LINEE GUIDA PER LA MODELLAZIONE DELLE MINACCE ED INDIVIDUAZIONE DELLE AZIONI DI MITIGAZIONE CONFORMI AI PRINCIPI DEL SECURE / PRIVACY BY DESIGN

SOMMARIO

LISTA DELLE TABELLE

LISTA DELLE FIGURE

1 INTRODUZIONE

1.1 Scopo

Gli obiettivi degli attacchi sono spesso vulnerabilità che si celano all'interno delle applicazioni software. La comunità OWASP (www.owasp.org) sottolinea la necessità di accrescere la consapevolezza sulla sicurezza delle applicazioni in quanto il software non sicuro mette a repentaglio le infrastrutture finanziarie, sanitarie, difensive, etc.

Questo documento vuole analizzare il contesto (processi, metodi e modelli) della Progettazione di Applicazioni Sicure con l'obiettivo di fornire le linee guida per la modellazione delle minacce e conseguente individuazione di azioni di mitigazione, in conformità con i principi "Secure/privacy by Design". Ciò costituisce una fase importante nel processo di individuazione preventiva dei requisiti di sicurezza e privacy.

1.2 Ambito di Applicabilità

Il presente documento si applica al contesto tecnologico dell'Agenzia per l'Italia Digitale (in seguito AgID) nell'ambito dei servizi previsti dal Contratto Esecutivo in [DA-8].

1.3 Struttura del documento

Il presente documento è articolato come segue:

- Il **Capitolo 5**, introduce i concetti base di security e privacy by design, analizza gli strumenti e i modelli a supporto della fase di progettazione del software sicuro, in essere e in divenire;
- Il **Capitolo 6**, definisce le linee guida per l'identificazione preventiva delle possibili minacce, delle relative azioni di mitigazione e per la valutazione e prioritizzazione delle minacce stesse;
- Il **Capitolo 7**, fornisce un caso d'uso applicativo in cui vengono impiegate le metodologie e gli strumenti di sicurezza indicati nel Capitolo 5 (Linee Guida).

2 DEFINIZIONI E ACRONIMI

2.1 Definizioni

Vocabolo	Descrizione
Ambiente di produzione	Agglomerato di sistemi, dispositivi hardware ed applicazioni in cui il software viene installato nella sua forma definitiva al fine di soddisfare le richieste dell'operatore o dell'utente finale.
Ambiente di sviluppo	Agglomerato di sistemi, dispositivi hardware ed applicazioni in cui il software viene progettato e creato.
Ambiente di test	Agglomerato di sistemi, dispositivi hardware ed applicazioni in cui il software creato viene testato.
Autenticazione	Processo attraverso il quale un sistema, un utente o un programma tenta di confermare la sua identità ad un altro sistema o applicazione.
Autorizzazione	Processo di definizione dei privilegi, ruoli e permessi di un utente su un sistema o un'applicazione.
Batch Job	Processo di scambio dati o informazioni che intercorre automaticamente, in periodi temporali prestabiliti, tra due sistemi, applicazioni o componenti.
Dati critici	Dati che hanno una rilevanza preponderante per l'immagine e l'operato aziendale (esempio cartellini di traffico telefonico).
Dati personali	Come da decreto legislativo 196/03: "qualunque informazione relativa a persona fisica, persona giuridica, ente od associazione, identificati o identificabili, anche indirettamente, mediante riferimento a qualsiasi altra informazione, ivi compreso un numero di identificazione personale".
Dati sensibili	Come da decreto legislativo 196/03: "Dati personali idonei a rivelare l'origine razziale ed etnica, le convinzioni religiose, filosofiche o di altro genere, le opinioni politiche, l'adesione a partiti, sindacati, associazioni od organizzazioni a carattere religioso, filosofico, politico o sindacale, nonché i dati personali idonei a rivelare lo stato di salute e la vita sessuale".
Eccezione	Occorrenza di una circostanza che altera o mira ad alterare il corso previsto o il normale operato di un sistema, di un'applicazione o di una sua componente.
Evento	Situazione riconducibile ad un'attività svolta o ad un'eccezione causata dall'utente, rilevante ai fini della sicurezza del sistema e dell'Information Security.
Identificazione	Meccanismo di convalida preventiva di un'azione.
Information Gathering & Disclosure	Processo relativo alla fuga di dati o informazioni, causato da bug o errori nel software.
Information Security	Insieme di controlli, policy, processi e procedure mirate a garantire la sicurezza delle informazioni in azienda.
Offuscatore	Software che converte il codice sorgente in forma difficilmente interpretabile o non interpretabile del tutto al fine di inibire l'utilizzo di tecniche di reverse engineering.

Vocabolo	Descrizione
Organizzazione	Ente locale o centrale della Pubblica Amministrazione
Reverse Engineering	Processo mirato a scoprire i principi tecnologici di un'applicazione attraverso la sua analisi strutturale.
Token	Valore generato per identificare univocamente una sessione interattiva.

Tabella 1 - Definizioni

2.2 Acronimi

Codice	Titolo
ACLs	Access Control List
AgID	Agenzia per l'Italia Digitale
API	Application Programming Interface
ASLR	Address Space Layout Randomization
BSIMM	Building Security in Maturity Model
CLASP	Comprehensive and Lightweight Application Security Process
CVSS	Common Vulnerability Scoring System
DFD	Data Flow Diagram
DPD	Data Protection Directive
DPIA	Data Protection Impact Analysis
DPO	Data Protection Officer
DREAD	Damage Potential, Reproducibility, Exploitability, Affected users, Discoverability
DS	Data Store
E	Entity
FIPP	Fair Information Practice Principles
FIPPs	Fair Information Practice Principles
FPFSD	framework for Privacy-Friendly System Design
GDPR	General Data Protection Regulation
HAZOP	Hazardous Operations
IOI	Item of Interest
IPSec	IP Security
LINDDUN	Linkability, Identifiability, Non Repudiation, Detectability, Disclosure of information, Content Unawareness e Policy and consent Non-compliance
MIC	Mandatory Integrity Control
MITM	Man In The Midle
MPRA	Multilateral Privacy Requirements Analysis
MS –SDL	Microsoft's Security Development Lifecycle

Codice	Titolo
NIST	National Institute of Standards and Technology
OCSE	Organizzazione per la Cooperazione e lo Sviluppo Economico
OWASP	Open Web Application Security Project
P	Process
P.A.S.T.A	Process for Attack Simulation and Threat Analysis
PAR	Privacy Awareness Requirements
PbD	Privacy by Design
PE	Privacy Engineering
PETs	Privacy-Enhancing Technologies
PI	Personal Informations
PII	Personale Indentifiable Information
PRET	Privacy Requirements Elicitation Technique
PriS	Incorporating Privacy Requirements into the System Design Process
ProPAn	Problem-based Privacy Analysis
ROI	Return On Investment
S.B.S.R.S.	Security Bulletin Severity Rating System
SAMM	Software Assurance Maturity Model
SDL	Security Development Lifecycle
SDLC	Software Development Life Cycle
SSDLC	Secure Software Development Life Cycle
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege
SW	Software
TCP	Trasmission Control Protocol
UML	Linguaggio di modellazione Unificata
XSS	Cross-site scripting

Tabella 2 - Acronimi

3 ESIGENZE ED AMBITI DI APPLICAZIONE

Secondo la fonte Gartner[1], anche in considerazione delle contromisure adottate negli ultimi anni per il controllo degli accessi alle infrastrutture e la messa in protezione dei dati, negli ultimi tempi, oltre il 75% degli attacchi sono stati indirizzati direttamente verso le applicazioni software, causando gravi danni di immagine e pesanti perdite finanziarie. Gli obiettivi degli attacchi sono le vulnerabilità che si celano all'interno di tale applicazioni. Le vulnerabilità applicative sono quasi sempre presenti poiché, fino ad ora, le politiche di qualità del software ed i relativi investimenti si sono concentrate soprattutto sulla correzione delle difettosità funzionali e sulle performance delle logiche applicative, trascurando l'attuazione di pratiche di progettazione e programmazione che garantiscono l'opportuna sicurezza.

Le vulnerabilità applicative vengono continuamente introdotte dagli stessi team di sviluppo interni alle organizzazioni, dai fornitori, dalle soluzioni open-source: alcuni analisti affermano che il 64% degli sviluppatori non sono confidenti di poter scrivere applicazioni sicure (fonte: Microsoft Developer Research[2]);

A titolo esplicativo, la seguente matrice riporta un indice quantitativo di effort in termini di costi necessario nella risoluzione delle problematiche di sicurezza applicativa:

		PROBLEMATICHE DI SICUREZZA (VULNERABILITA')				
		Scoperte in fase di progettazione e disegno	Scoperte in fase di sviluppo	Scoperte in fase di integrazione	Scoperte in fase di collaudo	Scoperte in fase di esercizio
DOVUTE A ERRORI	Architetturali	1X	5X	10X	20X	30X
	Di codifica		1X	10X	20X	30X
	Di integrazione			1X	10X	15X

Tabella 3 - Vulnerabilità dovute a errori

Il National Institute of Standards and Technology (NIST[3]) ha stimato che il costo del "code fixing" eseguito dopo il rilascio in produzione può risultare 30 volte il costo che si avrebbe in fase di progettazione.

Il numero delle tipologie di attacchi cresce in maniera esponenziale: dalle circa 2500 identificate nel 2000, si è passati ad oltre 50000 nel 2009 e continuamente se ne scoprono delle nuove (fonte: CERT[4]). Da qui l'inizio della diffusione delle prime e fondamentali best practices in materia di sicurezza applicativa, le prime tra tutte riconducibili ad una buona ingegnerizzazione del software, una piena comprensione delle minacce più comuni, compresi i difetti propri dei linguaggi di programmazione, ma soprattutto una considerazione della problematica fin dalle prime fasi del ciclo di sviluppo.

I costi aggiuntivi possono portare ad una perdita significativa di produttività e fiducia da parte degli utenti. L'adozione di un ciclo di vita sicuro del software (SSDLC) - ***Linee guida per l'adozione di un ciclo di sviluppo software sicuro - D01.Fase1.SP2*** [DR-1] - aiuta sistematicamente a considerare ed implementare opportune attività/metodologie di sicurezza nel corso di tutte le sue fasi: analisi, progettazione, sviluppo,

[1] https://www.gartner.com

[2] https://developer.microsoft.com/it-it/windows

[3] https://www.nist.gov/

[4] https://www.certnazionale.it/

test e manutenzione, assicurando che le vulnerabilità siano più facilmente individuabili e misurate prima della distribuzione dell'applicazione, riducendo così il costo totale dello sviluppo del software.

Esistono diversi modelli e metodologie di ciclo di vita di sviluppo del software, ma ciascuna di queste in generale consiste di una serie di step o fasi predefinite. Indipendentemente dal modello SDLC scelto, è necessario integrare il concetto di sicurezza al fine di garantire una appropriata protezione del sistema e delle informazioni che dovrà trasmettere, elaborare e memorizzare.

La progettazione di applicazioni software sicure può essere complessa e difficile, ma una buona strategia da adottare è quella di scomporre l'applicativo in piccole parti in modo da renderlo più facilmente analizzabile. Esistono importanti motivazioni che giustificano la necessità di realizzare un modello delle minacce, primo tra tutti la ricerca preventiva di difetti di sicurezza e a seguire, la finalizzazione dei requisiti di sicurezza nella progettazione di applicazioni più sicure.

4 PROGETTAZIONE DEL SOFTWARE SECURE/PRIVACY BY DESIGN

4.1 Processi di sviluppo del software sicuro

Questo capitolo illustra alcuni dei framework di processo di riferimento nello sviluppo sicuro delle applicazioni software, quali: il Software Assurance Maturity Model (SAMMM), il Building Security in Maturity Model (BSIMM), il Comprehensive and Lightweight Application Security Process (CLASP) e il ciclo di vita di sviluppo sicuro (SDL) di Microsoft (MS-SDL).

Lo scopo nel presentare diversi framework di sviluppo è quello di ottenere una migliore comprensione delle tecniche di Threat Modeling e di come questo si adatta alle fasi del ciclo di sviluppo del software.

4.1.1 Open Software Assurance Maturity Model (SAMM)

OpenSAAM[5] è un framework supportato da OWASP[6] (Open Web Application Security Project) che si basa su un insieme di procedure di sicurezza legate a quattro importanti funzioni di business critiche, coinvolte nello sviluppo del software, vale a dire Governance, Costruzione, Verifica e Distribuzione. Ciascuna funzione di business adotta tre pratiche di sicurezza e ciascuna di esse è suddivisa in tre livelli di maturità. La valutazione delle minacce è la prima pratica di sicurezza adottata durante la funzione di business "Costruzione". Questa utilizza il Threat modeling per identificare i potenziali rischi. OpenSAAM non si lega ad alcun approccio di modellazione delle minacce e raccomanda l'uso di STRIDE della Microsoft o TRIKE come possibili opzioni.

4.1.2 Building Security in Maturity Model (BSIMM)

L' iniziativa di sicurezza BSIMM[7] è stata progettata per aiutare i team di sviluppo software a comprendere e pianificare la sicurezza in un ciclo di vita di sviluppo delle applicazioni, studiando le pratiche di cinquantuno importanti iniziative di sicurezza software. Aziende come Google, Adobe, Intel, Visa, Nokia, Sony e Microsoft hanno partecipato alla ricerca guidata da Gary McGraw (leader esperto di settore nella sicurezza del software, vicepresidente della Security Technology presso la Synopsys Inc. SNPS, autorità riconosciuta a livello mondiale per la sicurezza del software e autore di otto libri tra i più venduti su questa tematica). La metodologia risultante ha unito le migliori pratiche (parere del team BSIMM) in un'unica iniziativa. Si tratta di dodici pratiche raggruppate in quattro domini, Governance, Intelligence, SSDL Touchpoint (pratiche associate all'analisi e alla garanzia di particolari manufatti e processi di sviluppo del software. Tutte le metodologie di sicurezza del software includono l'analisi dell'architettura, la revisione del codice e i test di sicurezza) e Deployment, utilizzate per organizzare le attività del framework di sicurezza del software. La prima pratica all'interno del dominio "Intelligence" è costituita dai modelli di attacco. Il Threat modelling viene utilizzato durante questa fase per modellare gli attacchi e per creare una base di conoscenza relativa all'applicazione.

BSIMM non è un approccio innovativo per lo sviluppo sicuro del software. Questo framework ha raccolto dati su attività effettivamente svolte dai leader dell'industria promuovendo poi quelle migliori e più utilizzate. Per tradizione, le società di software erano riluttanti a divulgare informazioni riguardo le pratiche interne. L'impiego della metodologia BSIMM sarebbe vantaggiosa per un'organizzazione che intende adottare un'iniziativa di sicurezza o migliorare/maturare le pratiche esistenti. Tuttavia, BSIMM non fornisce sufficienti informazioni di dettaglio riguardo il Threat Modeling.

[5] http://www.opensamm.org/

[6] https://www.owasp.org/images/6/6f/SAMM_Core_V1-5_FINAL.pdf

[7] https://www.bsimm.com/

4.1.3 Comprehensive, Light-weight Application Security Process (CLASP)

Il CLASP[8], è un altro framework di sicurezza supportato dall' OWASP che contiene best practices formalizzate per attuare la sicurezza, in modo strutturato e ripetibile, nei cicli di vita di sviluppo di software in essere o in divenire. Il framework esiste dal 2005, ma non sono stati registrati recenti aggiornamenti del progetto. È stato originariamente sviluppato dalla Secure Software Inc. e successivamente donato a OWASP che lo ha rilasciato come soluzione completa di sicurezza a favore delle organizzazioni. Insieme all'SDL di Microsoft, CLASP è stato riconosciuto come uno dei processi originali di alto profilo per lo sviluppo di software sicuro. CLASP, si basa su sette best practices che sono alla base di tutte le attività connesse alla sicurezza. La valutazione dell'applicazione è una di queste pratiche, che promuove un'analisi dei requisiti di sicurezza e la progettazione del sistema basata su l'utilizzo del Threat Modeling.

Il CLASP non è legato ad alcun metodo di modellizzazione specifico e non ha sviluppato un proprio approccio. Al fine di proporre un'iniziativa di sicurezza, a causa di una mancata evoluzione di questo progetto, è preferibile prendere in considerazione un framework più innovativo.

4.1.4 Microsoft's Security Development Lifecycle (SDL)

Il ciclo di vita di sviluppo sicuro (SDL[9]) è una metodologia introdotta dall'iniziativa "Trustworthy Computing" di Microsoft. SDL mira a ridurre i costi di manutenzione del software e ad aumentare l'affidabilità implementando la sicurezza in ciascuna fase del ciclo di vita dello sviluppo.

Il processo consiste in pratiche di sicurezza, raggruppate in sette fasi distinte: formazione, requisitazione, progettazione, implementazione, verifica, rilascio e monitoraggio/manutenzione.

Uno degli aspetti chiave dell'SDL è l'introduzione del Threat Modeling nella fase di progettazione, che promuove l'individuazione preventiva delle vulnerabilità presenti nelle applicazioni e alcune volte persino i potenziali difetti di progettazione. Queste informazioni vengono poi utilizzate per attuare eventuali piani di mitigazione o modifiche progettuali.

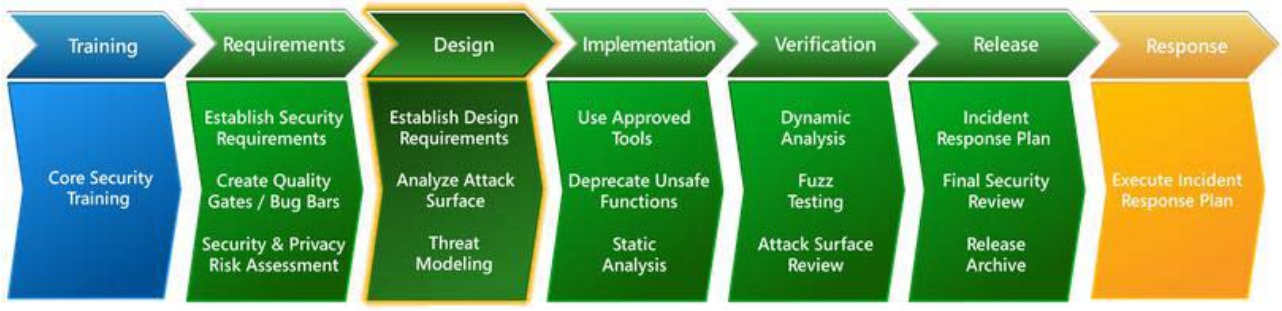


Figura 1 - Processo del ciclo di sviluppo sicuro di Microsoft

Sono disponibili diversi strumenti, non solo per il Threat Modeling ma per ciascun aspetto dell'SDL, tutorial online, documentazione, forum e blog di supporto. Il processo può essere utilizzato anche con le metodologie di sviluppo Waterfall e Agile e può essere applicato a qualsiasi piattaforma e implementato da qualsiasi organizzazione indipendentemente dalla sua dimensione.

L'obiettivo che tutte le metodologie di modellazione delle minacce condividono è lo sviluppo di un processo a passi iterativi che un team di sviluppo può facilmente seguire durante la valutazione di un sistema software.

[8] https://www.owasp.org/index.php/CLASP_Concepts

[9] https://www.microsoft.com/en-us/sdl/default.aspx

Microsoft ha sviluppato e pubblicato una propria metodologia di modellazione delle minacce denominata STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege) e diversi approcci per l'analisi dei rischi, tra cui la DREAD (Damage potential, Reproducibility, Exploitability, Affected users, Discoverability). Hussain, Erwin e Dunne hanno indicato la STRIDE come la metodologia di Threat modeling più ampiamente utilizzata [1].

Al termine dell'attività STRIDE, viene prodotto un elenco di vulnerabilità. Tali vulnerabilità vengono poi classificate secondo un indice di priorità (basso/medio/alto) utilizzando una tecnica di prioritizzazione delle contromisure attraverso la DREAD. Inoltre il prodotto finale dell'analisi STRIDE/DREAD può indirizzare anche in una fase successiva un eventuale Penetration Test (sulla base delle vulnerabilità riscontrate è possibile delimitare il perimetro che sarà poi oggetto di PT).

La Figura che segue illustra le fasi principali nella preparazione e nell'esecuzione di una modellazione delle minacce.

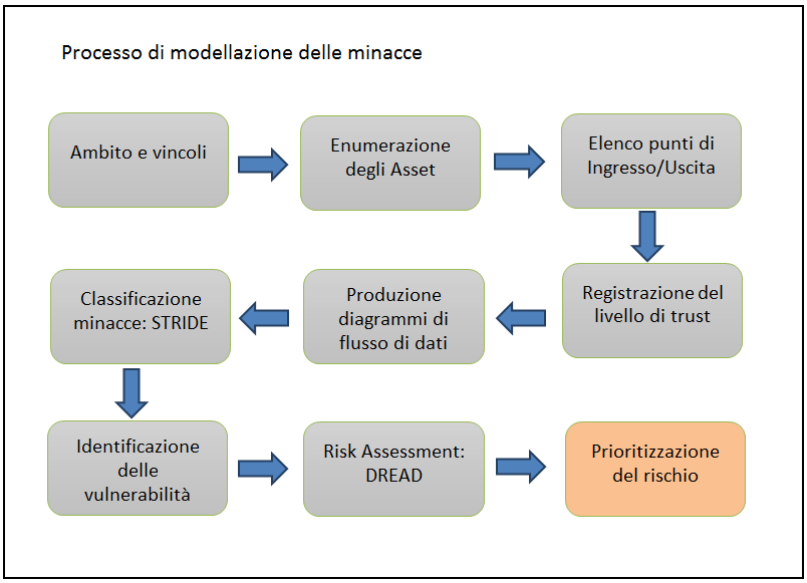


Figura 2 - SDL: Passi nella modellazione delle minacce

I processi riguardanti le diverse metodologie di Threat Modelling possono differire fra loro, ma hanno tutti in comune la raccolta delle informazioni sull'applicazione, sul suo ambiente, su come questa viene utilizzata e distribuita, come le vulnerabilità vengono identificate e come i rischi vengono valutati.

La tabella che segue riassume le caratteristiche principali di un processo di modellazione delle minacce:

Caratteristica/ambito	Descrizione
Ambito e vincoli	La modellazione delle minacce può essere un processo che richiede tempo, soprattutto se si modella un'applicazione di scala enterprise. Avere chiari gli obiettivi, aiuta a focalizzare l'attività di modellazione delle minacce. L'obiettivo del modello dovrebbe essere ben definito fin dall'inizio. Nel caso di sistemi più grandi, il processo può dare in modo più rapido maggior valore, riuscendo a limitare il campo di azione solo ad aree specifiche. Nella fase iniziale del processo di Threat Modelling, quando l'ambito e i vincoli sono ben definiti, informazioni aggiuntive come scenari d'uso, dipendenze e note di sicurezza forniscono supporto a una migliore comprensione del sistema stesso.
Scenari d'uso	Viene creato un elenco di possibili scenari di utilizzo o di trascorsi che dimostrano l'uso previsto dell'applicazione. Ciò può aiutare nell'identificazione di potenziali

	aree di abuso presenti nel sistema.
Dipendenze esterne/interne	Le applicazioni software spesso dipendono da altri sistemi o componenti. Questi possono avere un impatto diretto sulla sicurezza del sistema in oggetto. È importante elencare tali dipendenze e individuare l'impatto che questi possono avere sul sistema stesso. Alcuni esempi posso essere l'utilizzo di: Antivirus, Firewalls, librerie di terze parti e sistemi di autenticazione.
Note di sicurezza	Vengono registrati dettagli come le informazioni già conosciute relative agli aspetti di sicurezza di un sistema, eventuali ipotesi fatte sull'applicazione, o trade-off a livello di disegno. Queste note di sicurezza possono aiutare il processo decisionale.
Asset	Vengono elencati gli asset che potrebbero essere di interesse per un potenziale avversario. Questi possono essere di tipo fisico o logico. Potenziali esempi di asset sono: le credenziali di accesso, i dettagli di una carta di credito, le chiavi di crittografia o i dettagli dell'utente.
Punti di Ingresso/Uscita	I punti di ingresso/uscita sono quei punti del sistema dove i dati entrano o escono. Questi vengono talvolta indicati come "punti di attacco" poiché soggetti ad uno o più potenziali attacchi. Ciascuno di questi, corrisponde ad una parte del sistema per la quale dovrebbero essere implementate opportune misure di sicurezza.
Livelli di Fiducia o di Trust	Un livello di trust viene utilizzato per definire i privilegi che un'entità esterna deve avere per accedere al sistema. Questi possono essere classificati in base ai privilegi assegnati o alle credenziali fornite e fanno riferimento a punti di ingresso/uscita di risorse protette. Possibili esempi sono l'utente anonimo del sistema, l'utente autenticato del sistema e l'amministratore. I livelli di trust vengono applicati ad ogni punto di entrata/uscita.

