rettifica dei dati personali inesatti che lo riguardano senza ingiustificato ritardo. Tenuto conto delle finalità del trattamento, ha anche il diritto di ottenere l'integrazione dei dati incompleti.
Diritto di accesso	Il diritto di ottenere dal titolare del trattamento la conferma che sia o meno in corso un trattamento di dati personali che lo riguardano, e in caso positivo di ottenere l'accesso ai dati personali.

Il diritto di portabilità dice che per esempio una banca quando chiudete il conto per farlo in un'altra è obbligata a fare il passaggio dei dati per voi.

Il GDPR non impone normative tecniche precise ma è abbastanza generico e astratto, ma impone misure di sicurezza per non cadere in sanzioni:

Categoria	Descrizione
Privacy <i>by design</i> e <i>by default</i>	La soluzione con cui si trattano i dati personali deve, previe particolari configurazioni impostate dal Responsabile del trattamento o dal Titolare, prediligere i metodi che garantiscono il livello di privacy più alto. Deve, inoltre, prediligere che per impostazione predefinita siano trattati solamente i dati strettamente necessari per ogni specifica finalità del trattamento.
Pseudonimizzazione	Procedimento con il quale s'impedisce di identificare un individuo attraverso i suoi dati.
Accountability	Sapere rispondere e rendere conto dei risultati ottenuti o di quanto sia stato fatto in merito al trattamento dei dati personali.

alcune soluzioni reali invece per proteggere i dati sono:

- Anonimizzazione;
- Separazione;
- Conservazione cifrata e separata;
- Cancellazione sicura e che avviene appena non più necessari;
- Cifratura sia a riposo che quando trasmetti;
- Applicare il principio del need-to-know;
- Definire strategie di risposta agli incidenti e di prevenzione e gestione del rischio a livello aziendale;
- Richiedere e mantenere il minimo dei dati strettamente necessari.

In caso non si rispettino le soluzioni riportate sopra si può intaccare in un data breach, un data breach consiste in un evento per il quale si può compromettere la riservatezza, l'integrità e la disponibilità dei dati a perimetro.

Questa casistica può avvenire tramite alcuni di questi attacchi:

- comunicazione via email;
- comunicazione fisica/di persona;
- attacchi informatici fisici.

In caso di attacchi il Titolare e/o il Responsabile sono tenuti a notificarne l'avvenimento alle autorità entro 72 ore tramite lesson learning che riporta:

- la natura della violazione
- nome e i dati di contatto del DPO
- probabili conseguenze
- misure adottate

ma va anche documentato ogni violazione ai dati personali e tenere traccia delle comunicazioni tra le due figure.

Il GDPR indica anche le varie responsabilità, infatti il testo dice:

Chiunque subisca un danno materiale o immateriale causato da una violazione del presente regolamento ha il diritto di ottenere un risarcimento del danno dal titolare del trattamento o dal responsabile del trattamento.

Il titolare coinvolto risponde per il danno cagionato dalle sue azioni che violano il GDPR.

Se il responsabile non rispetta volutamente le indicazioni del titolare sarà lui a rispondere al risarcimento.

## PCI

L'ente che ha redatto i vari standard è il PCI Council, opera a livello globale a cui partecipano tutte le società che si occupano di pagamenti.

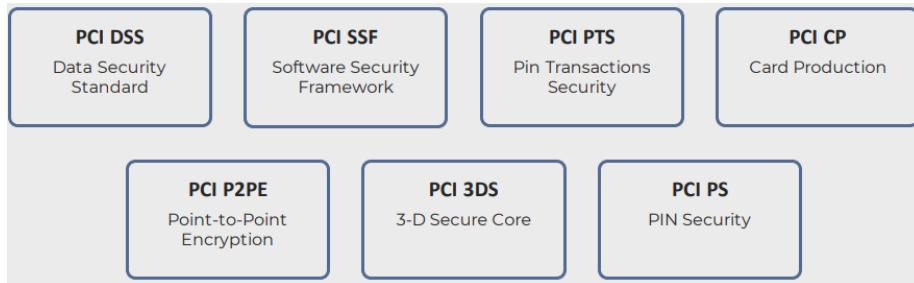
Va detto che in America è obbligatorio seguire le norme del PCI, in Europa no anche se fortemente consigliato visto che tutte le aziende che non lo fanno probabilmente non riusciranno a stringere patti con altri enti.

Il suo scopo è guidare l'adozione di standard uniformi al livello di sicurezza riguardo le informazioni e fornendo risorse per la messa in sicurezza dell'ecosistema dei pagamenti.

I compiti principali sono:

- Definire standard di sicurezza per i pagamenti a livello globale
- Validare e elencare prodotti e soluzioni che sono conformi con gli standard PCI (es. PCI SSC) e i relativi requisiti
- Qualificare organizzazioni
- Fornire un insieme di best practice per la sicurezza dei pagamenti

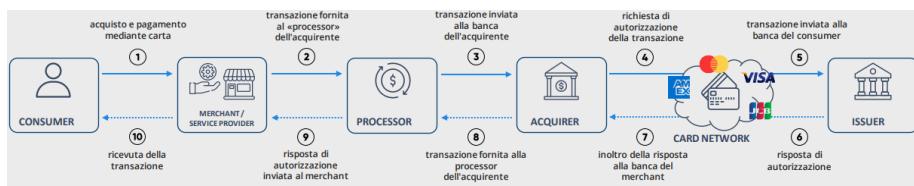
Gli standard rilasciati dal PCI Council sono 7 e sono:



Sono presenti 8 principali ruoli definiti da PCI:

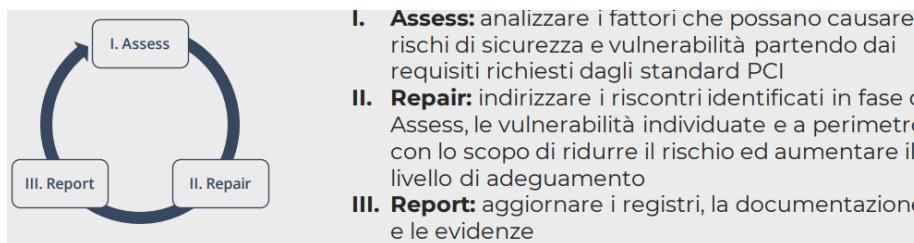


Ora possiamo vedere come i vari ruoli sono collocati nel processo di pagamento:

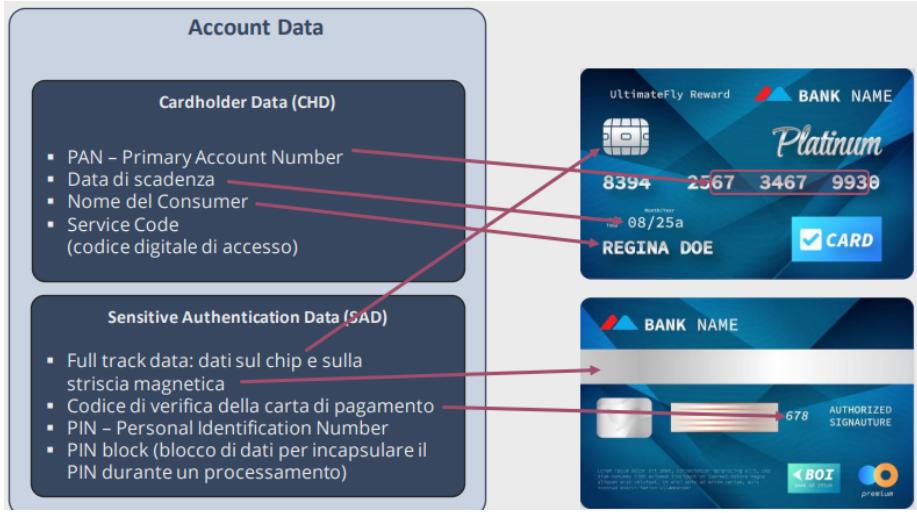


il card network cambia in base alla carta.

Per aderire al PCI è necessario attuare un processo continuo di compliance e di sicurezza, strutturato secondo le seguenti fasi:



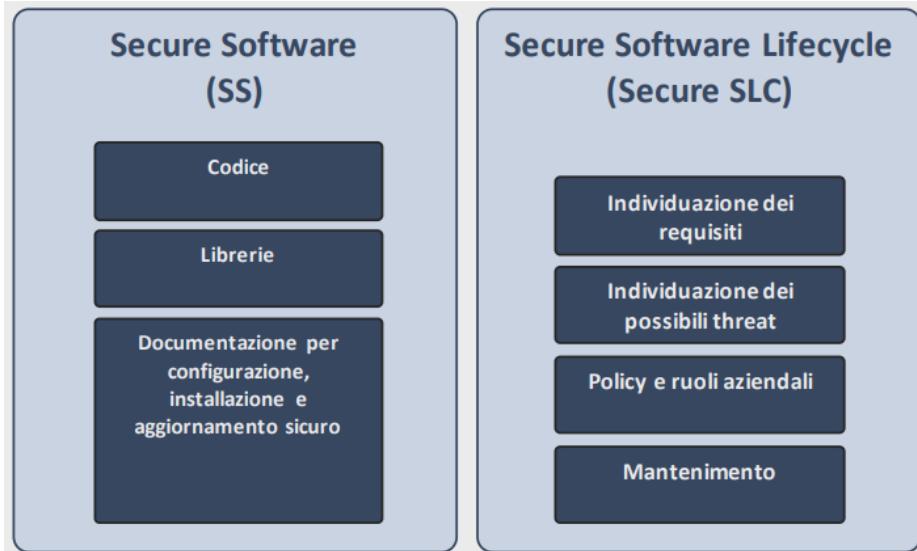
I dati coinvolti dallo standard PCI sono detti account data, e sono divisi fra cardholder data e sensitive authentication data:



## PCI SSF

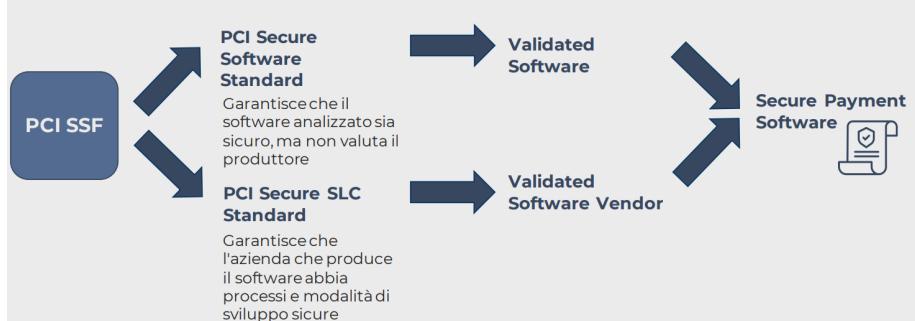
Il PCI SSF (Secure Software Framework) è uno standard PCI che norma lo sviluppo del software in maniera sicura. È composto da due parti, ognuna delle quali si occupa di differenti aspetti applicativi:

- PCI Secure Software
- PCI Secure Software Lifecycle



Seguendo il PCI SFF ti permettere di creare un secure payment software:

Un **Secure Payment Software** è un software validato secondo lo standard PCI SFF:



## PCI DSS

Lo scopo del PCI DSS è quello di mettere in sicurezza l'ambiente in cui vengono utilizzati gli Account Data normando quindi:

- ✓ Componenti del sistema, persone e processi che memorizzano, elaborano e trasmettono dati del titolare della carta e/o dati di autenticazione sensibili
- ✓ Componenti del sistema che potrebbero non memorizzare, elaborare o trasmettere Account Data ma hanno connettività non limitata a componenti del sistema che memorizzano, elaborano o trasmettono i suddetti
- ✓ Componenti del sistema, persone e processi che potrebbero influire sulla sicurezza degli Account Data

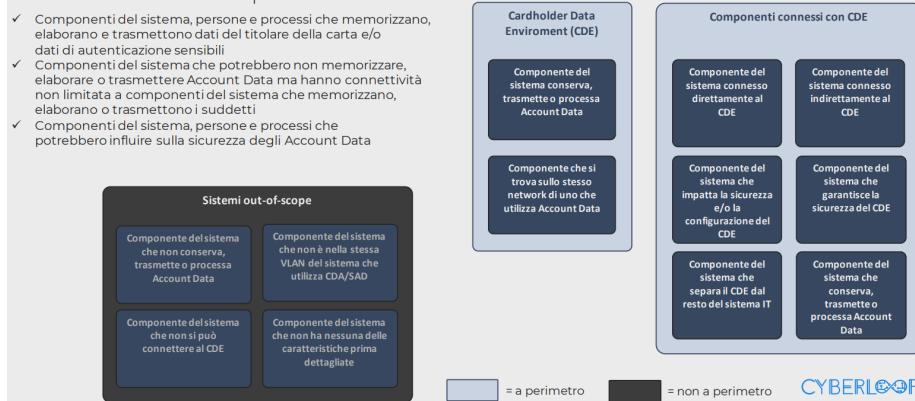


Figure che il PCI DSS controlla:

- Coloro che lavorano sui dati
- Coloro che non usano direttamente i dati ma ne hanno accesso
- Coloro che potrebbero mettere a rischio i dati anche se fuori dal loro ambiente

Lo standard PCI DSS migliora la sicurezza dei dati delle carte di pagamento attraverso misure come la protezione dei dati dei titolari di carta, il controllo degli accessi, il monitoraggio e il test delle reti, e la gestione delle vulnerabilità. Queste misure aiutano le organizzazioni a proteggere le informazioni sensibili da accessi non autorizzati e frodi.

## Standard ISO

ISO è un'organizzazione internazionale indipendente che sviluppa e pubblica standard tecnici per prodotti, servizi e sistemi a livello globale, al fine di garantire

la qualità, l'efficienza e la sicurezza; contribuendo a creare un ambiente di fiducia e cooperazione facilitando il commercio.

-  La Standardizzazione internazionale favorisce l'armonizzazione tra paesi, facilitando il commercio internazionale.
-  Le norme ISO migliorano la qualità dei prodotti e dei servizi, promuovendo l'innovazione e la competitività.
-  Contributo all'efficienza e alla compatibilità, riducendo le barriere tecniche e migliorando la compatibilità tra diverse tecnologie e sistemi.
-  Le norme ISO garantiscono la sicurezza degli utenti e dei consumatori, stabilendo criteri affidabili e universali.

Una norma ISO è un documento tecnico che stabilisce: requisiti, linee guida o specifiche per garantire la qualità, l'affidabilità e la sicurezza di prodotti, servizi o sistemi a livello internazionale.

Ogni norma è così strutturata: ISO 9001:2015 dove ogni elemento è:

Elemento	Descrizione
<b>ISO</b>	Indica che si tratta di una norma dell'International Organization for Standardization
<b>Numero</b>	È il numero di identificazione unico assegnato a questa specifica norma
<b>Anno</b>	È l'anno di pubblicazione o revisione della norma

Le norme ISO sono cruciali nel commercio internazionale, poiché fungono da linguaggio tecnico universale che supera barriere linguistiche e culturali; agendo come un "passaporto commerciale" che semplifica gli scambi internazionali.

In sintesi, le norme ISO contribuiscono a creare un ambiente di fiducia e cooperazione benefico per la comunità globale degli affari.

Per garantire che le norme ISO vengano rispettate esistono due tipi di audit:

## AUDITING INTERNO

Revisione interna dei propri processi con autovalutazione di una checklist dettagliata delle norme ISO pertinenti.  
Identificazione e correzione di eventuali non conformità.

## AUDITING ESTERNO

Presentazione del ruolo e delle responsabilità dell'Auditor esterno.  
Evidenziazione dell'obiettività e dell'indipendenza dell'Auditor esterno.

Quello interno viene fatto in maniera autonoma per prepararsi a quello esterno (come una simulazione d'esame) per poi passare a quello esterno dove viene in azienda una figura incaricata dalla ISO. Se l'auditor alla fine del controllo pensa che tutto sia in regola rilascia una certificazione.

Vediamo cos'è la certificazione:

# CERTIFICAZIONE

Ottenimento certificazione interessata.  
Presentazione delle eventuali non conformità/osservazioni/commenti.

Ora vediamo alcune certificazioni ISO:

Categoria	Descrizione
ISO 27001	Implementare un Sistema di Gestione della Sicurezza delle Informazioni (SGSI) all'interno di un'organizzazione, fornendo un approccio basato sul rischio per proteggere la confidenzialità, l'integrità e la disponibilità delle informazioni.
ISO 27002	Offrire linee guida dettagliate per l'implementazione di controlli di sicurezza delle informazioni. Fornisce un elenco completo di buone pratiche di sicurezza che le organizzazioni possono adottare per migliorare la sicurezza delle informazioni.
ISO 27003	Fornire linee guida per l'implementazione di un SGSI basato su ISO 27001.
ISO 27004	Definire gli indicatori chiave di performance (KPI) e le metriche per valutare l'efficacia di un SGSI implementato in conformità con ISO 27001.
ISO 27005	Offrire una guida per la gestione del rischio delle informazioni, fornendo un processo strutturato per identificare, valutare e gestire i rischi relativi alla sicurezza delle informazioni.

Tutti questi standard fanno parte della famiglia degli ISO 27000 che rappresenta un insieme di norme internazionali focalizzate sulla sicurezza delle informazioni. Essa fornisce un quadro strategico e operativo per sviluppare, implementare, monitorare e migliorare continuamente i processi di gestione della sicurezza delle informazioni all'interno di un'organizzazione.

La famiglia ISO 27000 si concentra sul proteggere le informazioni da minacce interne ed esterne, fornendo un approccio strutturato e basato su best practice.

Un'altra norma è la ISO 22301, dedicata al Business Continuity Management (BCM), processo strategico che mira a identificare, comprendere e mitigare i rischi che potrebbero minacciare la continuità delle operazioni di un'organizzazione.

Per implementare la norma bisogna seguire principi fondamentali:

Categoria	Descrizione
Analisi dei Rischi e delle Minacce	Effettuare un'analisi approfondita dei rischi e delle minacce che potrebbero interrompere le operazioni aziendali. Identificare scenari di crisi potenziali, valutando l'impatto e la probabilità di ciascuno
Sviluppo di Piani di Continuità Operativa (BCP)	Creare piani dettagliati che delineano le azioni da intraprendere in risposta a diverse situazioni di emergenza, specificando i ruoli e le responsabilità del personale durante la fase di risposta e ripristino.
Test e Esercitazioni Regolari	Condurre test e esercitazioni periodiche per garantire l'efficacia dei piani di continuità operativa.
Comunicazione e Coinvolgimento delle Parti Interessate	Stabilire un piano di comunicazione efficace per informare internamente ed esternamente le parti interessate durante un'incidente. Coinvolgere attivamente i dipendenti, i fornitori e altri stakeholder nella pianificazione della continuità.
Formazione del Personale	Fornire formazione regolare al personale per garantire una comprensione completa dei piani di continuità operativa. Assicurarsi che il personale sia a conoscenza dei propri ruoli e responsabilità durante situazioni di emergenza.

Nel dettaglio la creazione di BCP è un processo essenziale per garantire che un'organizzazione possa mantenere operativi i suoi processi critici in situazioni di emergenza o crisi.

## NIS2

Direttiva europea che aggiorna e amplia le misure di sicurezza per le reti e i sistemi informativi degli operatori di servizi essenziali e dei fornitori di servizi digitali, introducendo requisiti più rigorosi per la gestione del rischio e la cooperazione tra gli Stati membri.

Tra le principali novità proposte sono presenti:

- l'estensione del campo di applicazione a nuovi settori, come l'energia, i trasporti e la sanità
- l'introduzione di obblighi aggiuntivi per gli operatori di servizi essenziali e i fornitori di servizi digitali.



L'articolo 21 dice:

"Gli Stati membri provvedono affinché i soggetti essenziali e importanti adottino misure tecniche, operative e organizzative adeguate e proporzionate per gestire i rischi posti alla sicurezza dei sistemi informatici e di rete che tali soggetti utilizzano nelle loro attività o nella fornitura dei loro servizi, nonché per prevenire o ridurre al minimo l'impatto degli incidenti per i destinatari dei loro servizi e per altri servizi."

Sostanzialmente tutti i soggetti indicati sopra devono prendere misure per evitare rischi, queste misure sono definite e spiegate sempre nell'articolo 21. Ecco alcuni esempi:

- Misura 3: continuità operativa, disaster recovery, ecc;
- Misura 4: sicurezza della supply chain (catena di collaborazione fra aziende);
- Misura 8: Crittografia e cifratura;
- Misura 9: sicurezza risorse umane con controllo degli accessi e gestione utenti attivi (account di persone che non lavorano più li).

Un piano di adeguamento viene fatto per rendere conforme un'azienda al NIS2, le fasi sono 4, nel dettaglio:

1. Analisi: Per far sì che la sicurezza si adatti all'azienda si fanno delle interviste e valutazioni, è necessaria assoluta collaborazione fra tutti i soggetti per definire un piano di adeguamento efficace.
2. Definizione: Si definisce il piano completo tenendo in considerazione le criticità per il business individuate nelle fasi precedenti, prioritizzando, così, le attività (es: definizione dei ruoli, definizione di KPI di sicurezza cioè un indice per diverse casistiche come il numero di attacchi ricevuti).

Il NIS2 impone che anche aziende partner abbiano lo stesso livello di sicurezza.

3. Implementazione: si va ad implementare tutto quello concordato precedentemente, per farlo dobbiamo capire quanto c'è di differenza tra dove siamo ora e dove vogliamo arrivare, questo spazio si dice gap e può essere:
  - Gap Tecnologico: comporta la necessità, comunemente, di aggiornare il sistema IT dal punto di vista sia infrastrutturale sia software.
  - Gap Formativo: è un grosso ostacolo per la corretta messa in atto del piano definito, pertanto è necessaria una formazione adeguata al ruolo che si ricopre.
4. Monitoraggio: controllo che tutto sia rispettato, effettuare esami o simulazioni di attacco per vedere se i dipendenti sono preparati.

Per vedere gli impatti andare nelle slides:

Laboratorio di sicurezza dei sistemi informatici e privacy - 01 Normative - V1R0.pdf

Slides

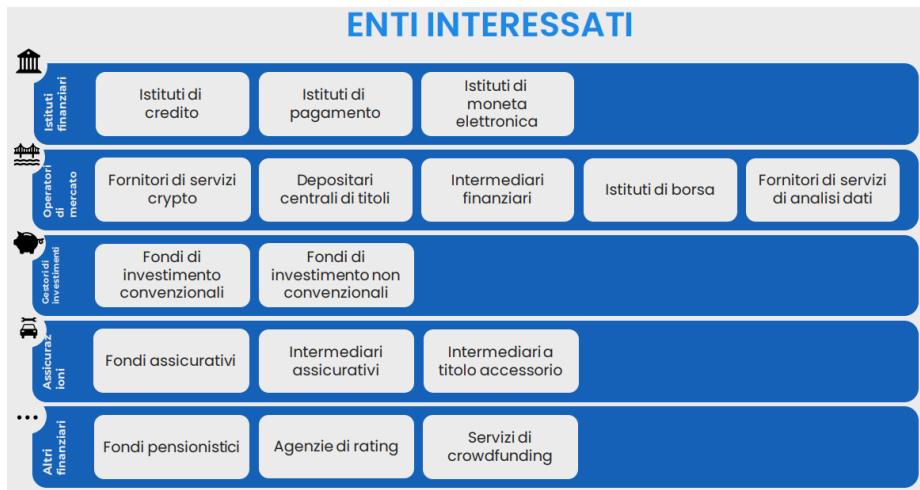
da pagina 82 a 90

### **Regolamento DORA**

Il Digital Operation Resilience Act (DORA), pacchetto europeo per la digitalizzazione del settore finanziario, mira a garantire adeguati meccanismi di salvaguardia in caso di attacchi informatici e a consolidare i requisiti per la prevenzione del rischio ICT nel settore finanziario

Pone particolare attenzione a cinque aree di sicurezza:

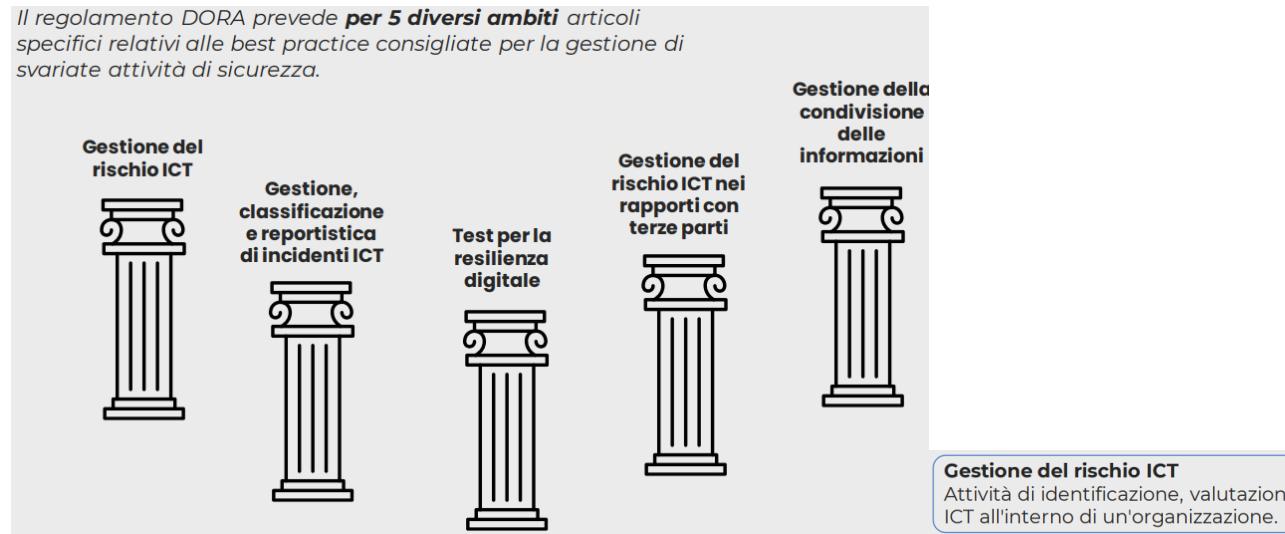
- Gestione del rischio ICT
- Gestione del rischio ICT nei rapporti con terze parti
- Governo della condivisione di informazioni tra entità finanziarie
- Formulazione di reportistica legata a importanti incidenti ICT
- Modalità di test per la verifica della resilienza digitale



Il DORA è nato proprio per creare una regolamentazione a parte per questo genere di enti per via della complessità di gestione e di protezione.