4.2 Modellazione e Individuazione delle minacce: Threat Modelling

I processi di sviluppo sicuro del software, analizzati nel capitolo precedente, prevedono la modellazione delle minacce e suggeriscono l'inclusione dell'attività di modellazione delle minacce nella metodologia, al fine di migliorare la pratica dell'identificazione delle vulnerabilità in materia di sicurezza del Software.

4.2.1 Introduzione e concetti base

Nella sua forma più elementare la modellazione delle minacce è un approccio strutturato che identifica potenziali minacce alla sicurezza, valutandone il rischio e fornendo le necessarie contromisure.

La modellazione delle minacce comporta l'identificazione degli asset in un processo strutturato, individuando le potenziali minacce su cui insistono, categorizzandole e determinando le opportune strategie di mitigazione.

Quando si approccia con la modellizzazione delle minacce, è importante avere una corretta comprensione della terminologia di base:

- **Asset**, è qualcosa di valore su cui un avversario pone particolare interesse. I dati presenti in un database sono un esempio di Asset.
- **Minaccia** (threat), è un evento che può o non essere dannoso all'origine ma che può danneggiare o compromettere un'attività a seguito di un attacco.
- **Vulnerabilità**, è un difetto nella sicurezza di una o più parti di un sistema che rende possibile una minaccia.
- **Attacco**, è un tentativo da parte di un avversario di sfruttare una vulnerabilità.
- **Rischio**, è la probabilità di essere bersaglio di un attacco.
- **Contromisura**, è un'azione o uno strumento che contrasta una minaccia e mitiga il rischio.

La modellazione delle minacce può essere definita come la "revisione sistematica delle caratteristiche e dell'architettura dell'applicazione da un punto di vista della sicurezza". Il processo fornisce un approccio strutturato per identificare e classificare le minacce basate sui componenti del software, sui flussi di dati e sui confini di fiducia (confini entro i quali esistono dei criteri di sicurezza).

La modellazione delle minacce è una parte importante del ciclo di vita del software che identifica le minacce che potrebbero non essere riconosciute nelle tradizionali sessioni di brainstorming sulla sicurezza.

A differenza delle tecniche di test del software come il penetration testing o fuzzing, la modellazione delle minacce può essere eseguita durante la fase di progettazione del sistema rendendola indipendente dallo sviluppo del codice. Introdotto nel ciclo di vita dello sviluppo del software, il Threat Modelling può contribuire a garantire la sicurezza già nella fase di progettazione e disegno di un'applicazione, riducendo i costi e minimizzando le necessarie successive correzioni di sicurezza nel progetto. Anche i sistemi in essere possono beneficiare di tale processo. Possono essere identificati nel sistema i problemi di sicurezza sconosciuti o non risolti e può essere applicata una classificazione del rischio alle vulnerabilità individuate. Il processo può anche essere adattato alle pratiche di sviluppo quali Agile.

4.2.2 Motivazioni nell'uso del Threat Model

4.2.2.1 Ricerca preventiva dei bug di sicurezza

Si pensi alla costruzione di una casa; le decisioni prese all'inizio del progetto impatteranno drasticamente sulla sicurezza del prodotto finale. La scelta di pareti in legno e un consistente numero di finestre al piano terra, espongono la casa a maggiori rischi rispetto ad una costruzione fatta in mattoni e con poche finestre. A seconda di dove si sta costruendo e di altri fattori, potrebbe anche essere una scelta ragionevole. Comunque, una volta effettuata la scelta, possibili futuri cambiamenti avrebbero sicuramente un impatto significativo sui costi. Certo, si possono sempre mettere delle inferiate sulle finestre, ma non sarebbe meglio concepire una soluzione più efficace e appropriata sin dall'inizio? È possibile applicare gli stessi tipi di compromessi alla tecnologia. La modellazione delle minacce aiuta a trovare problematiche di progettazione anche prima della stesura del codice e questa fase risulta essere il momento migliore per scoprire tali problemi.

4.2.2.2 Comprensione dei requisiti di sicurezza

Individuare le minacce e decidere come si intende procedere, rende chiari i requisiti. Con i requisiti più chiari, è possibile dedicare maggior energia ad un insieme consistente di funzionalità e proprietà di sicurezza. Esiste un'importante interazione tra requisiti, minacce e mitigazioni. Durante la fase di modellazione delle minacce, è possibile scoprire ad esempio, che alcune minacce non sono conformi ai requisiti aziendali e come tali potrebbero non essere prese in considerazione. Altresì i requisiti di sicurezza potrebbero dover tener conto di ulteriori minacce la cui risoluzione potrebbe però, essere troppo complessa e dispendiosa, pertanto, si dovrà scegliere tra l'indirizzare parzialmente tali minacce o accettarle comunicando che non è possibile affrontarle.

4.2.2.3 Ingegnerizzazione e rilascio di prodotti più sicuri

Considerando i requisiti e la progettazione effettuati all'inizio del processo, è possibile ridurre drasticamente le probabilità di dover ri-progettare o ricostruire il sistema, o manutenere un flusso costante di bug di sicurezza. L'effort risparmiato in tal senso potrebbe andare a favore di un processo di costruzione di un prodotto migliore, più veloce, più economico o più sicuro, lasciando spazio ad una maggiore concentrazione nella fase realizzativa dei requisiti funzionali.

4.2.3 Processo di modellazione del sistema da proteggere

Il processo di modellazione delle minacce s'identifica in un insieme di passi che realizzano dei sotto-obiettivi, piuttosto che come una singola attività. Essenzialmente, le domande da porsi al fine di identificare i sotto-obiettivi, sono:

- Cosa si sta realizzando?
- Cosa potrebbe andare storto una volta realizzato?
- Cosa è necessario fare per contrastare eventuali problemi?
- È stato svolto un lavoro di analisi accettabile?

Queste domane indirizzano puntualmente i quattro step del processo illustrato graficamente nella figura che segue e che possono essere così riassunti:

- Modellare il sistema che si sta costruendo, rilasciando o modificando.
- Identificare le minacce usando il modello e gli approcci descritti nel Paragrafo 5.2.4.
- Indirizzare le minacce utilizzando gli approcci descritti nel Paragrafo 5.3.
- Convalidare il lavoro per completezza ed efficacia sulla base delle best practices riportate nel Paragrafo 5.2.4.6.

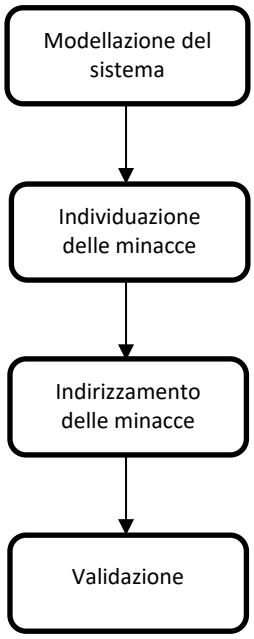


Figura 3 - I quattro step del Framework

Il framework si concretizza come una modalità strutturata per modellare le minacce.

4.2.3.1 Diagrammi DFD

I diagrammi sono un buon metodo per rappresentare ciò che si sta realizzando. Questi sono il modo più efficace per pensare a ciò che si sta costruendo. Ci sono diversi modi per diagrammare il software, e si può iniziare con un diagramma disegnato su una lavagna che rappresenta i flussi di dati e come questi attraversano il sistema.

La modellazione delle minacce si concentra sui dati, su come questi vengono fruiti (flussi) e su come si muovono tra i vari componenti del sistema. I diagrammi di modellazione forniscono una rappresentazione visiva del modo in cui i singoli sottosistemi operano e lavorano insieme. I diagrammi di flusso di dati (DFD)

vengono normalmente utilizzati in molti processi di modellazione delle minacce. Questi forniscono un modo coerente e compatto per modellare i flussi di dati presenti in un'applicazione attraverso l'utilizzo di sei forme distinte che rappresentano: il processo, i processi multipli, l'entità esterna, l'archivio dati, il flusso di dati e il perimetro privilegiato (Figura sottostante).

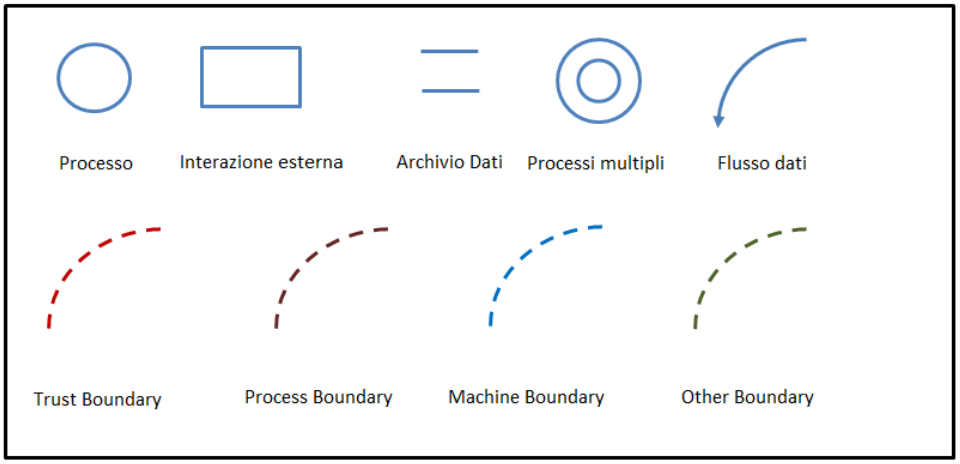


Figura 4 - Simbolismo nel Threat Modelling

I Trust boundaries (confini di fiducia) vengono identificati da una linea tratteggiata rossa, i Process Boundary (confini di processo) vengono invece identificati da una linea tratteggiata marrone, i Machine Boundary (confini della macchina) vengono identificati da una linea tratteggiata blu mentre gli altri confini vengono identificati da una linea tratteggiata verde.

L'utilizzo del colore consente ai team di riconoscere facilmente durante la valutazione del DFD del sistema quale tipo di confine il flusso di dati attraversa. Qui alcuni esempi di elementi che costituisco il DFD:

- Processo o Processi multipli
  - File di libreria dinamica
  - File eseguibile
  - Componente SW/FW
  - Web Service
  - Assemblies
  - Etc.
- Interazione o Entità esterna
  - Persona
  - Altro sistema
  - Sito web
  - Etc.
- Archivio dati
  - Database
  - File
  - Registro
  - Memoria condivisa
  - Code e Stack
  - Etc.
- Flusso dati
  - Chiamata a funzione

  - Traffico di rete
  - Etc.
- Trust boundaries
  - Perimetro del processo
  - File system
  - Etc.

Segue una tabella riepilogativa in cui, per ciascun elemento DFD viene riportato l'aspetto che assume nel diagramma, il significato ed alcuni brevi esempi:

ELEMENTO DFD	ASPETTO	SIGNIFICATO	ESEMPIO
Processo	Rettangolo arrotondato , cerchio o cerchio concentrico	Qualunque codice in esecuzione	Codice scritto in C, C#, Python o PHP, etc.
Flusso dati	Freccia	Comunicazione tra processi o tra processi e archivi di dati	Connessioni di rete, HTTP, RPC, LPC, etc.
Archivio dati	Due linee parallele con etichetta nel mezzo	Supporti di memorizzazione dati	File, database, registro di Windows, segmenti di memoria condivisi.
Entità esterna	Rettangolo con angoli retti	Persone o codice al di fuori del nostro controllo	Un utente, un sito web esterno.

Tabella 4 - Caratteristiche degli elementi DFD

L'impiego di diverse tipologie di diagramma concepiti come diversi blocchi di costruzione aiutano a modellare ciò che si sta realizzando.

Nell'esempio che segue, viene rappresentato un modello di una semplice applicazione web che implementa una su logica di business che interagisce con un browser web, con un server web e con un database (vedi figura a seguire).

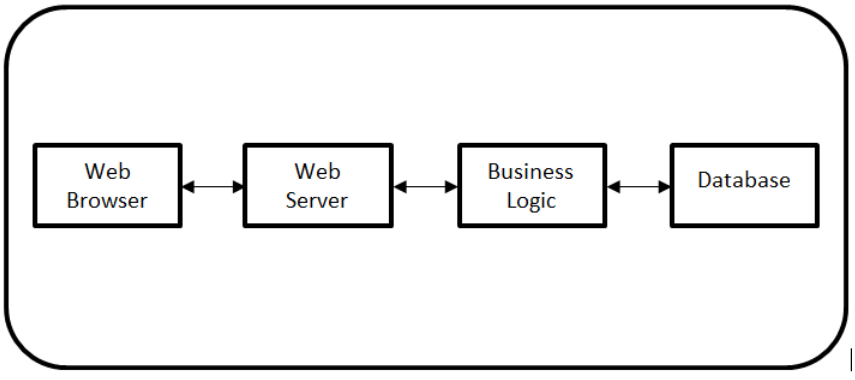


Figura 5 - Diagramma del sistema

Un modo semplice per migliorare il diagramma consiste nel delineare i confini per dare evidenza di "chi controlla cosa". Si può facilmente comprendere che le minacce che attraversano questi confini sono probabilmente le più importanti e possono essere un buon punto di partenza nel processo di identificazione. Questi confini prendono il nome di "trust boundaries" (confini di fiducia) e dovrebbero essere sicuramente disegnati ovunque esiste un controllo da parte di persone. Alcuni esempi significativi:

- Account (User ID sui sistemi Unix o i Security Identifiers su sistemi Microsoft Windows);
- Interfacce di rete;
- Macchine fisiche;
- Macchine virtuali;
- Perimetri organizzativi;
- Ovunque possa essere messa in discussione la diversificazione dei privilegi.

Nel diagramma riportato nella Figura che segue, si aggiungono i "trust boundaries" (rappresentati da rettangoli tratteggiati) e la descrizione di ciò che il perimetro contiene:

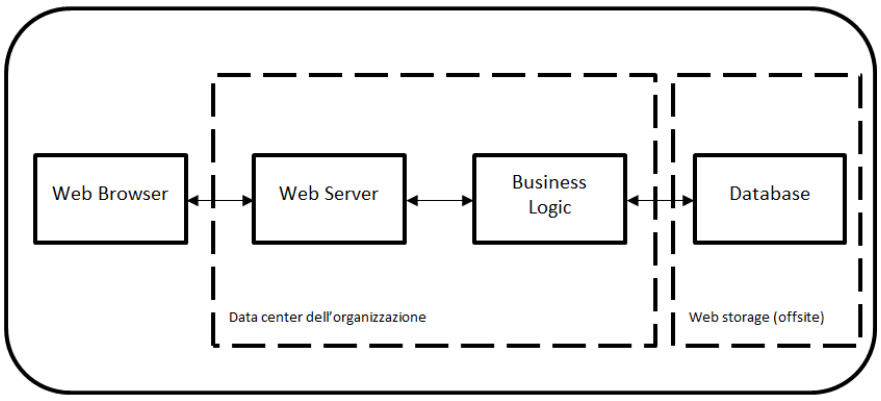


Figura 6 - Aggiunta dei "Trust boundaries" al diagramma

Quando il diagramma diventa più grande e più complesso, può essere molto utile numerare ogni processo, flusso dati e archivio dati presenti nel diagramma, come mostra la figura che segue (Ciascun "trust boundary" dovrebbe avere un identificativo univoco in rappresentanza del suo contenuto):

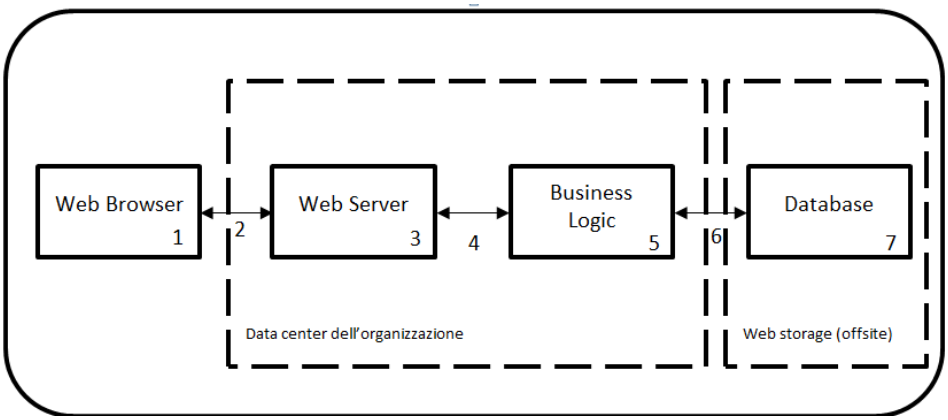


Figura 7 - Numerazione degli elementi del diagramma

Si deve pensare al diagramma del modello come parte integrante del processo di sviluppo, quindi deve essere messo sotto il controllo di versionamento così come tutto il resto del materiale relativo al progetto. A partire da questo modello, si procede con l'individuazione delle minacce sulla base delle metodologie e delle tecniche descritte nel paragrafo successivo.

4.2.4 Tecniche di modellazione e individuazione delle minacce

L'obiettivo che tutte le metodologie di modellazione delle minacce condividono è lo sviluppo di un processo di passi iterativi che un team può facilmente seguire durante la valutazione di un sistema software.

4.2.4.1 Microsoft SDL – STRIDE

La STRIDE è un processo metodologico che aiuta ad individuare le minacce di sicurezza in un sistema complesso. L'elemento mnemonico STRIDE è acronimo di Spoofing, Tampering, Repudiation, Information disclosure, Denial of service e Elevation of privilege.

Si riporta a seguire la descrizione delle classi di minacce previste:

- **Spoofing** (falsificazione di identità): è la pretesa di essere qualcos'altro o qualcun altro che non si è. Classifica l'insieme delle minacce che permettono ad un attaccante di interagire con il sistema utilizzando un'altra identità. Per esempio, un server di phising che pretende di essere il server della nostra banca.
- **Tampering** (alterazione dei dati): è l'alterazione di qualcosa che si presuppone non sia oggetto di modifica. Ciò può includere pacchetti di rete (sia fissa che mobile), dati persistiti su supporto di massa o in memoria nonché il codice applicativo. Classifica l'insieme delle minacce che permettono di modificare in modo fraudolento, dati e codice delle applicazioni. Per esempio, un attaccante sfruttando un bug software riesce a cambiare il codice di un'applicazione per aprire una back-door nel sistema.
- **Repudiation** (ripudio di una azione): significa dichiarare di non aver fatto qualcosa (indipendentemente dal fatto che sia stato fatto o meno). Classifica l'insieme delle minacce che permettono ad un attaccante di negare di aver compiuto un'azione sul sistema. Per esempio, un utente compie un'azione illegale sul sistema e il sistema non è in grado di rilevare l'azione o identificare l'utente.
- **Information Disclosure** (divulgazione di informazioni): riguarda l'esposizione delle informazioni a persone non autorizzate alla loro visione. Classifica l'insieme delle minacce che provocano l'esposizione di informazioni ad utenti/individui a cui non è consentito l'accesso in lettura/scrittura. Per esempio, un utente legge un file per il quale non ha ricevuto i diritti di lettura oppure un attaccante legge i dati in transito sulla rete.
- **Denial of Service** (diniego di servizio): sono attacchi designati all'interruzione del servizio da parte dei sistemi. Questi includono come effetto il crashing, il rallentamento che porta alla non usabilità del sistema e il riempimento degli storage. Classifica l'insieme delle minacce che permettono di negare o degradare la fornitura di un servizio. Per esempio, un attaccante invia molti pacchetti al fine di intasare la banda di rete di un server il quale non potrà a sua volta essere contattato e/o fornire i suoi servizi agli utenti.
- **Elevation of Privilege** (elevazione dei privilegi): avviene quando un programma o un utente è tecnicamente abilitato a fare cose che si presuppone non debba fare. Classifica l'insieme delle minacce che permettono ad un utente di ottenere privilegi non previsti per il suo ruolo. Per esempio, un utente anonimo sfrutta un bug software per ottenere i privilegi di amministratore.

Partendo dal diagramma (Figura 7):

- Come si fa a sapere se il browser web verrà utilizzato solo dalle persone che ci si aspetta?
- Cosa accade se qualcuno altera i dati presenti nel database?
- È corretto far transitare i dati da una componente all'atra del sistema senza che questi vengano cifrati?

Questi sono esattamente e rispettivamente esempi di spoofing, tampering e information disclosure che possono essere facilmente individuati con l'ausilio della STRIDE. Con una scarsa conoscenza della sicurezza,

ma con l'impiego delle giuste tecniche, è possibile trovare le minacce più importanti in modo veloce e con maggiore affidabilità. Se si sta impiegando un processo di modellizzazione delle minacce, la documentazione prodotta da tale processo, può aumentare il livello di fiducia nel realizzare un software più sicuro.

Per ciascuna minaccia vengono evidenziati gli elementi del diagramma su cui impattano (normalmente si ha maggiore impatto sul software, sui flussi dati o gli storage piuttosto che sui trust boundary).

La lista che segue fornisce alcuni esempi di minacce per ciascuna categoria (l'elenco non vuole essere esaustivo):

SPOOFING	Qualcuno potrebbe fingere di essere un altro utente del nostro sistema, quindi serve un modo per autenticare tutti gli utenti.
	Qualcuno potrebbe fingere di essere il nostro sito web, quindi è necessario assicurarsi di avere un certificato SSL e di utilizzare un singolo dominio per tutte le nostre pagine (per aiutare quel sottoinsieme di utenti che leggono gli URL per vedere se sono nel posto giusto).
	Qualcuno potrebbe mettere un collegamento nascosto in una delle nostre pagine, ad esempio logout.html o placeorder.aspx. Dobbiamo controllare il campo HTTP "Referer" prima di intraprendere qualsiasi azione. Non è una soluzione definitiva per contrastare il CSRF (Cross Site Request Forgery), ma è un inizio.

TAMPERING	Qualcuno potrebbe manomettere i dati del back-end.
	Qualcuno potrebbe manomettere i dati in transito tra il data center e il consumer.
	Chi sviluppa potrebbe rilasciare il codice del front-end dell'applicazione senza verificarlo, pensando che sia in fase di caricamento nell'area di staging. Ciò consentirebbe ad uno sviluppatore malintenzionato di aggiungere codice malevolo.

REPUDIATION	Le azioni precedenti potrebbero richiedere una attenta analisi per comprendere ciò che è accaduto. E' necessario porsi delle domande quali: Esistono i log di sistema? Nel log di sistema vengono registrate le giuste informazioni? Il log di sistema è protetto dal tampering?

INFORMATION DISCLOSURE	Cosa accade se qualcuno legge i dati presenti nel database? E' possibile che qualcuno possa connettersi al database per leggere o scrivere informazioni?

DENIAL OF SERVICE	Cosa accade se migliaia di utenti si connettono contemporaneamente alla nostra applicazione? E se il sistema va giù?

ELEVATION OF PRIVILEGE	Magari il front-end è l'unico punto di accesso al nostro sito da parte degli utenti, ma cosa lo impone? Cosa previene l'accesso diretto da parte degli utenti alla logica di business in esecuzione sul server o al caricamento di un nuovo codice? Se è presente un firewall, questo è stato correttamente configurato? Chi controlla l'accesso al database, o cosa accade se un impiegato commette un

	errore o premeditatamente modifica i file a livello di configurazione?

Microsoft ha inizialmente previsto l'applicabilità della STRIDE solo per i punti di ingresso/uscita del sistema in analisi. Tale approccio è stato poi affinato applicando la STRIDE a tutti gli elementi DFD del modello.

Segue una tabella che indica l'esposizione in termini di vulnerabilità secondo la STRIDE rispetto agli elementi DFD utilizzati nel processo di modellazione:

Elemento DFD	S	T	R	I	D	E
Entità Esterna	X		X			
Flusso Dati		X		X	X	
Archivio Dati		X	*	X	X	
Processo	X	X	X	X	X	X

Tabella 5 - STRIDE per elemento DFD

Le minacce STRIDE sono esattamente l'opposto di alcune delle proprietà che il nostro sistema dovrebbe avere, ovvero: autenticità, integrità, non ripudio, riservatezza, disponibilità e autorizzazione. La seguente tabella, mostra la correlazione tra la categoria di minacce STRIDE, la corrispettiva proprietà che si desidera mantenere, e gli elementi del sistema che sono tipicamente esposti alla classe di minacce indicata.

MINACCIA	PROPRIETA' VIOLATA (requisito o controllo di sicurezza)	TIPICA VITTIMA
	Autenticazione	Processo Entità esterna Persona
	Integrità e Privacy (*)	Processo Archivio dati Flusso dati
	Non ripudio (*)	Processo
	Confidenzialità e Privacy (*)	Processo Archivio dati Flusso dati
	Disponibilità e Privacy (*)	Processo Archivio dati Flusso dati
	Autorizzazione	Processo

Tabella 6 - STRIDE proprietà violate

(*) Articoli 25 e 32 del GDPR.

Ad esempio, alcuni elementi quali un flusso di dati tra due componenti, un processo o un archivio di dati; in base alla classificazione STRIDE sono potenzialmente vulnerabili a manomissioni, divulgazione di informazioni e negazione del servizio. Il processo di modellazione delle minacce prende in considerazione ogni minaccia per elemento e valuta se esistono tecniche di mitigazione adeguate per quel tipo di minaccia. Analizzando ciascun elemento del sistema come identificato nei DFD, viene creato un profilo di minaccia dell'applicazione.

4.2.4.1.1 Tecniche di mitigazione

Ogni minaccia viene mitigata o accettata. Per i non esperti di sicurezza, la STRIDE fornisce per ogni tipo di potenziale minaccia identificata una o più classificazioni delle tecniche di mitigazione da mettere in campo (vedi tabella che segue).

Tipo Minaccia	Tecnica di mitigazione o controlli di sicurezza
	Autenticazione
	Integrità
	Servizi di non ripudio
	Confidenzialità
	Disponibilità
	Autorizzazione

Tabella 7 - Tecniche di mitigazione

AUTENTICAZIONE – TECNICHE DI MITIGAZIONE DELLO SPOOFING

Le tecnologie per l'autenticazione di computer (o account di computer) includono quanto segue:

- IPSec,
- DNSSEC,
- SSH host keys,
- Kerberos authentication,
- HTTP Digest o Basic authentication,
- "Windows authentication" (NTLM),
- Sistemi PKI, come SSL o TLS con certificati.

Le tecnologie per l'autenticazione dei flussi a livello di bit (file, messaggi, ecc.) includono quanto segue:

- Digital signatures,
- Hashes.

I metodi per l'autenticazione delle persone possono coinvolgere uno qualsiasi dei seguenti elementi:

- Qualcosa che sai, come ad esempio una password,
- Qualcosa che hai, come una card di accesso,
- Qualcosa che sei, come ad esempio un dispositivo biometrico, comprese le fotografie,

- Qualcuno che conosci e che può autenticarti.

Le tecnologie per mantenere l'autenticazione tra le connessioni includono quanto segue:

- Cookies.

INTEGRITA' – TECNICHE DI MITIGAZIONE DEL TAMPERING

Le tecnologie per la protezione degli asset includono:

- ACLs o permissions,
- Digital signatures,
- Hashes,
- Windows Mandatory Integrity Control (MIC) feature,
- Unix immutable bits.

Le tecnologie per la protezione del traffico di rete:

- SSL,
- SSH,
- IPSec,
- Digital signatures.

NON RIPUDIO – TECNICHE DI MITIGAZIONE DELLA REPUDIATION

Le tecnologie che è possibile utilizzare per affrontare il problema del ripudio includono:

- Logging,
- Log analysis tools,
- Secured log storage,
- Digital signatures,
- Secure time stamps,
- Trusted third parties,
- Hash trees,
- Strumenti per la prevenzione delle frodi.

CONFIDENZIALITA' – TECNICHE DI MITIGAZIONE DELLA INFORMATION DISCLOSURE

Le tecnologie per la riservatezza includono:

- Protezione dei files:
  - ACLs/permissions,
  - Encryption,
  - Appropriata gestione delle keys.
- Protezione dei dati di rete:
  - Encryption,
  - Appropriata gestione delle keys.
- Protezione della comunicazione e delle headers di comunicazione:
  - Mix networks,
  - Onion routing,
  - Steganography.

DISPONIBILITA' – TECNICHE DI MITIGAZIONE DEL DENIAL OF SERVICE

Le tecnologie per la protezione degli asset includono:

- ACLs,
- Filters,
- Quotas (rate limiting, thresholding, throttling),
- High-availability design,
- Extra bandwidth (rate limiting, throttling),
- Cloud services.

AUTORIZZAZIONE – TECNICHE DI MITIGAZIONE DELL'ELEVATION OF PRIVILEGE

Le tecnologie per migliorare l'autorizzazione includono:

- ACLs,
- Group or role membership,
- Role based access control,
- Claims-based access control,
- Windows privileges (runas),
- Unix sudo,
- Chroot, AppArmor o altre unix sandboxes,
- The "MOICE" Windows sandbox pattern,
- Convalida degli input per uno scopo definito.

4.2.4.1.1.1 Indirizzamento dello Spoofing

La Tabella seguente elenca gli obiettivi di spoofing, le strategie di mitigazione per indirizzare lo spoofing e le tecniche per attuare tali mitigazioni:

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Spoofing di una persona	Identificazione e autenticazione (username e qualcosa che conosci/hai/sei)	Username, nomi reali, identificativi: • Password; • Token; • Biometria. Registrazione/Manutenzione/Scadenza.
Spoofing di un file su disco	Sfruttare il Sistema Operativo	• Percorsi assoluti; • Controllo ACL; • Assicurarsi che le pipes vengano create correttamente.
	Autenticatori crittografici	Firme digitali o autenticatori.
Spoofing di un indirizzo di rete	Crittografia	• DNSSEC; • HTTPS/SSL; • IPsec.
Spoofing di un programma in memoria	Sfruttare il Sistema Operativo	Molti sistemi operativi moderni hanno una qualche forma attuabile di identificazione dell'applicazione.

Tabella 8 - STRIDE: Indirizzamento dello Spoofing

Spoofing di una persona. Per evitare lo spoofing di una persona, è necessario che sia stato implementato un meccanismo di autenticazione e che a questa persona sia stato associato un nome utente univoco.

Spoofing di un file su disco. Quando si accede ad un file presente sul disco, non bisogna aprirlo utilizzando un percorso relativo come *open(file)*. Utilizzare il percorso assoluto *open(/percorso/assoluto/del/file)*. Se il file contiene dati sensibili, dopo averlo aperto, è necessario attuare un controllo di sicurezza sugli elementi del descrittore del file stesso (come il fully resolved name, i permessi e l'owner). Si potrebbe anche voler controllare il descrittore del file per evitare eventuali *race conditions.* Ciò vale doppiamente quando il file è un eseguibile, ma il controllo dopo l'apertura potrebbe essere complicato. Ciò, può aiutare a garantire che le autorizzazioni sull'eseguibile non possano essere modificate da parte di un utente malintenzionato. In ogni caso, è quasi sempre sconsigliabile invocare *exec()* con il parametro file specificato in modo relativo *./file*.

Spoofing di un indirizzo di rete. Nel caso di spoofing di un indirizzo di rete, è consigliabile l'impiego di protocolli come DNSSEC, SSL, IPsec o una combinazione di questi per assicurarsi di colloquiare con la controparte attesa.

Spoofing di un programma in memoria. Si tratta di programmi che si nascondono mostrandosi come programmi legittimi ma con l'intento di fare danni o trafugare informazioni. Questi sono un tipo di Trojan, i quali duplicano le azioni di processi esistenti che vengono poi eseguite inconsapevolmente dall'utente. E' necessario tenere sempre aggiornato il software come il sistema operativo e il browser web. Mantenere la connessione a Internet il più sicura possibile, tenendo sempre attivo un firewall. Sia i firewall hardware che software sono eccellenti strumenti per controllare il traffico Internet dannoso. E' opportuno installare inoltre un software antivirus o un Trojan remover.

4.2.4.1.1.2 Indirizzamento del Tampering

La Tabella seguente mostra in elenco gli obiettivi di tampering, le strategie di mitigazione per indirizzare il tampering e le tecniche per attuare tali mitigazioni.

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Tampering di un file	Sfruttare il Sistema Operativo	ACLs.
	Crittografia	• Firme digitali; • Keyed MAC.
Concorrenza nella creazione di un file (tampering del file system)	Utilizzo di una directory protetta da manipolazione arbitraria di un utente	• ACLs; • Utilizzo di strutture private di directory; • La randomizzare dei nomi dei file contrasta l'esecuzione di possibili attacchi.
Tampering dei pacchetti di rete	Crittografia	• HTTPS/SSL; • IPsec.
	Anti-pattern	Isolamento della rete.

Tabella 9 - STRIDE: Indirizzamento del Tampering

Tampering di un file. Se un attaccante possiede un account su una macchina, questo può facilmente portare un attacco di tampering su di un file che risiede sulla stessa macchina, oppure quando questo transita sulla rete per essere ricevuto.

Tampering della memoria. Ciò avviene quando un processo con privilegi minimi di trust o non, altera in qualche modo la memoria fisica della macchina. Per esempio, se si stanno prendendo dati da un segmento di memoria condivisa, esistono delle ACL tali da consentirne agli altri processi la sola lettura? Per le applicazioni web che acquisiscono dati tramite AJAX, assicurarsi di validare tali dati prima di darli in pasto alla logica di business.

Tampering del traffico di rete. La prevenzione del traffico di rete richiede una gestione sia dello spoofing che del tampering. Diversamente, qualunque intenzionato ad alterare i dati in transito potrebbe semplicemente far finta di essere all'altra estremità, portando invece un attacco di tipo MITM (Man In The Midle). La soluzione più comune per contrastare questo problema è utilizzare il protocollo SSL o l'IPsec (IP Security) come infrastruttura di comunicazione. Entrambi, SSL e IPsec indirizzano le problematiche legate alla confidenzialità e all'alterazione di informazioni e possono anche aiutare ad indirizzare lo spoofing.

Tampering del traffico di rete attraverso l'anti-pattern. L'isolamento della rete non assicura la protezione dalle minacce di tampering poiché generalmente non si riesce a mantenere la rete costantemente isolata.

4.2.4.1.1.3 Indirizzamento della repudiation

Indirizzare la repudiation in genere significa garantire che il sistema venga progettato prevedendo il tracciamento e la registrazione delle operazioni svolte (logging), garantendo inoltre che, tali registrazioni vengano protette e preservate. Alcuni di questi registri possono essere gestiti utilizzando un trasporto affidabile specifico. In questo senso, il syslog su UDP presenta forti lacune in termini di sicurezza; il syslog su TCP / SSL invece è notevolmente migliore.

La Tabella seguente mostra in elenco gli obiettivi di repudiation, le strategie di mitigazione per indirizzare la repudiation e le tecniche per attuare tali mitigazioni:

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Non utilizzare un meccanismo di log vuol dire non poter provare nulla.	Log	Assicurarsi di tracciare tutte le informazioni rilevanti dal punto di vista della sicurezza.
Esposizione dei Logs a eventuali attacchi	Proteggere i logs	• syslog su TCP / SSL; • ACL.
Il Log come canale di attacco	Informazioni dettagliate sul log	Documentare la progettazione del log sin dall'inizio del processo di sviluppo.

Tabella 10 - STRIDE: Indirizzamento della repudiation

Non avere un log significa non poter provare nulla. Prevedere e mantenere i log, significa, poter investigare su quanto accaduto, al fine di acquisire un valido riscontro, nel momento in cui qualcuno nega di aver ottenuto o fatto qualcosa.

Esposizione del log a possibili attacchi. Eventuali aggressori faranno del tutto per invalidare le informazioni contenute nel log, contrastando a volte l'operazione stessa di scrittura o forzando il "roll over" del registro, con il fine di rendere difficile l'individuazione dell'attacco. Questi inoltre, possono agire in modo tale da disattivare gli allarmi, facendo sì che, il vero attacco non venga alla luce.

Il log come canale di attacco. Da progettazione, è possibile che vengono raccolti dati provenienti da sorgenti 'malevoli' fuori dal nostro controllo, fornendo poi gli stessi dati a persone o sistemi che hanno dei privilegi di sicurezza. Un esempio di questa tipologia di attacco potrebbe essere l'invio di una e-mail indirizzata a "</html> destinatario@dominio.com", che causa problemi a quegli strumenti web-based che non prevedono l'HTML inline.

È possibile rendere il tutto più semplice scrivendo codice sicuro nell'elaborazione dei dati di log, dando chiara evidenza di ciò che questi non possono contenere, ad esempio "I dati di log sono tutti in chiaro, ed eventuali aggressori potrebbero inserire ciò che vogliono" oppure "I campi da 1 a 5 del tracciato del log sono sotto stretto controllo da parte del nostro software, mentre i campi da 6 a 9 sono facilmente iniettabili. Il campo 1 è un campo time GMT. I campi 2 e 3 sono indirizzi IP (v4 o v6) ... ".

4.2.4.1.1.4 Indirizzamento dell'information disclosure

La Tabella seguente mostra in elenco gli obiettivi dell'information disclosure, le strategie di mitigazione per indirizzare l'information disclosure e le tecniche per attuare tali mitigazioni:

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Monitoraggio della rete	Crittografia	• HTTPS/SSL; • IPsec.
Directory o nomi di file (per esempio "lettere-di-licenziamento/" o "NomeCognome.docx")	Sfruttare il Sistema Operativo	ACLs.
Contenuti di un file	Sfruttare il Sistema Operativo	ACLs.
	Crittografia	Crittografia dei file come PGP, crittografia del disco (FileVault, BitLocker).
API information disclosure	Progettazione	Attento controllo nella progettazione, considerando il passaggio dei parametri per indirizzo o valore.

Tabella 11 - STRIDE: Indirizzamento dell'Information disclosure

Monitoraggio della rete. Il processo di monitoraggio della rete, si avvale dell'architettura adottata nel realizzare la maggior parte delle reti, per monitorarne il traffico (In particolare, la maggior parte delle reti attuali inviano i pacchetti sulla rete e ciascun listener presente su ogni end-point deve decidere se il pacchetto è per lui importante o meno). Quando le reti vengono progettate in modi diversi, esistono svariate tecniche per tracciare il traffico che va verso o attraversa la stazione di monitoraggio. Se non viene indirizzato lo spoofing, così come il tampering, un attaccante può mettersi nel mezzo attuando lo spoofing su ciascun end-point. La mitigazione dell'information disclosure sulla rete richiede la gestione delle minacce di spoofing e di tampering. Se non si indirizza il tampering, ci sono diversi modi intelligenti per ottenere informazioni. Quindi di nuovo, l'SSL e l'IPSec sono le migliori scelte da mettere in campo.

Nomi che rivelano informazioni. Quando il nome di una directory o di un file di per se forniscono informazioni utili ad un possibile attaccante, il modo migliore per proteggersi è creare una directory padre con un nome innocuo e utilizzare le ACLs o i permessi del sistema operativo per proteggerle.

Contenuti sensibili nei file. Quando il contenuto di un file necessita di protezione, utilizzare le ACLs o la crittografia. Nel caso in cui la macchina (computer) dovesse cadere in mani non autorizzate, è necessario preventivamente utilizzare la crittografia al fine di proteggere tutti i dati. Le modalità di protezione crittografica che prevedono l'inserimento di una chiave o parola chiave da parte di una persona, sono più sicure ma meno convenienti. Esistono tecniche crittografiche per i file, filesystem e database, dipende da ciò che si deve proteggere.

API (Interfaccia di programmazione di una applicazione). Quando si progetta un'API, o diversamente si trasmettono informazioni oltre un confine di fiducia, è importante fare attenzione a quali informazioni si divulgano. È necessario partire dal presupposto che le informazioni fornite vengono passate ad altri, quindi

bisogna essere molto cauti e selettivi su ciò che viene fornito. Situazioni di errore generate da un sito web che mostrano il nome utente e la password di un database sono un esempio comune di questo problema.

4.2.4.1.1.5 Indirizzamento del denial of service

La Tabella seguente mostra in elenco gli obiettivi del Denial Of Service, le strategie di mitigazione per indirizzare il Denial Of Service e le tecniche per attuare tali mitigazioni:

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Saturazione della rete **(Network flooding)**	Verificare le risorse esauribili	• Risorse flessibili (Elastic resources); • Progettare pensando che il consumo di risorse da parte di un futuro utilizzatore malintenzionato possa essere alto o comunque superiore a quello presunto.
		ACLs di rete.
Risorse utilizzate da un programma	Progettazione attenta e cautelativa	Gestione di risorse flessibili.
	Evitare i moltiplicatori (fattori moltiplicativi)	Analizzare i punti in cui un eventuale attacco portato con uno sforzo minimo, potrebbe produrre una moltiplicazione nel consumo di CPU. Intervenire in modo tale da rendere più arduo il lavoro dell'attaccante abilitandone l'identificazione, come i client che applicano la crittografia o il login prima di procedere con il vero lavoro (ovviamente ciò non vuol dire che gli accessi non debbano essere crittografati).
Risorse di sistema	Sfruttare il Sistema Operativo	Usare le impostazioni del sistema operativo.

Tabella 12 - STRIDE: Indirizzamento del Denial of Service

Saturazione della rete. Se si dispone di strutture statiche di connessioni, cosa accade se queste si saturano? L'utilizzo di firewall può fornire un livello di protezione basato su ACLs per controllare l'accettazione (o l'invio) di dati e possono essere utili per mitigare gli attacchi di negazione di servizio di rete.

Individuazione delle risorse esauribili. Si possono identificare tre tipologie distinte di risorse esauribili: la prima è quella relativa alle risorse di rete; la seconda è quella relativa a quelle risorse direttamente gestite lato codice; la terza è quella relativa alle risorse gestite dal sistema operativo. In ogni caso, le risorse flessibili (elastic resources) risultano sempre essere una tecnica preziosa. Ad esempio, negli anni 90 alcuni stack TCP avevano un limite hardcoded di cinque connessioni TCP semiaperte (una connessione semiaperta è una connessione che viene aperta nel processo di avvio. Non bisogna preoccuparsi del fatto che ciò potrebbe non avere un senso, piuttosto bisognerebbe chiedersi il motivo per cui questo limite fu impostato a cinque). Oggi è spesso possibile ottenere risorse flessibili dai vari tipi di cloud provider.

Risorse di sistema. I sistemi operativi tendono ad avere limiti o quote per controllare il consumo di risorse a livello di codice utente. Considerare le risorse gestite dal sistema operativo, come la memoria o l'utilizzo del disco. Se il nostro codice viene eseguito su server dedicati, può essere ragionevole consentirgli di utilizzare

tutte le risorse di cui quel server dispone. Prestare attenzione a porre gli opportuni limiti al codice e assicurarsi di documentare ciò che si sta facendo.

Risorse del programma. Acquisire consapevolezza circa le limitazioni che si potrebbero avere nella gestione delle risorse applicative. Ad esempio, un attaccante potrebbe portarci ad un dispendio di lavoro e risorse inviando un flusso dati che richiedono costose operazioni crittografiche che potrebbero esporci a vulnerabilità di "Denial Of service".

4.2.4.1.1.6 Indirizzamento dell'elevation of privilege

La Tabella seguente mostra in elenco gli obiettivi dell'Elevation Of Privilege, le strategie di mitigazione per indirizzare l'Elevation Of Privilege e le tecniche per attuare tali mitigazioni.

OBIETTIVO DELLA MINACCIA	STRATEGIA DI MITIGAZIONE	TECNICA DI MITIGAZIONE
Confusione tra dati/codice	Usa strumenti e architetture che separano dati dal codice.	• Prepared Statement o Stored procedure per l'SQL; • Separatori chiari con forme canoniche; • Validare i dati prima di passarli al consumer.
Attacchi di corruzione del flusso di controllo e della memoria	Utilizzare un linguaggio di programmazione sicuro	Utilizzare un linguaggio di programmazione sicura nella stesura del codice ci protegge da intere classi di attacco.
	Sfruttare il sistema operativo per la protezione della memoria.	I sistemi operativi di ultima generazione possiedono meccanismi intrinsechi di protezione della memoria.
	Utilizzare il sandboxing	• I sistemi operativi moderni supportano il sandboxing in vari modi (AppArmor su Linux, AppContainer o il modello MOICE su Windows, Sandboxlib su Mac OS). • Non utilizzare per l'esecuzione l'account "nobody", crearne uno nuovo per ogni applicazione. Postfix e QMail sono buoni esempi del modello di un account per funzione.
Attacchi di command-injection	Fare attenzione	Validare l'input in termini di forma e dimensione attesi. Non sanitizzare. Tracciare l'input nel log e scartarlo se non viene riconosciuto.

Tabella 13 - STRIDE: Indirizzamento dell'Elevation of privilege

Confusione tra dati/codice. Accade spesso che i dati vengono trattati come codice. Attacchi come gli XSS sfruttano l'unione tra codice HTML e dati (un file html che contiene sia codice, come Javascript, che dati, come il testo da visualizzare e talvolta anche istruzioni di formattazione per il testo stesso). Esistono alcune strategie per affrontare questo problema. Il primo consiste nell'adottare modalità/strumenti che aiutano a mantenere separati codice e dati (ad esempio, i prepared statement in SQL indicano al database quali dichiarazioni aspettarsi e dove saranno posizionati i dati). Un'altra strategia è validare i dati prima di passarli. Ad esempio, se si stanno inviando dati ad una pagina web, è necessario assicurarsi che questi non contengano caratteri come <, >, # o & e quant'altro.

Attacchi di corruzione del flusso di controllo e/o della memoria. Questo insieme di attacchi generalmente sfrutta il "weak typing" e le strutture statiche presenti nei linguaggi simili al C per consentire ad un aggressore di introdurre del codice per poi eseguirlo. Se si utilizza un linguaggio sicuro, come Java o C#, molti di questi attacchi sono più difficili da portare. I sistemi operativi più moderni tendono a incorporare funzioni di protezione della memoria e di randomizzazione, come ad esempio l'Address Space Layout Randomization (ASLR). A volte queste funzioni sono opzionali e richiedono un compilatore o un linker switch. In molti casi, queste funzioni possono essere utilizzate gratuitamente e pertanto dovrebbero essere utilizzate. Ciò non è del tutto indolore, potrebbe essere necessario ricompilare, testare o fare altri piccoli interventi. L'ultima serie di controlli per contrastare la corruzione della memoria sono le sandbox. Le Sandbox sono funzioni del sistema operativo progettate per proteggere da un programma danneggiato il sistema operativo stesso e il resto dei programmi dell'utente in esecuzione su di esso.

Attacchi di command-injection. Gli attacchi di command-injection (iniezione di comando) sfruttano i dati di input come vettore di attacco (un aggressore fornisce un carattere di controllo, seguito da una serie di comandi). Per esempio, nella SQL injection, un apice chiude spesso un'istruzione dinamica SQL e quando si tratta di script shell unix, la shell può interpretare un punto e virgola come la fine dell'input, prendendo come comando qualsiasi cosa viene dopo.

4.2.4.2 Attack tree

L'attack tree rappresenta un'ulteriore metodologia per raccogliere e documentare i potenziali attacchi in modo strutturato e gerarchico.

Un attack tree è modellato attraverso una struttura ad albero i cui elementi base sono:

- Il nodo radice che rappresenta *l'obiettivo finale* dell'attaccante,
- I nodi figli che rappresentano i *sotto-goals* che concorrono al raggiungimento dell'*obiettivo finale*,
- Le foglie che rappresentano gli *attacchi*.