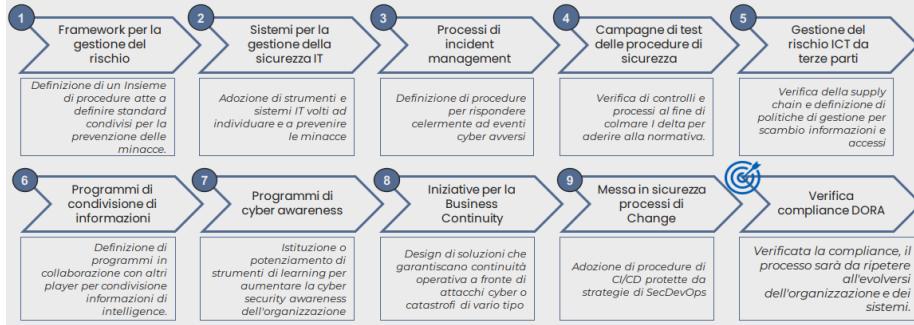
Questa normativa non sostituisce altre normative preesistenti come NIS2 ma le integra, infatti il DORA è maggiormente orientato alla assicurarsi la presenza di strategie, framework (modalità di lavoro che facilità) e processi di governo.

*Il regolamento DORA prevede **per 5 diversi ambiti** articoli specifici relativi alle best practice consigliate per la gestione di svariate attività di sicurezza.*



## DEFINIZIONE DELLE PRIORITÀ

Per gestire in maniera appropriata l'elevato numero di azioni necessarie a fini normativi nei tempi richiesti è fondamentale identificare un ordine che permetta quanto più possibile di fare sinergia tra i diversi filoni operativi, che si delineano in 10 macro categorie.



## Active directory security

Active Directory è un sistema server centralizzato, fondato sui concetti di dominio e di directory, ovvero un insieme di servizi di rete, meglio noti come directory service, gestiti da un domain controller. Tale sistema, inoltre, definisce la modalità con cui vengono assegnate agli utenti tutte le risorse di rete attraverso i concetti di: account utente, account computer, cartelle condivise,.. secondo l'assegnazione da parte dell'amministratore di sistema di Group Policy.

Active Directory Service è un'infrastruttura informativa condivisa per gestire elementi e risorse di rete in ambiente Windows, tra le principali caratteristiche vi sono le seguenti:

- Erogazione di un archivio centrale per l'identità e le informazioni sull'account
- Assegnazione di un identificatore univoco a ciascuno degli oggetti
- Mappatura dei nomi delle risorse di rete ai loro rispettivi indirizzi di rete
- Inclusione dei disposti di controllo dell'accesso, limitando la disponibilità delle informazioni della directory agli utenti autorizzati

## Architettura

Fisicamente, il sistema Active Directory è costituito da uno o più Windows Server che eseguono un servizio chiamato Controller di dominio.

Un Directory Service è un servizio che archivia e organizza gli oggetti in un archivio dati centralizzato e gerarchico:

- un oggetto directory è un oggetto che contiene uno o più attributi
- gli oggetti sono identificati da un nome univoco, possono essere creati, aggiornati ed eliminati nella directory

- I client possono recuperare il contenuto degli oggetti directory leggendo gli attributi di un oggetto specifico o eseguendo query per qualsiasi oggetto che corrisponda ai criteri specificati dal client

## Directory tree

Un concetto centrale nel servizio è la directory tree che possiede le seguenti caratteristiche:

1. Ogni oggetto ha un solo padre (tranne la radice dell'albero, il domain controller);
2. Ogni oggetto può avere zero o più oggetti figlio, dove a ciascuno viene assegnato un nome (RDN) univoco tra i fratelli;
3. Ogni oggetto della directory può essere identificato in modo univoco da tutti gli oggetti del servizio di directory dal suo nome distinto (DN), che si forma concatenandolo con l'RDN dell'oggetto.

## Dominio

Un dominio è un insieme di utenti e computer che condividono una namespace e un'infrastruttura di gestione comune. Tra le caratteristiche della struttura vi è:

- la presenza di almeno un computer membro dell'insieme che funge da Domain Controller (DC) che ospita un elenco per identificare tutti i membri del dominio
- la fruizione da parte del dominio di diversi servizi ai suoi client, principalmente relativi alla sicurezza e alla gestione
- l'attuazione da parte del DC dell'autenticazione dei membri, creando un'unità di fiducia per i suoi membri
- la condivisione del proprio identificativo tra tutti i membri

## Sicurezza

Esistono due oggetti fondamentali:

- Security Principal: è in identità associata ad un utente umano o ad un programma che può essere autenticato, tale identità deve possedere minimo due attributi: un nome e un id che lo identificano in modo univoco rendendolo significativo per il sistema.
- Security Identifier (SID): È l'id per poter identificare il security principal ed è quello che viene controllato per accedere agli oggetti. Spesso il SID corrisponde ad un utente umano ma può anche essere un pc o un servizio

## Autenticazione/Autorizzazione

Tutta l'infrastruttura permette agli utenti di effettuare l'accesso alle informazioni richieste mediante un'autenticazione e un'autorizzazione:

- I security principals del dominio sono tutti disponibili nel domain controller
- Il dominio funge da fonte primaria di identità per i clienti del dominio
- Il dominio, attraverso i protocolli di sicurezza pertinenti, fornisce la base per l'autenticazione all'interno di esso
- Durante il processo di autenticazione, il dominio fornisce informazioni di autorizzazione sotto forma di identità aggiuntive che rappresentano gruppi, consentendo di prendere decisioni di autorizzazione.

## LDAP

LDAP (Lightweight Directory Access Protocol) è un protocollo multipiattaforma utilizzato per :

- autenticazione ai servizi di directory
- comunicazione tra l'applicazione e i server dei servizi di directory che memorizzano e condividono informazioni su utenti, password e account di computer.

A causa dell'architettura di Active Directory, una volta violato un computer collegato a un dominio, l'attaccante può essere in grado di mappare la rete, individuare gli account e le risorse sensibili e stimare le vulnerabilità. Tale processo è reso possibile attraverso l'interrogazione del DC utilizzando il protocollo LDAP.

Il protocollo consente diversi tipi di autenticazione (RFC 2829), per esempio:

- Autenticazione anonima usata per disabilitare o aggiungere restrizioni di controllo degli accessi
- Autenticazione nome/password (non sicura e non adatta se si vuole diritto di riservatezza)
- Autenticazione mediante un digest (non espone la password in chiaro ma la password deve essere salvata in chiaro e precalcolato il digest lato server)
- Autenticazione client basata su certificati

## Connessioni tra domini

Ne esistono di diversi tipi:

- Unidirezionali: solo un dominio dei due della connessione, A e B, può accedere all'altro dominio; A accede a B ma non il contrario.
- Bidirezionale: Il dominio A si fida del dominio B e il dominio B si fida del dominio A. Questa configurazione significa che le richieste di autenticazione possono essere tra i due domini in entrambe le direzioni. Alcune relazioni bidirezionali possono essere di due tipi (transitive o intransitive) a seconda del tipo di fiducia che si sta creando
- Transitivo: può essere utilizzato per estendere le relazioni di fiducia con altri domini; se B si fida di C allora anche A può accedere a C perché si fida di B

- Intransitiva: può essere utilizzata per negare le relazioni di fiducia con altri domini

## Strumenti di sicurezza

### GPO

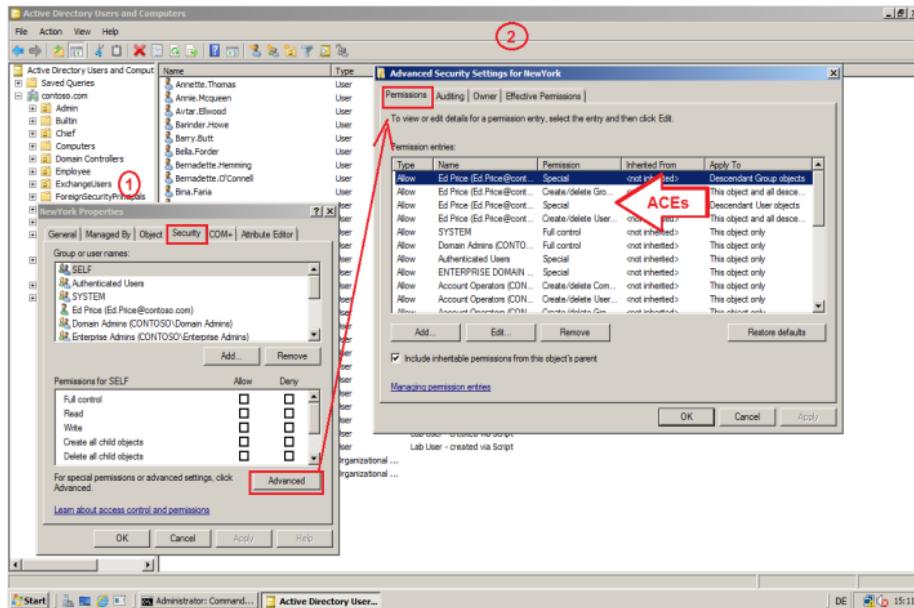
Un Group Policy Object (GPO) è un gruppo di impostazioni create utilizzando la Microsoft Management Console (MMC). Le GPO possono essere associate a uno o più contenitori di Active Directory, compresi siti, domini o unità organizzative (UO). I Group Policy Object , se utilizzati correttamente, consentono di aumentare la sicurezza dei computer degli utenti, di difendersi dalle minacce interne e dagli attacchi esterni e, in parte, di controllare ciò che gli utenti possono o non possono fare su un sistema informatico.

### ACL

Access Control Lists (ACLs) sono le impostazioni che definiscono le autorizzazioni in Active Directory.

Ciascuna di esse deve definire:

- chi può accedere
- a quale oggetto si può accedere
- il tipo di accesso (ACE) Ogni ACE si riferisce al security principal (Utenti, Gruppi e Processi) e definisce i diritti di accesso all'oggetto (consentito o negato) per quel security principal. Inoltre le ACL sono molto flessibili e possono essere aggiunte altre ACE in base alle necessità.



Un tipico ACE contiene le seguenti informazioni:

1. Un identificatore di sicurezza (SID), unico in tutto il dominio
2. Una maschera di accesso(32 bit) che definisce le operazioni consentite o negate
3. Un flag che indica il tipo di ACE ( per consentire, per negare o di controllo di sistema)
4. Un insieme di flag che determinano se i contenitori o gli oggetti figli possono ereditare l'ACE del loro genitore

**Come funziona** Quando un mandante di sicurezza invia una Richiesta di accesso per un oggetto, il SID della richiesta viene confrontato con l'Elenco di controllo degli accessi.

Se il SID corrisponde al SID presente nell'ACL, al mandante di sicurezza viene concesso l'accesso all'oggetto in base ai diritti predefiniti (es. lettura, scrittura, modifica, cancellazione, ecc.).

Esistono comunemente due tipi di ACL:

1. Discretionary access control list (DACL) che definisce i principi di sicurezza che vengono concessi o negati.
2. System access control list (SACL) che garantisce agli admin il privilegio di registrare i log e i tentativi di accesso agli oggetti protetti

## Aggiungere un PC

Quando un pc è aggiunto a dominio cosa succede?

1. Gli account utente del dominio diventano utenti validi del sistema e possono accedervi (a meno che non si applicano restrizioni).
2. Gli amministratori del dominio acquisiscono diritti amministrativi sul sistema.
3. Il computer stesso ottiene un account nel dominio e lo utilizza per autenticarsi.
4. Il nome del computer viene registrato nel DNS del dominio.
5. Le Group Policy definite nel dominio e indirizzate ai computer influiscono sul sistema.
6. Le Group Policy definite nel dominio e destinate agli utenti influiscono su qualsiasi utente del dominio che accede al computer.

## NTLM

Microsoft all'interno della propria architettura, dal 2008, ha creato un protocollo di autenticazione, denominato NT LAN Manager (NTLM), che consente l'autenticazione reciproca di computer e server diversi.

Il protocollo prevede l'autenticazione di un client mediante un nome utente e una password associata attraverso uno scambio di informazioni tra il dispositivo

dell'utente e un server, il quale conosce i dati di login e quindi può controllare l'accesso e autorizzarlo.



### Vantaggi

Un vantaggio di NTLM è che per l'autenticazione non bisogna inviare sulla rete password non protette. La trasmissione dal client al server avviene unicamente sotto forma di valore hash. Questo garantisce certamente un maggiore livello di sicurezza.

### Svantaggi

Se il valore hash dovesse venir intercettato, può venir meno la sicurezza promessa dal sistema. Infatti la password è crittografata tramite MD4, un metodo crittografico considerato ormai poco sicuro in quanto con i pc esistenti oggi tali valori hash possono essere decodificati abbastanza facilmente attraverso un attacco brute force.

## KERBEROS

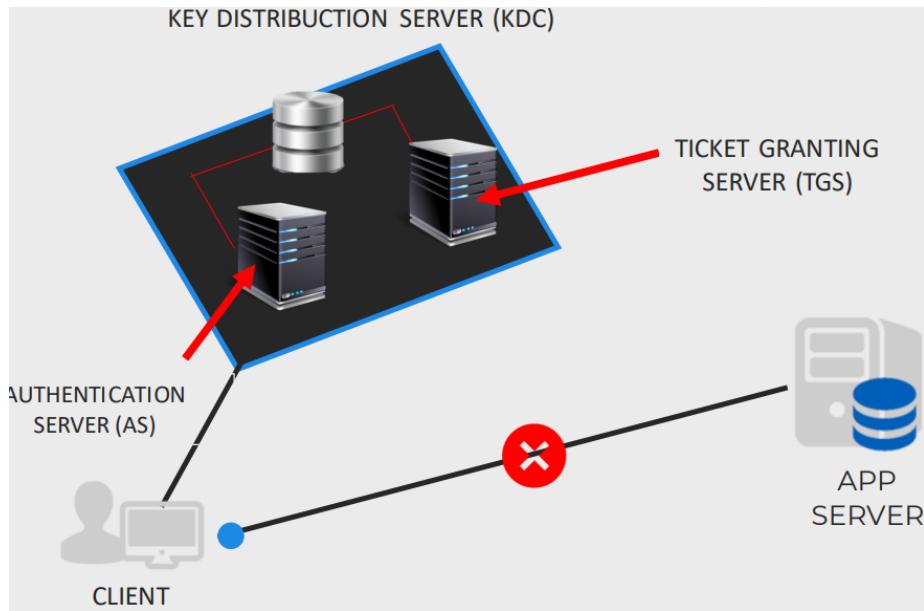
Kerberos, è un servizio di autenticazione che, consente autenticarsi attraverso una rete insicura in maniera sicura, si assegna una chiave unica a ciascun utente che si collega alla rete e incorporando poi queste chiavi nei messaggi inviati dagli utenti (chiave simmetrica) e per la verifica delle identità, si utilizzano delle codifiche crittografate e una terza entità, responsabile della validazione dell'autenticazione.

Il servizio è un progetto open source del Kerberos Consortium ed è la tecnologia di autorizzazione standard di Microsoft Windows introdotta già nelle prime apparecchiature di Windows 2000 in sostituzione di NTLM. Kerberos gestisce l'autenticazione Single Sing On gestendo le credenziali in tutta la foresta ogni volta che si cerca di accedere alle risorse, dopo l'accesso iniziale al dominio tramite Winlogon.

## Funzionamento

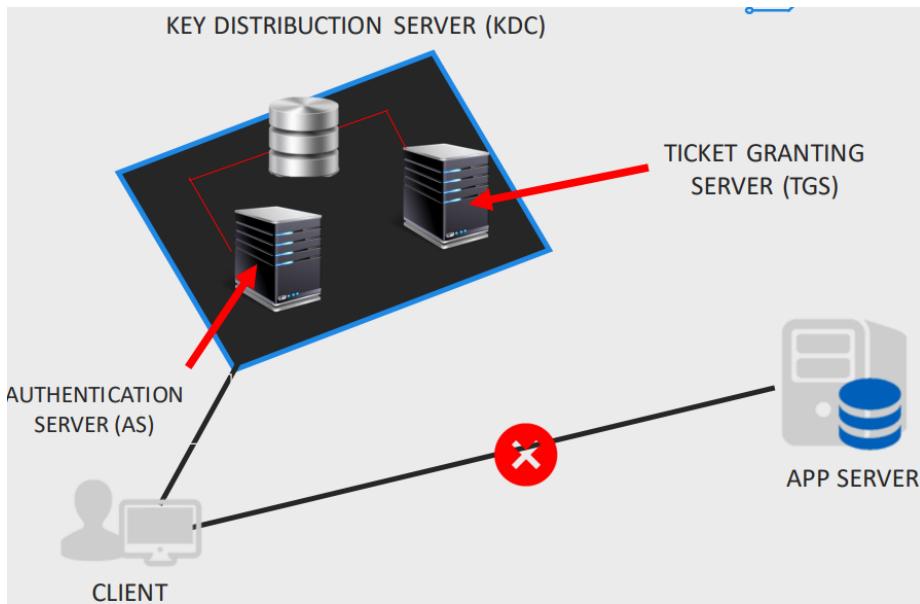
**Step 1** Client invia una richiesta (krb\_as\_req) all'AS chiedendo un Ticket per accedere all'App Server. Tale richiesta è strutturata in questo modo:

- Timestamp (cifrato con user psw)
- Username
- Numero Causale scelto dall'utente (user\_nonce)
- Nome del servizio richiesto(SPN)



**Step 2** Arrivata la richiesta all'AS recupera la password del client nel Db utilizzando lo username e decifra l'intero msg verificando la sua identità. Ed invia al client una risposta (KRB\_AS REP) che contiene:

1. Username client in chiaro
2. Il TGT (Ticket Granting Ticket) cifrato con una secret key
3. User data AS cifrato con la user psw



Il TGT è formato da:

- Username client
- Session key - Data di scadenza del TGT
- PAC (Privileged Attribute Certificate) contenente le informazioni utili sui privilegi dell'utente

User Data AS invece contiene:

- Session key
- Data di scadenza del TGT
- User nonce

**Step 3** Il client avendo ricevuto il TGT procede inviando la richiesta (KRB\_TGS\_REQ ) di accesso al servizio al TGS.

Tale richiesta sarà formata da:

1. Username e timestamp cifrato con la session key
2. Il TGT così com'è perchè non conosce la chiave per decifrarlo
3. User nonce
4. Nome del servizio richiesto(SPN)

**Step 4** Il TGS ricevute le informazioni invia una risposta (KRB\_TGS REP ) al Client contenente:

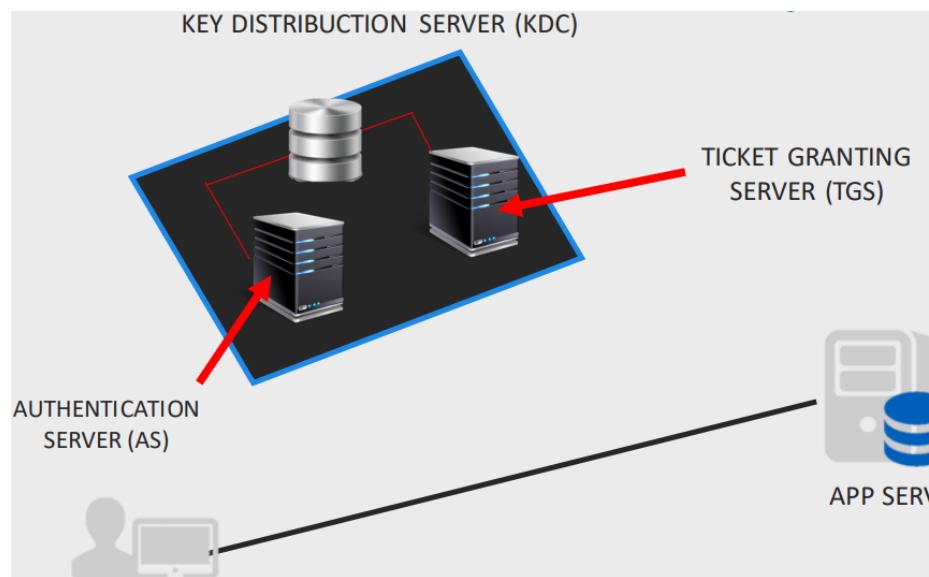
- Username in chiaro
- Il Ticket Granting Service cifrato con un'altra secret key
- User Data TGS cifrate con la session key

Il Ticket Granting Service è formato da:

- Username client - Service Session key
- Data di scadenza del Ticket - PAC User Data TGS invece contiene:
- Service Session key
- Data di scadenza del Ticket
- User nonce

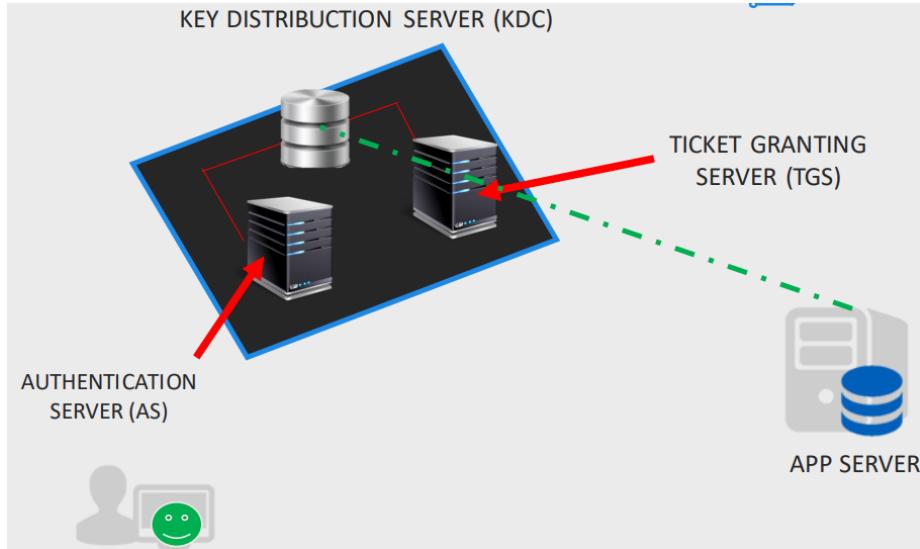
Il Client ricevuto Il Ticket Granting Service e interpretando la service session key invia all'App Server la richiesta di accesso alla risorsa (KRB\_AP\_REQ) che contiene:

- Ticket granting service che non può decifrare non conoscendo la chiave di decifratura
- Username e Timestamp cifrate con la service session key



**Step 5** L'App Server verifica il contenuto del Ticket Granting Service (più precisamente interpreta il PAC) potendo decifrare entrambi in quanto è a conoscenza di tutte le Secret Key essendo esse condivisa con il KDC.

E in caso di risposta affermativa dà il permesso al client di utilizzare il servizio.



Per ricordare:

Key Distribution Center (KDC) è la terza entità che verifica l'identità e rilascia i TGT.

Comprende il TGS e AS.

Ticket Granting Ticket (TGT) è il ticket che serve per accedere al TGS e dimostrare che si è identificati.

Ticket Granting Service (TGS) convalida il TGT e rilascia un ticket per accedere ad un servizio.

### Vantaggi

- Password non vengono mai inviate in rete in formato testuale.
- Le “chiavi segrete” sono trasmesse nel sistema solo in forma criptata.
- Tracciamento di chi ha richiesto che cosa e quando molto semplice
- Il servizio permette inoltre agli utenti e ai sistemi di servizio di autenticarsi a vicenda.
- A ogni passo del processo di autenticazione, sia gli utenti che i sistemi server sanno di avere a che fare con una controparte autentica.

### Svantaggi

- Kerberoasting
- Golden Ticket e Silver Ticket
- Le versioni più vecchie possono ancora essere utilizzate con la crittografia DES.
- Replay attacks

## Kerberoasting

Kerberoasting ci permette di decifrare le password degli account di servizio per accedere a un dominio Active Directory come qualsiasi utente autenticato.

Per farlo chiediamo al TGS ticket service (all'interno dell'Active directory) per gli account di servizio specificando il loro valore SPN. Quindi possiamo forzare questi ticket fino a quando non vengono violati, senza alcun rischio di rilevamento o blocco dell'account, ottenendo la password dell'account.

QUI [37 - 41]

## Silver Ticket

Attacco che comporta la compromissione delle credenziali e l'abuso del design del protocollo Kerberos.

Consente a un utente malintenzionato di falsificare i TGS per servizi specifici. I Tickets sono crittografati con l'hash della password per il servizio; pertanto, se un attaccante ruba l'hash per un account di servizio, può creare TGS Ticket per quel servizio.

QUI [42 - 45]

## Golden Ticket

Un Golden Ticket in Active Directory garantisce al beneficiario un accesso illimitato alla risorsa tramite compromissione del protocollo Kerberos.

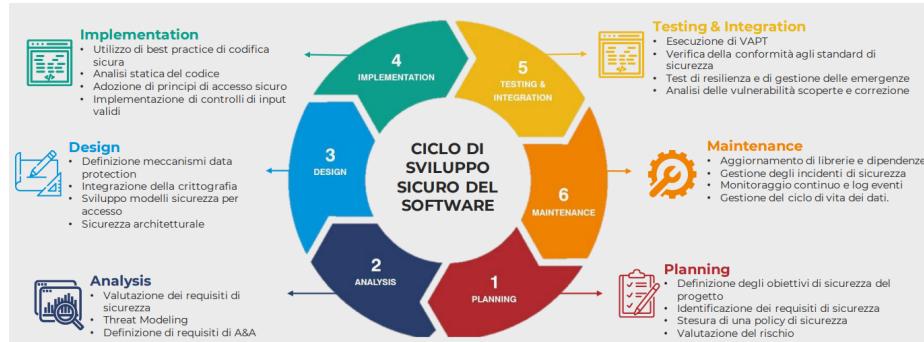
Avviene quando un attaccante compromette l'hash della password KRBTGT, nota solo al Key Distribution Center (KDC), e la utilizza per emettere i biglietti Kerberos riuscendo ad accedere a qualsiasi risorsa desiderata.

QUI [46 - 51]

La differenza principale tra un silver ticket e un golden ticket sta nel livello di accesso che concedono a un utente non autorizzato.

- Golden Ticket: ticket di autenticazione Kerberos contraffatto che concede all'utente l'accesso a qualsiasi servizio nel dominio. Viene creato falsificando un Ticket-Granting Ticket (TGT), che è il ticket iniziale che un utente ottiene dal Kerberos Authentication Server (KDC) dopo l'autenticazione. Un golden ticket è come una chiave passeggiatore che apre qualsiasi porta nel dominio.
- Silver Ticket: ticket di autenticazione Kerberos contraffatto che concede all'utente l'accesso solo a un servizio specifico nel dominio. Viene creato falsificando un Ticket-Granting Service (TGS) ticket, che è il ticket che un utente ottiene dal KDC per accedere a un servizio specifico. Un silver ticket è come una chiave contraffatta che apre solo una porta specifica nel dominio.

## Sicurezza applicativi web



La sicurezza va valutata in ogni fase non solo nell'implementazione.

## Standard

Ora vediamo una carrellata di standard di riferimento:

- CERT Secure Coding Standard: sviluppato appunto dal CERT e fra i vari principi si occupa di:
  - prevenzione delle vulnerabilità;
  - input validation (per evitare SQL injection) quindi controllo dell'input client/server side;
  - sicurezza comunicazioni;
- SSDF - Secure Software Development Framework: del NIST è un quadro di riferimento completo per garantire la sicurezza del software durante tutto il ciclo di sviluppo mostrato sopra, con enfasi sulla fase di progettazione;
- PCI DSS - Payment Card Industry Data Security: che dice come il software deve essere gestito per garantire un pagamento in rete sicuro;
- OWASP Application Security verification Standard: è un framework + standard di sicurezza progettato per valutare delle applicazioni web e dei servizi web.

## Best practice

### Deny by default

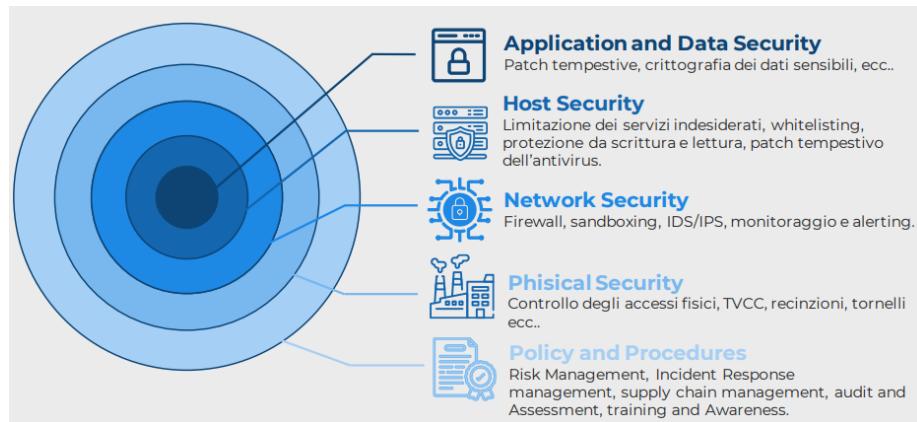
Per accedere ad una determinata risorsa devi essere in whitelist, quindi di base nessuno può accedere ed è il gestore che da il permesso a ciascun individuo.

### Least privilege principle

Si riducono al minimo i privilegi di ogni utente per evitare che chi non di dovere vada ad accedere a parti sensibili.

## Defense in Depth

Il principio è di mettere una sicurezza in più livelli in modo da fermare un malintenzionato molto prima



## Validate input

Consiste nel verificare e garantire che i dati inseriti in un'applicazione rispettino determinati criteri e regole.

La mancanza di una corretta validazione dell'input può portare a vulnerabilità di sicurezza, quali attacchi di injection, e compromettere l'integrità e la funzionalità dell'applicazione.

Contiene la data sanitization.

## Compiler Warnings

I "compiler warnings" sono avvertimenti emessi durante la compilazione del codice per segnalare possibili problemi o pratiche di programmazione rischiose.

Fondamentali per garantire la sicurezza del software, spesso questi messaggi identificano potenziali vulnerabilità, come l'uso di variabili non inizializzate o conversioni di tipo insicure, che potrebbero essere sfruttate in attacchi informatici.

## KISS

Acronimo che sta per “Keep It Simple, Stupid” che promuove la scrittura di codice semplice e diretto senza complessità non necessaria.

## Data Sanitization

Pulizia e validazione di dati per prevenire attacchi informatici.

Questo implica l'uso di tecniche come l'escape dei caratteri speciali, la validazione dei formati, il filtraggio dei caratteri pericolosi e l'implementazione di query parametrizzate nei database.

## OWASP TOP10

Comunità dedicata a indirizzare le organizzazioni nel sviluppare, acquistare e mantenere applicazioni e API affidabili e sicure.

Tutti gli strumenti, i documenti, i video, le presentazioni e i capitoli di OWASP sono gratuiti e aperti a chiunque sia interessato a migliorare la sicurezza delle applicazioni.

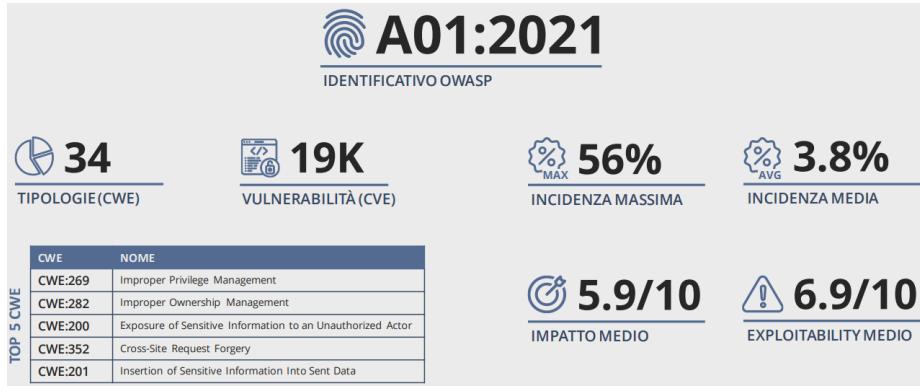
La fondazione OWASP è l'ente no-profit che garantisce il successo a lungo termine del progetto

Nel 2021 OWASP ha stilato una top 10 dei rischi di sicurezza più problematici

ID	SECURITY RISK
A01:2021	Broken Access Control
A02:2021	Cryptographic Failures
A03:2021	Injection
A04:2021	Insecure Design
A05:2021	Security Misconfiguration
A06:2021	Vulnerable and Outdated Components
A07:2021	Identification and Authentication Failures
A08:2021	Software and Data Integrity Failures
A09:2021	Security Logging and Monitoring Failures
A10:2021	Server-Side Request Forgery

### Broken Access Control

Il Broken Access Control si contestualizza all'interno del processo di verifica dell'identità e della liceità delle richieste, denominato controllo degli accessi e si concretizza nella sua elusione o, comunque, nella modifica dei controlli inizialmente implementati.



Le tipologie CWE sono le categorie di vulnerabilità CVE trovati.

L'incidenza indica quanti applicativi (nel pool analizzato) sono vulnerabili a questa falla.

L'impatto medio delle vulnerabilità sull'applicativo.

L'exploitability medio è quanto è facile sfruttare le vulnerabilità.

Alcune nozioni per capire meglio la vulnerabilità:

<b>PERMESSO</b>  Una concessione che viene assegnata a un utente, un processo o un sistema, per accedere a risorse o eseguire determinate azioni su un altro sistema o una rete.	<b>ACCESO AD UNA RISORSA</b>  Capacità di interagire in modo specifico con una risorsa digitale (ad esempio leggerla, modificarla o eseguendola).
<b>AUTENTICAZIONE vs AUTORIZZAZIONE</b>  L'autenticazione è il processo di verifica dell'identità di un'entità. L'autorizzazione è la policy per la quale viene definito chi può accedere a quali risorse ed in quali modalità.	<b>CONTROLLO DEGLI ACCESSI</b>  Processo di verifica dell'autorizzazione, che sfrutta le policy definite per determinare se un'entità può accedere o meno ad una specifica risorsa.

Il controllo degli accessi serve a garantire che gli utenti rispettino le regole stabilite, impedendo loro di compiere azioni non consentite. Quando ci sono problemi con questo tipo di controllo, possono verificarsi situazioni indesiderate come la divulgazione non autorizzata di informazioni, la manipolazione o la distruzione dei dati, oppure l'esecuzione di funzioni aziendali al di là dei permessi dell'utente. Bisogna stare attenti che le risorse che difendiamo non siano accessibili da altri canali rendendo inutile il controllo.

La definizione formale è:

Il Broken Access Control è una categoria di vulnerabilità ampiamente riscontrata in sistemi software, includendo anche applicativi Web, che consiste in problemi

legati ai permessi di accesso, aventi impatto sulla sicurezza di sistemi, processi e informazioni.

Nella tabella le principali contromisure riguardo broken access control:

CONTROMISURA	DETALIO	CONTROMISURA
Secure Environment	Un ambiente sicuro che è progettato e configurato per garantire la sicurezza delle informazioni e dei dati al suo interno. In un <i>secure environment</i> , vengono implementate misure di sicurezza per proteggere da accessi non autorizzati, manipolazioni indesiderate e altri rischi per la sicurezza.	Riutilizzo di componenti
Controlli pervasivi	Misure di sicurezza omnicomprensive per prevenire esposizione di dati sensibili e azioni non autorizzate	Deny-by-Default
Logging	Processo di registrazione sistematica e cronologica degli eventi rilevanti che si verificano in un sistema o in un'applicazione. Questi eventi possono includere attività degli utenti, errori, warning, operazioni di sistema, accessi, e altri eventi significativi.	Multi-factor authentication
		Privilegio minimo
		ABAC

## SCENARIO DI ATTACCO

### SCENARIO – PATH TRAVERSAL

#### CONTESTO

Un attaccante effettua una richiesta alla risorsa `./etc/shadow`, indicando una richiesta di accesso a un percorso di file specifico. La richiesta viene elaborata dal server senza che siano in atto controlli per prevenire l'accesso a cartelle esterne di sistema.

#### SCENARIO



In conclusione:

Il Broken Access Control è, ad oggi, un rischio significativo.

Per evitarlo, è cruciale seguire best practice di sicurezza durante l'intero ciclo di vita del software, comprese le fasi di analisi, progettazione e implementazione.

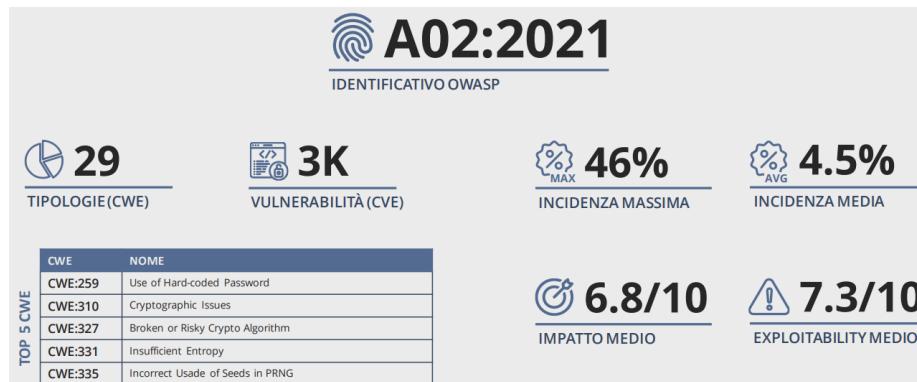
È essenziale applicare principi come il privilegio minimo e implementare un sistema solido di controllo degli accessi basato su profili autorizzativi.

L'identificazione e l'autenticazione sicure sono misure fondamentali per garantire l'efficacia dei processi autorizzativi. Validazioni robuste, eseguite sia lato client sia lato server, verificate ad ogni richiesta, risultano fondamentali.

## Cryptographic failures

I dati sensibili potrebbero essere esposti in diversi modi, tra questi spiccano implementazioni crittografiche errate in modo parziale o totale.

Errori legati alla crittografia spesso portano, infatti, all'esposizione di dati sensibili o addirittura alla compromissione del sistema, sia al momento della memorizzazione delle informazioni, sia al momento dell'invio.



Alcune nozioni per capire meglio la vulnerabilità:

<b>CRITTOGRAFIA</b> Branca della <i>Crittologia</i> che si occupa della conversione dei dati da un formato leggibile in un formato codificato, che può essere letto o elaborato solo dopo che è stato decrittato.	<b>CRITTOANALISI</b> Controparte della crittografia, è lo studio dei metodi per ottenere il significato di informazioni cifrate senza avere accesso all' <b>informazione segreta</b> che è di solito richiesta per effettuare l'operazione, come ad esempio una chiave crittografica.
<b>CRITTOGRAFIA SIMMETRICA</b> La <i>crittografia simmetrica</i> richiede lo scambio di una <b>chiave unica</b> di cifratura tra gli utenti interessati. Non permette di verificare l'autenticità del messaggio.	<b>CRITTOGRAFIA ASIMMETRICA</b> La <i>crittografia asimmetrica</i> utilizza <b>due chiavi asimmetriche</b> , una pubblica (usata per cifrare) e una privata (usata per decifrare), collegate tra loro in base a una funzione matematica. Permette di verificare l'autenticità del messaggio.

La definizione è:

Il Cryptographic Failures è una categoria di debolezze ampiamente riscontrata in sistemi software, includendo anche applicativi web, che consiste in problemi nella progettazione, nell'implementazione o nell'uso di algoritmi e protocolli crittografici, che causano impatti nella sicurezza di sistemi, processi e informazioni.

Il Cryptographic Failures di OWASP fornisce una guida dettagliata sugli errori crittografici più comuni e pericolosi che possono verificarsi nell'implementazione di funzionalità crittografiche all'interno di un'applicazione, i principali sono:

- Algoritmi crittografici deboli o insicuri: l'uso di algoritmi crittografici deboli o obsoleti può compromettere la sicurezza di un sistema. Ad esempio,

l'impiego di algoritmi come DES o MD5 è considerato insicuro.

- Problemi nella generazione e gestione delle chiavi: errori nella generazione, gestione o protezione delle chiavi crittografiche possono portare a vulnerabilità significative.
- Utilizzo improprio della crittografia: implementazione erronea della crittografia, come l'uso di modalità insicure o la mancata applicazione di funzioni di hashing, può compromettere la protezione dei dati.
- Mancanza di controllo dell'integrità dei dati: assenza di meccanismi per verificare l'integrità dei dati crittografati può consentire attacchi di manipolazione.
- Problemi di implementazione del protocollo: errori nell'implementazione di protocolli crittografici, come TLS/SSL, possono portare all'esposizione totale di dati sensibili.

Nella tabella le principali contromisure riguardo cryptographic failures:

CONTROMISURA	DETALLO
Algoritmi di cifratura sicuri	Utilizzare algoritmi di cifratura robusti raccomandati dalle ultime versioni degli standard di sicurezza.
Casualità	La casualità è cruciale per evitare prevedibilità nei dati criptati, quindi evitare la scoperta di pattern fissi da parte degli attaccanti.
Assenza di dati sensibili nei log e in cache	Assicurarsi che i dati sensibili non finiscano nella cache dei componenti del server o nei log del load balancer. Garantire che le copie temporanee o memorizzate nella cache dei dati sensibili siano protette dall'accesso non autorizzato
Backup	Effettuare backup periodici dei dati importanti. Condurre test periodici di ripristino dei dati per verificare l'efficacia dei backup e assicurarsi che siano ben custoditi.
Client-Side Data Protection	Impiegare header anti-caching per evitare la persistenza dei dati sensibili nella cache dei browser moderni. Assicurarsi che i dati autenticati siano eliminati dal DOM del browser alla fine della sessione o dopo la disconnessione dell'utente, oltre che le variabili <i>localStorage</i> , <i>sessionStorage</i> , <i>IndexedDB</i> e i cookie

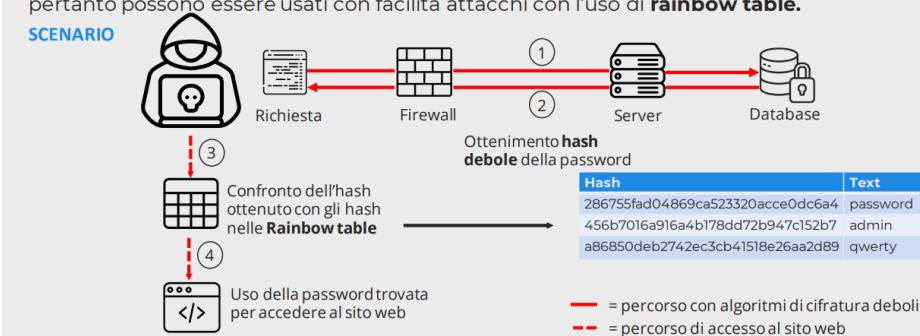
## SCENARIO DI ATTACCO

### SCENARIO – WEAK ENCRYPTION

#### CONTESTO

Le password degli utenti sono memorizzate all'interno del database tramite funzioni hash senza *salt*, pertanto possono essere usati con facilità attacchi con l'uso di **rainbow table**.

#### SCENARIO



In conclusione:

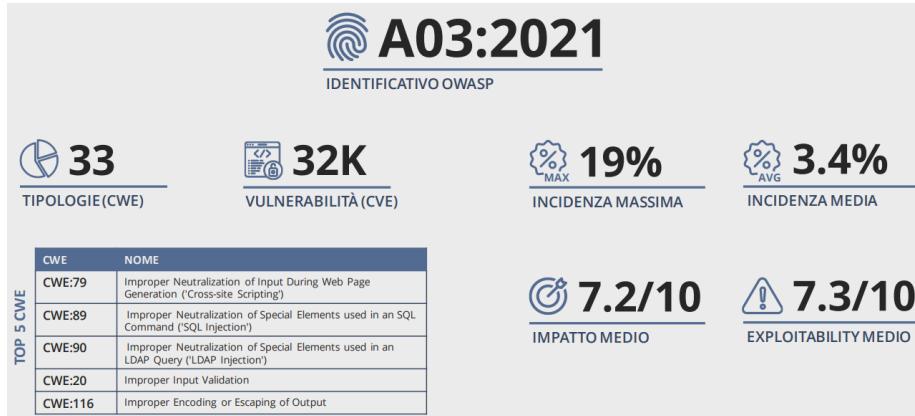
Il Cryptographic Failures rappresenta una delle minacce più rilevanti e pervasive nell'ambito della sicurezza delle applicazioni Web. La crittografia svolge un ruolo cruciale nella protezione dei dati sensibili e nella garanzia dell'integrità delle comunicazioni, ma errori nella sua implementazione possono comportare gravi conseguenze.

Il Cryptographic Failures può mettere a rischio la confidenzialità dei dati e la sicurezza delle applicazioni.

### Injection

Le vulnerabilità di tipo injection vedono la loro causa primaria nella gestione incorretta dei dati di input.

Gli attacchi di tipo injection si differenziano in base alla tecnologia colpita e alle modalità di esecuzione dell'attacco (SQL injection, OS command injection, Cross Site Scripting (XSS)).



Alcune nozioni per capire meglio la vulnerabilità:

<b>SOL Injection</b> È una tecnica di injection che sfrutta le vulnerabilità di sicurezza del codice applicativo che si collega al <b>database</b> contenente dati SQL. È tra le Injection più diffuse e famose oltre che datate. Il primo exploit è stato documentato per la prima volta nel 1998.	<b>Cross Site Scripting (XSS)</b> È una tecnica di injection che si verifica quando un attaccante inserisce codice malevolo all'interno di un'applicazione web al fine di obbligare un utente vittima ad eseguirne il contenuto sul proprio browser.
<b>Server Side Template Injection</b> Tecnica che sfrutta l'utilizzo della struttura di template per iniettare codice malevolo all'interno dello stesso, il quale successivamente viene eseguito lato server.	<b>OS Command Injection</b> Procedura di injection concepita con l'obiettivo finale di eseguire comandi arbitrari nel contesto del sistema operativo ospitante la tecnologia soggetta all'attacco.

La definizione è:

L'Injection è una tecnica di attacco informatico che consiste nell'inserimento di codice malevolo in un'applicazione, in un processo in esecuzione o in un database, al fine di modificarne il comportamento previsto.

Gli attacchi di tipo injection, variegati e strettamente legati alla tecnologia coinvolta, condividono il concetto fondamentale: manipolare la logica di esecuzione dell'applicativo attraverso l'inserimento di input controllato dall'utente malintenzionato.

Ad esempio, i buffer overflow sfruttano l'inserimento di una quantità eccessiva di dati per sovrascrivere parti del codice, potenzialmente compromettendo il servizio e permettendo l'esecuzione di codice malevolo. La radice del problema risiede nella mancanza di controlli sull'input; è essenziale verificarne la conformità alle aspettative e considerare non valido l'input non conforme.

Nella tabella le principali contromisure riguardo injection:

CONTROMISURA	DETTAGLIO
Mantenere i dati separati dai comandi e dalle query usando un'API sicura	È preferibile utilizzare l'API sicura fornita dalla tecnologia in uso piuttosto che utilizzare l'interprete per eseguire una determinata operazione sul OS.
Usare una validazione dell'input lato server con allow-list	Permette di precisare esattamente quali possibili forme può assumere l'input. Questa non è una difesa completa, poiché molte applicazioni richiedono caratteri speciali.
Escaping/encoding specifico	L'escaping/encoding adeguato al contesto permette di far sì che i possibili caratteri speciali all'interno di una stringa vengano trasformati, così da evitare possibili injection.
Utilizzare controlli al fine di limitare le possibili esposizioni di dati	Ad esempio: utilizzo di LIMIT in query SQL così da limitare il numero di record esposti in caso di SQL injection.

### SCENARIO – ATTACCO DI COMMAND INJECTION

#### CONTESTO

Un programma che consente agli utenti remoti di visualizzare il contenuto di un file.  
Segue il programma:

```
int main(char* argc, char** argv) {
    char cmd[CMD_MAX] = "/usr/bin/cat";
    strcat(cmd, argv[1]);
    system(cmd);
}
```

#### SCENARIO

Sebbene il programma sia apparentemente innocuo (consente solo l'accesso in sola lettura ai file), consente un attacco di tipo *command injection*, poiché se l'attaccante concatenasse al parametro legittimo altri comandi, questi verrebbero eseguiti a loro volta. **Possibile payload: file.txt;rm -rf /**

Il payload di esempio, una volta eseguito, in seguito alla visualizzazione del file fornito come argomento, porterebbe alla rimozione dell'intero root path.

## Insecure design

Si concentra sui rischi associati ai difetti di progettazione e architetturali. In particolare, questa categoria si dedica al threat modeling, all'utilizzo di design-pattern sicuri e alle scelte architetturali.

Uno dei fattori che contribuisce all'insecure design è la mancanza di un profilo di rischio a livello di business, che comporta una carenza di conoscenza dei livelli di sicurezza necessari.

Un design insicuro rimane vulnerabile anche in assenza di bug, poiché per sua natura manca delle protezioni necessarie a livello progettuale. Le conseguenze di una possibile falla nel design dipendono fortemente dall'origine del problema, ovvero dalla mancanza a livello progettuale. Tali conseguenze possono tradursi in ingenti perdite economiche, poiché risolvere una lacuna a livello progettuale richiede uno sforzo significativo.



Alcune nozioni per capire meglio la vulnerabilità:

<b>SECURE DESIGN</b> Un design sicuro implica l'integrazione della sicurezza fin dalle prime fasi del processo di sviluppo, evitando l'approccio di aggiungere misure di sicurezza in fasi successive. Questo approccio proattivo mira a prevenire vulnerabilità sin dall'inizio, promuovendo la creazione di sistemi più robusti e resilienti.	<b>THREAT MODELING</b> Processo proattivo di identificazione, valutazione e mitigazione delle potenziali minacce alla sicurezza di un sistema o applicazione durante la fase di progettazione. Coinvolge l'analisi degli scenari di attacco, la valutazione delle vulnerabilità e l'implementazione di contromisure per migliorare la resistenza contro le minacce informatiche.
<b>SECURE DEVELOPMENT LIFECYCLE</b> Il Secure Development Lifecycle (SDLC) è un approccio che integra pratiche di sicurezza fin dalle prime fasi dello sviluppo del software, garantendo la robustezza attraverso pianificazione, analisi dei requisiti, codifica sicura, test di sicurezza e valutazione continua.	<b>DESIGN PATTERNS</b> Un design pattern è una soluzione generica e riutilizzabile per problemi comuni nello sviluppo software. Fornisce un modello concettuale consolidato per affrontare specifiche sfide progettuali, promuovendo coerenza e migliorando la qualità del codice.

La definizione è:

L'Insecure Design si riferisce a un approccio di progettazione che presenta vulnerabilità e lacune nella sicurezza di un sistema, applicazione o prodotto. Questo tipo di design manca di controlli e misure di sicurezza efficaci, aumentando il rischio di esposizione a minacce informatiche e violazioni della sicurezza.

Nella tabella le principali contromisure riguardo insecure design:

CONTROMISURA	DETALLO
Stabilire e utilizzare un ciclo di vita di sviluppo sicuro	Implica l'adozione di pratiche e metodologie che integrano la sicurezza in ogni fase del processo di sviluppo del software. Ciò include la valutazione dei rischi, la definizione dei requisiti di sicurezza, la progettazione sicura, lo sviluppo sicuro, i test di sicurezza e la gestione delle vulnerabilità.
Stabilire e utilizzare librerie di design pattern sicuri o componenti pronti all'uso	Utilizzare librerie di design pattern sicuri o componenti già testati e approvati può contribuire a evitare la reintroduzione di vulnerabilità comuni. Ciò significa affidarsi a soluzioni consolidate e ben mantenute per funzionalità critiche del software.
Usare il threat modeling	Essendo un processo per identificare e valutare potenziali minacce e vulnerabilità in un'applicazione, ne consente un miglioramento della comprensione delle minacce specifiche, permettendo la messa a punto di contromisure mirate e la protezione delle parti più sensibili del sistema.
Integrare i controlli di plausibilità ad ogni livello della vostra applicazione (dal frontend al backend)	Implementare controlli di plausibilità in ogni strato dell'applicazione per garantire che i dati in ingresso siano validi e sicuri. Questo include la validazione lato client e la validazione lato server per garantire coerenza e sicurezza nei dati scambiati.

#### SCENARIO – PLUGIN VULNERABILE

##### CONTESTO

Un'applicazione sicura integra un plugin che, al contrario, risulta vulnerabile.