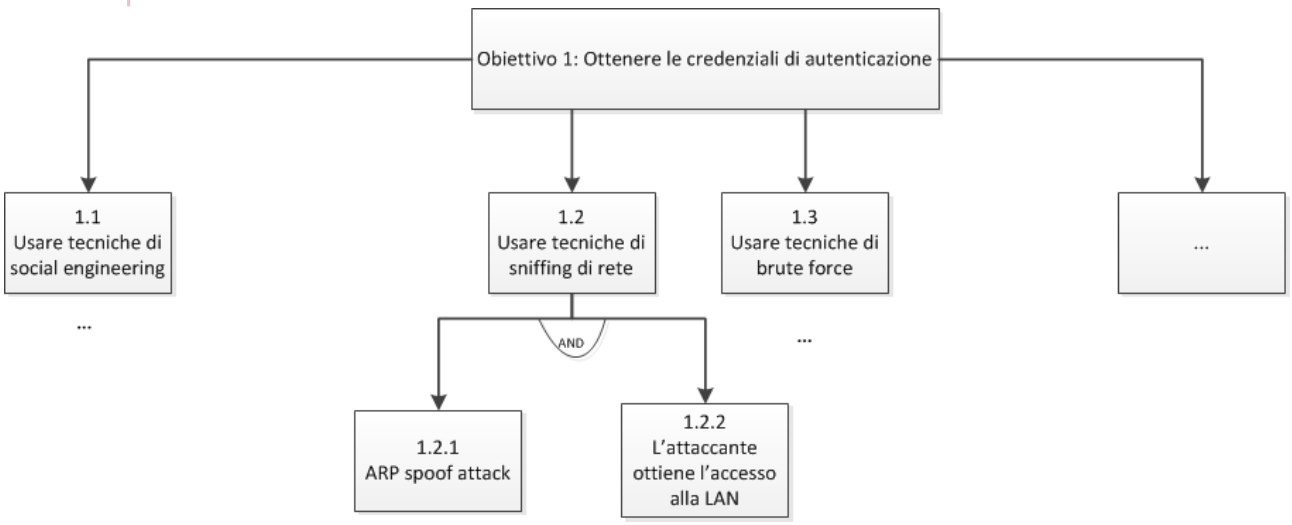
La modellazione segue la seguente logica di base:

- I "nodi OR" indicano le alternative, ossia modi indipendenti, per raggiungere un goal/sotto-goal.
- I "nodi AND" indicano i passi che concorrono al raggiungimento dello stesso goal/sotto-goal.

PASSO 1 – Modellazione degli attacchi

Si modellano quindi, tanti attack tree quanti sono gli obiettivi di un attaccante. In questa fase l'analista deve identificare i possibili obiettivi di attacco e, per ciascuno di essi, i passi attraverso cui realizzarli (la capacità di identificazione degli obiettivi e dei passi è quindi assolutamente soggettiva).

Si riporta di seguito, un esempio di modellazione di attack tree:



Nell'esempio in figura:

- Il nodo radice rappresenta l'obiettivo finale dell'attaccante che consiste nell' "Ottenere le credenziali di autenticazione",
- L'obiettivo finale può essere raggiunto in vari modi ("OR"):

  1.1 - "Usare tecniche di social engineering",

  1.2 - "Usare tecniche di sniffing di rete",

  1.3 - "Usare tecniche di brute force", ecc.

- Sulla base della modalità di Livello-1 scelta, saranno diversi gli elementi (nodi foglie) che concorreranno al raggiungimento dell'obiettivo radice. Ad esempio, nel caso si scelga il percorso 1.2 - "Usare tecniche di sniffing di rete" l'obiettivo si raggiunge (sub-goal) attraverso 2 step correlati ("AND"):
  1.2.1 - "ARP spoof attack";
  1.2.2 - "L'attaccante ottiene l'accesso alla LAN".

Lo stesso attack tree può anche essere rappresentato in forma testuale:

Obiettivo 1 - Ottenere le credenziali di autenticazione

1.1 - Usare tecniche di social engineering OR

…

1.2 - Usare tecniche di sniffing di rete OR

1.2.1 - ARP spoof attack AND

1.2.2 - L'attaccante ottiene l'accesso alla LAN

1.3 - Usare tecniche di brute force OR

…

…

PASSO 2 – Analisi degli attributi di sicurezza

Dopo aver modellato i possibili attacchi al sistema, è necessario analizzare gli attributi di sicurezza del sistema quali, ad esempio:

1. la possibilità o l'impossibilità dell'attacco (P=Possibile, I=Impossibile)
2. il costo dell'attacco (valore in euro, es. 10K)
3. gli strumenti necessari per realizzare l'attacco (AS=Attrezzature Specifiche, SAS=Senza Attrezzature Specifiche)

Per determinare il costo di un attacco:

1. si determina il valore per ciascun nodo foglia.
2. si determina il valore dei nodi non foglia. Questo è ricavato a partire dal valore dei loro figli.

L'analista dovrà avere la capacità di attribuire dei "buoni" valori per i nodi foglia (anche in questa fase la capacità di identificazione dei valori è assolutamente soggettiva).

PASSO 3 – Analisi dei risultati

L'analisi è volta a determinare:

1. il costo dell'attacco più economico (si analizzano i valori degli attributi a livello del nodo radice);
2. gli attacchi il cui costo non supera una certa soglia (si analizzano i sotto-alberi i cui nodi rappresentano una determinata proprietà/soglia);
3. l'attacco più economico che non utilizza attrezzature speciali (si analizzano i sotto-alberi i cui nodi rappresentano un determinato insieme di proprietà).

L'analisi dei risultati deve tenere conto delle caratteristiche del potenziale attaccante in quanto queste determinano quali parti dell'attack tree devono essere prese in considerazione. Attaccanti diversi hanno diversi livelli di abilità, accesso, avversione al rischio, soldi e così via. Se il potenziale attaccante è la criminalità organizzata, occorre considerare la possibilità di attacchi costosi e "mezzi illeciti" (corruzione, ricatto, ecc.) che espongono l'attaccante fino al rischio di andare in prigione. Se il potenziale attaccante è un terrorista, occorre considerare "mezzi illeciti" che espongono l'attaccante fino al rischio di morire per raggiungere l'obiettivo. Se l'attaccante è un casual hacker si esclude la possibilità di attacchi costosi e "mezzi illeciti" (corruzione, ricatto, ecc.).

PASSO 4 – Analisi What-if

L'analisi "what if" è costruita a partire dalle diverse ipotesi di adozione delle contromisure. Introducendo contromisure, cambiano i valori attribuiti nel secondo step della metodologia e quindi cambiano i risultati dell'analisi.

In conclusione, una delle caratteristiche di valore degli attack tree è che sono riutilizzabili. Tornando all'esempio inziale, una volta completato l'albero di attacco relativo a "Ottenere le credenziali di autenticazione", è possibile utilizzarlo in qualsiasi situazione in cui sia interesse dell'attaccante l'acquisizione delle credenziali di autenticazione. L'attack tree relativo a "Ottenere le credenziali di autenticazione" può inoltre diventare parte di un attack tree più grande.

4.2.4.3 TRIKE

TRIKE è un framework open-source per l'auditing della sicurezza da un punto di vista di risk management basato sulla generazione di modelli di minacce in modo affidabile e ripetibile. Il progetto ebbe inizio nel 2005 come tentativo di migliorare l'efficienza e l'efficacia delle esistenti metodologie di modellazione delle minacce e da allora viene attivamente aggiornato e utilizzato. I creatori di TRIKE hanno anche sviluppato strumenti a supporto di questa metodologia come il foglio di calcolo TRIKE. Questo strumento si focalizza sull'automazione della generazione delle minacce e non prevede alcun brainstorming. I team di sicurezza non hanno la necessità di individuare le possibili minacce in quanto tali minacce sono già predefinite. TRIKE può essere utilizzato anche da uno sviluppatore di sicurezza inesperto per trovare le vulnerabilità di sicurezza in modo efficace ed affidabile. Il team TRIKE ha adottato l'analisi HAZOP (Hazardous Operations),

ovvero, un metodo sistematico per identificare quali variazioni di processo devono essere mitigate. Questo framework può sostituire gli alberi di minaccia e d'attacco (attack tree).

TRIKE utilizza un approccio basato sul rischio con distinte implementazioni dei modelli di minacce e rischi. Approccia al Threat Modeling assumendo una posizione difensiva rispetto a quella di un attaccante. TRIKE ha anche proposto il concatenamento delle minacce, un'alternativa agli alberi delle minacce, nel tentativo di ridurre la natura ripetitiva di quest'ultimi.

4.2.4.4 P.A.S.T.A (Process for Attack Simulation and Threat Analysis)

Si tratta di un processo agnostico rispetto alla piattaforma, suddiviso in sette fasi distinte e applicabile alla maggior parte delle metodologie di sviluppo. Il suo obiettivo principale è quello di affrontare le minacce più gravi per un determinato obiettivo applicativo. Di seguito in elenco le fasi di cui sopra:

- Definizione degli obiettivi di business;
- Definizione dell'ambito tecnico;
- Scomposizione del sistema;
- Analisi delle minacce;
- Rilevamento della vulnerabilità;
- Enumerazione degli attacchi;
- Analisi del rischio/impatto.

Il processo è una combinazione di diversi approcci di modellizzazione delle minacce, definisce gli obiettivi di business, i requisiti di sicurezza e conformità e l'analisi dell'impatto sul business. Similmente al processo adottato da Microsoft, nella rappresentazione del sistema, l'applicazione viene ridotta in componenti discreti attraverso l'utilizzo di casi d'uso e DFD. Durante l'analisi delle minacce e delle vulnerabilità vengono utilizzati anche gli alberi delle minacce e i casi di abuso. Si calcola quindi l'impatto sul rischio e sul business e vengono individuate le necessarie contromisure.

4.2.4.5 AS/NZS 31000:2009 Risk Management

L'AS/NZS 31000:2009[10], pubblicato nel novembre 2009, è stato concepito sulla base dello standard australiano/Nuova Zelanda AS/NZS 4360:2004[11], il primo standard formale al mondo per la documentazione e la gestione dei rischi che fornisce linee guida generiche sulla gestione del rischio. L'approccio standard è semplice, flessibile e iterativo. Questo non vincola le organizzazioni ad adottare una particolare metodologia di gestione del rischio, a condizione che la metodologia adottata soddisfi le cinque fasi del processo AS/NZS 4360, ovvero:

- Stabilire il contesto: stabilire il dominio di rischio, definendo quali asset/sistemi sono importanti.
- Identificare i rischi: nell'ambito del rischio, quali specifici rischi sono in evidenza?
- Analisi dei rischi: esamina i rischi e determina se esistono controlli a supporto.
- Valutazione del rischio: determina il rischio residuo.
- Trattamento del rischio: descrive le modalità di trattamento dei rischi in modo da mitigare i rischi di business indicati.

La gestione del rischio AS/NZS 31000:2009 può essere utilizzata per classificare i rischi ai fini dei controlli di sicurezza. Tuttavia, per quanto riguarda l'enumerazione delle minacce manca di metodi strutturati, e pertanto potrebbe essere ritenuto inadeguato.

[10] https://infostore.saiglobal.com/preview/as/as30000/31000/31000-2009.pdf?sku=1378670

[11] http://www.fanarco.net/books/risk/AS-NZS_4360-2004_Risk_Management.pdf

4.2.4.6 Best practices di carattere generale

- VALIDARE E NON SANITIZZARE - Dobbiamo sapere cosa e quanto ci aspettiamo di ricevere e dobbiamo convalidare ciò che riceviamo. Se si riceve qualcos'altro rispetto al previsto, è necessario scartarlo restituendo un messaggio di errore. A meno che il nostro codice non sia perfetto, eventuali errori di sanitizzazione potrebbero produrre ancor più danno, in quanto, dopo aver scritto la funzione di sanitizzazione dell'input si farebbe affidamento su di essa.
- AFFIDARSI AL SISTEMA OPERATIVO - Affidarsi al sistema operativo è una buona pratica per i seguenti motivi:
    - Il sistema operativo mette a disposizione funzionalità di sicurezza che consentono di concentrarsi sui propri obiettivi.
    - Il sistema operativo funziona con privilegi che probabilmente non sono disponibili per il programma o l'aggressore.
    - Se l'aggressore controlla il sistema operativo, è probabile che abbiamo problemi ben più gravi, indipendentemente da ciò che il codice tenta di fare.

  Con tutti questi consigli sul fidarsi del sistema operativo, ci si potrebbe chiedere "quale bisogno c'è di modellare le minacce? Perché non affidarsi solo al sistema operativo?". Ebbene, sta a noi verificare di non impostare le autorizzazioni su un file a 777, o impostare gli ACL in modo tale da consentire la scrittura agli account "guest". Sta a noi scrivere codice che funzioni bene con i permessi di un utente normale piuttosto che con i permessi di un utente sandboxed.

- FILE BUGS - È bene includere le minacce tra i bugs e contrassegnarle come bug di sicurezza. Bisogna inoltre dare una priorità a questi bugs. I bugs di tipo "Elevation-of-privilege" sono quasi sempre riconducibili alla categoria di più alta priorità poiché, quando vengono sfruttati causano molti danni. I bugs di tipo "Denial of service" spesso vanno considerati per ultimi.
- VERIFICA DEL LAVORO SVOLTO - La convalida del modello è l'ultima cosa da fare come parte della modellazione delle minacce. Ci sono alcune attività da svolgere prima, ed è bene mantenerle allineate con l'ordine in cui è stato svolto il lavoro precedente. Pertanto, le attività di validazione includono il controllo del modello, la verifica di ciascuna minaccia e il controllo dei test. Probabilmente si desidera convalidare il modello anche una seconda volta, ovvero quando ci si avvicina al rilascio o all'installazione.
- CONTROLLO DEL MODELLO - È necessario assicurarsi che il modello finale corrisponda a quello costruito. Se così non fosse, come potremmo sapere se le minacce trovate sono giuste e rilevanti? Per fare ciò, è opportuno organizzare degli incontri durante i quali tutti, osservando e analizzando il diagramma, rispondano alle seguenti domande:
    - il diagramma è completo?
    - il diagramma è accurato?
    - il diagramma copre tutti le decisioni intraprese in termini di sicurezza?
    - è possibile procedere con la versione successiva del diagramma senza apportare modifiche?

  Una risposta affermativa a tutte le domande di cui sopra, indica che il diagramma è sufficientemente aggiornato e maturo per poter procedere. Diversamente sarà necessario apportare gli opportuni cambiamenti.

- DETTAGLI DEL DIAGRAMMA - Non disegnare mai un diagramma ad occhio, con un livello di dettaglio tale da rappresentare l'intero comportamento del sistema. Utilizzare un sotto diagramma che mostri il solo dettaglio di una particolare area del sistema stesso. Si deve cercare di filtrare ciò che non ha senso per il progetto. Per esempio, se si è davanti ad un processo molto complesso, forse tutto ciò che è in quel processo dovrebbe essere trattato in un diagramma, e tutto ciò che è al di fuori di esso in un altro. Se si ha un dispatcher o un sistema di code, questi sono un buon punto di suddivisione, e lo sono anche i database o i sistemi di fail-over. Forse ci sono ancora alcuni

elementi che devono essere maggiormente dettagliati? Bene, questi vanno filtrati. La cosa importante da ricordare è che il diagramma ha lo scopo di aiutare a comprendere e discutere il sistema.

4.3 Indirizzamento delle minacce

Una volta raccolte le minacce individuate in una o più liste, il passo successivo nel processo di modellazione è quello di scorrere l'elenco o gli elenchi indirizzando ciascuna minaccia. Ci sono quattro tipologie di azioni che si possono intraprendere per contrastare tali minacce:

- **Mitigazione** delle minacce - Si concretizza nel fare qualcosa per rendere più difficile la possibilità di poter essere sfruttate. Richiedere una password per controllare chi accede al sistema, mitiga la minaccia di spoofing. Aggiungere un controllo password che ne rafforza la complessità o la scadenza, riduce la probabilità che una password venga scoperta o venga utilizzata se rubata.
- **Eliminazione** delle minacce - Avviene quasi sempre eliminandone le caratteristiche. Se si presenta una minaccia in cui qualcuno ha accesso ad una funzione amministrativa di un sito web entrando in "/admin/URL", è possibile mitigarla impiegando delle password o altre tecniche di autenticazione, ma comunque, questa non verrà risolta. È possibile rendere più complessa la URL in modo da rendere meno probabile la possibilità che questa venga trovata, ma anche in questo caso la minaccia non verrà risolta. La si può eliminare rimuovendo l'interfaccia amministrativa, e gestendo l'attività di amministrazione tramite linea di comando. In questo caso esisterebbero comunque altre minacce associabili a come l'utente amministratore dovrebbe loggarsi per poter procedere con l'attività di amministrazione da linea di comando. Lo spostamento dell'interfaccia dall'http a linea di comando rende facile la mitigazione della minaccia controllando la superficie di attacco.
- **Spostamento** delle minacce - Consiste nel lasciare che qualcuno o qualcos'altro gestisca il rischio. Per esempio, potremmo demandare la gestione delle minacce relative all'autenticazione al sistema operativo, oppure rinforzare il perimetro di fiducia con un firewall. È anche possibile trasferire il rischio direttamente all'utente utilizzatore, chiedendogli di navigare attraverso una moltitudine di finestre di dialogo incomprensibili prima che questo possa effettivamente utilizzare il sistema. Ovviamente questa non vuole essere assolutamente una tra le migliori soluzioni, ma in alcuni casi esiste da parte degli utilizzatori una conoscenza tale da poterli rendere partecipi per convenire ad un compromesso di sicurezza. Se si pensa che esistano i presupposti per una soluzione del genere, si dovrebbe aiutare chi usa il sistema a prendere una decisione in tal senso.
- **Accettazione** della minaccia - È l'ultimo approccio per indirizzare le minacce. In alcuni casi, il costo nell'impedire a qualcuno di inserire una back-door nella scheda madre di un hardware aziendale potrebbe risultare elevato, quindi in questi casi si potrebbe scegliere di accettare il rischio. Una volta che questo viene accettato, non c'è più bisogno di preoccuparsene. A volte la preoccupazione indica che il rischio non è stato pienamente accettato o che l'accettazione del rischio non sia appropriata.

4.4 Valutazione del rischio: tecniche di Risk Ranking

4.4.1 DREAD

Microsoft ha sviluppato la metodologia DREAD (tabella che segue) per valutare ciascun rischio individuato durante l'attività STRIDE. Ad ogni rischio viene assegnato un punteggio DREAD da parte del team di sicurezza/sviluppo i quali realizzano e applicano il modello delle minacce.

Esistono diverse varianti del sistema di valutazione e prioritizzazione del rischio:

DREAD	DESCRIZIONE
	Classifica l'estensione del danno che si verifica se si sfrutta la vulnerabilità individuata.
	Classifica quanto spesso un tentativo di sfruttamento della vulnerabilità individuata

	funziona.
	Assegna un valore numerico allo sforzo necessario per sfruttare la vulnerabilità individuata. Inoltre, la possibilità di sfruttamento considera come condizioni preliminari che l'utente deve essere autenticato.
	Assegna un valore numerico che rappresenta la razione di istanze installate del sistema che potrebbero essere interessate se un exploit divenisse ampiamente disponibile.
	Misura la probabilità che una vulnerabilità possa essere trovata da ricercatori esterni della sicurezza e/o dagli hacker, se questa non viene risolta tramite patch.

Tabella 14 - Modello DREAD

Nella valutazione del rischio ad ogni componente DREAD viene assegnato un punteggio. I punteggi dei singoli componenti vengono quindi calcolati per dare un 'punteggio DREAD' totale.

Il rischio viene quindi determinato in base al valore che il punteggio "DREAD" assume rispetto ad intervalli di valori predefiniti. Il risultato finale è un elenco di vulnerabilità classificate per rischio.

Il processo di applicazione della metodologia DREAD è estremamente soggettivo e richiede le necessarie competenze. È consigliabile avere almeno un membro del team che abbia competenze sulla sicurezza per dare il necessario supporto nell'assegnazione dei punteggi DREAD.

Come fase finale del processo di modellazione delle minacce, viene attuata una valutazione del **rischio** (*risk assessment*), per dare una priorità a ciascuna vulnerabilità indentificata.

In generale, in termini quantitativi, il rischio è definito come il prodotto tra la probabilità di accadimento dell'evento e l'impatto:

Rischio = Probabilità x Impatto

In effetti, se almeno uno dei due termini del prodotto tende a zero, percepiamo il rischio come basso. Viceversa percepiamo un rischio grave quando ambedue i termini sono elevati.

Nella metodologia DREAD il concetto di "impatto" viene declinato in termini di:

1. danno (Damage)
2. utenti interessati (Affected Users)

mentre il concetto di "probabilità" viene declinato in termini di:

3. riproducibilità (Reproducibility),
4. sfruttabilità (Exploitability)
5. rilevabilità (Discoverability).

DREAD è appunto l'acronimo che "fattorizza" il rischio rispetto a queste 5 distinte categorie che caratterizzano la minaccia:

1. **Damage potential**: Quanto sarebbe rilevante il danno in caso di concretizzazione della minaccia[12]?
2. **Reproducibility**: Quanto è facile che la minaccia possa ripetersi?
3. **Exploitability**: Quanto tempo, sforzo e conoscenza è necessaria per concretizzare con successo la minaccia?
4. **Affected Users**: Nel caso in cui la minaccia si concretizzi, quale percentuale di utenti sarebbe coinvolta?

[12] Si ricordi che la minaccia è un evento potenziale, accidentale o deliberato. Se deliberato, la minaccia si configura come attacco.

5. **Discoverability**: Quanto è facile per un attaccante scoprire la minaccia?

A ciascuna categoria viene attribuito un peso. Il "DREAD score" è la media dei 5 pesi, ossia:

DREAD Score= (Damage + Reproducibility + Exploitability + Affected Users + Discoverability) / 5

Occorre quindi valutare e dare un peso numerico alle cinque categorie della tabella sopra mostrata. A seconda del dominio considerato, ci si può riferire o a una scala (semplificata) di tre soli valori o a una scala (più granulare) a dieci valori. I valori crescono rispettivamente al crescere del danno, della facilità di riproduzione, della facilità di sfruttamento, del numero di utenze coinvolte, della facilità di rilevamento.

A titolo di esempio, si consideri la categoria "Exploitability":

- nel caso si voglia adottare una scala a 3 valori si potrebbero voler considerare e pesare in modo diverso i seguenti casi (Quanto è difficile sfruttare la vulnerabilità?):
    - 1= L'attacco richiede una figura senior e una conoscenza profonda del sistema attaccato; un'utenza con diritti di amministrazione; dispendio di parecchie risorse per organizzare l'attacco (es. impiego di custom tool);
    - 2 = L'attacco richiede una figura senior; un'utenza autenticata con abilità non amministrative; tool di attacco disponibili in rete;
    - 3 = L'attacco è alla portata di una figura junior; senza autenticazione; con web browser.

- nel caso si voglia adottare una scala a 10 valori si potrebbero voler considerare e pesare in modo diverso i seguenti casi (Quanto è difficile sfruttare la vulnerabilità?):
    - 1 = Anche con conoscenza approfondita della vulnerabilità non si individua un percorso di attacco valido per l'exploit;
    - 2 = Sono richieste tecniche avanzate e tool custom. Sfruttabile solo dagli utenti autenticati;
    - 5 = La possibilità di exploit esiste, alla portata di skill medie da un'utenza autenticata;
    - 7 = La possibilità di exploit esiste, alla portata di un'utenza non autenticata;
    - 10 = Banale: basta un web browser.

Nel primo caso il calcolo del DREAD score:

DREAD Score= (Damage + Reproducibility + Exploitability + Affected Users + Discoverability) / 5

ricade in un numero compreso tra 1 e 3. Nel secondo caso ricade in un numero compreso tra 1 e 10.

Il risultato finale è, in entrambi i casi, come desiderato, un elenco di vulnerabilità classificate per rischio ossia ordinate per priorità di intervento (a valori bassi corrisponde priorità bassa, a valori alti priorità alta).

4.4.2 Security Bulletin Severity Rating System (S.B.S.R.S)

Come alternativa alla metodologia di analisi DREAD, Microsoft si avvale anche del "Security Bulletin Severity Rating System" per valutare le vulnerabilità nei prodotti Microsoft. Il sistema di classificazione raggruppa le vulnerabilità in una delle quattro categorie sotto riportate. Microsoft utilizza questo sistema per valutare la necessità di patch di sicurezza per i propri prodotti.

CLASSIFICAZIONE	DEFINIZIONE
CRITICA	Una vulnerabilità il cui sfruttamento potrebbe consentire l'esecuzione di codice senza alcuna interazione da parte dell'utente.

IMPORTANTE	Una vulnerabilità il cui sfruttamento potrebbe compromettere la riservatezza, l'integrità o la disponibilità dei dati degli utenti o l'integrità o la disponibilità delle risorse necessarie all' elaborazione.
MODERATA	L'impatto della vulnerabilità è mitigato in misura significativa da fattori quali i requisiti di autenticazione o l'applicabilità solo alle configurazioni non predefinite.
BASSA	L'impatto della vulnerabilità è ampiamente mitigato dalle caratteristiche del componente interessato. Microsoft raccomanda ai clienti di valutare se applicare l'aggiornamento di sicurezza ai sistemi interessati.