##### SCENARIO

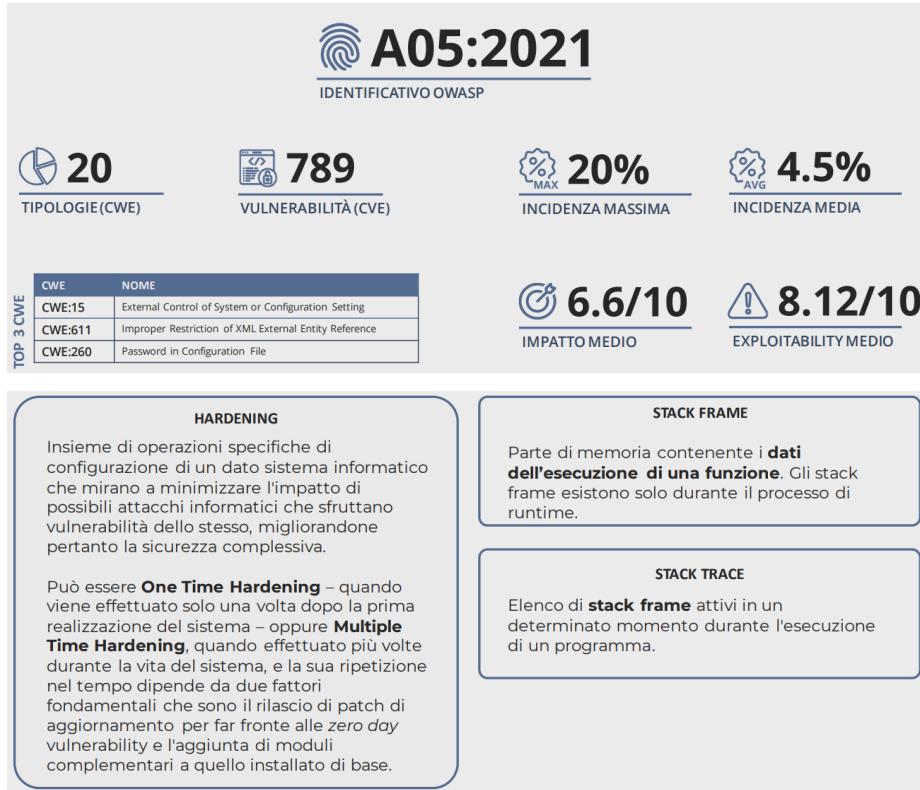
Il plugin esegue all'interno del contesto dell'applicazione, come ad esempio in un'applicazione Java, sia l'app sia il plugin potrebbero usare lo stesso classloader, violando il principio dei compartimenti isolati. A causa della condivisione dell'ambiente di esecuzione tra applicazione e plugin, l'attaccante può sfruttare le vulnerabilità del plugin per interagire con l'applicativo, intaccandone la sicurezza.



CYBERLIC

#### Security misconfiguration

la Security Misconfiguration, circa il 90% di applicazioni testate sono risultate vulnerabili ad essa. Non sorprende vedere questa categoria così rilevante, essendo i software sempre più configurabili e, spesso, lasciati dall'utente allo stato di default.



La definizione è:

La Security Misconfiguration è una categoria di vulnerabilità che si verifica quando un'applicazione, un server o qualsiasi componente di un sistema è configurato in modo errato, lasciando aperte vulnerabilità che potrebbero essere sfruttate dagli aggressori.

Tali configurazioni di sicurezza errate possono manifestarsi in vari modi, ad esempio:

- Permissive Configurations: Assegnare permessi eccessivi o non necessari a utenti, processi o risorse, consentendo l'accesso non autorizzato o il potenziale sfruttamento di vulnerabilità.
- Insecure Defaults: Utilizzare impostazioni di default che sono intrinsecamente insicure. Ciò potrebbe includere password deboli, configurazioni di crittografia insoddisfacenti o altre scelte di configurazione che rendono il sistema vulnerabile.
- Exposed Sensitive Information: Rivelare informazioni sensibili o dettagli implementativi nel codice, nei file di configurazione o altrove. Queste informazioni potrebbero essere sfruttate dagli aggressori per pianificare e condurre attacchi più mirati.

Nella tabella le principali contromisure riguardo la security misconfiguration:

CONTROMISURA	DETTAGLIO
Hardening ripetibile	Un processo di hardening ripetibile rende veloce e facile il deploy di un altro ambiente preconfigurato in modo sicuro. È consigliabile uniformare le configurazioni negli ambienti di sviluppo, QA e produzione, garantendo l'uso di credenziali distinte per ciascun ambiente. Automatizzare questo processo è cruciale per ridurre al minimo la fatica necessaria per istituire nuovi ambienti sicuri.
Piattaforma minimale	Implementare una piattaforma ridotta al minimo, priva di funzionalità, componenti, documentazione e esempi superflui. Eliminare o evitare l'installazione di funzionalità e framework non utilizzati.
Applicativo segmentato	Implementare un'architettura applicativa segmentata, in modo da fornire un'efficace e sicura separazione tra componenti o tenant, con segmentazione, containerizzazione, o cloud security groups (ACL).
Processo automatizzato	Strutturare un processo automatizzato per verificare l'efficacia delle configurazioni e impostazioni in tutti gli ambienti.

## SCENARIO – CREDENZIALI DI DEFAULT

**CONTESTO**  
Utilizzo di credenziali predefinite in una stampante Brother in rete, a causa di configurazioni iniziali di default.

**SCENARIO**  
Un attaccante riesce a collegarsi all'interfaccia di rete della stampante con credenziali di default, recuperabili online. Così facendo, l'attaccante riesce ad ottenere informazioni potenzialmente riservate, presenti ad esempio all'interno degli ultimi documenti stampati.

```

graph LR
    Attaccante((Attaccante)) -- "Inserimento credenziali di default" --> Interfaccia[Interfaccia di login]
    Interfaccia --> Stampante((Stampante))
    Stampante -- "Esfiltrazione documenti riservati" --> Attaccante
    style Attaccante fill:#ccc,stroke:#000,stroke-width:2px
    style Interfaccia fill:#fff,stroke:#000,stroke-width:2px
    style Stampante fill:#fff,stroke:#000,stroke-width:2px
  
```

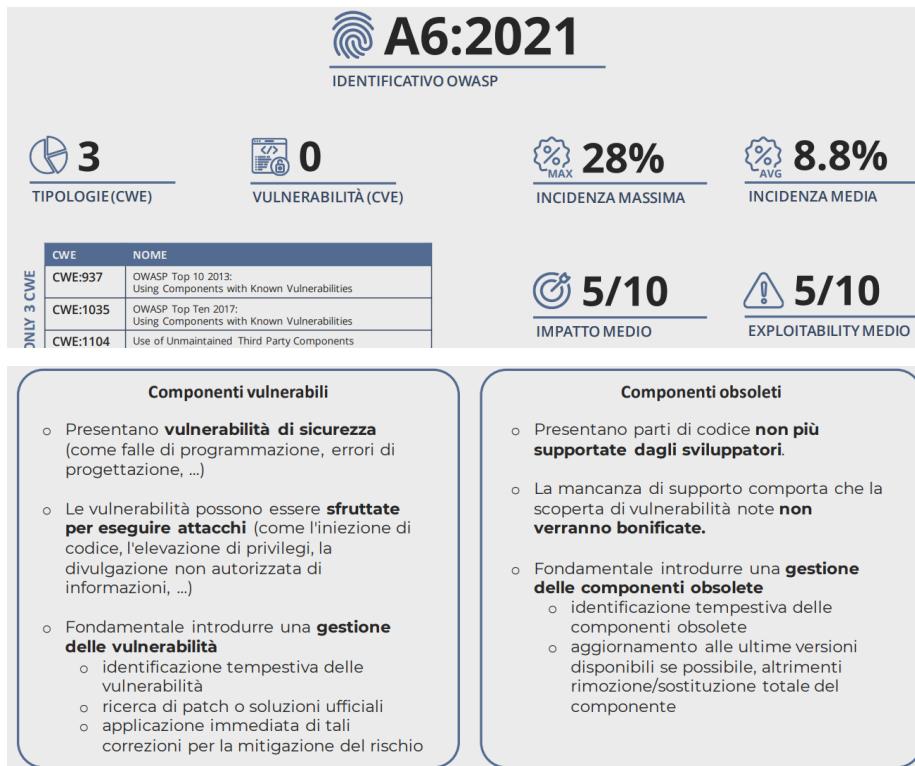
La Security Misconfiguration è una delle principali minacce alla sicurezza delle applicazioni Web e dei sistemi in generale. Gli attaccanti possono sfruttare configurazioni errate per ottenere accesso non autorizzato, eseguire attacchi di traversing directory, scoprire informazioni sensibili e altro ancora.

È essenziale eseguire regolarmente audit di sicurezza e revisioni delle configurazioni per identificare e correggere eventuali vulnerabilità di configurazione.

## Vulnerable and outdated components

La categoria Vulnerable and Outdated Components di OWASP riguarda una problematica nota, per la quale risulta complesso effettuare dei test mirati e, pertanto calcolarne il rischio.

È infatti l'unica categoria per cui non è associato nessun Common Vulnerability and Exposure (CVE).



La definizione è:

Il Vulnerable and Outdated Components è una minaccia alla sicurezza che si verifica quando un'applicazione utilizza componenti software, come librerie o framework, che contengono vulnerabilità di sicurezza note o sono datati e non più supportati.

Si vuole sottolineare, inoltre, che le applicazioni spesso dipendono da componenti di terze parti per funzionare in modo efficiente. Tali componenti potrebbero contenere vulnerabilità di sicurezza o potrebbero essere obsolete. In tal caso, possono costituire una potenziale via di accesso per gli attaccanti. Gli aggressori possono sfruttare le vulnerabilità nelle versioni obsolete o noti problemi di sicurezza nelle componenti utilizzate per compromettere l'applicazione e, in ultima analisi, il sistema sottostante.

Il Vulnerable and Outdated Components di OWASP si presenta se:

- Non si conoscono le versioni di tutti i componenti utilizzati (sia lato client che lato server). Questo include i componenti utilizzati direttamente così come le dipendenze annidate.
- Se il software è vulnerabile, non supportato o non aggiornato. Questo include i sistemi operativi, i server web, i database management system (DBMS), le applicazioni, API e tutti i componenti, ambienti di esecuzione e librerie.
- In caso non venissero effettuate scansioni periodiche di sicurezza e non si consultassero i bollettini di sicurezza relativi ai componenti utilizzati.
- In caso non fosse previsto un piano efficace di patch management relativamente alla piattaforma sottostante, i framework, e le dipendenze in modo tempestivo e basato sul rischio.
- In assenza di test di specifici per verificare la non regressione del software a seguito di aggiornamenti di librerie.

Nella tabella alcune contromisure:

CONTROMISURA	DETTAGLIO
Rimozione e pulizia	<b>Rimuovere</b> dipendenze, funzionalità, componenti, file e documentazione non utilizzati per <b>ridurre la complessità</b> e la superficie di attacco dell'applicazione.
Monitoraggio delle versioni e vulnerabilità note	<b>Inventariare continuamente</b> le versioni dei componenti sia lato client che lato server, come framework e librerie, utilizzando strumenti come OWASP Dependency Check, retire.js, ecc. <b>Monitorare costantemente fonti di vulnerabilità</b> come Common Vulnerability and Exposures (CVE) e il National Vulnerability Database (NVD).
Sicurezza nell'acquisizione dei componenti	Acquisire i componenti solo da <b>fonti ufficiali</b> attraverso link sicuri per garantire l'integrità del software. Preferire <b>pacchetti firmati</b> per ridurre il rischio di includere componenti modificati o dannosi.
Gestione delle componenti obsolete	Controllare le librerie e i componenti che non sono più mantenuti o che non ricevono più patch di sicurezza per le vecchie versioni. Quando possibile, <b>preferire librerie supportate</b> e frequentemente aggiornate con patch di sicurezza.

### Identification and authentication failures

Precedentemente denominata Broken Authentication. Questo termine si riferisce a una serie di vulnerabilità e inefficienze che possono emergere nei processi di identificazione e autenticazione, i quali sono fondamentali per garantire l'accesso sicuro e autorizzato ai sistemi e alle applicazioni.



La definizione è:

Identification and Authentication Failures riguarda situazioni in cui i meccanismi di autenticazione e identificazione di un'applicazione non sono implementati in modo sicuro o sono vulnerabili. Questo può includere problemi come la gestione inadeguata delle sessioni, l'uso debole di password, la mancanza di controlli di autenticazione multi-fattore, errori nelle risposte di autenticazione e altri problemi legati all'identificazione e alla verifica delle identità degli utenti.

Nella tabella alcune contromisure:

CONTROMISURA	DETALIO
Utilizzo di HTTPS per pagine autenticate	Tutte le pagine di accesso e autenticate devono essere accessibili esclusivamente tramite TLS o altro trasporto sicuro.
Verifica della password attuale	Prima di consentire il cambio della password, è fondamentale autenticare l'utente richiedendo la password attuale.
Protezione contro CSRF e session hijacking	Richiedere le credenziali correnti prima di aggiornare informazioni sensibili o eseguire transazioni sensibili per prevenire attacchi CSRF e session hijacking.
Limitazione dell'accesso fisico	Proteggere il browser dell'utente da accessi temporanei per prevenire l'esecuzione di azioni non autorizzate.
Protezione dell'ID di sessione	Evitare il compromesso dell'ID di sessione per assicurare un ambiente autenticato sicuro.

## SCENARIO – USER ENUMERATION ATTACK SULLA FUNZIONALITÀ DI LOGIN

### CONTESTO

Un'applicazione web mette a disposizione un portale di login per autenticare l'utente verso la piattaforma.

### SCENARIO

La risposta del server alla richiesta di login fornisce messaggi diversi per account esistenti e non esistenti oppure la temporizzazione della risposta varia a seconda dell'esistenza dell'account, consentendo a un attaccante di enumerare gli username validi.



## Software and data integrity failures

Si focalizza sulle relazioni di fiducia senza verifiche di integrità in contesti di aggiornamento software, gestione di dati critici e CI/CD pipelines.



<b>Serializzazione</b> Processo di conversione di un oggetto o dati in un formato serializzato, spesso binario, che può essere trasferito o archiviato facilmente. La deserializzazione è il processo inverso, in cui i dati serializzati vengono ricostruiti nell'oggetto originale.	<b>CI/CD Pipeline</b> Pratiche di sviluppo del software. Una pipeline CI/CD è un flusso automatizzato che include integrazione continua, test automatici e distribuzione continua, riducendo il tempo tra lo sviluppo di nuove funzionalità e la loro effettiva implementazione nel software in produzione.
<b>Integrità dei dati</b> La sicurezza e la coerenza dei dati vengono mantenute attraverso il controllo e la protezione da modifiche non autorizzate, corruzioni o accessi non autorizzati. Garantire l'integrità dei dati è fondamentale per assicurare che le informazioni mantengano la loro precisione e affidabilità.	<b>Autenticità dei dati</b> Rappresenta la verifica che i dati provengano da una fonte attendibile e che non siano stati alterati durante la loro trasmissione o conservazione. Assicurare l'autenticità dei dati contribuisce a prevenire manipolazioni e garantisce che le informazioni siano affidabili e originali.

La definizione è:

La classe di vulnerabilità Software and Data Integrity Failures secondo i dati forniti è caratterizzata da situazioni in cui il codice e l'infrastruttura non proteggono adeguatamente contro violazioni dell'integrità del software e dei dati.

Queste vulnerabilità possono derivare da pratiche come l'inclusione di funzionalità da fonti non attendibili, la mancanza di verifica dell'integrità durante gli aggiornamenti del software, e la presenza di vulnerabilità di deserializzazione non sicura.

Nella tabella alcune contromisure:

CONTROMISURA	DETALLO
Firme digitali	Implementare firme digitali o meccanismi simili per verificare che il software non sia stato alterato e provenga dalla fonte attesa.
Utilizzo di componenti fidati	Utilizzare repository attendibili per le librerie e le dipendenze. Possibilmente, ospitare un repository interno noto per validare i componenti.
Utilizzare strumenti di sicurezza della catena di distribuzione del software	Integrare strumenti per identificare e correggere vulnerabilità nelle dipendenze del progetto.
Revisione del codice e delle configurazioni	Effettuare revisioni regolari del codice e delle configurazioni per identificare e correggere potenziali vulnerabilità.

CONTROMISURA	DETALIO
Segregazione, configurazione e controllo degli accessi nelle pipeline	Configurare la pipeline CI/CD con controlli di accesso appropriati, segregazione e configurazione sicura per prevenire accessi non autorizzati.
Utilizzare solo formati di serializzazione sicuri	Evitare l'uso di formati di serializzazione vulnerabili e optare per formati sicuri come JSON o XML.
Integrità di dipendenze e codice in build	Utilizzare strumenti di sicurezza della catena di distribuzione per verificare l'integrità del codice e delle dipendenze durante la fase di build.
Verifica dei pacchetti di aggiornamento	Applicare firme digitali o verifiche di integrità ai pacchetti di aggiornamento del software per garantire l'origine e l'integrità degli aggiornamenti.
Limitare l'accesso alle classi deserializzabili e utilizzare controlli di sicurezza come white-list	Applicare restrizioni sull'accesso alle classi deserializzabili e utilizzare white-list per specificare quali classi possono essere deserializzate.

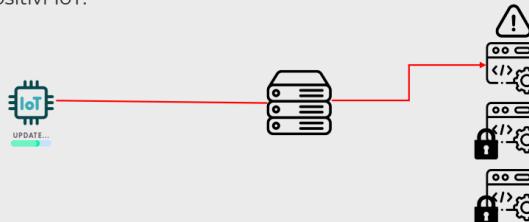
#### SCENARIO – Aggiornamento non Firmato su Dispositivi IoT

##### CONTESTO

Alcuni dispositivi IoT installati in una rete eseguono aggiornamenti del firmware senza verificarne la firma.

##### SCENARIO

Sfruttando la carenza di verifica delle firme digitali, un attaccante crea un firmware malevolo destinato a dispositivi IoT.



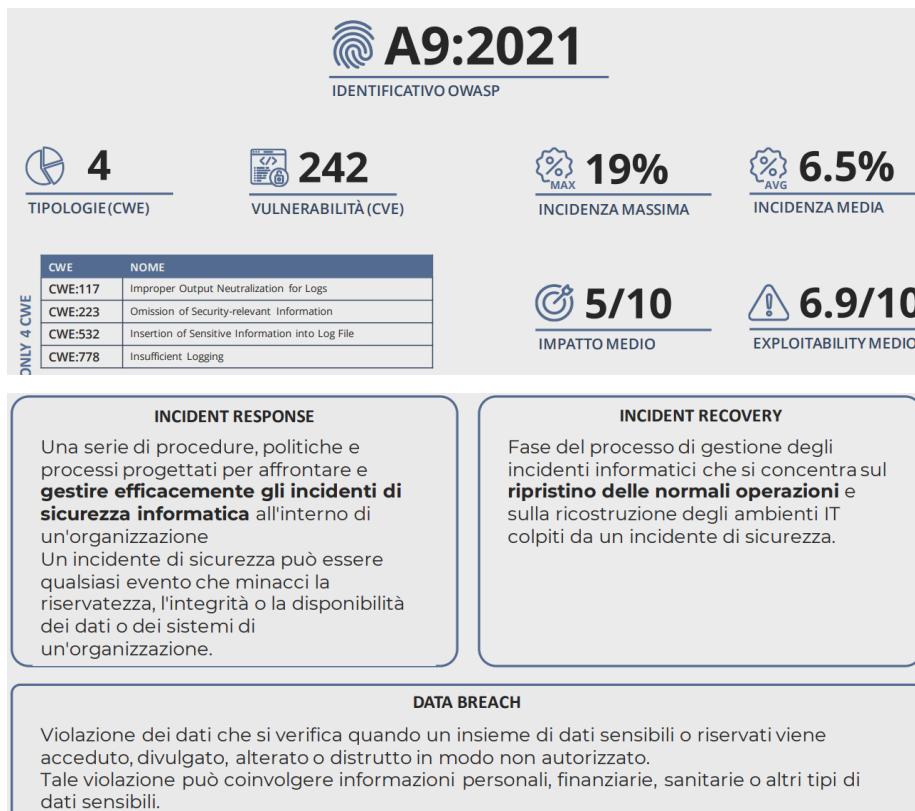
La gestione delle vulnerabilità di software e integrità dei dati richiede un approccio strategico. L'implementazione di firme digitali, controlli di accesso e la verifica delle dipendenze sono essenziali per mitigare rischi come l'esecuzione di codice malevolo o la compromissione dei dati.

Processi di revisione e monitoraggio continuo sono fondamentali per individuare e correggere potenziali vulnerabilità. L'adozione di pratiche sicure nella catena di distribuzione del software, inclusa la protezione delle pipeline CI/CD, contribuisce a prevenire accessi non autorizzati e modifiche indesiderate.

Mantenere le librerie e i framework aggiornati è cruciale per beneficiare delle ultime correzioni di sicurezza. In sintesi, un approccio completo e proattivo è essenziale per garantire la robustezza e l'integrità dei sistemi software.

## Security logging and monitoring failures

Una delle principali sfide nella sicurezza delle applicazioni riguarda l'insufficienza o la totale mancanza di logging e monitoraggio.



La definizione è:

I Security Logging and Monitoring Failures si presentano quando non viene svolto in maniera efficace un monitoraggio di attività sospette o potenzialmente dannose. Le applicazioni sicure, infatti, devono essere in grado di registrare in modo adeguato le attività rilevanti, come gli eventi di sicurezza, gli accessi, le modifiche e altri comportamenti significativi.

Nella tabella alcune contromisure:

CONTROMISURA	DETALGO
Registrazione e Monitoraggio	Assicurarsi che tutti i login, il controllo degli accessi e gli errori derivanti dalla verifica degli input lato server possano essere registrati con un contesto utente sufficiente.
Sicurezza dei dati di log	Codificare correttamente i dati di log per prevenire injection o attacchi ai sistemi di registrazione o monitoraggio.
Integrità delle transazioni di alto valore	Assicurarsi che le transazioni di alto valore abbiano un audit trail. Implementare controlli di integrità, come tabelle append-only in un database, per prevenire manipolazioni o cancellazioni non autorizzate.
Monitoraggio delle attività sospette	I team DevSecOps dovrebbero stabilire sistemi di monitoraggio di allerta efficaci. Rilevare e affrontare rapidamente attività sospette per garantire una risposta tempestiva a potenziali minacce.
Piano di Incident Response e Recovery	Stabilire o adottare un piano di incident response e recovery. Seguire framework riconosciuti, come il National Institute of Standards and Technology (NIST) 800-61r2 o versioni successive, per garantire una risposta strutturata agli incidenti e la capacità di recuperare rapidamente da eventi avversi.

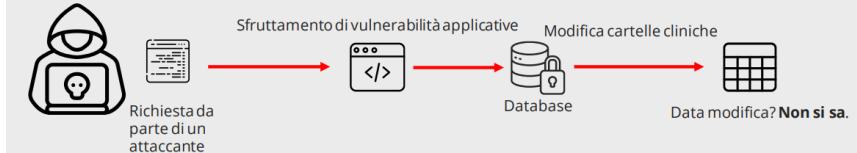
## SCENARIO – ASSENZA LOGGING E MONITORAGGIO

### CONTESTO

Un amministratore di un sito web di un fornitore di servizi sanitari non ha potuto rilevare una violazione a causa di mancanza di logging e monitoraggio.

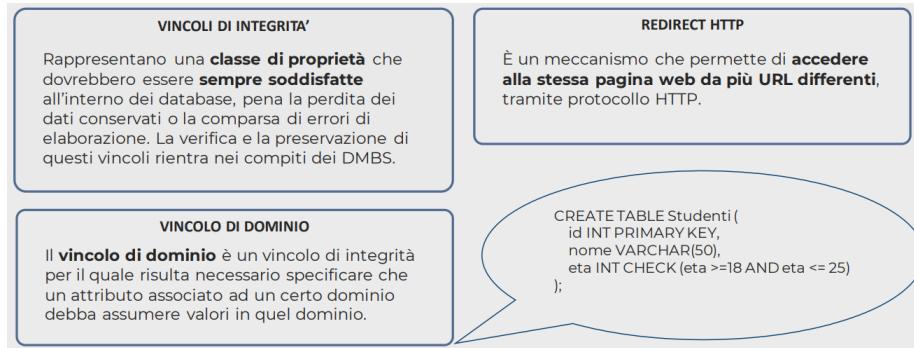
### SCENARIO

Una terza parte ha informato il fornitore del piano sanitario che un attaccante aveva acceduto e modificato migliaia di cartelle cliniche di più di 3,5 milioni di persone. Un'indagine post-incidente ha rilevato che gli sviluppatori del sito web non avevano corretto delle vulnerabilità significative presenti. Poiché non c'era nessuna forma di logging o monitoraggio del sistema, la violazione dei dati potrebbe essere stata in corso dal 2013, un periodo di più di sette anni.



## Server-side request forgery





La definizione è:

Il Server-Side Request Forgery (SSRF) è un vettore di attacco che sfrutta un'applicazione per interagire con la rete interna/esterna o la macchina stessa. Si manifesta attraverso l'errata gestione degli URL. Questo può coinvolgere immagini su server esterni, Web Hook personalizzati e richieste interne per interagire con altri servizi.

Il flusso comune di SSRF coinvolge la prima richiesta, spesso HTTP, seguita da una seconda richiesta che può utilizzare diversi protocolli e schemi.

A seconda delle funzionalità e dei requisiti dell'applicazione, SSRF può verificarsi in due casi principali:

- Vincolo di Dominio/IP Identificato e Fidato:
- L'applicazione può inviare richieste solo a domini o IP identificati e fidati. o Libero Accesso a IP o domini esterni: l'applicazione può inviare richieste a qualsiasi indirizzo IP o nome di dominio esterno.

Nella tabella alcune contromisure:

CONTROMISURA	DETTAGLIO	CONTROMISURA
Validazione dell'Input	Implementare controlli rigorosi sulla validità degli indirizzi IP e dei nomi di dominio forniti dagli utenti, assicurandosi che rispettino i formati attesi.	Utilizzo di Token di Autenticazione Casuale
Verifica dell'Indirizzo IP	Verificare che l'indirizzo IP fornito sia pubblico, escludendo quelli associati a reti private. Ciò previene l'accesso non autorizzato a risorse interne.	Limitazioni del Protocollo
Convalida del Nome di Dominio	Verificare che i nomi di dominio forniti siano presenti in una lista di allow-list di domini fidati, riducendo il rischio di collegamenti a risorse non autorizzate. <i>Se un'applicazione può inviare richieste a qualsiasi IP o dominio</i>	Controllo dei Redirect HTTP
		Monitoraggio Costante

La gestione efficace delle vulnerabilità SSRF richiede una strategia bilanciata, adottando una doppia strategia.

A livello di rete, implementare regole firewall "deny by default" e suddividere le reti per limitare l'accesso.

Sul fronte applicativo, usare le allow list per ridurre il rischio. In scenari di comunicazione con risorse specifiche, le allow list sono efficaci. Un'alternativa alle allow list sono i token di sicurezza, poiché garantiscono un altro livello di sicurezza.

La consapevolezza del personale è cruciale, evita input URL completi e garantisce la conformità ai protocolli. Un approccio integrato tra rete e applicazione è essenziale per difendersi da minacce SSRF.

## Crittografia

Disciplina che racchiude i principi/mezzo per trasformare i dati al fine di nascondere il contenuto e impedirne l'uso e modifiche non autorizzate.

Garantisce:

- confidenzialità: i dati non vengono visti da chi non autorizzato;
- integrità: i dati non vengono alterati;
- autenticità: si è certi che i dati provengono davvero dalla fonte.

Alcune terminologie:

- Messaggio in chiaro: il messaggio di partenza che si vuole proteggere con cifratura
- Messaggio cifrato: output di un algoritmo di crittografia, appare con una serie di cifre casuali.
- Chiave: Un parametro utilizzato da un algoritmo per effettuare un'operazione crittografica (es. cifratura di un messaggio in chiaro o decifratura di un messaggio cifrato). È detta pubblica se è possibile divulgarla pubblicamente senza compromettere la sicurezza dei dati cifrati, privata altrimenti.
- Schema crittografico: insieme di trasformazioni specificate senza ambiguità che richiede la collaborazione di due o più parti al fine di ottenere un servizio (crittografico).
- Security strength: numero, indicato in bit, che indica la quantità di lavoro necessaria per violare un algoritmo crittografico; se il suo valore è  $S$  bit sono necessari  $2^S$  operazioni.
- Criptoperiodo: L'arco di tempo in cui una determinata chiave è autorizzata all'uso o in cui le chiavi di un determinato sistema possono rimanere in vigore.

## **Algoritmi a chiave simmetrica**

Classe di algoritmi dove viene definita e usata una sola chiave per cifrare e decifrare, è necessario condividerla fra attori della comunicazione.

Vantaggi:

- Alta efficienza in termini di costo computazionale
- Accelerazione hardware presente all'interno della maggior parte delle CPU moderne
- Le quantità di dati cifribili con una chiave sono molto ampie

Svantaggi:

- Soffrono del problema dello scambio della chiave: per iniziare una comunicazione sicura, è necessario avere a disposizione una chiave condivisa con la controparte

### **Cifratura a blocchi**

L'algoritmo divide i dati da cifrare in blocchi di dimensione fissa, successivamente con la chiave cifriamo e decifriamo ciascun blocco.

### **Cifratura a flusso**

Tramite la chiave simmetrica generiamo un flusso di bit pseudo-casuali, ogni bit del messaggio in chiaro verrà messo a XOR con il flusso appena ottenuto. Per decifrare facciamo le stesse operazioni.

Un problema è che un malintenzionato potrebbe modificare il messaggio originale andando a cambiare un simbolo del testo cifrato.

### **Operazioni cifrari a blocchi**

Prima di vedere le operazioni vediamo due definizioni:

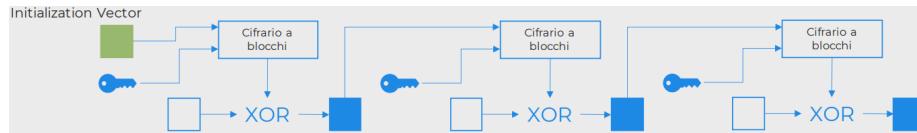
- Nonce: valore che deve essere utilizzato una volta sola. Deve essere diverso per ogni comunicazione, tuttavia non ha altri tipi di requisiti.
- Initialization Vector: deve essere casuale e non deve avere nessun tipo di legame con gli IV utilizzati in comunicazioni passate.
- 

Per esempio, se un cifrario richiede l'utilizzo di un Nonce, potremo generare un valore casuale per la prima comunicazione e incrementarlo di un valore fisso per tutte le comunicazioni successive. Questo non sarebbe possibile con un cifrario che richiede l'utilizzo di un IV, perciò sarebbe necessario generare una valore casuale ad ogni nuova comunicazione, altrimenti il livello di sicurezza offerto dall'algoritmo sarebbe compromesso.

**ECB** Electronic Code Block, ogni blocco viene cifrato con la stessa chiave, avendo poi più blocchi cifrati uguali, per questo non si consiglia di utilizzare la stessa chiave per ogni blocco rendendo il cifrario debola ad attacchi basati su crittoanalisi.

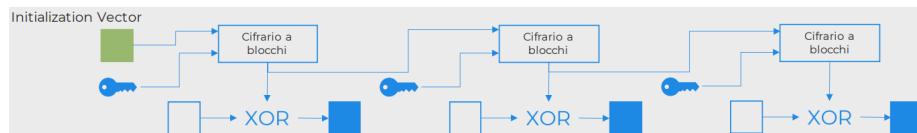
**CBC** Cipher Block Chaining, il primo blocco in chiaro viene messo a XOR con un initialization vector, ottenendo un blocco cifrato che verrà messo a XOR con il prossimo in chiaro e così via.

**CFB** Cipher Feedback Mode, si genera un cifrario a blocchi (creato da chiave e initialization vector) che si mette a XOR con il primo blocco in chiaro ottenendo un blocco cifrato, successivamente si rigenera il cifrario a blocchi con il blocco appena creato e la chiave, che verrà messo a XOR con il prossimo blocco non cifrato, e così via.



**OFB** Output Feedback Mode, come il precedente ma invece di utilizzare il blocco cifrato per creare un altro cifrario a blocchi ma utilizziamo il cifrario appena utilizzato con la chiave per crearne uno nuovo.

In questo modo non propaghiamo errori in un bit su tutto il messaggio.



**CTR** Counter mode, utilizziamo un nonce, uguale per tutti i blocchi, e un contatore unico per ciascun blocco.

Il nonce e il contatore vengono cifrati con il cifrario a blocchi; l'output viene messo a XOR con il blocco in chiaro per generare il blocco cifrato.

Il vantaggio che questa modalità offre è la possibilità di parallelizzare completamente la cifratura dei vari blocchi in quanto non c'è nessuna dipendenza fra i diversi blocchi.

### Data at rest e data in transit

Data at rest intende le info. memorizzate su un dispositivo o database, i data in transit intende i dati mentre sono trasmessi.

## **XTS**

Modalità per proteggere specificamente i dati at rest.

Utilizza due chiavi:

- principale: uguale per tutti i dati;
- specifica: per ogni blocco (si utilizza il numero di settore).

## **Authenticated Encryption**

Fornisce una classe di algoritmi detta Authenticated Encryption with Associated Data (AEAD). La parte del nome Associated Data fa riferimento alla possibilità di includere nel messaggio una parte di dati che non verranno cifrati.

Tramite funzioni MAC (per garantire autenticità) e algoritmi con chiave simmetrica garantisce confidenzialità e autenticità.

Gli algoritmi sono:

- AES-GCM
- AES-CCM
- ChaCha 20-Poly 1305

## **DES, TDES, 2TDES**

Data Encryption Standard, utilizza una cifratura a 56 bit, al giorno d'oggi non è sicura.

Triple DES, utilizza tre chiavi a 56 bit ed agno blocco viene applicato un algoritmo di DES di cifratura utilizzando la prima chiave, l'algoritmo di decifrazione DES usando la seconda chiave, l'algoritmo di cifratura DES usando la terza chiave.

Nella cifratura 2TDES viene eseguita la stessa procedura di TDES ma la prima e la terza chiave utilizzate sono la stessa, risultando in una chiave di lunghezza effettiva di 112bit.

## **RC4**

Nonostante sia usato nell'active directory non è più consigliato da usare.

## **AES**

Insieme di algoritmi, ad oggi è il cifrario a blocchi più utilizzato.

Ha tre versioni (AES-128,AES-192,AES-256) e ciascuna delle tre versioni fornisce un livello di security strength pari alla lunghezza della chiave.

## **ChaCHa 20-Poly 1305**

Si tratta di uno schema AEAD basato sul cifrario a flusso ChaCHa20 e la funzione Poly1305.

Il NIST non ha fornito alcuna indicazione sulla sua sicurezza, riconosciuto dall'IETF ed è stato indicato come unico altro utilizzabile all'interno del protocollo TLS 1.3

Questo cifrario risulta più efficiente in termine di costo computazionale su quei sistemi che non forniscono accelerazione hardware per AES, ad esempio alcuni microprocessori utilizzati in ambito embedded e IoT.

### Algoritmi a chiave asimmetrica

Si utilizzano due chiavi diverse, una per cifrare e una per decifrare. Una delle due chiavi è pubblica ed è condivisibile su un canale pubblico senza problemi; l'altra è privata non deve essere condivisa con nessuno ed è legata matematicamente alla prima.

Questi algoritmi garantiscono confidenzialità/ integrità. Infatti il mittente dovrà usare la chiave pubblica del destinatario per cifrare il messaggio. In questo modo, solo il legittimo possessore della chiave privata correlata matematicamente con la chiave pubblica utilizzata sarà in grado di decifrare il messaggio ricevuto.



Per garantire l'autenticazione il mittente utilizzerà la propria chiave privata per cifrare il messaggio (o digest ottenuto da una funzione hash). Dopodiché invierà il messaggio, la firma ottenuta e la propria chiave pubblica. In questo modo il ricevente potrà usare la chiave pubblica per decifrare la firma, confrontarla con il messaggio ricevuto e, se questa corrisponde, saprà che l'unica persona in grado di generare quella firma è il legittimo detentore della chiave privata associata alla chiave pubblica che ha ricevuto. Questo processo è utilizzato per la firma digitale.



Vantaggi:

- È possibile inviare la chiave pubblica su un canale insicuro, per cui non presenta il problema dello scambio della chiave

Svantaggi:

- Sono più costosi a livello computazionale rispetto agli algoritmi a chiave simmetrica
- La quantità di dati che è possibile cifrare è limitata dalla grandezza della chiave

### **RSA**

Basato sulla complessità computazionale del problema di fattorizzazione dei numeri primi, garantisce confidenzialità durante la trasmissione e autenticazione di documenti o dati tramite firma digitale.

### **DSA**

Si basa sulla complessità computazionale del problema logaritmo discreto. A differenza di RSA, non viene utilizzato per la trasmissione di chiavi su canali insicuri ma solamente per le sue applicazioni in ambito di autenticazione e firma digitale.

### **ECDSA**

Elliptic Curve Digital Signature Algorithm, si basa sulle proprietà delle curve ellittiche su campi finiti. Come DSA, è un algoritmo specifico per la firma digitale e viene usato solo in ambito di autenticazione e firma digitale.

Prima abbiamo parlato di curve ellittiche, la crittografia con queste curve (ECC) è un approccio alla crittografia a chiave pubblica basato sulla struttura algebrica delle curve ellittiche su campi finiti. L'ECC consente chiavi più piccole rispetto alla crittografia a chiave pubblica tradizionale, consentendo di aumentare il livello di sicurezza senza aumentare eccessivamente il costo computazionale.

### **Key transport e agreement**

Si può utilizzare un algoritmo a chiave asimmetrica per cifrare una chiave condivisa, che verrà poi utilizzata per cifrare il resto della comunicazione con un cifrario a chiave simmetrica.

In questo modo otteniamo:

- L'alta efficienza computazionale di un algoritmo a chiave simmetrica
- La possibilità di cifrare grandi quantità di dati con una stessa chiave
- La possibilità di scambiare la chiave su un canale insicuro

Quando la chiave viene simmetrica viene generata da una delle due parti e trasmessa si dice key transport, se entrambe le parti contribuiscono in egual modo alla generazione si parla di key agreement.

Il key agreement è più sicuro perché permette l'implementazione di proprietà perfect forward secrecy (anche alla compromissione di una chiave in futuro i messaggi passati non sono intaccati).

## **Diffie-Hellman**

Permette a due controparti di una comunicazione, entrambe in possesso di una coppia di chiavi pubbliche e private, di stabilire un segreto condiviso attraverso una rete insicura.

La versione di DH chiamata Ephimeral (DHE) implementa la perfect forward secrecy generando una coppia di chiavi pubblica e privata nuova per ogni nuova comunicazione. Esiste anche una versione del protocollo basata sull'algebra delle curve ellittiche, chiamata ECDH o ECDHE.

Immune agli attacchi "Man in the Middle" dopo la generazione delle chiavi. Tuttavia, è vulnerabile se un agente terzo falsifica le informazioni pubbliche all'inizio e inganna le due controparti.

## **Funzioni di hash**

Una funzione di hash è unidirezionale e restituisce un output di lunghezza fissa ed è chiamato per l'appunto digest. Quindi anche ad un cambio minimale dell'input con lunghezza variabile ottengo un output completamente diverso.

Come già detto le funzioni di hash sono unidirezionali, quindi non è possibile risalire agli input partendo dal digest.

Per essere conformi nell'ambito crittografico la funzione di hash deve avere le seguenti caratteristiche:

- Resistenza alla preimmagine: cioè deve essere difficile a livello computazionale risalire ad un input.
- Resistenza alla seconda preimmagine: difficile trovare un secondo input che produca lo stesso hash.
- Resistenza alla collisione: improbabile avere due input che formino lo stesso hash.

Le funzioni di hash vengono utilizzate per verificare l'integrità del messaggio, infatti ricevuto un messaggio e il suo hash il destinatario calcolerà l'hash a sua volta con la stessa funzione e la confronta con quella ricevuta.

Oltre a ciò garantisce l'autenticazione, (Message Authentication Code), cioè l'input della funzione hash richiede, oltre al messaggio, anche una chiave segreta e condivisa fra mittente e destinatario, dopodiché il procedimento rimane uguale a quello per verificare l'integrità.

L'hash può tornare utile per salvare password in un database, rendendole criptate in caso di furto di dati e anche per firma digitale.

**MD4/MD5** Producono un output di 128bit. A causa delle vulnerabilità riscontrate nel corso degli anni, MD4 e MD5 non sono più considerati sicuri per l'ambito della crittografia.

Ancora presenti in sistemi active directory con protocollo NTLM

**SHA-1** Produce un output di 160bit. Al giorno d'oggi il livello di sicurezza garantito da questa funzione non è più considerato adatto all'ambito della crittografia, pertanto non dovrebbe essere utilizzato per nuove applicazioni.

**SHA-2** Indica un insieme di funzioni che sono considerate sicure, la funzione più utilizzata fra queste è SHA-256, che come indicato dal nome produce un output di 256 bit.

Alcune versioni erano vulnerabili ad attacchi di tipo length extension, un attacco mirato a sistemi che firmano un dato aggiungendo un segreto e poi calcolando l'hash di segreto + messaggio.

L'attacco può essere utilizzato su funzioni hash come MD5 e SHA-1, che per via del loro funzionamento interno, dividono l'input in blocchi e combinano l'output del blocco precedente con il blocco corrente per produrre il nuovo output.

L'attacco permetterà quindi di aggiungere dati arbitrari al messaggio in chiaro e di calcolare un nuovo MAC valido utilizzando l'hash intercettato, anche senza essere a conoscenza del segreto utilizzato.

Questo tipo di attacco riduce il livello di sicurezza di alcune funzioni della famiglia SHA-2, infatti sono stati introdotti gli algoritmi SHA-512/224 e SHA-512/256 che, attraverso il troncamento annullano l'efficacia dell'attacco.

**SHA-3** Altra famiglia di funzioni, standardizzata nel 2015 dal NIST in seguito ad un concorso indetto per identificare un algoritmo alternativo a quello utilizzato nelle funzioni SHA-2, in modo da avere un sostituto pronto qualora una nuova vulnerabilità compromettesse la sicurezza di quest'ultimo.

## Funzioni MAC

**HMAC** Keyed-Hash Message Authentication Code (HMAC) è un MAC che utilizza una funzione hash e una chiave per produrre un digest che permetta di garantire l'integrità e l'autenticità di un messaggio.

**CMAC** Cipher-block-chaining-based MAC, è un MAC standardizzato dal NIST che usa alla sua base un algoritmo a blocchi come TDES o AES, secondo la modalità di operazione CBC. La Security Strength dell'algoritmo MAC in questo caso è considerata pari a quella del cifrario a blocchi sottostante.

**KMAC** KMAC, è un MAC basato su SHA3. Prende il nome dell'algoritmo alla base di SHA3 ovvero il Keccak; può supportare un security strength fino a 256bit, a patto che sia utilizzata una chiave di pari lunghezza.

**GMAC** Galois MAC (GMAC) come CMAC, ma utilizza un cifrario a blocchi in modalità GCM.

## **Man in the middle**

Forma di attacco informatico in cui un aggressore intercetta e modifica la comunicazione tra due parti, facendo in modo che entrambe credono di comunicare direttamente tra loro quando in realtà tutte le informazioni passano attraverso l'attaccante.

Per prevenire attacchi di questo tipo, è stata introdotta l'autenticazione basata sui certificati.

## **Certificati**

Documento digitale emesso da una Certificate Authority con una parte pubblica e una parte privata. Non è possibile falsificare un certificato in quanto non è in possesso della chiave privata che la CA ha utilizzato per firmarlo.

La parte pubblica contiene informazioni sull'ente a cui è stato rilasciato il certificato e che è autorizzato ad utilizzarlo, come un indirizzo internet per cui può essere utilizzato (es: \*.google.it), informazioni sull'ente che ha firmato e rilasciato il certificato, il periodo di validità del certificato, la firma del certificato e la chiave pubblica.

La parte privata la chiave privata.

## **CA**

Certification Authority, cioè l'ente che firma i certificati e li rende effettivamente validi.

Le CA accettate vengono configurate a livello di S.O. o a livello applicazione (browser).

## **TLS**

Il protocollo Transport Layer Security (TLS) è il protocollo utilizzato durante le comunicazioni su internet per garantire Confidenzialità, Integrità ed Autenticazione dei dati.

Utilizza una combinazione di tutte le tecnologie e gli algoritmi visti durante questa lezione:

- Algoritmi di Key Exchange o Key Agreement per lo scambio della chiave
- Algoritmi a chiave simmetrica per la cifratura della comunicazione
- Funzioni Hash e MAC per la verifica dell'integrità
- Certificati e Firma Digitale per l'autenticazione della controparte

La scelta di quali algoritmi utilizzare per ciascuno scopo è negoziata durante una fase iniziale detta di «handshake».

## Sviluppo sicuro

O secure coding, si riferisce alle pratiche di sviluppo per minimizzare la presenza di vulnerabilità.

Lo scopo principale è la prevenzione in tutte le fasi di vita di un software.

In ogni fase dello sviluppo ci sono dei piccoli passi da seguire:

- governance: bisogna redigere alcune regole che verranno seguite per tutto lo sviluppo;
- design: fare valutazioni sui requisiti di sicurezza, come che protocolli usare. Si può usare il threat modeling, cioè l'insieme dei pericoli che potrebbero avvenire e eventuali soluzioni;
- implementation: utilizzare le best practices dello sviluppo sicuro;
- verification: verifichiamo tramite test il codice;
- operation: gestione dell'applicativo in running e monitoraggio delle vulnerabilità.

## Standard

Ora vedremo le principali organizzazioni di riferimento e relativi standard.

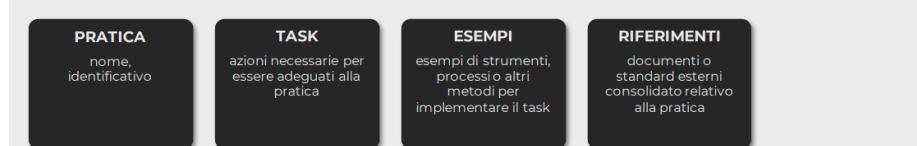
### NIST SP

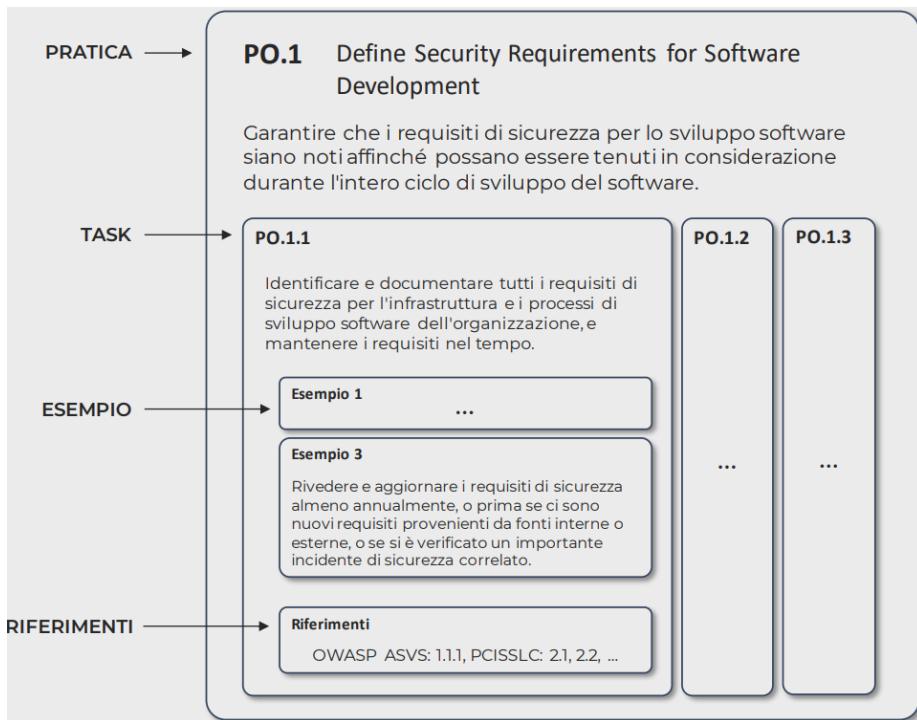
Il NIST Special Publication (SP) 800-218, noto anche come Secure Software Development Framework (SSDF) è stato concepito come un quadro di riferimento completo per garantire la sicurezza del software durante il ciclo di vita dello sviluppo.

Le pratiche del framework sono divise in 4 gruppi:

1. Prepare the organization;
2. Protect the software;
3. Produce well-secured software;
4. Respond to vulnerabilities.

Il contenuto di ogni pratica viene formalizzato sulla base dei seguenti elementi:





## PCI DSS

Il PCI DSS, acronimo di Payment Card Industry Data Security Standard, è uno standard di sicurezza delle informazioni creato per garantire la protezione dei dati delle carte di pagamento.

Composto da 12 requisiti che richiedono implementazione di processi, criteri o soluzioni specifiche; l'unico riguardante lo sviluppo sicuro è il 6.

Nello specifico il Requisito (6.5.3) dice:

Gli ambienti pre-produzione sono separati dagli ambienti di produzione e la separazione è applicata tramite controllo degli accessi.

Gli approcci definiti per procedure di test sono:

- 6.5.3.a: Esaminare le politiche e le procedure per verificare che siano definiti processi per separare l'ambiente di pre-produzione dall'ambiente di produzione tramite controllo degli accessi
- 6.5.3.b: Esaminare la documentazione della rete e le configurazioni di sicurezza per verificare che l'ambiente di pre-produzione sia separato dall'ambiente/i di produzione.
- 6.5.3.c: Esaminare le configurazioni del controllo degli accessi per verificare che siano in atto controlli che garantiscano la separazione tra l'ambiente di pre-produzione e l'ambiente/i di produzione.

## OWASP ASVS

ASVS è un framework e standard di sicurezza progettato per valutare la sicurezza delle applicazioni web e dei servizi web.

Definisce tre livelli di verifica della sicurezza (più è alto il livello più è alta la sicurezza):

1. destinato a livelli bassi e verificabile tramite penetration test
2. destinato ad applicazioni con dati sensibili: richiedono protezione ed è il livello, consigliato per la maggior parte delle app
3. destinato ad applicazioni più critiche che:
  - gestiscono transazioni ad alto valore,
  - contengono dati medici sensibili,
  - richiedono il massimo livello di fiducia,

	L1	L2	L3
Verificare che i token di sessione basati su cookie abbiano l'attributo 'Secure' impostato.	✓	✓	✓
Verificare che l'applicazione non registri credenziali o dettagli di pagamento. I token di sessione dovrebbero essere memorizzati nei registri solo sotto forma di hash, in modo irreversibile.	✓	✓	✓
Verificare che i security logs siano protetti dall'accesso e dalla modifica non autorizzati.	✓	✓	
Verificare che sia in uso uno strumento di analisi del codice che possa rilevare codice potenzialmente dannoso.		✓	
Verificare che l'applicazione non accetti upload di file di grandi dimensioni che potrebbero riempire lo spazio di archiviazione o causare denial of service.	✓	✓	✓
Verificare che l'applicazione abbia un sistema alerting configurabile quando vengono rilevati attacchi automatizzati o attività insolite.	✓	✓	
Verificare che i token di sessione siano generati utilizzando algoritmi crittografici approvati.	✓	✓	
Verificare che vengano eseguiti backup regolari dei dati più importanti e che ne venga eseguito il test di restore.		✓	

## SEI CERT SCS

Il SEI CERT Secure Coding Standard è un insieme di linee guida e best, practice pratiche e efficaci, sviluppate dal Software Engineering Institute (SEI) per promuovere:

- la scrittura di codice sicuro e affidabile
- ridurre il rischio di vulnerabilità di sicurezza nel software

## **CIS SCS**

Il CIS Secure Coding Standard è un insieme di linee guida e best practice sviluppate dal Center for Internet Security (CIS) per promuovere la scrittura di codice sicuro e resistente agli attacchi.

Questo standard fornisce una serie di regole e raccomandazioni progettate per ridurre il rischio di vulnerabilità di sicurezza nel software durante tutte le fasi del ciclo di vita del software.

### **Principi di progettazione sicura**

#### **Shift left**

Approccio che sposta l'attenzione sulla sicurezza fin dalle prime fasi del ciclo di vita del software.

Efficace nel garantire che la sicurezza sia presa in considerazione da subito nella progettazione, riducendo così i costi e il rischio di possibili violazioni.

#### **Deny by default**

L'idea chiave è che tutte le richieste di accesso o le connessioni sono respinte automaticamente, a meno che non siano esplicitamente autorizzate da regole specifiche.

#### **Least privilege**

Questo concetto si basa sull'idea di ridurre al minimo i privilegi concessi a ciascun utente, processo o sistema, al fine di mitigare il rischio di potenziali minacce alla sicurezza.

#### **Defense in depth**

Strategia che prevede la creazione di strati multipli di difesa, combinando misure tecniche, procedurali e fisiche per proteggere un sistema da minacce esterne e interne.

Si mira a ridurre la possibilità di un compromesso o di danni attraverso la diversificazione delle difese lungo l'intera infrastruttura.

#### **Security by design**

Approccio di sviluppo dove le misure di sicurezza sono integrate nella progettazione e nell'architettura di un sistema fin dall'inizio.

Essenziale per creare sistemi software robusti e resilienti, non seguendo questo approccio si avranno applicazioni vulnerable-by-design.

Approccio a 6 step:



## Best practices di sviluppo

### Validate input

Verificare e garantire che i dati inseriti in un'applicazione rispettino determinati criteri e regole.

Prevenzione verso attacchi di injection

### Compiler warnings

Avvertimenti emessi durante la compilazione del codice per segnalare possibili problemi o pratiche di programmazione rischiose.

### KISS

"Keep It Simple, Stupid" promuove la scrittura di codice semplice e diretto, minimizzando la complessità non necessaria.