Tabella 15 - Sistema di classificazione del S.B.S.R.S.

Questo sistema di rating può anche essere adattato per classificare le vulnerabilità identificate dalla STRIDE. L'obiettivo finale è quello di produrre un elenco di prioritizzazione delle vulnerabilità a supporto del processo decisionale. Sebbene Microsoft non utilizzi più l'analisi DREAD, le motivazioni potrebbero essere giustificate dal fatto che si tratta di un processo di valutazione del rischio più adatto ai non esperti di sicurezza o a team novizi nella modellazione delle minacce rispetto al Security Bulletin Severity Rating System. L'analisi DREAD prevede un calcolo basato sulla valutazione di diversi aspetti della vulnerabilità, quali il danno potenziale o la riproducibilità. Il Security Bulletin Severity Rating System classifica le vulnerabilità in una di quattro categorie. Entrambi si basano sul parere del singolo individuo/team che effettua la valutazione del rischio. La DREAD ha perfezionato il suo sistema di valutazione del punteggio riducendo la probabilità di variazione dei risultati ed offre un approccio strutturato che i team possono adottare. Il processo è inoltre ben documentato e relativamente facile da attuare. Questa è una delle possibili motivazioni per cui scegliere l'analisi DREAD rispetto a S. B. S. R. S.

4.4.3 Altri processi di valutazione del rischio

Esistono altre opzioni disponibili per la valutazione del rischio, come l'US-CERT Vulnerability Metric, che utilizza una metrica quantitativa e valuta la gravità di una vulnerabilità assegnandole un valore compreso tra 0 e 180. Il Common Vulnerability Scoring System (CVSS) mira a fornire un framework open per misurare l'impatto delle vulnerabilità IT, mentre la scala SANS Critical Vulnerability Analysis classifica le vulnerabilità utilizzando diversi fattori chiave e variando il grado di peso. Il processo di valutazione del rischio da adottare dipende essenzialmente dalla scelta individuale del team che realizza il Threat Model.

4.5 Privacy by Design

4.5.1 Introduzione e concetti base

La garanzia di sicurezza di un'organizzazione rappresenta la base per la protezione della privacy degli individui. La privacy può essere compromessa a causa di un errore nella sicurezza, tuttavia, la privacy si riferisce all'utilizzo improprio e non, delle informazioni da parte di utenti autorizzati. La privacy è una politica di gestione delle informazioni più che una politica di controllo accesso. Insieme, privacy e sicurezza, rappresentano la base per creare un rapporto di fiducia. Una sicurezza solida è la base della protezione della privacy. La privacy richiede una sicurezza effettiva, ma quest'ultima da sola, non garantisce una privacy effettiva. La sicurezza garantisce ad esempio il controllo accessi alle risorse di un'organizzazione attraverso un'istruzione di controllo accessi come: all'entità X è consentito eseguire l'operazione Y sulla risorsa Z che tradotta in un esempio pratico, potrebbe essere: "Il personale del settore finanziario può interrogare la base dati della contabilità".

La privacy si riferisce alla relazione tra un'organizzazione che raccoglie informazioni e il proprietario delle informazioni raccolte. Per stabilire e gestire questa relazione, è necessario creare una politica di riservatezza formata da diverse istruzioni. Un'istruzione della politica di riservatezza definisce:

- I tipi di informazioni raccolte e quelle accessibili;
- Chi può accedere alle informazioni raccolte;
- Per quali scopi è possibile accedere alle informazioni.

Tale istruzione può assumere la seguente forma: all'entità X è consentito eseguire l'operazione Y sulla risorsa protetta Z del proprietario A per lo scopo B solo se il proprietario A esprime il proprio consenso esplicito. Una possibile traduzione di tale concetto in un esempio pratico, potrebbe essere: "I membri del settore di ricerca e sviluppo possono consultare i dati del modello di utilizzo di un singolo individuo per lo sviluppo di un nuovo prodotto se la persona in questione ha espresso esplicitamente il proprio consenso a rilasciare i propri dati su tale modello". Da notare che questa istruzione comprende la scelta da parte del proprietario delle informazioni di stabilire come utilizzare le sue informazioni. È possibile, inoltre, applicare anche altre dipendenze per l'accesso alle informazioni personali. Ad esempio, una norma può imporre alcune restrizioni sull'utilizzo delle informazioni raccolte.

La privacy by Design è emersa come un approccio proattivo, integrativo e creativo per rafforzare i requisiti di privacy nelle prime fasi della progettazione applicativa. Tra le sfide legate all'ingegneria della privacy by design vi è la mancanza di metodologie olistiche, sistematiche e integrative che affrontino la complessità e la variabilità della privacy e sostengano la traduzione dei suoi principi fondamentali nelle attività di ingegneria. Per certi versi questo è comprensibile poiché l'approccio è stato sviluppato per tener conto di una serie di fonti e standard. Tuttavia, ne consegue che i suoi principi fondanti sono dati ad un alto livello di astrazione senza fornire a corredo metodologie e linee guida per ottenere requisiti concreti in materia di privacy. I principi fondamentali della Privacy by Design si basano sui Fair Information Practice Principles (FIPP) e fungono da framework universale per l'integrazione della privacy in tre principali aree di applicazione: tecnologie dell'informazione e della comunicazione, aree di business, progetti fisici e infrastrutturali.

La Privacy Engineering si è affermata come una nuova disciplina che mira ad applicare principi e processi di ingegneria nello sviluppo, nell'implementazione e manutenzione dei sistemi, in modo sistematico e ripetibile, per raggiungere un livello accettabile di protezione della privacy. Per distinguere questi concetti; la Privacy by Design (PbD) intende spiegare "Cosa fare" per raggiungere un livello adeguato di protezione della privacy, mentre la Privacy Engineering (PE) intende spiegare "Come farlo" definendo la privacy come un attributo di qualità nell'ingegneria dei sistemi. In altre parole, la PE si concentra sullo sviluppo e la valutazione di metodi, tecniche e strumenti che identificano e affrontano in modo sistematico le problematiche legate alla privacy durante il processo di sviluppo dei sistemi.

La tabella che segue, sintetizza i 'concetti' alla base della Privacy:

Concetto di Privacy	Descrizione
Personal Identifiable Information (PII)	Informazioni che possono essere ricondotte ad un individuo.
Data Subject	Individuo collegato alla PII.
Item of Interest (IOI)	Informazioni relative ad un individuo (ad esempio soggetti, messaggi, azioni, ecc.).
Unlinkability	Non essere in grado di distinguere se due IOI sono correlati.
Anonymity	Non essere in grado di identificare il soggetto all'interno di un gruppo di soggetti.
Plausible deniability	Essere in grado di negare di aver eseguito un'azione.

Undetectability	Non essere in grado di distinguere se esiste un IOI.
Unobservability	Impossibilità nell'essere rintracciabili da tutti i soggetti coinvolti.
Confidentiality	Restrizioni autorizzate all'accesso e alla divulgazione delle informazioni.
Awareness	Essere consapevoli delle conseguenze della condivisione delle informazioni personali (PI).
Compliance	Aderenza alle normative e alle politiche interne di una organizzazione.

Tabella 16 - Concetti alla base della Privacy

Similmente alla miriade di normative sulla privacy disponibili, ci sono stati diversi tentativi di strutturare e classificare i concetti di privacy. Di seguito si riportano alcuni esempi di tassonomie basate su due approcci distinti:

- Classificazione dei concetti di privacy da un punto di vista giuridico;
- Classificazione dei concetti di privacy da un punto di vista di ingegneria del software.

Tassonomia di Solove. Presenta una tassonomia delle violazioni della privacy da un punto di vista legale. Anche se questa non tratta la privacy digitale, ma descrive la privacy in generale, fornisce comunque alcune informazioni utili in materia. Solove[13] opera una distinzione tra quattro gruppi di attività di base dannose:

- Raccolta dei dati. Include due tipi di violazioni della privacy: il controllo inteso come "osservazione, ascolto o registrazione delle attività di un individuo", e, l'investigazione che consiste in varie forme di sondaggio per ottenere informazioni.
- Trattamento dei dati. Include cinque tipi di violazioni dei dati raccolti al punto precedente: aggregazione (ovvero combinazione di dati relativi a un individuo), identificazione (ovvero collegamento dei dati per identificare un individuo), negligenza (poca attenzione nella protezione dei dati memorizzati), uso secondario (ovvero utilizzo dei dati per scopi diversi da quelli per i quali sono stati raccolti) ed esclusione (ovvero quando l'interessato non è a conoscenza dei dati che gli altri hanno su di esso).
- Diffusione dei dati. Include sette categorie di violazioni: violazione della riservatezza (ovvero non mantenere riservate le informazioni di una persona), divulgazione (cioè rivelare informazioni "sensibili" veritiere su una persona), esposizione (cioè rivelare le nudità, il dolore o le caratteristiche fisiche di una persona), maggiore accessibilità (cioè amplificare l'accessibilità dei dati), appropriazione (cioè l'uso della propria identità per perseguire un'altra finalità).
- Invasione. A differenza dei gruppi precedenti, non riguarda necessariamente le informazioni personali, ma degli elementi che limitano la sfera personale e decisionale (ad esempio atti invasivi che violano la tranquillità di una persona e atti invasivi che impattano sulle decisioni private di una persona).

Linee guida FIPPs (Fair Information Practice Principles). La Privacy and Personal Information Protection raccoglie un insieme di indicazioni proposte dalla Federal Trade Commission degli Stati Uniti. Queste possono essere considerate come il fondamento di tutta la legislazione vigente negli Stati Uniti, in materia di protezione dei dati. Sono state utilizzate per la definizione delle linee guida dell'OCSE (Organizzazione per

[13] https://en.wikipedia.org/wiki/Daniel_J._Solove

la Cooperazione e lo Sviluppo Economico) e per la direttiva europea sulla protezione dei dati. I principi si basano su cinque distinte categorie:

- Avviso/Consapevolezza. I consumatori dovrebbero essere adeguatamente informati prima di procedere nella raccolta delle informazioni personali.
- Scelta/Consenso. I consumatori devono essere in grado di scegliere come devono essere utilizzati i propri dati personali. In particolare laddove si fa un uso secondario dei dati (ad esempio, registrazione ad una mailing list o trasferimento di informazioni a terzi).
- Accesso/Partecipazione. Una persona dovrebbe essere in grado di accedere ai dati che la riguardano e di contestarne l'accuratezza e la completezza.
- Integrità/Sicurezza. I dati devono essere accurati e sicuri.
- Enforcement/Redress. Dovrebbero essere messe in atto misure di enforcement per garantire il rispetto dei FIPP.

Le FIPP sono utilizzate anche per definire altre tassonomie di privacy. Se ne cita qualcuna a titolo di esempio:

- linee guida Microsoft per la privacy;
- tassonomia definita da Anton et al.[13]. Questa tassonomia però, non contiene solo i cinque FIPP come obiettivi di protezione della privacy, ma include anche una serie di obiettivi di vulnerabilità alla privacy che sono correlati alle minacce esistenti. Questi obiettivi di vulnerabilità includono il monitoraggio delle informazioni, l'aggregazione, la memorizzazione, il trasferimento di informazioni, la raccolta e la personalizzazione.

European Data Protection Legislation. La legislazione è una questione complessa, spesso vaga e formulata in modo ambiguo; il che la rende molto difficile da attuare. La privacy non richiede solo misure tecnologiche, ma anche misure organizzative. Inoltre, è difficile prevedere tutti i potenziali domini e contesti (e le relative normative) in cui un prodotto software verrà poi utilizzato. Tuttavia, alcune disposizioni legislative in materia di protezione dei dati possono, con un minimo sforzo, essere integrate nella progettazione del sistema.

Guarda e Zannone [2] riassumono la Direttiva Europea sulla Protezione dei Dati nei seguenti nove principi:

- Elaborazione corretta e lecita. La raccolta e il trattamento dei dati personali non devono interferire in modo irragionevole con la privacy delle persone interessate né interferire in modo irragionevole con la loro autonomia e integrità e devono essere conformi al quadro giuridico generale.
- Consenso. I dati personali devono essere raccolti e trattati solo previo esplicito consenso al loro trattamento da parte degli interessati.
- Finalità. I dati personali devono essere raccolti per finalità specifiche, lecite e legittime e non devono essere trattati per finalità non compatibili con quelle per cui sono stati raccolti.
- Minimalità. La raccolta e il trattamento dei dati personali sono limitati al minimo necessario per il raggiungimento delle finalità specifiche. Ciò include che i dati personali vengono conservati solo per il tempo necessario a raggiungere lo scopo specifico.
- Informazione minima. La divulgazione di dati personali a terzi è limitata e avviene solo a determinate condizioni.
- Qualità dell'informazione. I dati personali devono essere accurati, pertinenti e completi rispetto alle finalità per le quali sono raccolti e trattati.
- Controllo dell'interessato. L'interessato deve essere in grado di controllare e condizionare il trattamento dei suoi dati personali.
- Sensibilità. Nel trattamento dei dati personali è necessario applicare misure di protezione più rigorose sui dati ritenuti particolarmente sensibili per il soggetto interessato.
- Sicurezza delle informazioni. I dati personali devono essere trattati in modo da garantire un livello di sicurezza adeguato ai rischi connessi al trattamento e alla natura dei dati stessi.

Poiché il DPD è stato creato in un momento in cui Internet era ancora agli inizi, nel 2012 è stata elaborata una proposta di riforma della legislazione attuale per rafforzare i diritti della privacy online. Questa riforma indirizza:

- il "diritto all'oblio" ovvero, l'obbligo di fornire esplicitamente il consenso necessario al trattamento dei dati;
- il "diritto di portabilità dei dati" che consente un accesso più facile ai propri dati e una maggiore trasparenza sul modo in cui questi vengono gestiti.

Anche la responsabilità di coloro che trattano i dati personali è accresciuta dall'attuazione di principi quali "Privacy by Design".

La direttiva relativa alla privacy e alle comunicazioni elettroniche è stata promulgata nel 2002 e completa la direttiva DPD in quanto si concentra sulla protezione dei dati nell'era digitale. Essa disciplina il settore delle comunicazioni elettroniche ed è stata modificata nel 2009. È principalmente nota per la richiesta di consenso da parte dell'utente nella memorizzazione dei cookie, ed è quindi spesso indicata come la "direttiva sui cookie". Inoltre, la direttiva disciplina lo spam online imponendo un regime di "opt-in", in base al quale le e-mail indesiderate possono essere inviate solo previo accordo del destinatario. La direttiva comprende anche la conservazione e il trattamento dei dati relativi al traffico e dei dati relativi all'ubicazione. I dati relativi al traffico dovrebbero essere cancellati o resi anonimi non appena non sono più necessari ai fini della trasmissione. Il trattamento di tali dati può avvenire solo quando questi vengono resi anonimi o quando l'interessato ha dato il suo consenso.

Sebbene la legislazione sulla protezione dei dati sia complessa e spesso ambigua, alcune norme possono essere automatizzate nei sistemi software. Negli ultimi anni è emersa una ricerca che si propone di estrarre diritti e obblighi dai documenti legali e che fornisce la tracciabilità tra le politiche sulla privacy (scritte) e le loro controparti software implementate.

4.5.1.1 Proprietà

In quanto concetto astratto e soggettivo, la declinazione della privacy varia a seconda delle problematiche sociali e culturali, delle discipline di studio, degli interessi degli stakeholder e del contesto applicativo. Le norme di privacy più comuni sono volte a consentire agli individui di controllare, modificare, gestire e cancellare informazioni su se stessi e decidere quando, come e in quale misura tali informazioni possono essere comunicate agli altri.

La privacy si basa prevalentemente su due modelli di tutela:

- *hard privacy* (la privacy quale libertà negativa). Si basa sul concetto di libertà (e del relativo diritto) definendo un perimetro entro cui l'individuo può agire al riparo da invasioni esterne. Questa libertà dal controllo si esprime in una sfera di libertà di scelte e di comportamenti. Nella modellazione delle minacce è necessario tener conto delle seguenti entità: il fornitore di servizi, il titolare dei dati e l'ambiente con cui interagisce.
- *soft privacy* (privacy quale libertà positiva). Contrariamente al primo, si basa sul presupposto che l'interessato abbia concesso il controllo dei propri dati personali a terzi, e debba fidarsi dell'onestà e della competenza dei responsabili del trattamento. L'obiettivo di questo modello è quindi, di fornire la sicurezza dei dati ed elaborare questi con finalità e consenso specifici, tramite politiche, controllo degli accessi e audit. Il modello prevede che l'interessato fornisca i suoi dati personali e il responsabile del trattamento di tali dati è responsabile della loro protezione. Di conseguenza, si applica un modello di minaccia più debole, che include diverse parti con diversi poteri.

Alla base del modello di privacy, ci sono alcune delle classiche proprietà di sicurezza quali:

- **confidenzialità,** garantisce che le informazioni siano accessibili solo da parte di persone autorizzate;
- **integrità,** garantisce l'accuratezza e la completezza delle informazioni e dei metodi di elaborazione;

- **disponibilità** (o resistenza alla censura), garantisce che le informazioni siano accessibili agli utenti autorizzati;
- **non ripudio,** garantisce che non si sia in grado di negare ciò che si è fatto.

Le caratteristiche di queste proprietà si trovano nella norma ISO 17799[14]. A queste si aggiungono ulteriori proprietà quali:

- **Unlinkability**. Si riferisce alla capacità di nasconde il legame tra due o più azioni (ad esempio, nascondere i link tra due messaggi anonimi inviati dalla stessa persona), identità (ad esempio, due pagine web visitate da parte dello stesso utente o due persone collegate da una relazione di amicizia in un social network) o informazioni (ad esempio, voci presenti in due distinti database relativi alla stessa persona). La unlinkability consiste nel separare due o più elementi di interesse, detti IOI, (quali ad esempio, soggetti, messaggi, azioni, etc). Questa separazione assicura che all'interno del sistema (comprendente questi ed eventualmente altri elementi), l'aggressore non sia in grado di identificare in modo efficace se questi IOI sono collegati o meno.
- **Anonymity**. Si riferisce alla capacità di nascondere il legame tra un'identità e un'azione o un'informazione (ad esempio, il mittente anonimo di una e-mail, l'autore di un testo, la persona che accede ad un servizio, la persona a cui si riferisce una voce presente in un database). Tale proprietà assicura che un eventuale attaccante non possa identificare in modo efficace il soggetto. L'anonimato può essere ricondotto anche alla precedente proprietà (unlinkability).
- **Pseudonymity**. Suggerisce che è possibile costruire una reputazione su uno pseudonimo e utilizzare pseudonimi multipli per scopi diversi.
- **Plausible deniability**. Si riferisce alla capacità di ostacolare la possibilità da parte di un attaccante di dimostrare che un soggetto conosce, ha fatto o ha detto qualcosa.
- **Undetectability and unobservability**. Si riferisce alla capacità di nascondere le attività dell'utente. L'*undetectability* è vista come l'impossibilità di rilevare un elemento di interesse (IOI) da un punto di vista di un attaccante, il che significa che quest'ultimo non può distinguere quando l'elemento esiste da quando invece non esiste. L'*unobservability* è vista come l'impossibilità di osservare un elemento di interesse (IOI) da parte di tutti i soggetti non coinvolti.
- **Confidentiality**. Si riferisce alla capacità di nascondere il contenuto dei dati o di controllare la divulgazione degli stessi (ad esempio, il trasferimento di e-mail crittografate, l'applicazione del controllo di accesso a un documento classificato o a un database contenente informazioni sensibili). NIST descrive la riservatezza come la capacità di mantenere le restrizioni autorizzative all'accesso e alla divulgazione delle informazioni, compresi i mezzi per proteggere la privacy e le informazioni proprietarie. Sebbene la riservatezza sia una proprietà di sicurezza, come si evince dalla definizione, essa è importante anche per preservare le proprietà dell'anonimato e di inscindibilità.
- **Content awareness**. Si riferisce alla capacità di garantire che gli utenti abbiano la consapevolezza dei loro dati personali e di limitare l'uso delle informazioni necessarie per consentire l'esecuzione della funzione a cui si riferiscono. Quanto più sono elevate le informazioni personali identificabili divulgate dagli interessati, tanto maggiore è il rischio di violazione della privacy. L'utente deve essere consapevole delle conseguenze della condivisione delle informazioni. Queste conseguenze possono riferirsi alla violazione della privacy così come a risultati indesiderati fornendo informazioni incomplete o errate.
- **Policy and consent compliance**. Si riferisce alle politiche in atto e alle proprietà di conformità sul consenso in merito al trattamento dei dati personali per informare l'interessato sulla politica di privacy del sistema, o consentire allo stesso di specificare i consensi in conformità con la legislazione, prima che gli utenti accedono al sistema.

[14] https://www.iso.org/standard/39612.html

Le proprietà sopra esposte possono essere considerate tutte proprietà tipiche di sicurezza. Relativamente ai modelli di hard/soft privacy possono essere inoltre così suddivise:

- unlinkability, anonimity, pseudonymity, plausible deniability, undetectability and unobservability, confidentiality possono essere considerate come proprietà di hard privacy;
- content awareness, policy and consent compliance, possono essere considerate come proprietà di soft privacy.

4.5.1.2 Principi

La **Privacy by Design** è un concetto sviluppato alla fine degli anni 90 da Ann Cavoukian [3], commissario per l'informazione e la privacy dell'Ontario. Si tratta di un approccio ingegneristico che si concentra sull'intero processo a partire dai principi di privacy e protezione dei dati.

La tutela della privacy non dovrebbe essere garantita solo dal rispetto delle norme e dai quadri normativi, in quanto non vi è alcuna utilità nelle norme giuridiche di codifica rigorosa se il sistema stesso non ha una solida base per la sicurezza e la tutela della privacy.

La privacy by Design può essere raggiunta applicando i sette principi su cui si basa:

- **Proattivo non reattivo, preventivo non correttivo**: le minacce alla privacy dovrebbero essere anticipate e prevenute, piuttosto che corrette dopo che queste si sono verificate.
- **Privacy come impostazione predefinita**: la privacy dovrebbe essere lo standard. I dati personali dovrebbero essere protetti automaticamente, anche senza alcuna azione esplicita da parte dell'individuo interessato.
- **Privacy incorporata nella progettazione**: la privacy non dovrebbe essere considerata un elemento aggiuntivo, ma dovrebbe essere integrata nella progettazione e nell'architettura dei sistemi software e delle attività di business in generale.
- **Piena funzionalità - somma positiva, non somma zero**: la privacy dovrebbe coesistere con altri interessi di business. Dovrebbero tuttavia essere evitati compromessi non necessari (ad esempio privacy vs. sicurezza o privacy vs. performance). Si dovrebbe cercare di ottenere una somma positiva.
- **Sicurezza end-to-end - Tutela dell'intero ciclo di vita**: la privacy richiede sicurezza durante l'intero ciclo di vita dei dati personali, garantendo una gestione sicura e completa di tutti i dati.
- **Visibilità e trasparenza**: gli obiettivi di privacy dichiarati dall'organizzazione e conseguentemente adottati dai sistemi, devono essere visibili e trasparenti agli utenti.
- **Rispetto per la privacy degli utenti**: devono essere applicate misure adeguate per responsabilizzare l'utente nel trattamento dei propri dati.

Il panorama tecnologico della privacy è stato classificato da Gürses [14] secondo tre paradigmi che tuttavia, non si escludono a vicenda:

- **Privacy come controllo**. In quanto controllo, mira a fornire agli interessati un mezzo per controllare la divulgazione dei propri dati. Rientrano in questa categoria anche gli strumenti applicati dall'organizzazione per definire e applicare politiche di sicurezza dei dati e prevenire l'abuso di accessi non autorizzati. Esempi di tecnologie correlate, includono ad esempio, le impostazioni relative alla privacy, il controllo dell'accesso e la verifica dei dati.
- **Privacy come riservatezza**. Questo paradigma impedisce la divulgazione delle informazioni o perlomeno ne minimizza il più possibile la divulgazione al fine di evitare di collegarle all'interessato. Le tecnologie di esempio sono: i protocolli di autenticazione anonimi e le reti di comunicazione anonime.
- **Privacy come pratica**. Si concentra maggiormente sull'aspetto sociale della privacy e mira a rendere più trasparenti i flussi di informazioni attraverso strumenti di sensibilizzazione e feedback.