### Data sanitization

La Data Sanitization è il processo di pulizia e validazione dei dati per prevenire attacchi informatici come SQL injection e cross-site scripting.

Un esempio è la parametrizzazione delle query che consiste nel separare i dati dalle istruzioni SQL.

### Linee guida di programmazione sicura

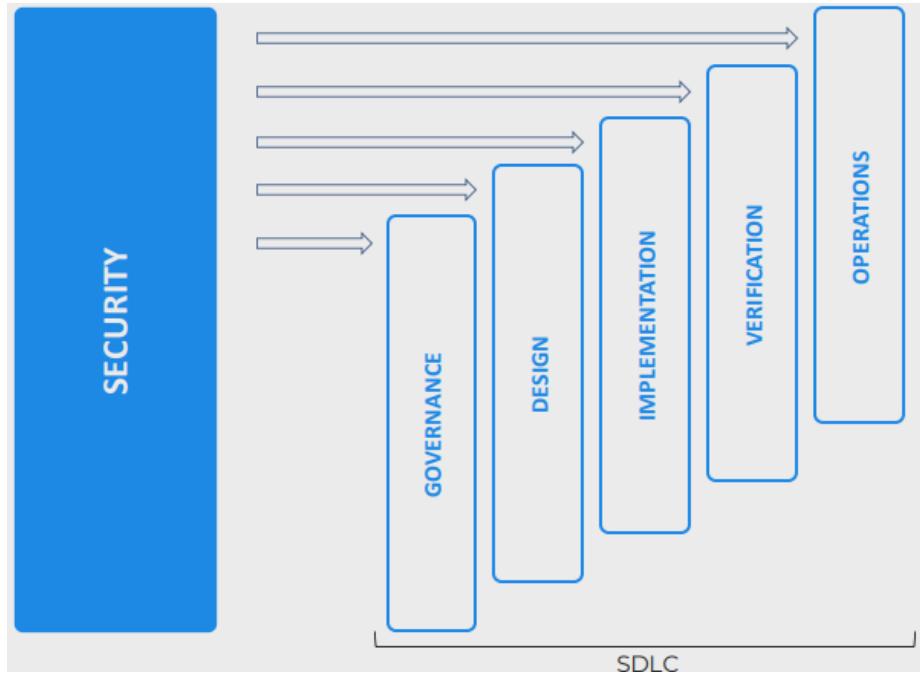
Le linee guida possono essere generali o specifiche per l'OOP:

- generali da 39 a 68,
- specifiche da 69 a 76,

QUI

## SSDLC

Il ciclo di vita dello sviluppo del software (SDLC) è un processo strutturato che consente lo sviluppo di software di alta qualità, a basso costo e nel minor tempo possibile. Il Secure SDLC (SSDLC) integra la sicurezza nel processo.



Alcuni concetti del SSDLC sono: Shift left, Defense in depth e Security-by-design.

## OWASP SAMM V.2

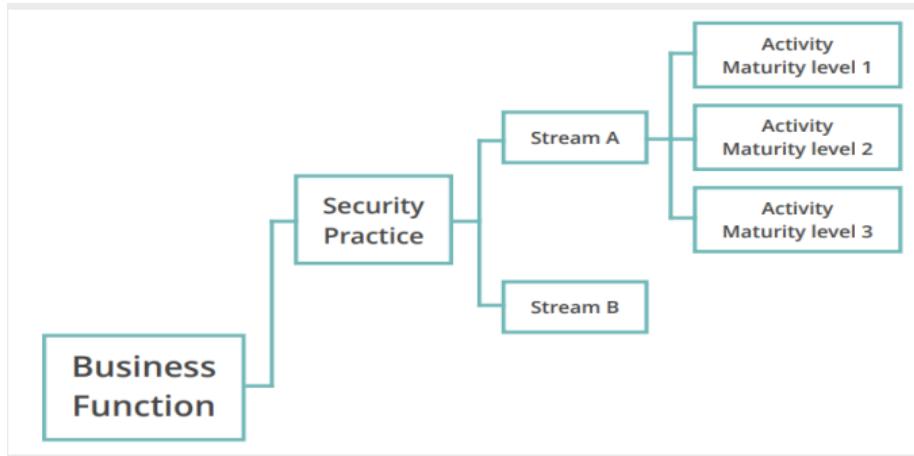
Software Assurance Maturity Model, framework per il SSDLC.

SAMM supporta l'intero ciclo di vita del software ed è agnostico, pensato per essere evolutivo e guidato dal rischio, poiché non esiste una singola ricetta che funzioni per tutte le organizzazioni.

Lo standard è:

- Misurabile
- Eseguibile
- Versatile

Basato su 15 pratiche di sicurezza raggruppate in 5 funzioni aziendali, con ogni pratica che contiene un insieme di attività strutturate in 3 livelli di maturità, ogni livello indica la “facilità”.



Ogni funzione aziendale (fase del SSDLC) è una categoria di attività che qualsiasi organizzazione coinvolta nello sviluppo software deve soddisfare in qualche misura.

Ogni fase ha tre pratiche, cioè attività correlate alla sicurezza che garantiscono garanzia per la funzione correlata.

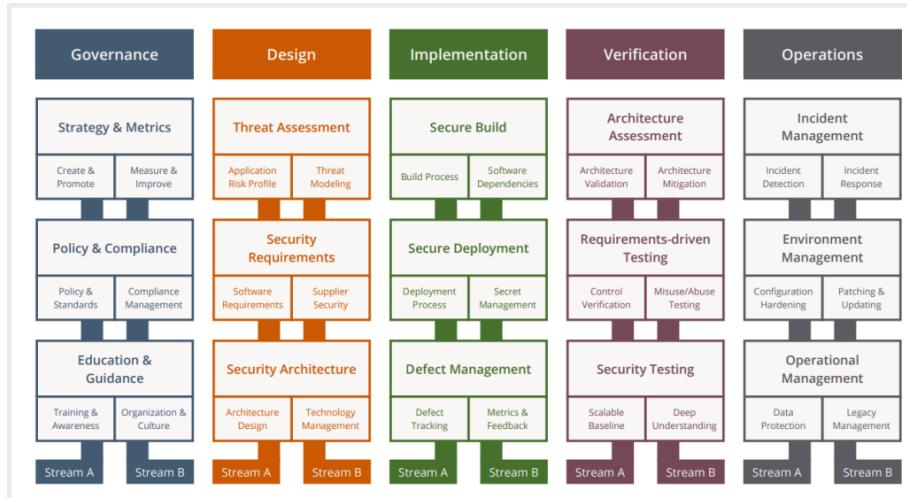
Le pratiche hanno attività, raggruppate e divise in due filoni detti stream.

Gli stream coprono diversi aspetti di una pratica e hanno i propri obiettivi, allineando e collegando le attività nella pratica attraverso i diversi livelli di maturità.

I livelli di maturità sono come degli obiettivi, dove ogni livello ha obiettivi progressivamente più sofisticati con attività specifiche e metriche di successo più stringenti.

In generale i livelli rappresentano:

- 0: pratica non soddisfatta
- 1: comprensione iniziale della pratica
- 2: aumento dell'efficienza/efficacia della pratica
- 3: padronanza completa della pratica



Le 5 fasi con le loro 3 pratiche con le rispettive attività divise nei 2 stream

## Fasi / Funzioni aziendali

### Governance

Concentrazione sui processi e sulle attività relative a come un'organizzazione gestisce le attività complessive di sviluppo software.

Le "Practices" sono:

- Strategia e Metriche: costruisce un piano complessivo per le attività di sviluppo software sicuro.
- Stream A: create and promote di una roadmap per la sicurezza delle applicazioni; per definire obiettivi e allineare le parti.
- Stream B: misurare e migliorare la roadmap misurando le prestazioni nell'organizzazione.

Stream A	Stream B
<b>Livello di maturità 1:</b> identificare gli obiettivi e i mezzi per misurare l'efficacia del programma di sicurezza.	
Identificare i fattori trainanti dell'organizzazione in relazione alla sua tolleranza al rischio.	Definire metriche con un'analisi sull'efficacia e l'efficienza del Programma di Sicurezza delle Applicazioni.
<b>Livello di maturità 2:</b> Stabilire una roadmap strategica unificata per la sicurezza del software all'interno dell'organizzazione.	
Stabilire una strategia unificata per la sicurezza delle applicazioni.	Stabilire obiettivi e KPI per misurare l'efficacia del programma.
<b>Livello di maturità 3:</b> Allineare gli sforzi di sicurezza con gli indicatori organizzativi rilevanti e i valori degli asset.	
Allineare il programma di sicurezza delle applicazioni per supportare la crescita dell'organizzazione.	Influenzare la strategia basandosi sulle metriche e sulle esigenze organizzative.

- Politiche e Conformità: guida il rispetto degli standard e delle normative.
- Stream A: policy & standard da gestire e fornire per l'integrazione nel SDLC.
- Stream B: compliance management cioè individuare e fornire i requisiti di conformità per l'integrazione nel SDLC.

Stream A	Stream B
<b>Livello di maturità 1:</b> Identificare e documentare i fattori di governance e conformità rilevanti per l'organizzazione.	
Determinare una base di sicurezza che rappresenti le politiche e gli standard dell'organizzazione.	Identificare i driver e i requisiti di conformità di terze parti e mapparli alle politiche e agli standard esistenti.
<b>Livello di maturità 2:</b> Stabilire una base di sicurezza e conformità specifica dell'applicazione.	
Sviluppare requisiti di sicurezza applicabili a tutte le applicazioni.	Pubblicare requisiti specifici di conformità per le applicazioni e indicazioni per i test.
<b>Livello di maturità 3:</b> Misurare l'adesione alle politiche, agli standard e ai requisiti di terze parti.	
Misurare e riportare lo stato dell'aderenza individuale delle applicazioni alle politiche e agli standard.	Misurare e riportare la conformità individuale delle applicazioni ai requisiti di terze parti.

- Educazione e Orientamento: aumenta la conoscenza nell'organizzazione riguardo al software sicuro.
- Stream A: formazione e sensibilizzazione sulla sicurezza del software tra gli stakeholder.
- Stream B: organizzazione e cultura aziendale si concentrano sulla promozione della sicurezza delle applicazioni nell'organizzazione per il successo di un progetto SDLC.

Stream A	Stream B
<b>Livello di maturità 1:</b> identificare gli obiettivi e i mezzi per misurare l'efficacia del programma di sicurezza.	
Identificare i fattori trainanti dell'organizzazione in relazione alla sua tolleranza al rischio.	Definire metriche con un'analisi sull'efficacia e l'efficienza del Programma di Sicurezza delle Applicazioni.
<b>Livello di maturità 2:</b> Stabilire una roadmap strategica unificata per la sicurezza del software all'interno dell'organizzazione.	
Stabilire una strategia unificata per la sicurezza delle applicazioni.	Stabilire obiettivi e KPI per misurare l'efficacia del programma.
<b>Livello di maturità 3:</b> Allineare gli sforzi di sicurezza con gli indicatori organizzativi rilevanti e i valori degli asset.	
Allineare il programma di sicurezza delle applicazioni per supportare la crescita dell'organizzazione.	Influenzare la strategia basandosi sulle metriche e sulle esigenze organizzative.

## Design

Riguarda i processi e le attività relativi a come un'organizzazione definisce gli obiettivi e crea software all'interno dei progetti di sviluppo.

Le "Practices" sono:

- Valutazione delle minacce/Threat assessment: concentrazione sull'identificazione delle potenziali minacce nelle applicazioni.
- Stream A: Application Risk Profile, identifica quali applicazioni possono rappresentare una minaccia per l'organizzazione se venissero attaccate o violante.
- Stream B: Threat Modeling, supporto al team di sviluppo software al fine di capire quali rischi sussistano in ciò che sta venendo sviluppato, cosa potrebbe andare storto e come i rischi possano essere mitigati o risolti.

Stream A	Stream B
<b>Livello di maturità 1:</b> Identificazione a titolo di impegno migliore delle minacce di alto livello per l'organizzazione e i singoli progetti. Viene eseguita una valutazione di base del rischio dell'applicazione per comprendere la probabilità e l'impatto di un attacco.	Effettuare un modello delle minacce basato sul rischio con il miglior impegno possibile, utilizzando il brainstorming e i diagrammi esistenti con semplici checklist di minacce.
<b>Livello di maturità 2:</b> Standardizzazione e analisi a livello aziendale delle minacce legate al software all'interno dell'organizzazione. Comprendere il rischio per tutte le applicazioni nell'organizzazione centralizzando l'inventario del profilo di rischio per gli interessati.	Standardizzare la formazione sulla modellazione delle minacce, i processi e gli strumenti per scalare su tutta l'organizzazione.
<b>Livello di maturità 3:</b> Miglioramento proattivo della copertura delle minacce in tutta l'organizzazione. Rivedere periodicamente i profili di rischio delle applicazioni a intervalli regolari per garantire precisione e riflettere lo stato attuale.	Ottimizzazione continua e automazione della tua metodologia di modellazione delle minacce.

- Requisiti di sicurezza: concentrazione sulla definizione di requisiti di sicurezza appropriati per il software e per i fornitori di software.
- Stream A: Requisiti Software, specificano gli obiettivi e le aspettative per proteggere il servizio e i dati al centro dell'applicazione.
- Stream B: Sicurezza del fornitore, riguarda i requisiti relativi alle organizzazioni fornitrice all'interno del contesto di sviluppo dell'applicazione.

Stream A	Stream B
<b>Livello di maturità 1:</b> Considerare esplicitamente la sicurezza durante il processo di definizione dei requisiti software. Gli obiettivi di sicurezza dell'applicazione a alto livello vengono mappati sui requisiti funzionali.	Valutare il fornitore in base ai requisiti di sicurezza dell'organizzazione.
<b>Livello di maturità 2:</b> Aumentare la granularità dei requisiti di sicurezza derivati dalla logica aziendale e dai rischi conosciuti. Sono disponibili requisiti di sicurezza strutturati e utilizzati dai team di sviluppo.	Includere la sicurezza negli accordi con i fornitori al fine di garantire la conformità ai requisiti dell'organizzazione.
<b>Livello di maturità 3:</b> Imporre un processo di requisiti di sicurezza per tutti i progetti software e le dipendenze di terze parti. Costruire un framework di requisiti per consentire ai team di prodotto di utilizzarlo.	Garantire una corretta copertura della sicurezza per i fornitori esterni fornendo obiettivi chiari.

- Architettura della sicurezza: concentrazione sulla gestione dei rischi architettonici per la soluzione software.
- Stream A: Progettazione dell'Architettura, un buon design influenza significativamente la sicurezza
- Stream B: Gestione della Tecnologia, comprende i framework e altre tecnologie usate che sono il pilastro di qualsiasi soluzione software e vanno

esaminati per garantire sicurezza

Stream A	Stream B
<b>Livello di maturità 1:</b> Inserire la considerazione di orientamenti proattivi sulla sicurezza nel processo di progettazione del software.	
I team vengono formati sull'uso dei principi di base della sicurezza durante la fase di progettazione.	Raccogliere informazioni sulle tecnologie, i framework e le integrazioni all'interno della soluzione complessiva per identificare i rischi.
<b>Livello di maturità 2:</b> Direct the software design process toward known secure services and secure-by-default designs.	
Stabilire pattern di progettazione comuni e soluzioni di sicurezza per l'adozione.	Standardizzare le tecnologie e i framework da utilizzare in tutte le diverse applicazioni.
<b>Livello di maturità 3:</b> Controllare formalmente il processo di progettazione del software e convalidare l'utilizzo di componenti sicuri.	
Le architetture di riferimento vengono utilizzate e valutate continuamente per l'adozione e l'adeguatezza.	Imporre l'uso di tecnologie standard su tutti gli sviluppi software.

## Implementation

Focalizzata sul modo in cui un'organizzazione costruisce e distribuisce i componenti software e i relativi difetti.

Le attività qui dentro impattano gli sviluppatori nella vita quotidiana.

Le "Practices" sono:

- Secure Build: creazione di un processo di compilazione ripetibile e che tiene conto della sicurezza delle dipendenze dell'applicazione.
- Stream A: Processo di Compilazione, se coerente garantisce che il software che stai distribuendo sia prevedibile e direttamente collegato al codice sorgente.
- Stream B: Dipendenze Software, le attività in questo filone aiutano a creare una visione delle librerie esterne e assicurano che la loro robustezza sia adeguata dal punto di vista della sicurezza.

Stream A	Stream B
<b>Livello di maturità 1:</b> Il processo di compilazione è ripetibile e consistente.	
Creare una definizione formale del processo di compilazione in modo che diventi consistente e ripetibile.	Creare registri con la lista dei materiali delle tue applicazioni e analizzarli in modo opportunistico.
<b>Livello di maturità 2:</b> Il processo di compilazione è ottimizzato e completamente integrato nel flusso di sviluppo.	
Automatizzare la tua pipeline di compilazione e garantire la sicurezza degli strumenti utilizzati. Aggiungere controlli di sicurezza nella pipeline di compilazione.	Valutare le dipendenze utilizzate e garantire una reazione tempestiva alle situazioni che rappresentano un rischio per le tue applicazioni.
<b>Livello di maturità 3:</b> Il processo di compilazione aiuta a prevenire che difetti conosciuti entrino nell'ambiente di produzione.	
Definire controlli di sicurezza obbligatori nel processo di compilazione e garantire che la creazione di artefatti non conformi fallisca.	Analizzare le dipendenze utilizzate per individuare problemi di sicurezza in modo comparabile al proprio codice.

- Secure Deployment: si aumenta la sicurezza e l'integrità delle applicazioni sviluppate e delle distribuzioni software.
- Stream A: Processo di Distribuzione, rimozione degli errori automatizzando il processo di distribuzione il più possibile e condizionando il successo ai risultati dei controlli integrati di verifica della sicurezza; favorisce la separazione dei compiti rendendo responsabili del rilascio persone adeguatamente formate e non sviluppatori.
- Stream B: Gestione dei Segreti, protezione della privacy e dell'integrità dei dati sensibili necessari per il funzionamento delle applicazioni negli ambienti

Stream A	Stream B
<b>Livello di maturità 1:</b> I processi di distribuzione sono completamente documentati.	
Formalizzare il processo di distribuzione e proteggere gli strumenti e i processi utilizzati.	Introdurre misure di protezione di base per limitare l'accesso ai segreti di produzione.
<b>Livello di maturità 2:</b> I processi di distribuzione includono tappe di verifica della sicurezza.	
Automatizzare il processo di distribuzione su tutte le fasi e introdurre test di verifica della sicurezza sensati.	Fare injection dei segreti dinamicamente durante il processo di distribuzione da archivi sicuri e monitorare tutti gli accessi umani ad essi.
<b>Livello di maturità 3:</b> Il processo di distribuzione è completamente automatizzato e incorpora la verifica automatica di tutte le tappe critiche.	
Verificare automaticamente l'integrità di tutto il software distribuito, indipendentemente dal fatto che sia di produzione, sviluppato internamente o esternamente.	Migliorare il ciclo di vita dei segreti delle applicazioni generandoli regolarmente e garantendo un uso adeguato.

- Defect Management: concentrazione sulla gestione dei difetti di sicurezza nel software e sulle metriche associate.

- Stream A: Tracciamento delle vulnerabilità, gestione della raccolta e il follow-up di tutti i potenziali problemi in un pezzo di software.
- Stream B: Metriche e Feedback, traccia i difetti per guidare il miglioramento delle attività di sicurezza all'interno dell'organizzazione tramite feedback.

Stream A	Stream B
<b>Livello di maturità 1:</b> Tutte le vulnerabilità sono tracciate all'interno di ciascun progetto.	
Introdurre un tracciamento strutturato dei difetti di sicurezza e prendere decisioni informate basate su queste informazioni.	Rivedere regolarmente i difetti di sicurezza precedentemente registrati e trarre vantaggio da successi rapidi basati su metriche di base.
<b>Livello di maturità 2:</b> Il tracciamento delle vulnerabilità viene utilizzato per influenzare il processo di distribuzione.	
Valutare tutti i difetti di sicurezza su tutta l'organizzazione in modo coerente e definire SLA per specifiche classi di gravità.	Raccogliere metriche standardizzate di gestione dei difetti e utilizzarle anche per la prioritizzazione delle iniziative guidate centralmente.
<b>Livello di maturità 3:</b> Il tracciamento dei difetti attraverso più componenti viene utilizzato per aiutare a ridurre il numero di nuovi difetti.	
Imporre gli SLA predefiniti e integrare il sistema di gestione dei difetti con altri strumenti rilevanti.	Migliorare continuamente le metriche di gestione dei difetti di sicurezza e correlarle con altre fonti.

## Verification

Si concentra sul controllo e verifica degli artefatti prodotti durante lo sviluppo del software; include test, attività di revisione e valutazione.

Le "Practices" sono:

- Valutazione dell'Architettura: convalida della sicurezza e della conformità dell'architettura del software e dell'infrastruttura di supporto.
- Stream A: Validazione dell'Architettura, verifica la sicurezza del software e dell'architettura di supporto, verificando che i componenti dell'architettura dell'applicazione e dell'infrastruttura soddisfino gli obiettivi e dei requisiti di sicurezza.
- Stream B: Mitigazione dell'Architettura, garantisce che tutte le minacce identificate durante la Valutazione delle Minacce siano adeguatamente mitigate.

Stream A	Stream B
<b>Livello di maturità 1:</b> Revisionare l'architettura per assicurarsi che le mitigazioni di base siano in atto per i rischi tipici.  Identificare i componenti dell'architettura dell'applicazione e dell'infrastruttura e rivedere il loro soddisfacimento dei requisiti di sicurezza di base.	Revisione ad hoc dell'architettura per identificare minacce di sicurezza non mitigate.
<b>Livello di maturità 2:</b> Rivedere la completa fornitura di meccanismi di sicurezza nell'architettura.  Validare i meccanismi di sicurezza dell'architettura.	Analizzare l'architettura rispetto alle minacce conosciute.
<b>Livello di maturità 3:</b> Rivedere l'efficacia dell'architettura e fornire feedback per migliorare l'architettura della sicurezza.  Revisione dell'efficacia dei componenti dell'architettura.	Incorporare i risultati della revisione dell'architettura nell'architettura aziendale, nei principi e nei modelli di progettazione dell'organizzazione, nelle soluzioni di sicurezza e nelle architetture di riferimento.

- Testing guidato dai Requisiti: utilizzo di test di sicurezza positivi (verifica dei controlli) e negativi (test di abuso) basati sui requisiti (storie degli utenti).
- Stream A: Verifica dei Controlli, convalida che i controlli di sicurezza e i requisiti siano soddisfatti attraverso test
- Stream B: Test di Misuse/Abuse,sfrutta il fuzzing, casi di uso improprio/abuso e l'identificazione di qualsiasi funzionalità o risorsa nel software che può essere abusata per individuare debolezze nelle funzionalità da attaccare in un'applicazione.

Stream A	Stream B
<b>Livello di maturità 1:</b> Identificare opportunisticamente vulnerabilità di base e altri problemi di sicurezza.  Effettuare test per i controlli di sicurezza del software.	Effettuare test di fuzzing per la sicurezza.
<b>Livello di maturità 2:</b> Effettuare una revisione dell'implementazione per scoprire rischi specifici dell'applicazione rispetto ai requisiti di sicurezza.  Derivare casi di test dai requisiti di sicurezza conosciuti.	Creare e testare casi di abuso e test di difetti dovuti alla logica di business.
<b>Livello di maturità 3:</b> Mantenere il livello di sicurezza dell'applicazione dopo correzioni di bug, modifiche o durante la manutenzione.  Eseguire il testing di regressione (con test di unità di sicurezza).	Denial of service test

- Testing di Sicurezza: rilevazione e risoluzione di problemi base di sicurezza attraverso l'automazione, consentendo ai test manuali di concentrarsi su vettori di attacco più complessi in modo da scoprire vulnerabilità tecniche e nella logica aziendale.
- Stream A: Baseline Scalabile, uso di strumenti di test automatizzati specifici

- dell'applicazione che integrano la validazione della sicurezza nel processo di compilazione e distribuzione; si favorisce la larghezza
- Stream B: Comprensione Approfondita, esecuzione di test di sicurezza manuali e complessi su componenti ad alto rischio; favorisce la profondità.

Stream A – Create & Promote	Stream B – Measure & Improve
<b>Livello di maturità 1:</b> identificare gli obiettivi e i mezzi per misurare l'efficacia del programma di sicurezza.	
Identificare i fattori trainanti dell'organizzazione in relazione alla sua tolleranza al rischio.	Definire metriche con un'analisi sull'efficacia e l'efficienza del Programma di Sicurezza delle Applicazioni.
<b>Livello di maturità 2:</b> Stabilire una roadmap strategica unificata per la sicurezza del software all'interno dell'organizzazione.	
Stabilire una strategia unificata per la sicurezza delle applicazioni.	Stabilire obiettivi e KPI per misurare l'efficacia del programma.
<b>Livello di maturità 3:</b> Allineare gli sforzi di sicurezza con gli indicatori organizzativi rilevanti e i valori degli asset.	
Allineare il programma di sicurezza delle applicazioni per supportare la crescita dell'organizzazione.	Influenzare la strategia basandosi sulle metriche e sulle esigenze organizzative.

**VAPT** Vulnerability Assessment and Penetration Testing, è un processo completo che identifica le potenziali vulnerabilità e che valuta la capacità del sistema di resistere agli attacchi informatici, fornendo così un quadro dettagliato e approfondito dello stato di sicurezza del sistema stesso

Usa due approcci distinti:

- Vulnerability Assessment: dedicato all'identificazione, alla quantificazione e alla classificazione delle vulnerabilità presenti nel sistema.
- Penetration Testing: "ethical hackers", tentano di sfruttare le vulnerabilità identificate per penetrare nel sistema

## Operations

Comprende attività necessarie per garantire che la riservatezza, l'integrità e la disponibilità siano mantenute per tutta la durata operativa di un'applicazione e dei dati associati ad essa.

Le "Practices" sono:

- Incident management: attività svolte per migliorare la capacità dell'organizzazione di individuare e rispondere agli incidenti di sicurezza.
- Stream A: Rilevamento degli Incidenti, processo di determinare se un evento rilevante per la sicurezza identificato è effettivamente un incidente di sicurezza.

- Stream B: Risposta agli Incidenti, si agisce nel momento in cui si riconosce e si verifica l'esistenza di un incidente di sicurezza.

Stream A	Stream B
<b>Livello di maturità 1:</b> Rilevamento e gestione degli incidenti con il massimo impegno.	
Utilizzare i dati di log disponibili per effettuare il rilevamento con il massimo impegno di possibili incidenti di sicurezza.	Identificare ruoli e responsabilità per la risposta agli incidenti.
<b>Livello di maturità 2:</b> Processo formale di gestione degli incidenti istituito.	
Seguire un processo stabilito e ben documentato per il rilevamento degli incidenti, con enfasi sull'valutazione automatizzata dei log.	Stabilire un processo formale di risposta agli incidenti e garantire che il personale sia adeguatamente formato per svolgere i propri ruoli.
<b>Livello di maturità 3:</b> Gestione degli incidenti matura.	
Utilizzare un processo gestito in modo proattivo per il rilevamento degli incidenti.	Impiegare un team dedicato e ben addestrato per la risposta agli incidenti.

- Environment management: descrive le attività proattive svolte per migliorare e mantenere la sicurezza degli ambienti in cui operano le applicazioni dell'organizzazione.
- Stream A: Hardening delle Configurazioni, gestione da parte dell'organizzazione delle configurazioni legate alla sicurezza in tutti gli elementi dello stack tecnologico; con l'accento posto sugli elementi di terze parti.
- Stream B: Patching & Aggiornamenti, gestione da parte dell'organizzazione dei patch e degli aggiornamenti per tutti gli elementi dello stack tecnologico.

Stream A	Stream B
<b>Livello di maturità 1: patch e hardening a best-effort</b>	
Effettuare hardening delle configurazioni con il massimo impegno, basato sulle informazioni prontamente disponibili.	Effettuare il patching dei componenti di sistema e delle applicazioni con il massimo impegno disponibile.
<b>Livello di maturità 2: Processo formale con basi stabilite.</b>	
Effettuare un hardening coerente delle configurazioni, seguendo le linee guida e gli standard stabiliti.	Effettuare regolarmente il patching dei componenti di sistema e delle applicazioni, su tutto lo stack. Garantire la consegna tempestiva dei patch ai clienti.
<b>Livello di maturità 3: Conformità con un processo in continuo miglioramento applicato con rigore.</b>	
Monitorare attivamente le configurazioni per individuare eventuali non conformità alle basi stabilite e gestire le occorrenze rilevate come difetti di sicurezza.	Monitorare attivamente lo stato degli aggiornamenti e gestire i patch mancanti come difetti di sicurezza. Ottenere proattivamente informazioni sulle vulnerabilità e sugli aggiornamenti per i componenti.

- Operational management: si concentra sulle attività di supporto operativo

necessarie per mantenere la sicurezza durante tutto il ciclo di vita del prodotto.

- Stream A: Protezione dei Dati, garantire che l'organizzazione protegga adeguatamente i dati in tutti gli aspetti della loro creazione, gestione, archiviazione e elaborazione.
- Stream B: Gestione delle Legacy, identificazione, gestione e tracciamento di sistemi, applicazioni, dipendenze delle applicazioni e servizi che non sono più utilizzati, la successiva rimozione migliora la gestibilità dell'ambiente e riduce la superficie di attacco dell'organizzazione, consentendo risparmi diretti e indiretti.

Stream A	Stream B
<b>Livello di maturità 1:</b> Pratiche Fondamentali.	
Implementare pratiche di protezione dei dati di base.	Decommissionare le applicazioni e i servizi non utilizzati come identificato. Gestire gli aggiornamenti/migrazioni dei clienti individualmente.
<b>Livello di maturità 2:</b> Processi Gestiti e Reattivi	
Sviluppare un catalogo dei dati e stabilire una politica di protezione dei dati.	Sviluppare processi di dismissione ripetibili per sistemi/servizi non utilizzati e per la migrazione dalle dipendenze legacy. Gestire le roadmap di migrazione legacy per i clienti.
<b>Livello di maturità 3:</b> Monitoraggio Attivo e Risposta repentina	
Automatizzare il rilevamento della non conformità alle politiche e verificare periodicamente la conformità. Rivedere e aggiornare regolarmente il catalogo dei dati e la politica di protezione dei dati.	Gestire proattivamente le roadmap di migrazione, sia per le dipendenze non supportate in fase di fine vita, sia per le versioni legacy del software fornito.

## Sistemi di autenticazione e controllo degli accessi

Il loro scopo è quello di garantire la sicurezza delle informazioni sensibili, mantenendo l'integrità dei sistemi informatici.

### Sicurezza password

Le password devono seguire certe regole per evitare che vengano ottenute tramite attacchi brute force, come lunghezza, caps, non deve contenere dati personali o certi caratteri.

Password	Lunghezza	Numeri	Caratteri speciali	Maiuscola
marco.rossi2024?	✓	✓	✓	✗
iosonounapassword	✓	✗	✗	✗
Marc20!	✗	✓	✓	✓
Ros2024mar	✓	✓	✗	✓
hGt738!HKos45@"	✓	✓	✓	✓

### Password salt

Modo per salvare correttamente le password nei database evitando il problema degli hash uguali se due utenti usano la stessa password.

Si aggiunge il salt, sequenza di dati casuali e univoci per ogni password, prima di eseguire l'hashing della password in chiaro.

### Sistemi a sfida

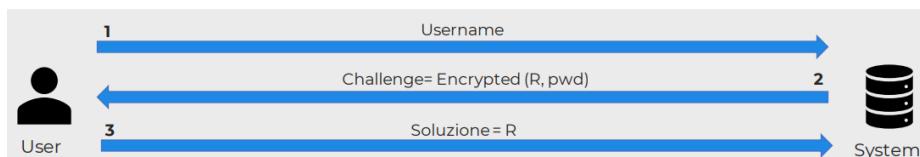
Metodo di autenticazione che protegge le password durante la trasmissione sulla rete.

All'invio di una richiesta l'utente ottiene una sfida crittografica da decifrare con la propria password, poi risponde al sistema con la soluzione della sfida, senza mai inviare la password in rete.

Si garantisce maggior sicurezza (evitando intercettazioni) e riservatezza, a scapito di una complessità maggiore e possibili vulnerabilità se la sfida non è implementata correttamente.

### Sfida simmetrica

1. [UTENTE] Richiesta di accesso.
2. [SISTEMA] Generazione e invio sfida.
3. [UTENTE] Decifrazione con password e invio soluzione.
4. [SISTEMA] Verifica soluzione



### Sfida asimmetrica

1. [UTENTE] Richiesta di accesso tramite certificato.
2. [SISTEMA] Generazione sfida tramite chiave pubblica utente e invio.
3. [UTENTE] Decifratura con chiave privata e invio soluzione.
4. [SISTEMA] Verifica soluzione



## OTP

Una One-Time Password (OTP) è una sequenza di caratteri, numeri o simboli utilizzata per l'autenticazione o per accedere a risorse protette su Internet una sola volta per sessione/transazione.

Generata tramite:

- Sincronizzazione temporale: si parla di Time-Based One-Time Password (TOTP) creati con algoritmi basati sul tempo e usati dei 2FA.

Sono molto sicuri (chiavi valide solo per piccoli intervalli) e facili da implementare ma necessitano che entrambi gli attori abbiano un orologio sincronizzato.

- Calcoli matematici: uno è l'HMAC-based One-Time Password (HOTP).

Funziona in questo modo:

- l'utente e il sistema condividono una chiave da usare per l'HMAC;
- l'utente e il sistema condividono un contatore per convalidare/generare l'OTP e incrementato dopo ogni transazione;
- l'utente usa HMAC e la chiave per generare OTP;
- l'utente invia OTP e il server lo controlla con quello calcolato da lui.
- si aggiorna il contatore.

Usando questo metodo si supera la dipendenza del tempo ed è perfetto in ambienti offline ma si ha una complessità non differente ed è vulnerabile ad attacchi di riproduzione.

- Challenge: coinvolgono l'invio di una sfida unica dal validatore al generatore OTP, che usa la sfida e altri dati per generare l'OTP; aumentando la sicurezza dell'autenticazione (l'OTP si genera solo in risposta ad una specifica sfida).

I passi sono:

1. Il validatore invia una sfida univoca al generatore dell'OTP.
2. Il generatore dell'OTP utilizza la sfida ricevuta (e dati) per generare l'OTP e lo invia al validatore.

- Il validatore confronta l'OTP ricevuto con il valore atteso, che è derivato dalla stessa sfida inviata al generatore

### Token keys

**Hardware** Dispositivi fisici che generano e forniscono One-Time Password (OTP) per l'autenticazione, di solito sono USB, smart card o portachiavi.

Estremamente portabili ed evitano accessi non autorizzati, visto che l'OTP è generato e memorizzato direttamente sul dispositivo.

Soggetti a smarrimento o furto e richiedono una distribuzione fisica ai singoli utenti.

**Software** Applicazioni software che generano OTP per l'autenticazione, e installate su dispositivi digitali.

Offrono la convenienza di essere installati su dispositivi già in possesso dell'utente e supportano una varietà di metodi di generazione di OTP, come TOTP (Time-Based One-Time Password) o HOTP (HMAC-Based One-Time Password).

Vulnerabili a minacce informatiche

## AUTH OUT-OF-BAND (OOBA)

Autenticazione che coinvolge la comunicazione di informazioni sensibili tramite canali separati o "fuori banda" rispetto al canale principale utilizzato per l'autenticazione.

Le informazioni sensibili vengono trasmesse tramite un canale diverso rispetto a quello principale utilizzato per l'interazione utente-sistema.

Esempio: con 2FA un utente potrebbe ricevere un codice OTP tramite SMS o tramite un'applicazione di autenticazione separata per completare il processo di login.

OOBA molto spesso è un requisito normativo e protegge da attacchi di phishing, ma la sua enorme complessità nell'implementazione porta a costi aggiuntivi e possibili ritardi.

## Single Sign-on (SSO)

Autenticazione che utilizza un'unica procedura di autenticazione, semplificando l'accesso a molti servizi che usano la stessa autenticazione.

Ne esistono di 3 tipi:

- SSO Federato: permette agli utenti di accedere a diverse applicazioni e servizi di varie organizzazioni utilizzando un'unica autenticazione.

<b>Pro:</b> <ul style="list-style-type: none"> <li>○ Interoperabilità tra organizzazioni</li> <li>○ Standard</li> <li>○ Esperienza utente migliorata</li> <li>○ Controllo e privacy</li> <li>○ Riduzione dei costi</li> </ul> <b>Contro:</b> <ul style="list-style-type: none"> <li>○ Complessità tecnica</li> <li>○ Possibili problemi di latenza</li> <li>○ Gestione delle federazioni</li> </ul>	<b>Protocolli utilizzati:</b> <ul style="list-style-type: none"> <li>○ SAML</li> <li>○ OpenID Connect</li> <li>○ OAuth 2.0</li> </ul>
--	---

2. SSO Centralizzato: un unico sistema di identità controlla l'accesso a multiple applicazioni e servizi all'interno di un'organizzazione.

<b>Pro:</b> <ul style="list-style-type: none"> <li>○ Accesso semplificato</li> <li>○ Gestione centralizzata</li> <li>○ Sicurezza e Compliance</li> <li>○ Costi ridotti</li> </ul> <b>Contro:</b> <ul style="list-style-type: none"> <li>○ Single point of failure</li> <li>○ Scalabilità</li> <li>○ Rischi di sicurezza centralizzati</li> <li>○ Limitazioni in interoperabilità con sistemi esterni</li> </ul>	<b>Protocolli utilizzati:</b> <ul style="list-style-type: none"> <li>○ SAML</li> <li>○ OAuth 2.0</li> </ul>
--	---

3. SSO Cooperativo: condivide un meccanismo di autenticazione tra applicazioni correlate, senza protocolli federativi. Adatto per ambienti in cui tutte le applicazioni sono gestite dalla stessa organizzazione o entità.

<b>Pro:</b> <ul style="list-style-type: none"> <li>○ Integrazione semplificata</li> <li>○ Meno dipendenze esterne</li> <li>○ Sicurezza e Compliance</li> <li>○ Costi ridotti</li> </ul> <b>Contro:</b> <ul style="list-style-type: none"> <li>○ Limitata a un unico ecosistema</li> <li>○ Difficoltà nella scalabilità con nuove app</li> </ul>	<b>Protocolli utilizzati:</b> <ul style="list-style-type: none"> <li>○ LDAP</li> <li>○ Kerberos</li> <li>○ Protocolli Proprietari</li> </ul>
--	--

## Identity e Service Provider

Identity Provider (IdP) consente agli utenti di autenticarsi e dimostrare la propria identità in modo sicuro e affidabile.

Responsabile della creazione, archiviazione, manutenzione e gestione delle identità digitali degli utenti e della gestione delle loro credenziali (password, politiche di sicurezza).

Tre categorie:

1. Social Media Identity Provider: come Facebook, Google e LinkedIn che permettono agli utenti di utilizzare le proprie credenziali di accesso per accedere ad altre applicazioni e siti web.
2. Governative Identity Provider: gestiti dal governo/agenzie governative che forniscono servizi di autenticazione e gestione delle identità digitali per i cittadini all'interno di una specifica giurisdizione.
3. Enterprise Identity Provider: utilizzati dalle organizzazioni e aziende per gestire l'accesso degli utenti alle proprie risorse digitali, sia interne che esterne.

Service Provider (SP), offre servizi (come applicazioni web, servizi cloud, risorse IT) interni di un'azienda agli utenti autenticati tramite IdP.

## SAML

Security Assertion Markup Language, protocollo open standard per lo scambio di dati di autenticazione e autorizzazione tra un Identity Provider (IdP) e un Service Provider (SP).

Si utilizza per implementare:

- Single Logout (SLO): permette agli utenti di disconnettersi da tutti i servizi contemporaneamente.
- SSO: utilizzo di un unico set di credenziali, gestite da un IdP, per accedere a diversi servizi.
- Federazione delle identità: consente a più organizzazioni di condividere identità in modo sicuro.

Estremamente sicuro, interoperabile ed efficiente per l'utente ma anche complesso, dipendente da XML e lento (gestione di firme e XML).

## Fasi

1. Richiesta di Autenticazione:
  1. un utente vuole accedere ad un servizio del SP;
  2. l'SP verifica per sessioni attive, se non è autenticato crea una AuthnRequest;
  3. l'utente viene reindirizzato dall'IdP appropriato con l'AuthnRequest.
2. Reindirizzamento all'IdP:
  1. l'AuthnRequest è inviata all'IdP (con GET o POST);
  2. si firma la richiesta per sicurezza;
3. Autenticazione presso IdP
  1. l'IdP riceve la AuthnRequest e estrae le informazioni necessarie;
  2. l'utente inserisce le sue credenziali (username+psw, OTP, ecc);

3. l'IdP crea o recupera una sessione esistente per l'utente, registrando l'evento di autenticazione.
4. Generazione Asserzione SAML (documento XML inviato tram IdP e SP)
  1. l'IdP prepara una risposta SAML (Response) che include un'asserzione (Assertion), l'asserzione può contenere:
    - Statements (uno o più) relative all'identità del user.
    - Attributi.
    - Condizioni di validità.
5. Invio Asserzione al SP
  1. l'IdP invia la Response al SP, precedentemente firmata digitalmente.
6. Verifica Asserzione e Autorizzazione
  1. ricevuta la Response l'SP decodifica i dati e verifica la firma;
  2. si controllano le condizioni di validità (audience e temporalità);
  3. si estraggono le informazioni per determinare i privilegi di accesso;
  4. si concede l'accesso.

## **OAuth 2.0**

Protocollo di autenticazione che consente alle applicazioni di ottenere accesso limitato a servizi online per conto di un utente.

Basato sull'emissione di token da parte di un server di autorizzazione.

Componenti:

- Resource Owner: l'utente con i dati che l'app vuole.
- Client: l'applicazione che vuole i dati.
- Authorization Server: colui che emette i token client dopo il consenso del proprietario.
- Resource Server: colui che ha le risorse dell'utente e accetta i token

Il protocollo permette una delega sicura e flessibile controllando gli accessi (revocabili sempre) ma è estremamente complesso implementarlo e ogni implementazione cambia.

## **Fasi**

1. Richiesta di Autorizzazione: l'applicazione che richiede l'accesso (Client) alle risorse dell'utente, invia una richiesta di autorizzazione al Resource Owner, direttamente o tramite Authorization Server.
2. Autenticazione dell'Utente e Concessione dei Permessi: dopo l'autorizzazione del Resource Owner il Client riceve una concessione di autorizzazione dall'Authorization Server che prova che l'utente ha concesso l'accesso.

Le concessioni possono essere:

- Authorization Code: codice temporaneo.
  - Implicit: token fornito senza uno step intermedio.
  - Password: username e password del Resource Owner.
  - Client Credentials: per le autorizzazioni server-to-server dove il client può autenticarsi direttamente.
3. Richiesta del token di accesso: il client si autentica presso l'Authorization Server, dopo la conferma presenta la concessione per ottenere un token per accedere alle risorse.
  4. Emissione Token: dopo la presentazione del token l'Authentication Server controlla le credenziali del Client e la sua concessione. Se tutto è regolare viene emesso il token di accesso.
  5. Accesso alle risorse: ora il Client può accedere alle risorse, per farlo invia una richiesta HTTPS con il token al Resource Server.
  6. Convalida Token: il Resource Server verifica la firma, la scadenza del token e che autorizzazioni ha il token.

## OpenID Connect

Protocollo basato su OAuth 2.0 che permette di verificare l'identità di un utente basandosi sull'autenticazione eseguita da un Identity Provider (IdP).

OpenID Connect estende e modifica il flusso di OAuth 2.0 aggiungendo autenticazione esplicita dell'utente tramite l'ID token e Claims (coppie "nome-valore" che forniscono dati di attributo sull'utente che è stato autenticato).

il protocollo permette un meccanismo standardizzato con una facile integrazione con altri servizi e con scambio di informazioni dettagliate, ideale anche per i SSO; ma la sua complessità tecnica e l'overhead di prestazioni lo rendono molto complicato.

### Fasi

1. Richiesta di Autorizzazione: l'applicazione che richiede l'accesso (Client) alle risorse dell'utente, invia una richiesta di autorizzazione al Resource Owner, direttamente o tramite Authorization Server, contenente parametri specifici come il parametro scope contenente il valore openid.
2. Ricezione della Concessione di Autorizzazione: il Client ottiene una concessione dal Authentication Server rappresentata da un Authentication Code, elemento temporaneo utilizzato per chiedere un ID token e token di accesso.
3. Richiesta del Token di Accesso: Il Client usa l'ID client e la chiave segreta, forniti dal Authentication Server alla registrazione, per autenticarsi a quest'ultimo

4. Emissione del Token di Accesso:
5. Accesso alle risorse:
6. Convalida del Token di Accesso:

## LDAP

Protocollo di rete utilizzato per accedere e gestire servizi di directory. Leggero e tipicamente utilizzato per gestire utenti e gruppi e per autenticare gli utenti in un dominio.

Il protocollo è pensato per organizzare i dati in maniera gerarchica tramite una Directory Information Tree (DIT) ed essere ottimizzato per la lettura rispetto che per le modifiche.

L'autenticazione è fatta usando il bind, che lega gli utenti al server con le loro credenziali.

LDAP ha i seguenti elementi:

- Directory System Agent (DSA): un server di directory che esegue LDAP nella propria rete.
- Directory User Agent (DUA): accede ai server DSA come client
- Distinguished Name (DN): contiene il percorso alla struttura della directory, utilizzato da LDAP per raggiungere l'informazione.
- Relative Distinguished Name (RDN): il nome distinto relativo, ovvero ogni componente del percorso del DN.
- Application Programming Interface (API): l'interfaccia di programmazione delle applicazioni che consente ai servizi di comunicare con altri prodotti o servizi anche senza sapere come sono stati adottati.

Pro:

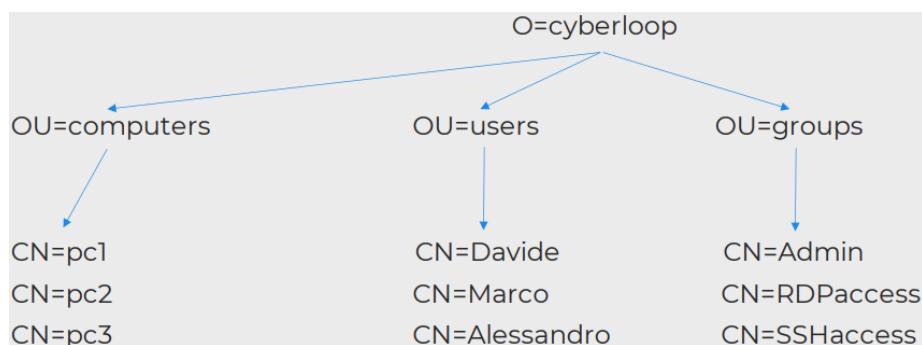
- Struttura Gerarchica: Utilizza una struttura a albero (DIT) per un'organizzazione logica e facile navigazione dei dati.
- Standardizzazione: È uno standard aperto e ampiamente utilizzato, con numerose implementazioni e documentazione disponibile.
- Scalabilità: È progettato per essere scalabile, consentendo l'aggiunta di nuovi utenti e risorse senza compromettere le prestazioni.
- Sicurezza: Offre funzionalità di sicurezza come autenticazione, autorizzazione e crittografia delle comunicazioni.
- Integrazione: Può essere integrato con una vasta gamma di applicazioni e servizi.

Contro:

- Possibile Single Point of Failure: Un'interruzione del server LDAP può causare problemi a tutti i servizi che dipendono da esso.
- Rischio di Consistenza dei Dati: Esiste il rischio di inconsistenza dei dati se le modifiche non vengono sincronizzate correttamente tra i server.

## Componenti

- Domain Access Component (dc): permette agli utenti di accedere rapidamente ai servizi e alle risorse all'interno della directory LDAP utilizzando nomi di dominio familiari.
- Organization Name (o): sottoclasse all'interno del Distinguished Name (DN), serve come punto di partenza per le ricerche all'interno della directory LDAP.
- Organizational Unit (ou): sottoclasse di O, contiene i Common name.
- Common name (cn): utilizzato per identificare il nome di un gruppo o di un account utente personale



## RADIUS

Remote Authentication Dial-In User Service, protocollo di rete utilizzato per l'autenticazione, l'autorizzazione e l'accounting degli utenti che si connettono e accedono a una rete. Viene utilizzato per gestire accessi remoti e VPN, centralizzando la gestione delle credenziali e le policy di sicurezza.

Pro:

- Centralizzazione dell'Autenticazione: RADIUS consente di centralizzare l'autenticazione degli utenti, semplificando la gestione delle credenziali e garantendo un controllo più stringente sull'accesso alla rete.
- Scalabilità: È in grado di gestire grandi volumi di autenticazioni e connessioni contemporaneamente, rendendolo adatto per reti di grandi dimensioni.

- Sicurezza: Offre opzioni per la crittografia dei dati sensibili durante la trasmissione, migliorando la sicurezza delle comunicazioni tra client e server RADIUS.
- Logging e Reporting: Supporta il tracciamento e il logging delle attività degli utenti attraverso il processo di accounting, fornendo informazioni dettagliate sull'utilizzo della rete.

Contro:

- Complessità di Configurazione: La configurazione e l'implementazione di un server RADIUS possono essere complesse e richiedere competenze tecniche avanzate.
- Single Point of Failure: Un server RADIUS centralizzato rappresenta un single point of failure per l'intera rete, il che significa che un malfunzionamento del server potrebbe impedire l'accesso degli utenti alla rete.
- Limitazioni della Banda Larga: Il protocollo RADIUS può causare congestione di rete e ritardi nelle autenticazioni in caso di traffico elevato o di una larghezza di banda limitata.

Fasi

1. Identificazione: l'utente fornisce un'identificazione univoca, per iniziare il processo di autenticazione.
2. Autenticazione delle Credenziali: l'utente fornisce le proprie credenziali di accesso, per dimostrare di essere l'utente legittimo associato all'identificazione fornita.
3. Verifica delle Credenziali: le credenziali fornite vengono confrontate con quelle memorizzate nel sistema per verificare la loro correttezza e autenticità. Questo processo può coinvolgere la crittografia, l'hashing o altri metodi per proteggere le credenziali durante la trasmissione e il confronto.
4. Autorizzazione: dopo l'autenticazione si può essere autorizzato ad accedere alle risorse o ai servizi richiesti, controllando le regole di sicurezza e policy.
5. Registrazione dell'Accesso: viene registrato l'accesso dell'utente, memorizzando informazioni come l'orario e la data dell'accesso, l'indirizzo IP del dispositivo utilizzato, i dettagli dell'operazione eseguita e altre informazioni pertinenti per la sicurezza e l'auditabilità del sistema.

RADIUS consiglia, ma non obbliga, altre tipologie di autenticazione:

- Autenticazione del Dispositivo: nel contesto dell'accesso remoto o dei servizi online, può essere richiesta anche l'autenticazione del dispositivo utilizzato dall'utente (tramite certificato digitale, codice di autenticazione generato dal dispositivo o altre informazioni che confermano l'identità del dispositivo).

- Autenticazione Multifattore (MFA): per l'accesso a servizi sensibili o critici, può essere richiesta l'autenticazione multifattore, che richiede più di una forma di autenticazione per confermare l'identità dell'utente.

RADIUS per autenticare permette due approcci:

- Password Authentication Protocol (PAP): Il client RADIUS inoltra l'ID utente e la password dell'utente remoto al server di autenticazione RADIUS. Se le credenziali sono corrette, il server autentica l'utente e il client RADIUS abilita l'utente remoto a connettersi alla rete.
- Challenge Handshake Authentication Protocol (CHAP): o handshake a tre vie, si basa sull'uso di un segreto condiviso crittografato tra client e server, rispetto al precedente è più sicura (crittografa gli scambi e si possono eseguire autenticazioni ripetute durante la sessione).

## FIREWALL, IDS E IPS

I firewall proteggono le reti da accessi non autorizzati e attacchi malintenzionati. Agiscono come la prima linea di difesa, filtrando il traffico in entrata e in uscita basandosi su regole definite.

Possono essere hardware o software, o una combinazione di entrambi. Monitorano e controllano il traffico in entrata e uscita basandosi su regole.

### Firewall 1° Gen

Chiamati "Packet Filter Firewall", filtrano il traffico basandosi su regole statiche predefinite, come indirizzi IP e numeri di porta.

Sono semplici e veloci con costi relativamente bassi ma si paga con un controllo limitato sul contenuto e vulnerabilità a spoofing e attacchi basati su protocolli.

### Firewall 2° Gen

Chiamati "Stateful inspection Firewall", introducono il controllo dello stato delle connessioni, permettendo una migliore ispezione del traffico di rete, monitorando l'intera sessione.

Rispetto ai primi offrono un monitoraggio migliore e un filtraggio avanzato, sono adatti alle reti dinamiche. Le limitazioni si sentono sui costi e la complessità di gestione.

### Funzionalità Firewall

- Filtrano il traffico andando a controllare ogni pacchetto ed evitare che escono o entrano se non conformi a certe regole.