4.5.1.3 Riferimenti normativi

Il 27 aprile 2016 il Parlamento Europeo ha approvato il Regolamento generale sulla protezione dei dati personali afferenti a persona fisica (General Data Protection Regulation[15]) che abroga la direttiva 95/46/CE, al fine di armonizzare le legislazioni dei singoli paesi, consolidare il diritto alla privacy dei cittadini dell'UE e garantire un'adeguata sicurezza dei dati sensibili trattati dalle aziende.

Il regolamento si applica a tutti gli enti pubblici e le imprese che trattano dati personali e dati personali classificati (sensibili, biometrici, sanitari e giudiziari, etc.) e/o che raccolgono grandi quantità di dati personali.

Tra gli elementi di maggiore innovazione ed interesse troviamo:

- La responsabilizzazione del Titolare nell'adozione di comportamenti proattivi tali da dimostrare l'attuazione di misure concrete individuate attraverso una valutazione dell'impatto dei trattamenti previsti (Data Protection Impact Analysis - DPIA);
- La nomina del Responsabile della protezione dei dati (Data Protection Officer - DPO) con il compito di vigilare, sui processi interni alla struttura, l'osservanza del Regolamento, fornire pareri in merito alla DPIA, indicare le misure tecniche e organizzative per attenuare i rischi per i diritti e gli interessi delle persone interessate, fungere da punto di contatto con l'autorità di controllo;
- La comunicazione all'Autorità Garante di eventuali violazioni della sicurezza dei dati personali (Data Breach);
- Il principio di "privacy by design" che prevede l'integrazione delle attività volte alla protezione dei dati personali in tutte le fasi del ciclo di vita dei sistemi e delle applicazioni IT, dalla fase di progettazione, messa in esercizio, utilizzo e dismissione finale;
- Il principio di "privacy by default" che prevede il rispetto dei principi generali della protezione delle informazioni, quali la minimizzazione dei dati e la limitazione delle finalità, nelle impostazioni dei servizi e dei prodotti che trattato dati personali.

La protezione della privacy richiesta dal GDPR presuppone quindi programmi di conformità sostenuti da tutta l'azienda, in modo da integrare i requisiti di sicurezza dei dati in tutte le fasi di ogni processo aziendale, dalla progettazione al rilascio.

L'Articolo 25 del GDPR[16] – **Data protection "by default" "by design"** – chiede al titolare del trattamento di mettere in atto misure tecniche e organizzative adeguate:

- volte ad attuare in modo efficace i principi di protezione dei dati, quali la pseudo-anonimizzazione e la minimizzazione, e a integrare nel trattamento le necessarie garanzie al fine di soddisfare i requisiti del regolamento e tutelare i diritti degli interessati (by design);
- per garantire che siano trattati, by default, solo i dati personali necessari per ogni specifica finalità del trattamento. Tale obbligo vale per la quantità dei dati personali raccolti, la portata del trattamento, il periodo di conservazione e l'accessibilità. In particolare, dette misure garantiscono che, by default, non siano resi accessibili dati personali a un numero indefinito di persone fisiche senza l'intervento della persona fisica.

Nel ciclo di vita del software in sicurezza, deve essere posta quindi maggiore attenzione quando si sviluppano applicativi che tratteranno dati personali, inserendo controlli mirati per ciascuna fase, secondo le best practices di sicurezza e secondo le indicazioni fornite dall'analisi dei rischi privacy (DPIA). Tali controlli devono essere formalmente verificati in fase di test e collaudo.

Altri elementi da considerare durante le fasi del ciclo di sviluppo software, si evincono dall'Articolo 32 del GDPR[17] – **Sicurezza del trattamento** – nel quale viene chiesto al titolare e al responsabile del trattamento

[15] https://gdpr-info.eu/

[16] https://gdpr-info.eu/art-25-gdpr/

[17] https://gdpr-info.eu/art-32-gdpr/

di mettere in atto misure tecniche e organizzative adeguate per garantire un livello di sicurezza adeguato al rischio, che comprendono, tra le altre, se del caso:

- la pseudo-anonimizzazione e la cifratura dei dati personali;
- la capacità di assicurare su base permanente la riservatezza, l'integrità, la disponibilità e la resilienza dei sistemi e dei servizi di trattamento;
- la capacità di ripristinare tempestivamente la disponibilità e l'accesso dei dati personali in caso di incidente fisico o tecnico;
- una procedura per testare, verificare e valutare regolarmente l'efficacia delle misure tecniche e organizzative al fine di garantire la sicurezza del trattamento.

4.5.2 Best practices per il trattamento dei dati personali

- **Ridurre al minimo i dati personali utilizzati -** *Ridurre l'impatto dei rischi limitando la gestione di dati personali a ciò che è strettamente necessario per raggiungere lo scopo definito.*
  - Comprovare che i dati personali siano sufficienti, pertinenti e non eccessivi rispetto all'intento; altrimenti, non raccogliere i dati.
  - Comprovare che i dati personali non rivelino (direttamente o indirettamente) l'origine razziale o etnica, le opinioni politiche, filosofiche o religiose, l'appartenenza sindacale, le informazioni sulla salute o le informazioni sulla vita sessuale di un individuo, tranne che per circostanze eccezionali.
  - Comprovare che i dati personali non si riferiscono a reati, sentenze penali o misure di sicurezza.
  - Evitare la raccolta di dati personali aggiuntivi.
  - Limitare la trasmissione di documenti elettronici contenenti dati personali allo stretto necessario.
  - Eliminare i dati personali che non sono più utili o le richieste di un soggetto dal sistema in esercizio o dai backup, se necessario.

- **Gestire i periodi di conservazione dei dati personali -** *Ridurre l'impatto dei rischi assicurando che i dati personali non vengano mantenuti per più di quanto necessario.*
  - Definire periodi di conservazione dei dati personali limitati nel tempo e appropriati allo scopo dell'elaborazione.
  - Verificare che l'elaborazione possa rilevare la scadenza del periodo di conservazione.
  - Verificare che l'elaborazione consenta la cancellazione dei dati personali a scadenza del periodo di conservazione e che il metodo scelto per l'eliminazione sia appropriato ai rischi legati alle libertà civili e la privacy dei soggetti interessati.
  - Eliminare immediatamente i dati personali quando il periodo di conservazione scade.

- **Informare i soggetti e ottenere il consenso -** *Consentire ai soggetti interessati di effettuare una scelta libera, specifica e informata.*
  - Determinare se l'elaborazione si basa su una base giuridica diversa dal consenso.
  - Determinare i mezzi pratici da attuare per ottenere il consenso degli interessati.
  - Assicurare che il consenso sia ottenuto prima che inizia l'elaborazione.
  - Assicurare che il consenso sia ottenuto liberamente.
  - Assicurare che il consenso sia ottenuto in modo notificato e trasparente in termini di finalità del trattamento.
  - Assicurare che il consenso sia ottenuto per uno scopo specifico.

- **Partizionare i dati personali - Ridurre la possibilità che i dati personali possano essere correlati e che possa verificarsi una violazione di tutti i dati personali.**

- Identificare i dati personali utili solo per ogni processo aziendale.
- Separare i dati utili di ogni processo in modo logico.
- Comprovare regolarmente che i dati personali sia partizionati in modo efficace e che i destinatari e le interconnessioni non siano stati associati.

- **Cifrare i dati personali -** *Rendere incomprensibili i dati personali a chiunque senza autorizzazione di accesso.*
  - Determinare tutto ciò che deve essere crittografato (incluso dischi rigidi, file, dati provenienti da un database o canali di comunicazione) in base alla forma in cui sono memorizzati i dati personali, i rischi individuati e le prestazioni richieste.
  - Scegliere il tipo di crittografia (simmetrica o asimmetrica) in base al contesto e ai rischi individuati.
  - Adottare soluzioni di crittografia basate su algoritmi pubblici notoriamente forti.
  - Definire misure per garantire la disponibilità, l'integrità e la riservatezza delle informazioni necessarie per recuperare le informazioni perse (incluse le password di amministratore e dati di ripristino).

- **Anonimizzare i dati personali -** *Eliminare le caratteristiche che identificano i dati personali.*
  - Determinare ciò che deve essere anonimo in base al contesto, alla forma in cui vengono memorizzati i dati personali (compresi i campi del database o estratti dai testi) e rischi individuati.
  - Anonimizzare permanentemente i dati che richiedono tale criterio di protezione in base alla forma dei dati (inclusi database e record testuali) e i rischi individuati.
  - Se questi dati non possono essere anonimizzati in modo permanente, scegliere strumenti (inclusi la cancellazione parziale, la cancellazione, la ricerca di hashing e l'indice) che rispondano innanzitutto alle esigenze funzionali.

4.5.3 Tecniche di modellazione e individuazione delle minacce

4.5.3.1 LINDDUN

La privacy sta diventando una questione chiave nell'e-society. È della massima importanza che la privacy sia integrata quanto prima nel ciclo di vita del software di sviluppo. LINDDUN[18] è una metodologia di analisi delle minacce alla privacy e supporta gli analisti nell'individuare i requisiti di riservatezza.

LINDDUN è un nome mnemonico sviluppato da Mina Deng [15] per il suo dottorato di ricerca alla Katholieke Universiteit di Leuven, Belgium. Questa è una metodologia speculare alla modellazione delle minacce STRIDE (STRIDE-per-element) e tratta le violazioni delle seguenti proprietà sulla privacy:

- Collegabilità (Linkability);
- Identificabilità (Identifiability);
- Non ripudio (Non Repudiation);
- Rilevabilità (Detectability);
- Divulgazione di informazioni (Disclosure of information);
- Inconsapevolezza sul contenuto (Content Unawareness);
- Inaderenza alla politica sul conseno (Policy and consent Non-compliance).

[18] https://www.linddun.org/

LINDDUN viene presentato come un approccio completo alla modellazione delle minacce con un metodo di individuazione di processi, minacce e requisiti. Può essere ragionevole utilizzare LINDDDUN come framework per l'identificazione delle minacce sulla privacy, in sostituzione o in aggiunta alla STRIDE.

In primo luogo, viene creato un diagramma di flusso dei dati (DFD), una rappresentazione grafica strutturata del sistema che utilizza quattro tipi principali di elementi: entità, archivi dati, flussi di dati e processi. Ciascun tipo di elemento DFD viene associato a una serie di categorie di minacce alla privacy (sono state identificate sette categorie di alto livello di minacce alla privacy: **L**-Linkability, **I**-Identifiability, **N**-Non Repudiation, **D**-Detectability, **D**-Disclosure of information, **U**-Content Unawareness e **N**-Policy and consent Non-compliance). Per identificare le minacce che insistono sul sistema analizzato, per ciascun elemento è necessario esaminare le minacce corrispondenti alle categorie di cui sopra.

La tabella seguente, mostra la correlazione tra le minacce di privacy previste da LINDDUN e le tipologie di elementi DFD sopra descritte:

Elemento DFD	L	I	N	D	D	U	N
Archivio dati	X	X	X	X	X		X
Flusso dati	X	X	X	X	X		X
Processo	X	X	X	X	X		X
Entità	X	X				X	

Tabella 17 - Minacce LINDDUN per elemento DFD

La metodologia LINDDUN supporta l'analista fornendo una serie di alberi di minaccia che descrivono i percorsi d'attacco più comuni per ogni possibile combinazione tra le tipologie di minaccia e gli elementi DFD. Basandosi su questi alberi, l'analista potrà documentare le minacce identificate, utilizzando scenari di casi di abuso per descrivere in dettaglio i possibili attacchi. Le minacce vengono quindi considerate prioritarie in base al loro rischio. Tuttavia non fornisce esplicitamente un supporto per l'analisi del rischio. Le minacce che derivano da tale processo, possono quindi essere tradotte in requisiti di riservatezza. Infine, LINDDUN fornisce un elenco di soluzioni per la privacy al fine di mitigare le minacce individuate.

La tabella seguente, riporta gli obiettivi di privacy basati sule varie tipologie di minaccia previste in LINDDUN, dove (**E**-Entity, **DF**-DataFlow, **DS**-DataStore, **P**-Process).

Minacce LINDDUN	Obiettivo elementare a tutela della privacy
Linkability of (E,E)	Unlinkability of (E,E)
Linkability of (DF,DF)	Unlinkability of (DF,DF)
Linkability of (DS,DS)	Unlinkability of (DS,DS)
Linkability of (P,P)	Unlinkability of (P,P)
Identifiability of (E,E)	Anonymity/pseudonymity of (E,E)
Identifiability of (E,DF)	Anonymity/pseudonymity of (E,DF)
Identifiability of (E,DS)	Anonymity/pseudonymity of (E,DS)
Identifiability of (E,P)	Anonymity/pseudonymity of (E,P)
Non-repudiation of (E,DF)	Plausibledeniability of (E,DF)
Non-repudiation of (E,DS)	Plausibledeniability of (E,DS)

Non-repudiation of (E,P)	Plausibledeniability of (E,P)
Detectability of DF	Undetectability of DF
Detectability of DS	Undetectability of DS
Detectability of P	Undetectability of P
Information Disclosure of DF	Confidentiality of DF
Information Disclosure of DS	Confidentiality of DS
Information Disclosure of P	Confidentiality of P
Content Unawareness of E	Content awareness of E
Policy and consent Noncompliance of the system	Policy and consent compliance of the system

Tabella 18 - obiettivi di privacy basati sule varie tipologie di minaccia previste in LINDDUN

4.5.3.1.1 Tecniche di mitigazione

Nella metodologia LINDDUN, le proprietà e la corrispettive minacce alla privacy vengono classificate come hard e soft privacy. La tabella a seguire evidenzia tale classificazione:

Controlli di privacy	Minaccia alla privacy
Hard privacy	
Unlinkability	Linkability
Anonymity & Pseudonymity	Indentifiability
Plausible deniability	Non repudiation
Undetectability & unobservability	Detectability
Confidentiality	Disclosure of information
Soft privacy	
Content awareness	Content Unawareness
Policy and consent compliance	Policy and consent non-compliance

Tabella 19 - LINDDUN Hard & Soft privacy

LINDDUN fornisce per ogni tipo di potenziale minaccia identificata una o più classificazioni delle tecniche di mitigazione da mettere in campo attraverso una mappatura tra obiettivi e tecniche di miglioramento della privacy (PETs):

	Tecniche di mitigazione	U	A	P	D	C	W	O
Anonymity system	• Mix-networks (1981) • DC-networks (1985) • ISDN-mixes • Onion Routing (1996) • Crowds (1998) • Single proxy (90s) (Penet pseudonymous remailer (1993-1996), Anonymizer, SafeWeb) • Anonymous Remailer (Cipherpunk Type 0, Type 1, Mixmaster Type 2 (1994), Mixminion Type 3 (2003)) • Low-latency communication (Freedom	X	X			X		

	Network (1999-2001), Java Anon Proxy (JAP) (2000), Tor (2004))							
	• DC-net & MIX-net + dummy traffic • ISDN-mixes	X	X		X	X		
	• Broadcast systems + dummy traffic	X	X		X			
Privacy preserving authentication	• Private authentication • Anonymous credentials (single show, multi show)	X	X					
	• Deniable authentication	X	X	X				
	• Off-the-record messaging	X	X	X		X		
Privacy preserving cryptographic protocols	Multi-party computation (Secure function evaluation)	X				X		
	Anonymous buyer-seller watermarking protocol	X	X			X		
Information retrieval	Private information retrieval + dummy traffic	X	X		X			
	Oblivious transfer	X	X			X		
	Privacy preserving data mining	X	X			X		
	• Searchable encryption • Private search		X			X		
Data anonymization	• K-anonymity model • l-Diversity	X	X					
Information hiding	Steganography	X	X		X			
	Covert communication	X	X		X			
	Spread spectrum	X	X		X			
Pseudonymity systems	Privacy enhancing identity management system	X	X					
	User-controlled identity management system	X	X					
	Privacy preserving biometrics	X	X			X		
Encryption techniques	Symmetric key & public key encryption					X		
	Deniable encryption			X		X		
	Homomorphic encryption					X		
	Verifiable encryption					X		
Access control techniques	Context-based access control					X		
	Privacy-aware access control					X		
Policy and feedback tools	Policy communication (P3P)							X
	Policy enforcement (XACML, EPAL)							X
	Feedback tools for user privacy awareness						X	
	Data removal tools (spyware removal, browser cleaning tools, activity traces eraser, harddisk data						X	

	eraser)							

Tabella 20 - Mappatura tra obiettivi e tecniche di miglioramento della privacy

* **U**–Unlinkability, **A**–Anonymity/Pseudonymity, **P**–Plausible deniability, **D**–Undetectability/unobservability, **C**–Confidentiality, **W**–Content Awareness, **O**–Policy and consent compliance of the system

4.5.3.2 PRoPAN

Beckers et al. [4] ha creato un approccio in quattro fasi per l'identificazione semiautomatica delle minacce alla privacy. Il metodo ProPAn (Problem-based Privacy Analysis) contribuisce nella produzione dei requisiti di protezione della privacy ed è un approccio basato su problemi per l'identificazione semiautomatica delle minacce alla privacy durante l'analisi dei requisiti dei sistemi software. L'obiettivo di questa metodologia è quello di assistere gli ingegneri del software nella requisitazione al fine di ottenere le seguenti informazioni:

- conoscenza del settore rilevante per la privacy;
- dati personali trattati;
- requisiti di riservatezza.

La metodologia consiste in quattro fasi:

- Disegno del diagramma di contesto e dei diagrammi dei problemi,
- Aggiunta dei requisiti di privacy al modello,
- Generazione di grafici delle minacce alla privacy,
- Analisi dei grafici delle minacce alla privacy.

Riferimento bibliografico: Kristian Beckers, Stephan Faßbender, Maritta Heisel, and Rene Meis. A Problem-based Approach for Computer Aided Privacy Threat Identification. In Privacy Technologies and Policy, volume 8319 of LNCS, pages 1–16. Springer, 2014.

4.5.3.3 PriS

PriS [5] è un metodo di ingegnerizzazione dei requisiti di sicurezza, che integra i requisiti di riservatezza già nelle fasi iniziali del processo di sviluppo del sistema. PriS considera i requisiti relativi alla privacy come obiettivi organizzativi che devono essere soddisfatti e adotta l'uso di modelli di processi di privacy come un modo per:

1. descrivere l'effetto dei requisiti relativi alla privacy sui processi aziendali;
2. facilitare l'identificazione dell'architettura di sistema che meglio supporta i processi aziendali in relazione agli aspetti di privacy.

In questo modo, PriS fornisce un approccio olistico che va dagli obiettivi di alto livello ai sistemi informatici "rispettosi della privacy". Il metodo si articola in quattro fasi:

- individuare gli obiettivi di tutela della privacy,
- analizzare l' impatto degli obiettivi di tutela della privacy sui processi organizzativi,
- modellare i processi interessati utilizzando modelli di tutela della privacy
- identificare le tecniche che meglio supportano o implementano i processi summenzionati.

Riferimento bibliografico: Christos Kalloniatis, Evangelia Kavakli, and Stefanos Gritzalis. Addressing privacy requirements in system design: the PriS method. Requirements Engineering, 13(3):241–255, 2008.

4.5.3.4 FPFSD

Spiekermann e Cranor [6] hanno sviluppato un framework denominato (FPFSD). Questo è un framework per la progettazione di sistemi rispettosi della privacy (framework for privacy-friendly system design) in cui si attua una distinzione tra due diversi approcci alla privacy:

- Privacy-by-policy, introduce l'approccio di notifica e consenso basato sui principi di Fair Information Practice Principles (FIPP) e implica che l'utente sia informato su quali informazioni vengono utilizzate e perché. Inoltre, l'utente può decidere di non fornire dati.
- Privacy-by-architecture, incoraggia l'archiviazione dei dati presso il cliente invece di far archiviare le informazioni riservate dalle aziende stesse.

Riferimento bibliografico: Sarah Spiekermann and Lorrie F. Cranor. Engineering privacy. IEEE Transactions on Software Engineering, 35(1):67–82, 2009.

4.5.3.5 MPRA

Gürses [7] propone una tecnica multilaterale per l'analisi dei requisiti in materia di tutela della privacy (multilateral privacy requirements analysis technique). Si articola in tre fasi principali: analisi degli stakeholder, analisi funzionale e analisi della privacy. In primo luogo vengono determinati gli attori e le parti interessate. Per ciascuno di questi, vengono determinati gli obiettivi funzionali che vengono poi collegati alle ipotesi di dominio. Viene inoltre creato un modello informativo. Nella fase finale, le problematiche sulla privacy sono determinate in base a ciascun stakeholder, in relazione al modello informativo e agli obiettivi funzionali individuati nella fase precedente. Tali problematiche possono anche tradursi in minacce alla privacy e documentate come casi di abuso. Infine, le problematiche (o minacce) vengono trasformate in obiettivi di tutela privacy come un'approssimazione giustificabile.

Riferimento bibliografico: Fahriye Seda Gürses. Multilateral privacy requirements analysis in online social network services. PhD thesis, Department of Computer Science, KU Leuven, 2010.

4.5.3.6 Privacy in the Cloud

Mouratidis et al. [8] pone in particolare l'attenzione sulle applicazioni cloud proponendo un framework che supporta l'elicitazione dei requisiti di sicurezza e privacy. Il framework è composto da un linguaggio (basato su Secure Tropos) e da un processo (basato su PriS) a supporto dell'analisi della sicurezza e della privacy. Il processo si articola in tre fasi. Il primo passo (facoltativo) è la catalogazione delle minacce alla sicurezza e alla privacy, che mira a creare un punto di riferimento basato sull'esperienza passata da utilizzare poi successivamente. In secondo luogo, l'attività di analisi della sicurezza e della privacy consiste in due sotto-attività: la definizione del contesto organizzativo, in cui vengono identificati gli obiettivi organizzativi, gli attori e le dipendenze, i piani e le risorse e gli obiettivi di sicurezza e privacy; e la definizione delle possibili problematiche in materia di sicurezza e privacy, che consiste nell'identificare i requisiti, le misure e i meccanismi di sicurezza e privacy. Il terzo e ultimo passo è la selezione del provider dei servizi cloud in base al grado di soddisfazione dei meccanismi di sicurezza e privacy ottenuti dai potenziali fornitori cloud. Sebbene si tratti di un framework che fornisce procedure dettagliate per analizzare i requisiti di sicurezza e privacy, non è disponibile alcuna conoscenza reale della sicurezza e della privacy.

Riferimento bibliografico: Haralambos Mouratidis, Shareeful Islam, Christos Kalloniatis, and Stefanos Gritzalis. A framework to support selection of cloud providers based on security and privacy requirements. Journal of Systems and Software, pages 2276–2293, 2013.

4.5.3.7 Adaptive privacy

Omoronyia et al. [9] propone un framework di riferimento per supportare la divulgazione selettiva delle informazioni personali da parte di applicazioni software in un contesto in continuo cambiamento. Il framework si concentra sui requisiti di sensibilizzazione alla privacy (PAR - privacy awareness requirements) e descrive:

1. come identificare gli attributi da monitorare per rilevare le minacce alla privacy,
2. come scoprire le minacce alla privacy prima della divulgazione delle informazioni personali,
3. come determinare la gravità, così come i benefici relativi a una minaccia scoperta.

Il quadro normativo richiede tuttavia che i requisiti di riservatezza di tutti gli agenti siano già definiti e non fornisce indicazioni su come ottenerli.

Riferimento bibliografico: Inah Omoronyia, Luca Cavallaro, Mazeiar Salehie, Liliana Pasquale, and Bashar Nuseibeh. Engineering adaptive privacy: on the role of privacy awareness requirements. In Proceedings of the 2013 International Conference on Software Engineering, ICSE '13, pages 632–641. IEEE Press, 2013.

4.5.3.8 STRAP

STRAP [10] è un framework di analisi della privacy che è stato sviluppato sulla base dei risultati dell'analisi di sei framework esistenti (modelli di rischio [16], Patrick e Kenny [17], framework i* [18], Langheinrich [19], Bellotti e Sellen [20], e valutazione euristica [21]). Si tratta di un metodo a cinque fasi, che consiste in una analisi orientata agli obiettivi, analisi della vulnerabilità, perfezionamento e progettazione degli obiettivi, valutazione del progetto e iterazione. La fase di analisi della vulnerabilità è la fase principale in cui vengono combinati i framework esistenti. L'analisi si basa su una serie di domande analitiche per determinare la cattura e l'uso delle informazioni. In secondo luogo, viene utilizzata l'euristica per identificare potenziali problemi basati su difetti e requisiti comuni. Queste euristiche possono essere classificate secondo le FIPP degli Stati Uniti: notice/awareness, choice/consent, integrity/security, enforcement/redress. Sebbene queste categorie, basate sulla legislazione e sugli orientamenti in materia di protezione dei dati, siano molto importanti dal punto di vista della tutela della privacy, non tengono conto delle proprietà fondamentali quali l'anonimato, la non collegabilità e la non rilevabilità.

Riferimento bibliografico: Carlos Jensen. Designing for privacy in interactive systems. PhD thesis, Georgia Institute of Technology, 2005.

4.5.3.9 Microsoft privacy guidelines

Le linee guida sulla privacy fornite da Microsoft descrivono alcuni concetti basilari di privacy [11], come diversi tipi di consenso o concetti di minimizzazione dei dati. Inoltre, per gli scenari selezionati vengono presentate alcune linee guida riguardanti i seguenti principi: avviso, scelta, trasferimento successivo, accesso, sicurezza e integrità dei dati. Tuttavia, contiene solo un elenco piatto degli orientamenti richiesti e raccomandati e non intende descrivere un approccio più strutturato. Queste linee guida possono essere utilizzate come ispirazione per determinare le possibili minacce, ad esempio per ampliare il catalogo degli alberi delle minacce[19].

Riferimento bibliografico: Privacy guidelines for developing software products and services, version 3.1. Technical report, Microsoft Coorporation, Sept 2008. http://download.microsoft.com/download/0/8/2/082448D8-2AED-45BC-A9A0-094840E9E3A2/Microsoft_and%20Privacy_guidelines_for_developers.doc.

4.5.3.10 PRET

Miyazaki et al. [12] ha definito una tecnica di elicitazione dei requisiti di riservatezza informatizzata, denominata (PRET - privacy requirements elicitation technique). Questa tecnica restituisce i requisiti necessari affinché il sistema sia conforme alla legge, sulla base di un questionario compilato dall'ingegnere di sistema.

Riferimento bibliografico: Seiya Miyazaki, Nancy Mead, and Justin Zhan. Computer-aided privacy requirements elicitation technique. Asia-Pacific Conference on Services Computing (APSCC'08), pages 367–372, 2008.

[19]http://download.microsoft.com/download/0/8/2/082448D8-2AED-45BC-A9A0-094840E9E3A2/Microsoft_and%20Privacy_guidelines_for_developers.doc.

5 LINEE GUIDA PER L'INDIVIDUAZIONE E LA RIVISITAZIONE DEI REQUISITI DI SICUREZZA E DI PRIVACY APPLICATIVI

5.1 Linee guida per la modellazione delle minacce

5.1.1 Identificazione degli obiettivi di sicurezza

La sicurezza dei dati è assicurata quando vengono assicurate tre caratteristiche di fruizione di questi ultimi, che sono la riservatezza, l'integrità e la disponibilità.

Per raggiungere la sicurezza vengono eseguite azioni, che in caso di:

- riservatezza, proteggono i dati al fine di contrastare la divulgazione non autorizzata;
- integrità, contrastano le modifiche non autorizzate dei dati;
- disponibilità, contrastano la indisponibilità malevola dei dati/servizi.

Gli obiettivi specifici di sicurezza sono un sottoinsieme degli obiettivi di progetto e dovrebbero essere utilizzati per guidare gli sforzi impiegati nella modellazione delle minacce.

Identificare i principali obiettivi di sicurezza permette di concentrarsi con maggiore attenzione sulle aree da proteggere. Ad esempio, se si identificano i dettagli del profilo cliente come dati riservati, che devono essere protetti, è possibile esaminare la modalità di archiviazione sicura di tali dati e il modo in cui l'accesso a tali dati viene controllato e verificato.

Per determinare gli obiettivi di protezione, occorre porsi le seguenti domande:

- quali dati occorre proteggere?
- Esistono requisiti di conformità? I requisiti di conformità possono includere criteri di protezione, leggi sulla privacy, regolamenti e standard.
- Esistono requisiti di qualità specifici del servizio? I requisiti di qualità del servizio includono tipicamente la disponibilità e i requisiti prestazionali.
- Esistono beni immateriali che devono essere protetti? Tali beni includono ad esempio, la reputazione dell'organizzazione, le informazioni commerciali sensibili e la proprietà intellettuale.

Di seguito sono riportati alcuni esempi di obiettivi di sicurezza comuni:

- impedire agli aggressori di ottenere dati sensibili, inclusi i codici di accesso e le informazioni sul profilo.
- Soddisfare gli accordi a livello di servizio per la disponibilità delle applicazioni.
- Proteggere la credibilità dell'organizzazione.

5.1.2 Creazione di un disegno ad alto livello dell'applicazione

La modellazione delle minacce è un processo iterativo di analisi, dove ad ogni ciclo si scende sempre più in dettaglio, identificando di livello in livello le funzionalità chiave dell'applicazione, le sue caratteristiche ed i dati da proteggere.

Per avere una panoramica dell'applicazione occorre:

- Disegnare lo scenario di sviluppo dall'inizio alla fine;
- Identificare i ruoli;
- Identificare gli scenari d'uso più significativi;
- Identificare le tecnologie;
- Identificare i meccanismi di sicurezza.

Di seguito sono riportati i dettagli di ciascuna fase.

Disegnare lo scenario di sviluppo dall'inizio alla fine - La prima attività consiste in una modellazione ad alto livello dell'applicazione (composizione e struttura dell'applicazione, relativi sottosistemi e caratteristiche di distribuzione). Dopo il primo disegno, si aggiungono i dettagli sui meccanismi di autenticazione, autorizzazione e comunicazione. Da notare che quando si inizia la modellazione non sempre si è a conoscenza di tutti i dettagli per cui deve essere sempre possibile ritornare sull'aspetto sotto analisi in un secondo momento per approfondire ulteriormente.

La figura che segue riporta l'esempio di un diagramma iniziale che mostra l'architettura di una applicazione con alcuni dettagli.

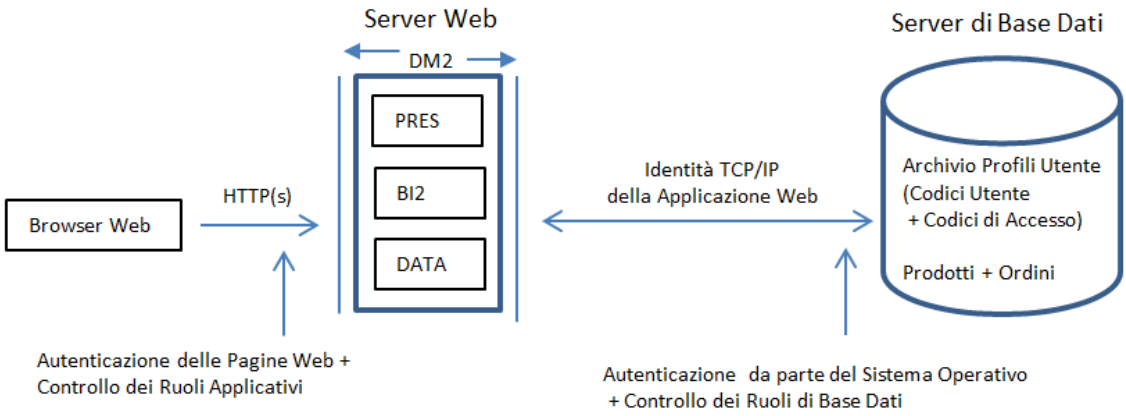


Figura 8 - Esempio di disegno architetturale di una applicazione

In generale il disegno architetturale deve evidenziare i seguenti elementi:

- Topologia fisica e logica dei componenti: il collocamento dei server in rete (Intranet, Extranet e accesso a Internet). Iniziare con le topologie di rete logiche, per poi visualizzare le relative topologie fisiche quando si dispone di tali dettagli. È possibile aggiungere o rimuovere le minacce a seconda della scelta di topologie fisiche.
- Livelli logici: il livello di presentazione (front-end), il livello di business (back-end) e i livelli di accesso ai dati. Successivamente occorre rifinire per includere, una volta noti, i limiti fisici del server;
- Componenti chiave: i componenti importanti all'interno di ogni livello logico. In questa fase è possibile includere i limiti reali del componente e di processo una volta conosciuti;
- Servizi chiave: eventuali servizi importanti. Una volta noti, da mostrare come processi;
- Porte e protocolli di comunicazione: i server, i componenti e i servizi che comunicano tra di loro e come lo fanno. Da mostrare le specifiche dei pacchetti informativi in entrata e in uscita, una volta individuati;
- Identità: le identità utente principali usate all'interno dell'applicazione e gli eventuali profili di servizio rilevanti;
- Dipendenze esterne: le dipendenze dell'applicazione da eventuali sistemi esterni. Elencare queste dipendenze è utile per individuare eventuali vulnerabilità che potrebbero insorgere in un secondo momento se alcune assunzioni fatte inizialmente sul generico sistema esterno dovessero risultare non più vere o in qualche modo cambiate.

È importante revisionare il disegno del sistema nel corso del tempo per verificare se tutti gli elementi individuati siano ancora come descritti, se devono essere cambiati o se hanno bisogno di un ulteriore livello di dettaglio.

5.1.2.1 Identificazione dei Ruoli

È importante identificare i ruoli all'interno dell'applicazione, ossia, chi può fare cosa.

La fase di identificazione dei ruoli è utilizzata sia per determinare ciò che dovrebbe accadere (accesso alle risorse autorizzate come stabilito per lo specifico ruolo) che per determinare ciò che non dovrebbe accadere (accesso a risorse per le quali non si ha l'autorizzazione).

L'assegnazione dei ruoli deve essere centralizzata e a ciascun ruolo deve essere associato un profilo 'autorizzativo' che regolamenta i comandi, le transazioni e gli accessi ai dati.

Per ridurre la superficie d'attacco è necessario stabilire quali devono essere i privilegi minimi per ogni componente dell'applicazione. Nella gestione della sicurezza delle risorse (quali DB, Active Directory, file, etc.), i ruoli applicativi devono essere quindi stabiliti a partire dalla logica di business con l'obiettivo di delimitare il perimetro di azione nell'accesso alla risorsa.

5.1.2.2 Identificare gli Scenari d'Uso Chiave

In questa fase occorre identificare le principali funzionalità e modalità d'uso e dettagliare gli aspetti legati alle attività di creazione, lettura, aggiornamento e cancellazione dei dati. Le caratteristiche chiave vengono spesso descritte nel contesto dei casi d'uso e permettono di far capire come l'applicazione è destinata ad essere utilizzata e come può essere utilizzata in modo improprio. I casi d'uso consentono di identificare i flussi di dati e di focalizzarsi sull'analisi di eventuali minacce nelle fasi successive di dettaglio della modellizzazione.

5.1.2.3 Identificare le Tecnologie

Occorre identificare tutte le tecnologie utilizzate e le loro caratteristiche: Sistemi operativi; Server Web; Server di Base Dati; le tecnologie utilizzate per implementare la presentazione dei dati a livello utente, per gestire le regole di business, per l'accesso ai dati sottostanti e il linguaggio di sviluppo utilizzato.

L'identificazione delle tecnologie consente di concentrarsi in un momento successivo alla modellizzazione delle minacce che possono nascere, legate alle specifiche tecnologie in uso. Inoltre, aiuta a determinare le eventuali tecniche di mitigazione del rischio.

5.1.2.4 Identificare Meccanismi di Sicurezza Applicativa

Un'altra fase importante è l'identificazione dei meccanismi di sicurezza applicativa, in particolare occorre analizzare i seguenti aspetti:

- Validazione input e dati;
- Autenticazione;
- Autorizzazione;
- Gestione della configurazione;
- Dati sensibili;
- Gestione della sessione;
- Crittografia;
- Manipolazione dei parametri;
- Gestione delle eccezioni;
- Audit e gestione dei log.

Lo scopo dell'identificazione di questi elementi è quello di aggiungere il più possibile ulteriori dettagli, anche poco conosciuti. Ad esempio è utile documentare come l'applicazione viene autenticata dalla base dati o come gli utenti vengono autorizzati, oppure quali sono le aree preposte all'autenticazione e l'autorizzazione.

5.1.3 Scomposizione dell'applicazione

La scomposizione dell'applicazione è utile per scoprire le minacce e le vulnerabilità del sistema. Nell'ottica di sicurezza, i componenti più importanti sono:

- confini di fiducia (trust boundaries);
- flussi di dati;
- punti di ingresso (entry points);
- punti di uscita (exit points).

5.1.3.1 Confini di fiducia (Trust boundaries)

Identificare i confini di fiducia dell'applicazione aiuta a concentrare l'analisi sulle aree di maggiore interesse. I confini di fiducia evidenziano dove cambiano i livelli di fiducia. In quest'ambito, la fiducia è intesa in chiave di riservatezza e integrità. Ad esempio, una modifica nei livelli di controllo di accesso all'applicazione, dove è necessario un livello o un privilegio specifico per accedere a una risorsa o un'operazione, sarebbe una modifica del livello di fiducia. Un altro esempio, potrebbe essere in un punto di entrata nell'applicazione ove è necessario filtrare i dati di accesso.

Per identificare i confini di fiducia occorre:

- Iniziare individuando i confini del sistema esterno. Ad esempio, l'applicazione può scrivere un file sul server X, può effettuare chiamate al database sul server Y e può chiamare il servizio Web Z. Questo definisce il tuo limite di sistema.
- Identificare i punti di controllo di accesso o i luoghi chiave in cui l'accesso richiede privilegi aggiuntivi o l'appartenenza ad un dato ruolo. Ad esempio, l'accesso ad una pagina particolare, potrebbe essere limitata ai soli dirigenti, nel qual caso richiederebbe un accesso autenticato e inoltre che l'utente ricopra un certo ruolo.
- Identificare i confini di fiducia da una prospettiva di flusso di dati. Per ogni sottosistema, considerare se il flusso di dati a monte o l'input dell'utente sia affidabile e se non lo è, considerare in che modo il flusso di dati e l'input possono essere autenticati e autorizzati. Conoscere quali punti di ingresso esistono tra i confini di fiducia, consente di concentrare l'identificazione delle minacce in questi punti chiave di accesso.

Alcuni esempi di confini di fiducia sono: un firewall perimetrale, il confine tra il web server e il server di base dati, punti di ingresso di componenti aziendali che espongono dati privilegiati, dunque, protetti da ulteriori controlli di accesso, il limite tra l'applicazione e i servizi di terze parti.

5.1.3.2 Flussi di Dati

È importante tracciare il flusso dei dati all'interno dell'applicazione dal punto di ingresso al punto d'uscita. Questa attività è necessaria per comprendere come interagisce l'applicazione con i sistemi esterni, i sistemi client e come interagiscono i componenti interni. È importante anche, prestare particolare attenzione al flusso di dati che attraversa i confini di fiducia e come tali dati vengono convalidati nel punto di entrata. Inoltre occorre fare molta attenzione ai dati sensibili e come questi attraversano il sistema, se passano attraverso una rete e/o se sono persistenti. Un buon approccio è quella di analizzare il flusso dei dati tra i singoli sottosistemi a partire dal livello più alto e poi via via a scendere ai livelli più bassi.

5.1.3.3 Punti d'Ingresso (Entry Points)

I punti di ingresso dell'applicazione servono anche come punti di ingresso per gli attacchi. Il front-end di una applicazione web che è in ascolto di richieste http è un esempio di punto di ingresso vulnerabile agli attacchi. Questo punto di ingresso è destinato ad essere esposto agli utilizzatori. Altri punti di ingresso, come i punti di accesso interni esposti dai sotto componenti negli strati dell'applicazione, possono esistere solo per supportare la comunicazione interna con altri componenti. Tuttavia, occorre conoscere dove sono

localizzati e quali tipi di input ricevono nel caso in cui un aggressore riesca ad aggirare l'interfaccia dell'applicazione e attaccare direttamente un punto di ingresso interno.

A titolo di esempio si elencano di seguito ulteriori esempi di Entry Points:

- Front-end applicativo (form di Login, form di Ricerca);
- Funzioni applicative (funzione di login, web service esposti, ..);
- Console di amministrazione del database;
- External reporting;
- Porte TCP/UDP (network socket).

5.1.3.4 Punti di Uscita (Exit Points)

È importante identificare da dove l'applicazione invia i dati all'utente o ai sistemi esterni, dando priorità ai punti di uscita in cui l'applicazione scrive dati che includono l'input proveniente dall'utente o dati provenienti da fonti non attendibili, ad esempio basi dati condivise.

A titolo di esempio si elencano di seguito ulteriori esempi di Exit Points:

- File di Log;
- Database;
- Interfacce esterne (es. web service).

5.1.4 Identificazione delle minacce

In questa fase, è possibile individuare minacce e attacchi che potrebbero compromettere l'applicazione e gli obiettivi di sicurezza. Il processo di identificazione consiste in sessioni di brainstorming tra i team di sviluppo e test. Idealmente, il team di lavoro è costituito da architetti applicativi, professionisti della sicurezza, sviluppatori, tester e amministratori di sistema.

Esistono due approcci per affrontare questa fase:

- Iniziare, elencando minacce e attacchi comuni. Con questo approccio, si inizia con un elenco di minacce comuni raggruppate per categorie di vulnerabilità. Successivamente, occorre adattare tale elenco alla propria architettura. Ad esempio, utilizzare gli scenari identificati per esaminare i flussi di dati, prestando particolare attenzione ai punti di ingresso e in particolare a quelli che attraversano i confini di fiducia. In questo modo si potranno eliminare immediatamente alcune minacce, in quanto non applicabili ai casi d'uso.
- Utilizzare un approccio basato su domande. Un approccio basato su questionari può aiutare a identificare le minacce e i possibili attacchi. La categorizzazione STRIDE si basa su categorie di minacce molto estese, quali spoofing (assunzione impropria di identità), manomissione di dati, ripudio, divulgazione indesiderata di informazioni e interruzione di servizio. Il modello STRIDE deve essere usato per porre domande relative a qualsiasi aspetto dell'architettura e del design dell'applicazione. Questo è un approccio basato su obiettivi, in cui si prendono in considerazione tutti gli obiettivi di un possibile aggressore (punto di vista di un attaccante).

Per identificare le minacce, si esaminano tutti i livelli dell'applicazione, livello per livello e funzione per funzione. Ponendo l'attenzione sulle categorie di vulnerabilità, ci si concentra sulle aree in cui vengono spesso effettuati errori di sicurezza. Occorre identificare le potenziali minacce e le possibili azioni che un aggressore potrebbe tentare di utilizzare per sfruttare le vulnerabilità a cui l'applicazione è esposta. Durante questa attività di identificazione delle minacce si eseguono le seguenti attività:

- Identificazione delle minacce e degli attacchi comuni.
- Identificazione delle minacce annidate nei casi d'uso.
- Identificazione delle minacce annidate nei flussi di dati.

5.1.4.1 Identificazione delle minacce e attacchi comuni

Esistono una serie di minacce e attacchi comuni che si basano su vulnerabilità di carattere comune. Questa sezione identifica una serie di domande chiave da porsi per ciascuna categoria.

Autenticazione:

- Come potrebbe un aggressore rubare una identità?
- Come potrebbe un utente malintenzionato accedere all'archivio delle credenziali?
- Come potrebbe un aggressore portare un attacco? Come vengono memorizzate le credenziali dell'utente e quali criteri di codici di accesso vengono applicati?
- Come può un utente malintenzionato modificare, intercettare o eludere il meccanismo di reimpostazione delle credenziali dell'utente?

Autorizzazione:

- Come potrebbe un utente malintenzionato influenzare i controlli di autorizzazione per accedere a operazioni privilegiate?
- Come potrebbe un utente malintenzionato elevare i privilegi?

Input e dati di convalida:

- Come potrebbe un utente malintenzionato iniettare comandi SQL?
- Come potrebbe un utente malintenzionato eseguire un attacco di cross-site scripting?
- Come potrebbe un utente malintenzionato ignorare la validazione degli input?
- Come potrebbe un utente malintenzionato inviare un input non valido per influenzare la logica di protezione adottata sul server?
- Come potrebbe un utente malintenzionato inviare un errore di input per bloccare l'applicazione?

Gestione della configurazione:

- Come potrebbe un utente malintenzionato accedere alle funzioni di amministratore delle configurazioni?
- Come potrebbe un utente malintenzionato accedere ai dati di configurazione dell'applicazione?

Dati sensibili:

- Dove e come l'applicazione memorizza i dati sensibili?
- Quando e in quale punto i dati sensibili vengono passati attraverso la rete?
- Come potrebbe un utente malintenzionato visualizzare i dati sensibili?
- Come potrebbe un utente malintenzionato manipolare i dati sensibili?

Gestione delle sessioni:

- Si utilizza un algoritmo di crittografia personalizzato e ci si fida di tale algoritmo?
- Come potrebbe un aggressore prendere il controllo di una sessione utente?
- Come potrebbe un utente malintenzionato visualizzare o modificare lo stato della sessione di un altro utente?

Crittografia:

- Di cosa ha bisogno un aggressore per superare il meccanismo di crittografia adottato?
- Come potrebbe un utente malintenzionato ottenere l'accesso alle chiavi crittografiche?
- Quali standard crittografici si stanno utilizzando? Quali sono gli attacchi noti su questi standard?
- Si vuole adottare un proprio meccanismo di crittografia?

- In che modo la tipologia di distribuzione potenzialmente influenzerà la scelta dei metodi crittografici?

Manipolazione dei parametri:

- In che modo un aggressore potrebbe manipolare i parametri per influenzare la logica di protezione implementata sul server?
- Come potrebbe un utente malintenzionato manipolare i dati sensibili presenti nei parametri?

Gestione delle eccezioni:

- Come potrebbe un utente malintenzionato bloccare l'applicazione?
- Come potrebbe un utente malintenzionato ottenere dettagli utili ai propri fini?

Revisione e registrazione (audit):

- Come potrebbe un aggressore coprire le sue tracce?
- Come si può dimostrare che un utente malintenzionato (o un utente legittimo) ha eseguito azioni specifiche?

5.1.4.2 Identificazione delle potenziali minacce annidate nei casi d'uso

Occorre esaminare i casi d'uso chiave, che sono stati individuati nella fasi precedenti, per capire il modo in cui un utente potrebbe influenzare malevolmente o involontariamente l'applicazione ad eseguire un'operazione non autorizzata o a divulgare dati riservati o privati. In questa fase ci si pongono domande immedesimandosi nella figura dell'aggressore. Alcuni esempi di domande da porsi:

- Come può un utente iniettare un input dannoso in questo caso d'uso?
- I dati vengono pubblicati in base all'input da parte dell'utente o in base all'input non validato da parte dell'utente?
- In che modo un aggressore potrebbe manipolare i dati della sessione?
- Come potrebbe un utente malintenzionato ottenere dati sensibili quando vengono trasmessi attraverso la rete?
- In che modo un aggressore potrebbe eludere i controlli di autorizzazione?

5.1.4.3 Identificazione delle potenziali minacce annidate nei flussi di dati

Occorre rivedere i casi d'uso e gli scenari chiave e analizzare i flussi di dati, in particolare, i flussi di dati tra i singoli componenti dell'architettura. Il flusso di dati che attraversa i confini di fiducia è richiede particolare attenzione. Nella stesura del codice, si deve assumere che, tutti i dati esterni al confine di fiducia dell'applicativo siano dannosi. Nel codice si dovrebbe eseguire una validazione dei dati molto esaustiva. Per identificare le minacce associate ai flussi di dati, ci si deve porre le seguenti domande:

- Quale è il percorso del flusso di dati dal front-end al back-end dell'applicazione?
- Quali componenti chiamano altri componenti?
- Quale aspetto hanno i dati validi?
- Dove viene eseguita la validazione dei dati?
- Quali sono i vincoli imposti ai dati?
- Come vengono validati i dati in base ai parametri previsti in termini di lunghezza, intervallo valori, formato e tipo?
- Quali dati sensibili vengono trasmessi tra i componenti dell'applicazione e attraverso le reti, e come vengono protetti durante il transito?

L'uso della documentazione quali ad esempio, i diagrammi DFD e i diagrammi di sequenza UML, aiuta nell'analisi dell'applicazione e nell'identificazione dei flussi di dati.

5.1.4.4 Esplorare minacce ulteriori usando gli alberi delle minacce/attacchi

Vedi Paragrafo 5.2.4.2.