- Prevenzione delle Intrusioni, se avanzati permettono tramite sistemi di prevenzione delle intrusioni (IPS) di bloccare attacchi.
- Controllo dell'Accesso alle risorse di rete.
- Registrazione e Reporting di eventi di rete per monitorare l'attività e facilitare l'identificazione e l'analisi delle minacce.

## Configurazioni dei Firewall

### Statica

Implica regole fisse che non cambiano automaticamente in risposta al traffico di rete o ad altri fattori esterni.

Estremamente semplici nella gestione con una prevedibilità elevata e overhead basso.

Si consiglia l'uso in piccole-medie imprese con modifiche alla rete rare e traffico prevedibile e limitato.

### Dinamica

Permette al firewall di adattare le sue regole in tempo reale in risposta a cambiamenti nel traffico di rete o ad altri criteri specificati.

Flessibili, se cambia la rete spesso, e si adattano per riconoscere e reagire a pattern inusuali al costo di un elevato costo computazionale.

SI consiglia l'uso in aziende grandi e servizi cloud con traffico alto e variabile e sempre esposto ad a minacce.

### Regole

Set di istruzioni configurate in un firewall per regolare il traffico di rete in entrata e uscita in base a criteri specifici; basate su criteri specifici

Servono per decidere quali dati possono attraversare il firewall.

Più in alto è definita la regola, maggiore è la sua priorità

**Default deny** O "deny-all" o "implicit deny", stabilisce che, per impostazione predefinita, tutti gli accessi alla rete sono negati.

A meno che non esista una regola specifica che autorizzi esplicitamente un particolare tipo di traffico NULLA può attraversare il traffico.

Massima sicurezza e minimizzazione dei rischi ma limitante se non configurato correttamente con blocchi di traffico legittimo.

**Whitelist e Blacklist** Le whitelist sono un elenco di entità che sono esplicitamente autorizzate a interagire con la rete.

Molto sicure ma vanno aggiornate continuamente.

Le blacklist sono un elenco di entità che sono esplicitamente bloccate.

Più flessibili ma meno sicure.

**Ingress** Prima a linea di difesa contro attacchi esterni e accessi non autorizzati, gestisce il traffico di rete che entra nella rete da fonti esterne.

Vengono implementate tramite firewall, IPS e ACL

**Egress** Previene la perdita di dati sensibili e monitora le comunicazioni in uscita, blocca o limita il traffico verso destinazioni non fidate.

Vengono implementate tramite firewall e IDS.

## Architetture

### Termini utili

DMZ, Demilitarized Zone, rete fisica o logica separata che funge da strato intermedio tra la rete interna sicura di un'organizzazione e il resto del mondo esterno.

In questa zona si inseriscono servizi accessibili dall'esterno (FTP, Posta, ecc) senza esporre tutta la rete.

Screening router, router configurato con regole di filtraggio dei pacchetti, che determinano quali dati possono passare tra la rete esterna, la DMZ e la rete interna.

Primo punto di difesa che filtra tutto in e out.

Bastion Host, sistema fortemente securizzato, ottimizzato per resistere agli attacchi. Sono implementate misure di sicurezza robuste (sistemi di rilevamento delle intrusioni, firewall e software di monitoraggio).

Fornisce un livello aggiuntivo di sicurezza, agendo come un punto di ingresso sicuro per gli utenti esterni, si colloca fuori dal firewall esterno ma è comunque collegato alla rete interna.

### Packet Filtering Firewall

Screening router al gateway di Internet configurato in modo da filtrare o bloccare determinati pacchetti in base a regole definite.

In questa architettura il testing e il logging sono limitati con l'aggiunta di una gestione delle regole difficili.

## Dual-homed Gateway Firewall

Sistema in cui il dispositivo di sicurezza possiede due interfacce di rete, ognuna collegata a una rete distinta.

Intermediario stretto tra la rete interna e Internet, non permettendo il traffico diretto tra le due, ma gestendo e filtrando ogni comunicazione.

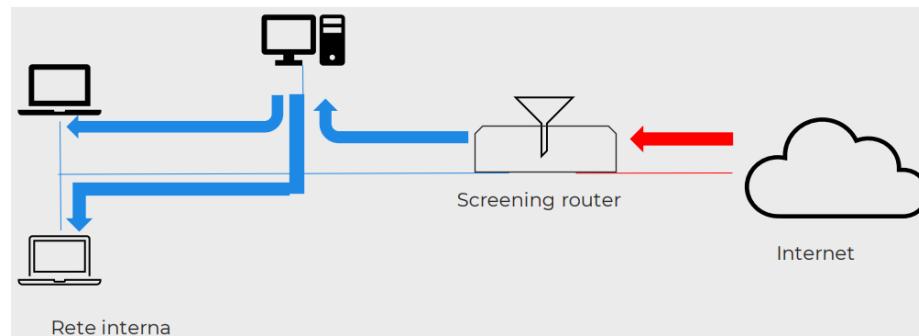
Isolamento tra reti, forzando il traffico attraverso il firewall, con un controllo del traffico migliorato.

## Screened Host Firewall

Sistema in cui un Bastion Host è protetto da uno Screening Router.

Intermediario tra Internet e la rete interna, limitando l'accesso diretto alla rete interna e filtrando il traffico verso e dal Bastion Host.

Isolamento tra reti, forzando il traffico attraverso il Bastion Host, con un controllo del traffico migliorato.

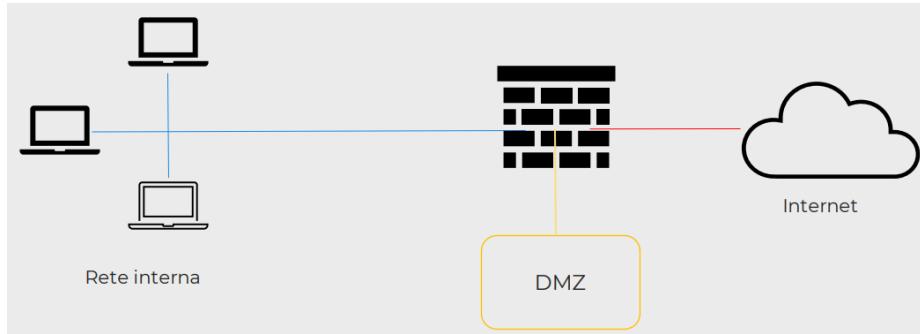


## Screened Subnet Firewall

Impiega un singolo firewall dotato di tre interfacce di rete. Questa configurazione consente di separare e gestire in modo efficace il traffico destinato a diversi segmenti di rete.

Ogni interfaccia porta a:

1. Interfaccia Pubblica: collega firewall e Internet.
2. Interfaccia DMZ: collega firewall e zona demilitarizzata.
3. Interfaccia Intranet: collega firewall e reti interne.



## Gateway

### Livello circuito

Un Circuit Level Gateway è un tipo di firewall che funziona a livello di sessione del modello OSI.

Gateway tra l'utente e il destinatario di una sessione, esaminando solo i pacchetti TCP handshaking e non il contenuto.

Garantisce che la sessione sia sicura e che i pacchetti vengano trasmessi durante una sessione valida.

Senza ispezionare i contenuti il firewall è efficiente e veloce con uso di CPU limitato; ma l'ispezione superficiale limita la sicurezza e protezione.

USATO SE SI VUOLE + VELOCITÀ

### Livello applicazione

Application Level Gateway o proxy firewall, è un firewall che filtra il traffico a livello di applicazione.

Esamina ogni pacchetto applicazione.

Il controllo dettagliato migliora la sicurezza permettendo un logging avanzato e un controllo granulare; ma le prestazioni diminuiscono e aumenta la complessità e il costo.

USATO SE SI VUOLE + SICUREZZA

## Proxy e Reverse Proxy

### Proxy

Application Gateway, dispositivo che funge da intermediario tra utenti finali (interni o esterni) e servizi.

Lo scopo principale è migliorare la sicurezza, la gestione e l'efficienza del traffico di rete.

Il proxy permette di filtrare contenuti specifici, tramite politiche varie (filtr URL, controlli contenuto, ecc), fare caching dei contenuti richiesti spesso e fare controllo di accesso.

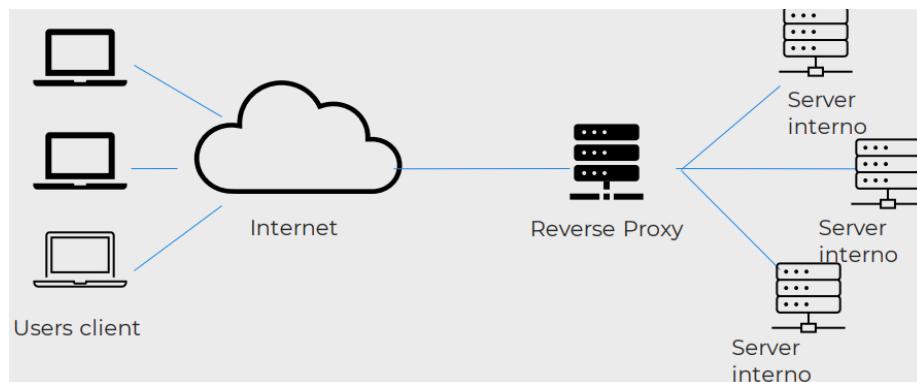
Alcuni esempi di utilizzo sono:

- bloccare siti non sicuri per gli studenti;
- evitare che i dipendenti vadano sui social durante il lavoro;

### Reverse

Proxy posizionato davanti ai server web e che gestisce le richieste internet che arrivano a questi server.

Serve per bilanciare il carico, offuscare l'infrastruttura e decodificare richieste SSL/TLS.



### IDS e IPS

Intrusion Detection System, sistema di rilevamento delle intrusioni che monitora il traffico di rete e/o i sistemi informatici per attività sospette e allerte.

Intrusion Prevention System, sistema di prevenzione delle intrusioni che non solo rileva le intrusioni, ma agisce attivamente per bloccarle o prevenirle.

Progettati per adattarsi e rispondere dinamicamente alle minacce emergenti, analizzando pattern rilevando nuove minacce.

Incorporano capacità di machine learning, in modo da apprendere continuamente dalla rete comportamenti normali o dannosi.

Entrambi segnalano immediatamente, IPS in più blocca attivamente le attività nocive.

Entrambi possono integrarsi con:

- Firewall: firewall diventa il primo livello e IDS/IPS rileva quello che passa per minacce più profonde.

- Antimalware: si crea una protezione contro il malware più completa con IDS/IPS che monitora il traffico e l'antimalware esamina file e processi.
- SIEM: IDS/IPS fornisce i dati al SIEM che analizza e visualizza informazioni di sicurezza per identificare tendenze e orchestrare risposte su larga scala.

## VLAN

Virtual Local Area Network, tecnologia di rete che consente di suddividere una rete fisica in più segmenti logici separati.

Isola il traffico, VLAN differenti non vedono il loro traffico, migliora la sicurezza e la gestione del traffico.

La sicurezza migliora perché:

- Controllo degli Accessi: è possibile definire politiche di sicurezza specifiche per ogni VLAN
- Riduzione della Superficie di Attacco
- Segmentazione di Rete

## Modern Firewall

### NGFW

Un Firewall di Nuova Generazione (NGFW) rappresenta un'evoluzione del tradizionale firewall stateful. È progettato per bloccare le minacce moderne con funzionalità avanzate che vanno oltre la semplice ispezione dei pacchetti.

Caratteristiche:

- Deep Packet Inspection (DPI): analizza anche i contenuti del payload dei pacchetti per rilevare malware e tentativi di intrusione.
- Controllo delle Applicazioni Integrato: identifica e filtra il traffico per applicazione.
- IPS: inclusa per bloccare attacchi noti e sconosciuti in tempo reale.
- Utilizzo di informazioni sempre aggiornate

### FWaaS

Firewall as a Service, soluzione di firewall gestita e distribuita tramite il cloud, che protegge senza hardware dedicato o gestione on premise.

Scalabile, con costi bassi e sempre accessibile.

## **Zero Trust**

Modello che presuppone che le minacce possano originarsi sia all'interno che all'esterno e che nulla debba essere fidato implicitamente.

Si verifica continuamente e tutti devono essere autorizzati e autenticati con il principio di privilegio minimo; si segrega la rete.

## **RED TEAMING E BLUE TEAMING**

### **Penetration Test**

Attacco organizzato e autorizzato, mirato a testare l'infrastruttura IT e le sue difese al fine di determinare la suscettibilità a vulnerabilità di sicurezza.

Durante il processo viene preparata una documentazione dettagliata degli step, composti da test automatici e manuali, e dei risultati ottenuti.

Con lo scopo ultimo di fornire dettagli sulla soluzione di mitigazione di una vulnerabilità rilevata.

Non è compito del penetration tester rimediate fisicamente ad una vulnerabilità identificata.

### **Inherent risk**

Livello di rischio presente quando sono stati implementati gli adeguati security controls.

Si può decidere di accettare, trasferire, mitigare con:

- assicurazioni
- firmare contratti con service provider
- implementare misure preventive o recovery plan

Il rischio intrinseco (inherent risk) è il livello base del rischio associato a un'attività o processo prima che vengano implementati dei controlli per mitigarlo. È il rischio inevitabile che deriva semplicemente dal fare qualcosa.

### **Vulnerability Assessment**

Processo che si occupa di definire, identificare e classificare vulnerabilità in un sistema.

Di solito eseguito con l'aiuto di strumenti automatici detti Vulnerability Scanner

In questo caso NON viene fatto l'exploit della vulnerabilità (al contrario del PT).

## **Prospettive di test**

### **External PT**

Si testa il perimetro esterno andando a simulare pienamente un attacco.

Viene fatto in maniera furtiva, si testano meccanismi di protezione, rumorosa, concentrazione sull'identificazione di vulnerabilità, o ibrida.

Lo scopo ultimo è l'accesso agli host, ottenere info sensibili e accedere alla rete interna.

### **Internal PT**

Si testa la rete interna con un host interno alla rete; si può fare anche su sistemi isolati dalla rete principale.

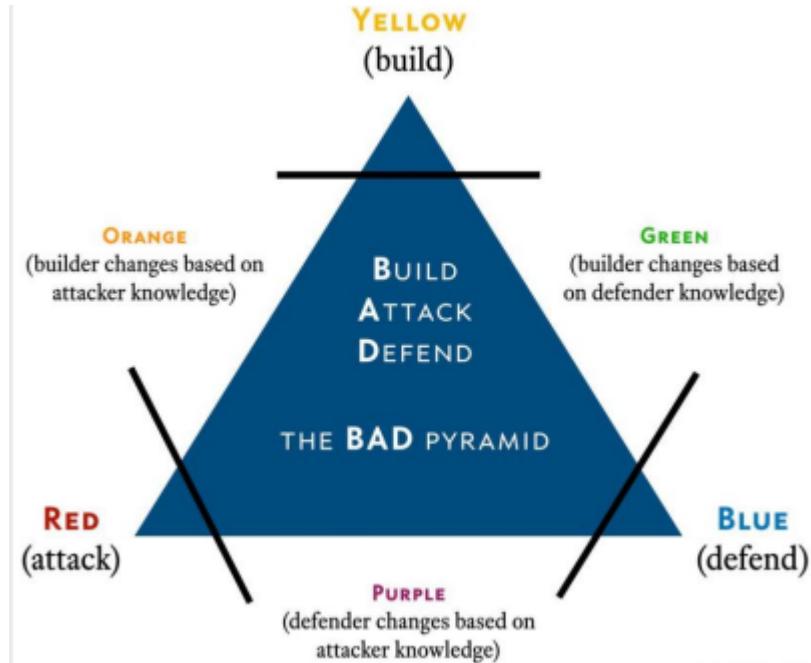
Lo scopo ultimo è l'accesso a servizi sensibili ed effettuare lateral movement, quindi controllare se è possibile spostarsi all'interno della rete.

## **Testing Methods**

I PT possono essere di diverso tipo e variano in base a quante informazioni iniziali si hanno:

- Blackbox: fornite poche informazioni, come indirizzi IP, domini, subnets. Simulazione 1:1 di un attacco in blind.
- Greybox: fornite alcune informazioni aggiuntive rispetto al Blackbox testing, come url. Soluzione ibrida.
- Whitebox: accesso a tutte le informazioni possibili, vista dell'intera infrastruttura, codice sorgente di applicativi e credenziali. Concentrazione sull'identificazione di vulnerabilità.

## Red, Blue, Purple Teams



### Red

Emulano Tattiche Tecniche Procedure (TTP) in modo più realistico possibile. Pratica simile ma non identica al penetration test, il red teaming è un'attività più completa che include tattiche, tecniche e procedure. Basato su obiettivi mirati.

### Blue

Security team interno incaricato della difesa da attaccanti reali e red team. Si differenzia dal classico security team interno all'azienda in quanto mirato a vigilanza costante.

### Purple

Il loro obiettivo è ottimizzare l'efficacia dei team di red e blue, integrano le tattiche e i controlli del blue team con le minacce e le vulnerabilità individuate dal red team.

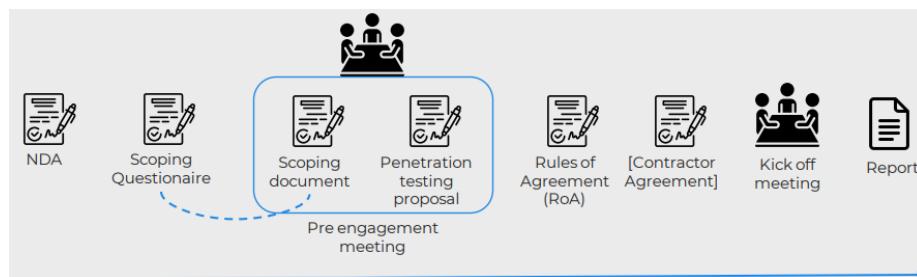
## Penetration Testing Process

Il processo di PT non rappresenta una ricetta fissa e non è una guida step-by-step; perché ogni azienda ha un'infrastruttura unica con obiettivi di testing unici.

Per definirlo nel migliore dei modi si usano stage indipendenti (non sono step, non vanno seguiti in maniera rigida). Possono essere:

### Pre Engagement

Fase preliminare, si determinano alcune figure all'interno dell'azienda per firmare documenti al fine di rispettare le norme vigenti, tutelare ambo le parti e siglare come verrà svolto il test.



Iniziamo con il NDA, che serve per proteggere la riservatezza delle informazioni scambiate durante il penetration test.

Poi si passa allo Scoping Questionnaire, insieme strutturato di domande progettate per raccogliere informazioni specifiche e cruciali relative agli obiettivi, ai requisiti e alle condizioni del progetto.

In soldoni si identificano il perimetro, le risorse coinvolte e le aspettative del cliente.

Passo successivo è il Meeting, dove si discutono tutti gli elementi rilevanti ed essenziali con il cliente prima del penetration test (chiarimento info scoping questionnaire, metodologie PT, ecc).

Le informazioni raccolte durante il meeting sono integrate con quelle del questionario di scoping.

Successivamente si firma il ROA, Rule of Agreements, accordo tra l'organizzazione che richiede il penetration test e il team che lo condurrà; stabilisce le regole, le aspettative e le responsabilità di entrambe le parti.

Se si effettua un physical PT è necessario il Contractor Agreement.

Dopo la firma di tutti i documenti contrattuali viene fatto il kick-off meeting, riunione in presenza.

Durante questo incontro, si stabilisce:

- natura del PT e come si svolgerà;
- gestione vulnerabilità critiche;
- rischi del PT.

## Information Gathering

Fase cardine in cui vengono raccolte tutte le informazioni disponibili e ottenibili associate al target; usate come input per le successive fasi.

Ha 4 categorie:

1. OSINT: pratica che consiste nel collezionare informazioni da fonti pubbliche.

Di solito si trovano informazioni sui componenti o sui dipendenti utili per accedere all'infrastruttura o discovery di superficie di attacco e vulnerabilità.

2. INFRASTRUCTURE ENUMERATION: fase in cui si indaga la posizione della compagnia in internet ed intranet. Si indaga la struttura e si sviluppa un overview dell'infrastruttura.
3. SERVICE ENUMERATION: fase in cui vengono identificati servizi che ci permettono di interagire con host e server attraverso network.
4. HOST ENUMERATION: fase in cui vengono esaminati i singoli host interni al perimetro d'attacco. Utili per exploitation, lateral movement, privilege escalation.

Si identificano SO, servizi utilizzati con versione e componenti di comunicazione; è possibile farlo all'interno dell'host stesso e si chiama Pillaging.

Uno strumento open-source utile per fare Port Scanning è NMAP, che può effettuare host-discovery, scansione di porte e vulnerability assessment.

Per funzionare e determinare quale protocollo un sistema usa invia una richiesta, del protocollo da cercare, ad una porta e in base alla risposta (o se proprio non risponde) possiamo sapere diverse cose:

Probe Response	Assigned State
TCP SYN/ACK response	Open
TCP RST response	Closed
No Response	Filtered
Errore ICMP unreachable	Filtered

UDP SCAN (-sU):	
Probe Response	Assigned State
Any UDP response (unusual)	Open
No Response	Open filtered
ICMP unreachable error (Type 3 code 3)	closed
Altri errori ICMP	Filtered

Source img: <https://www.dpstele.com/snmp/transport-requirements-udp-tcp.php>

Nel caso non si riceva risposta in una scansione UDP, potrebbe essere perché porta è filtrata o perché pacchetto non conteneva dati attesi e quindi è stato ignorato dal servizio che l'ha ricevuto. Per questo motivo con UDP è più efficace inviare pacchetti noti per il servizio che ci si aspetta di trovare.

## Vulnerability Assessment

Fase in cui si analizzano i dati e le informazioni ottenute da Information Gathering e si traggono conclusioni.

Le conclusioni devono essere confermate

Una mappa di esecuzione è:

1. Vulnerability research: si cercano vulnerabilità, exploit e misconfigurazioni già note e riportate.
2. Trovata una vulnerabilità possiamo domandarci che cosa l'ha causata o la sta causando.
3. Realizzazione di una Proof-Of-Concept (POC), prototipo che mira a testare la presenza della vulnerabilità.
4. Test della POC per confermare (diverso da sfruttamento della vulnerabilità).
5. Si riparte.

L'utilizzo di una Proof Of Concept è un attacco che mira alla mera verifica della presenza di una vulnerabilità o misconfigurazione senza lo sfruttamento di questa.

Il suo obiettivo è provare che security hole esiste, così che sviluppatori e amministratori possano validarla, riprodurla e testarne le mitigazioni.

### **Exploitation**

Fase in cui si sfruttano le vulnerabilità o debolezze rilevate.

### **Post Exploitation**

Fase che segue l'avvenuto exploit di un target system, con l'obiettivo di reperire informazioni sensibili e rilevanti per la sicurezza locale o del target business.

Spesso richiede privilegi più elevati di quelli di un utente standard.

Include:

- Evasive testing: se è compreso nelle regole di ingaggio perché un singolo comando può generare allarme e comportare quarantena dell'host

Rende il test più realistico possibile simulando un vero attacco e mettendo alla prova gli strumenti di rilevazione e difesa dell'infrastruttura.

- Pillaging: Da dentro il sistema si indaga ruolo dell'host in network e trust-relationship con altri componenti del network, si cercano info sensibili o utili per lateral movement.

Ora che si ha una nuova overview possiamo eseguire una vuln assessment.

- Persistenza: gli aggressori implementano backdoor o meccanismi di accesso costante per mantenere l'accesso non autorizzato al sistema anche dopo che la vulnerabilità iniziale è stata corretta.

- Data Exfiltration: si cerca di trovare informazioni sensibili, per testare sistemi di Data Loss Prevention o EDR.
- Privilege escalation: si aumenta il livello di accesso e i privilegi su un sistema compromesso attraverso sfruttamento di vulnerabilità software, debolezze nella configurazione del sistema o ingegneria sociale.

Un esempio è l'utilizzo di sudo bad privilege configuration, che utilizza il file sudoers che contiene la allow list dei comandi, per esempio vim con privilegi massimi, permettendo di usare una shell di root.

### **Lateral Movement**

Fase in cui si testa cosa può fare un attaccante con l'intero network corporativo una volta ottenuto accesso ad un sistema.

Ci si muove all'interno della rete per attaccare altre macchine o per raggiungere obiettivi più importanti.

### **Post-Engagement**

Si concludono i test e si esegue una pulizia accurata, rimuovendo strumenti e script utilizzati e ripristinando eventuali modifiche alla configurazione.

È fondamentale documentare dettagliatamente tutte le attività svolte per facilitare le operazioni di pulizia.

Le informazioni sensibili dei clienti non verranno mantenute alla conclusione del test

Dopo la pulizia si consegna il report, che include:

- Attack chain in caso di compromissione completa o accesso interno alla rete dall'esterno, con dettagli tecnici e guida step-by-step a fini di riproducibilità.
- Executive summary, report non tecnico per audience manageriale.
- Lista di finding con: Risk rating (es. CVSS score), Impatto, Raccomandazioni di rimedio o mitigazione, References.

Durante la revisione del rapporto con il cliente, si dovrebbero affrontare le domande e le preoccupazioni, concentrandosi sulle scoperte rilevanti.

Si passa poi alla chiusura progetto e passaggi post valutazione, come ulteriori test per verificare l'efficacia delle soluzioni adottate per risolvere le vulnerabilità individuate.

### **SOC**

il Security Operations Center è il centro nevralgico per la monitorizzazione e l'analisi delle attività di sicurezza informatica di un'organizzazione il cui obiettivo

è il rilevamento, la valutazione e la risposta a minacce informatiche in tempo reale.

## **Incident Response**

Processo utilizzato per gestire la rilevazione, l'analisi e la mitigazione degli incidenti di sicurezza informatica.

Le strategie di risposta sono:

- Preparazione: Formazione del team di risposta e definizione di protocolli di comunicazione e risposta.
- Identificazione e Contenimento: Rapido isolamento delle risorse colpite per prevenire ulteriori danni.
- Eradicazione e Recupero: Eliminazione della minaccia dal sistema e ripristino delle operazioni normali.
- Revisione: Analisi del gesto e aggiornamento delle procedure di sicurezza per prevenire incidenti futuri.

## **Hands On**

### **Hands On - Kerberoasting Attack**

Nelle slides dalla pagina 36 alla 41

### **Hands On - Golden/Silver Ticket Attack**

Nelle slides dalla pagina 42 alla 49

### **Hands On - Broken Access Control**

Nelle slides dalla pagina 3 a 6

### **Hands On - Cryptographic failures**

Nelle slides dalla pagina 7 a 10

### **Hands On - Injection**

Nelle slides dalla pagina 11 a 14

### **Hands On - Identification and authentication failures**

Nelle slides dalla pagina 15 a 18