5.1.5 Identificazione delle vulnerabilità

Così come è stato fatto nel processo di identificazione delle minacce, si fornisce di seguito una rassegna delle diverse categorie di vulnerabilità. In questa fase, l'obiettivo è quello di analizzare le vulnerabilità (le minacce sono state già trattate nel paragrafo precedente) partendo dal principio che per poter analizzare correttamente un'applicazione è necessario valutare le sue vulnerabilità ad ogni livello.

Di seguito alcuni esempi di domande da porsi a secondo della fase.

Autenticazione:

- I nomi utente e le password sono inviati in chiaro su un canale non protetto? Si utilizza una crittografia ad hoc per informazioni sensibili?
- Le credenziali sono memorizzate? Se vengono memorizzate, come vengono memorizzate e protette?
- Viene imposto l'uso di strong authentication? Quali altre norme sull'uso dei codici di accesso vengono applicate?
- Come vengono verificate le credenziali?
- Come viene identificato l'utente autenticato dopo l'accesso iniziale?

Occorre rivedere l'autenticazione cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Le credenziali di autenticazione o cookie di autenticazione vengano passate su collegamenti di rete non crittografati, che possono portare al furto delle credenziali o della sessione;
- Si fa uso di codici d'accesso e procedure di accesso deboli, che possono portare ad un accesso non autorizzato;
- Si fa uso di dati personali come forma di autenticazione.

Autorizzazione:

- Quali controlli di accesso vengono utilizzati nei punti di accesso dell'applicazione?
- L'applicazione prevede l'uso di ruoli? Se utilizza ruoli, sono sufficientemente granulari ai fini del controllo degli accessi?
- L'inserimento erroneo di un codice di accesso fallisce in maniera sicura?
- È possibile limitare l'accesso alle risorse di sistema?
- Vengono attuate politiche restrittive nell'accesso alla base dati?
- Quale procedura di autorizzazione viene adottata a livello di base dati?

Occorre esaminare l'autorizzazione cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Si fa uso di ruoli e profili utente sovra privilegiati;
- Granularità di ruoli insufficiente;
- Non viene attuata alcuna restrizione di accesso alle risorse di sistema o viene attuata limitatamente ad alcuni profili.

Inserimento e Validazione Dati:

- Occorre identificare le vulnerabilità nella convalida dei dati di ingresso e dei dati in generale, domandandosi quanto segue:

  - I dati di ingresso vengono tutti validati?
  - I dati di ingresso, vengono validati in termini di lunghezza, intervallo, formato e tipo?
  - Si fa affidamento sulla sola validazione lato client?
  - Un aggressore potrebbe iniettare comandi o dati dannosi nell'applicazione?
  - Non si usano particolari accorgimenti nella scrittura dei dati all'interno delle pagine Web o piuttosto si preferisce codificarli in HTML per impedire gli attacchi di cross-site scripting?
  - L'input, viene validato prima di essere utilizzato nella composizione delle istruzioni SQL al fine di impedirne l'iniezione di malware?
  - I dati vengono validati al punto di ingresso del componente destinatario in virtù del fatto che oltrepassano sia il confine di fiducia del mittente che del destinatario?
  - Ci si può fidare dei dati presenti nel database?
  - Si accettano come input, i nomi dei file, gli URL e i nomi utente? Sono state considerate le problematiche di canonicalizzazione?
- Occorre rivedere la convalida degli input cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:
  - Ci si affida soltanto alla validazione lato client;
  - Viene eseguito un controllo a posteriori dei dati già immessi anziché un filtraggio dinamico al momento dell'inserimento;
  - Si fa uso di dati non validati indirizzati a pagine Web;
  - Si fa uso di input non validato per generare query SQL;
  - All'interno del codice sorgente, si fa uso di tecniche di accesso ai dati non sicure, che possono aumentare il rischio di minaccia da iniezione SQL;
  - Le decisioni intraprese nelle procedure di sicurezza si basano sui nomi di file, URL o nomi utente forniti in input.

Gestione di Configurazione:

- Come vengono protette le interfacce di amministrazione remote?
- Si proteggono gli archivi di configurazione?
- I dati di configurazione sensibili vengono crittografati?
- I privilegi di amministratore vengono separati?
- Vengono usati profili utente di processo e servizio con privilegi limitati?

Rivedere la gestione delle configurazioni cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- I dati confidenziali di configurazione, quali le stringhe di connessione e le credenziali del profilo utente di servizio, vengono memorizzati in chiaro;
- Non esiste protezione negli aspetti di gestione della configurazione dell'applicazione, incluse le interfacce di amministrazione;
- Si fa uso di profili utente di processo e profili utente di servizio con privilegi non strettamente necessari.

Dati Sensibili:

- I dati confidenziali vengono memorizzati in archivi di persistenza?
- I dati sensibili, come vengono storicizzati?
- I dati confidenziali presenti in memoria, vengono storicizzati?
- Si trasferiscono dati sensibili attraverso la rete?
- I dati sensibili vengono registrati nei log dell'applicazione?

Rivedere i dati sensibili cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- I dati confidenziali vengono memorizzati quando non è necessario memorizzarli;
- I dati confidenziali vengono direttamente memorizzati nel codice dell'applicazione;
- I dati confidenziali vengono memorizzati in chiaro;
- I dati sensibili vengono passati in chiaro sulle reti.

Gestione della Sessione:

- Come vengono generati i cookie di sessione?
- Come vengono scambiati gli identificatori di sessione?
- Come viene protetto lo stato della sessione mentre attraversa la rete?
- Come viene protetto lo stato della sessione per impedire il furto di sessione?
- Come viene protetto lo stato della sessione?
- La durata della sessione viene limitata?
- In che modo l'applicazione esegue l'autenticazione utilizzando l'archivio delle sessioni?
- Le credenziali sono trasmesse attraverso la rete e vengono mantenute all'interno dell'applicazione? Se sì, come vengono protette?

Controllare gli aspetti di gestione della sessione con il fine di individuare una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Gli identificatori di sessione vengono passati su canali non crittografati;
- La durata della sessione è prolungata;
- Gli stati di sessione non vengono protetti;
- Gli identificatori di sessione vengono passati nelle stringhe di query.

Crittografia:

- Quali algoritmi e tecniche di crittografia vengono usate?
- Vengono impiegati algoritmi di crittografia personalizzati?
- Perché vengono utilizzati determinati algoritmi?
- Quale è la durata di validità prevista per le chiavi crittografiche e come queste vengono protette?
- Quante volte vengono riciclate le chiavi?
- Come vengono distribuite le chiavi crittografiche?

Occorre esaminare la crittografia cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Vengono impiegati algoritmi crittografici personalizzati;
- Si fa uso di un algoritmo errato o di una chiave di dimensione troppo piccola;
- Le chiavi crittografiche non vengono protette;
- Si fa uso della stessa chiave per un periodo di tempo prolungato.

Manipolazione dei Parametri:

- Tutti i valori dei parametri di ingresso vengono validati?
- Tutti i valori dei parametri inseriti nei campi della pagina GUI, nei dati dei cookie e nelle intestazioni http, subiscono un processo di validazione?
- I dati sensibili vengono trasferiti attraverso i parametri?
- L'applicazione rileva la manomissione di parametri?

Esaminare gli aspetti di gestione dei parametri cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Non è possibile validare tutti i parametri di ingresso. Ciò rende l'applicazione suscettibile ad attacchi di tipo "interruzione di servizio" e ad attacchi indirizzati alla modifica del codice sottostante, quali SQL injection e XSS.
- I dati sensibili vengono inclusi nei cookie non crittografati. I dati del cookie possono essere modificati lato client o possono essere acquisiti e modificati in quanto vengono passati attraverso la rete.
- I dati sensibili vengono inclusi nelle stringhe di query e nei campi delle pagine di front-end. Le stringhe di query e i campi all'interno di pagine di front-end sono facilmente modificabili lato client.
- Si fa affidamento sulle informazioni presenti nell'intestazione HTTP. Queste informazioni sono facilmente modificabili lato client.

Gestione delle Eccezioni:

- In che modo l'applicazione gestisce le condizioni di errore?
- Viene mai permesso alle eccezioni di propagarsi fino al client?
- Che tipo di dati presentano i messaggi di eccezione?
- Vengono rivelate troppe informazioni al client?
- Dove vengono registrati i dettagli delle eccezioni? I file di log vengono protetti?

Occorre esaminare i meccanismi di gestione delle eccezioni cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Non è possibile convalidare tutti i parametri di ingresso;
- Vengono rivelate troppe informazioni al client.

Audit and Gestione dei Log:

- Sono state individuate delle attività chiave per l'audit?
- L'attività di audit copre tutti i livelli e i server dell'applicazione?
- Come vengono protetti i file di log?

Occorre esaminare i meccanismi di logging cercando una corrispondenza tra le seguenti vulnerabilità di carattere comune:

- Mancata revisione e registrazione (audit) dei tentativi d'accesso falliti;
- Mancata protezione dei file di audit;
- Mancata revisione e registrazione (audit) nei vari livelli del server dell'applicazione.

Indicazioni aggiuntive. Dopo aver completato l'attività di modellazione delle minacce, si procede come segue:

- Se si vuole descrivere il modello delle minacce in un documento, mantenere il documento di facile lettura in modo da potere essere consultato frequentemente. I contenuti chiave dovrebbero includere gli obiettivi di sicurezza, gli scenari chiave, le risorse protette, un elenco di minacce e un elenco di vulnerabilità.
- Utilizzare le vulnerabilità per contribuire a predisporre la progettazione e l'implementazione della sicurezza.
- Utilizzare le vulnerabilità per pianificare e implementare il test di sistema.
- Tracciare e aggiornare l'elenco di vulnerabilità in un sistema di tracciamento.
- Se sono state identificate delle minacce a cui sono state attribuite una priorità molto alta, ma per le quali non sono state individuate delle vulnerabilità corrispondenti, è necessario decidere se o non

indagare ulteriormente con il rischio di essere esposti a possibili attacchi, oppure di continuare l'analisi alla ricerca di una possibile vulnerabilità.
- Comunicare le informazioni acquisite ai membri del team di lavoro.

5.2 Identificazione del Processo di Sviluppo del Software Sicuro

L'applicazione di un processo di gestione del rischio nello sviluppo di un sistema abilita le organizzazioni a bilanciare i requisiti per la protezione delle informazioni e degli asset proprietari con il solo costo di implementare le strategie di controllo della sicurezza e di mitigazione attraverso l'SDLC. Il processo di gestione del rischio, identifica le attività e gli asset critici, nonché le vulnerabilità sistemiche a cui è esposta l'organizzazione. I rischi sono spesso condivisi in tutta l'organizzazione e non sono specifici per sole determinate architetture di sistema.

Alcuni dei vantaggi apportati con l'integrazione degli aspetti di sicurezza nel ciclo di vita di sviluppo del sistema, sono:

- L'individuazione preventiva e la mitigazione delle vulnerabilità e dei problemi di sicurezza presenti nella configurazione dei sistemi, con conseguente riduzione dei costi per l'implementazione dei controlli di sicurezza e delle tecniche di mitigazione delle vulnerabilità;
- La consapevolezza delle potenziali sfide ingegneristiche dovute ai controlli di sicurezza obbligatori;
- L'identificazione dei servizi di sicurezza condivisi e riutilizzo delle strategie e degli strumenti di sicurezza che riducono i costi di sviluppo e migliorano la condizione di sicurezza complessiva del sistema, attraverso l'applicazione di metodi e tecniche collaudate;
- La facilitazione nell'attuazione delle decisioni prese da parte dei dirigenti, attraverso l'applicazione tempestiva di un processo completo di gestione del rischio;
- La documentazione di importanti decisioni di sicurezza prese durante il processo di sviluppo, per informare la direzione sulle considerazioni di sicurezza intraprese durante tutte le fasi dello sviluppo;
- Il miglioramento dell'organizzazione e della fiducia dei suoi utenti nel promuovere l'adozione e l'uso dei propri sistemi;
- Una migliore interoperabilità e integrazione dei sistemi che sarebbe difficile raggiungere se la sicurezza fosse considerata separatamente ai vari livelli.

Uno studio della Forrester Consulting sullo stato della sicurezza applicativa ha riportato che le organizzazioni che implementano un processo MS-SDL hanno mostrato risultati di ROI migliori rispetto agli altri approcci metodologici. Anche, la Aberdeen Group ha dimostrato come l'adozione di un processo MS-SDL aumenti la sicurezza e riduca la gravità e il costo degli incidenti di vulnerabilità, generando al contempo un ritorno sugli investimenti (quattro volte maggiore) rispetto ad altri approcci di sicurezza adottati nello sviluppo di software.

MS-SDL è supportato da una rilevante quantità di risorse, tra cui documentazione, tutorial e strumenti software a supporto. Tale ricchezza di informazioni e strumenti rende sicuramente SDL un'opzione interessante per le organizzazioni che intendono adottare nuove iniziative di sicurezza del software.

La modellazione delle minacce è un approccio per analizzare la sicurezza di un'applicazione. Si tratta di un approccio strutturato che consente di identificare, quantificare e affrontare i rischi di sicurezza associati a un'applicazione. La modellazione delle minacce non è un approccio alla revisione del codice, ma integra il processo di revisione del codice da un punto di vista della sicurezza. L'inclusione della modellazione delle minacce nell'SDLC può contribuire a garantire che le applicazioni vengano sviluppate con la sicurezza integrata fin dall'inizio (Secure by Design/ Secure by default). Questo, in combinazione con la documentazione prodotta nell'ambito del processo di modellazione delle minacce, può fornire al revisore una maggiore comprensione del sistema. Consente inoltre al revisore di vedere dove si trovano i punti di accesso all'applicazione e quali sono le potenziali minacce associabili a ciascun punto di accesso. Il concetto di modellazione delle minacce non è nuovo, ma negli ultimi anni si è verificato un chiaro cambiamento di mentalità. La modellazione delle minacce, oggi guarda ad un sistema dal punto di vista di un potenziale

attaccante, piuttosto che dal punto di vista del difensore. Microsoft è stata forte sostenitrice del processo negli ultimi anni. Hanno fatto della modellazione delle minacce una componente fondamentale del loro SDLC, che sostengono essere una delle ragioni della maggiore sicurezza dei loro prodotti negli ultimi anni.

Quando l'analisi del codice sorgente, ad esempio di applicazioni esistenti, viene eseguita al di fuori dell'SDLC, i risultati della modellazione delle minacce, aiutano a ridurre la complessità dell'analisi del codice sorgente, promuovendo un approccio maggiormente approfondito. Invece di rivedere tutto il codice sorgente con uguale attenzione, è possibile assegnare una priorità alla revisione di sicurezza del codice, basandosi sul risultato ottenuto dal processo di modellazione che individua le minacce a più alto rischio.

Questi, i vantaggi introdotti dal processo di modellazione delle minacce:

- la conferma dell'idoneità degli elementi di sicurezza individuati da attuare;
- l'individuazione di eventuali lacune nelle caratteristiche di sicurezza da attuare;
- l'identificazione di eventuali altri elementi di sicurezza;
- l'identificazione dei requisiti di policy e di processo;
- l'identificazione dei requisiti da inserire nelle operazioni di sicurezza;
- l'identificazione dei requisiti in materia di tracciamento e monitoraggio;
- arrivare ai casi di abuso, se utilizzati, secondo la metodologia Agile;
- la comprensione dei requisiti di business continuity;
- la comprensione dei requisiti in materia di capacità e disponibilità.

L' esecuzione del processo di modellazione delle minacce, in fase di progettazione, aiuta nella:

- identificazione delle vulnerabilità che devono essere colmate a livello di progettazione e di inserimento nella fase di realizzazione;
- identificazione dei beni informativi che necessitano di controlli di sicurezza;
- mappatura dei controlli di sicurezza, identificati in controlli tecnico/amministrativi/fisici a seconda dei casi (questa attività può essere svolta anche a livello di architettura, ma farlo a livello di progettazione aiuta ad essere più precisi);
- identificazione dei casi di "test di sicurezza"/"scenari di test di sicurezza" nella verifica dei requisiti di sicurezza.

La modellazione delle minacce è il processo di valutazione e documentazione dei rischi associati alla sicurezza di un particolare sistema e/o applicazione software. Mediante l'adozione di opportune tecniche già discusse in precedenza nel documento, è possibile identificare strategie di mitigazione efficaci per contrastare potenziali minacce a cui potrebbe essere soggetta l'applicazione. La modellazione delle minacce consente inoltre, di giustificare l'introduzione o l'eliminazione di eventuali feature all'interno dell'applicazione oltre che governare, al fine di proteggere gli asset dell'applicazione, l'introduzione di nuove policy o pratiche di sicurezza all'interno del sistema. La categorizzazione delle minacce di sicurezza può essere ottenuta mediante l'adozione di un modello denominato STRIDE che è l'acronimo che riunisce la gamma dei rischi a cui può essere soggetta l'applicazione e per i quali deve essere protetta.

Nell'ambito della fase progettuale dell'SDL, nel processo di definizione dei requisiti di sicurezza, un modello come STRIDE può essere di aiuto nel definire i pattern di attacco, tra cui estrarre il modello di attacco (ovvero il sottoinsieme dei possibili attacchi) per il nostro sistema applicativo. Lo STRIDE ha l'indubbio vantaggio di non essere eccessivamente astratto ma è piuttosto facilmente riconducibile a situazioni reali. In gran parte della letteratura il modello STRIDE viene definito come un sistema per modellare le minacce. Ma in realtà (in accordo con Gary McGraw) si ritiene più opportuno pensare che STRIDE sia un modello relativo ai possibili attacchi, legando la "minaccia" agli attori (umani e non) che sono invece gli artefici degli attacchi stessi. Le sei categorie di rischi a cui la STRIDE afferisce (Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, and Elevation of privilege) consentono di identificare le vulnerabilità ed i possibili vettori di attacco nelle applicazioni software.

Le linee guida per la modellazione delle minacce ispirata a MS-SDL si suddivide nelle seguenti fasi:

- Identificazione degli asset. Cosa il sistema dovrebbe proteggere?
- Creazione di una panoramica dell'architettura. Concentrarsi sui confini di fiducia, ovvero sui flussi di dati scambiati tra componenti posseduti da un'entità e componenti posseduti da un'altra entità.
- Scomposizione del sistema in sotto-componenti fino al livello più basso possibile.
- Identificazione delle minacce. Utilizzare la STRIDE o un albero delle minacce per facilitare l'enumerazione delle stesse.
    - STRIDE è un acronimo di spoofing, tampering, repudiation, information disclosure, denial of service e elevation of privilege. Queste descrizioni di vulnerabilità non sono intese come categorie che si escludono a vicenda, ma piuttosto come una tecnica euristica per l'enumerazione. Esaminare ciascun componente, concentrandosi sui confini di fiducia, e valutare se questo presenta vulnerabilità precedentemente indicate.
    - Gli Attack tree, possono eventualmente sostituire la STRIDE.
- Documentazione delle minacce.
- Valutazione delle minacce secondo il modello DREAD, un acronimo che indica il danno potenziale, la riproducibilità, l'utilizzabilità, gli utenti interessati, l'esposizione. Le minacce che si collocano ai primi posti in ciascuna di queste categorie dovrebbero avere una priorità più elevata.

5.3 Modellazione e Individuazione delle minacce con STRIDE

Un evidente vantaggio dell'approccio STRIDE è l'indipendenza dal codice. Ciò è vantaggioso in quanto aiuta a identificare i problemi di sicurezza nella fase di analisi e progettazione del sistema software. Inoltre consente a tutto il team (oltre agli sviluppatori) di partecipare alla definizione del modello delle minacce del sistema stesso.

Il processo può portare benefici laddove non si dispone di un esperto di sicurezza all'interno del proprio team. L'impiego della STRIDE consente a questi team di individuare le vulnerabilità e le tecniche di mitigazione da attuare per difendersi dalle eventuali minacce identificate.

Le organizzazioni, possono inoltre beneficiare del processo sia per i nuovi progetti che per i progetti in essere, in cui potrebbero esistere delle vulnerabilità non identificate all'interno del codice. L'implementazione di una metodologia che identifica e classifica le minacce è un processo ripetibile strutturato che può portare benefici a qualsiasi tipo di progetto.

Il risultato finale è, un elenco di vulnerabilità che i team di sviluppo e gli stakeholder dei prodotti possono quindi valutare, al fine di effettuare una accurata valutazione dei rischi, che insistono sulle vulnerabilità individuate (Risk Assessment).

La STRIDE è stata identificata come metodologia leader di Threat Modeling nell'industria del software.

Il successo di questa metodologia è dovuto ad un approccio ben strutturato alla modellazione delle minacce, ed è un eccellente supporto ed una risorsa per gli utenti.

A volte la STRIDE viene indicata come "categorie STRIDE" o " tassonomia STRIDE". Tuttavia si evidenzia che la STRIDE non nasce come strumento di categorizzazione, ma con l'obiettivo di aiutare a trovare i possibili attacchi.

5.4 Valutazione del rischio derivante dalle minacce individuate con DREAD

La metodologia DREAD è stata sviluppata da Microsoft nell'ambito della definizione del Security Development Lifecycle e del Threat Modeling. L'adozione della metodologia DREAD prodiga i seguenti benefici:

1. è utile per focalizzarsi sui reali rischi di una minaccia specifica;
2. obbliga a considerare fattori aziendali come la criticità del sistema e l'impatto sul business;

3. le cinque categorie sono tra loro scarsamente correlate (una di esse non implica le altre): considerare fattori indipendenti è un'ottima garanzia per formulare una corretta valutazione del rischio.
4. è largamente impiegata (ad es. OWASP[20] e OpenStack[21]) .

5.5 Modellazione e Individuazione delle minacce di privacy con LINDUN

La privacy è un aspetto molto importante, soprattutto nella società di oggi, dove i dati personali sono onnipresenti. Purtroppo questa, viene spesso trascurata durante lo sviluppo del software. Nonostante emergono diverse metodologie orientate alla produzione di requisiti di tutela della privacy (vedi paragrafo 5.5.3), queste non soddisfano appieno quelle che sono le aspettative, o comunque, non sono in grado di fornire una guida metodologica sostanziale da adottare nel corso dell'analisi o mancano del necessario supporto alla tutela della privacy. Per colmare questa lacuna, si propone la metodologia LINDDUN, descritta nei capitoli precedenti. LINDDUN si ispira infatti a STRIDE, ovvero, un approccio consolidato per la modellazione e l'identificazione delle minacce alla sicurezza. Inoltre, la metodologia LINDDUN è stata costruita sulla base delle classificazioni esistenti in materia di tutela della privacy. Anche se LINDDUN non è una tecnica di conformità, attua diversi principi imposti dalla legislazione sulla protezione dei dati (ad esempio consenso, minimalizzazione, sensibilizzazione, ecc.) e richiama esplicitamente l'attenzione sulla necessità di conformità normativa. Inoltre, le proprietà sulla privacy, riportate nel paragrafo 5.5.1.1, costituiscono la base delle categorie di minacce gestite da LINDDUN. Infine, LINDDUN aderisce anche ai principi della Privacy by Design in quanto mira a introdurre la privacy nelle prime fasi del ciclo di vita di sviluppo del software.

[20] https://www.owasp.org/index.php/Threat_Risk_Modeling

[21] https://wiki.openstack.org/wiki/Security/OSSA-Metrics#DREAD

6 UN ESEMPIO APPLICATIVO: CASO D'USO "EASY WEB SITE"

Lo use case si riferisce a un classico sito web disponibile su Internet che implementa un servizio per i propri clienti, i quali vi accedono attraverso un browser.

A titolo di esempio, si suppone che:

- il sito web acceda a una base dati di tipo SQL sia in lettura sia in scrittura;
- il sito web esponga funzionalità sia per i clienti del servizio sia per gli amministratori del servizio;
- l'utenza non autenticata (ad esempio, gli anonymous users) non possa accedere al sistema.

Sulla base delle assunzioni fatte, andiamo a rappresentare il sistema in oggetto attraverso un diagramma, facendo uso del simbolismo DFD, tipicamente utilizzato nella modellazione delle minacce (vedi paragrafo 5.2.3.1). Il diagramma che segue, mostra una scomposizione del sistema in oggetto ponendo in evidenza quelle che sono le sue componenti principali (Browser Client, Web Server e SQL Database), i confini di fiducia (Generic Trust Border Boundary e Internet Boundary) nonché i flussi dati di interscambio tra le singole componenti del sistema ([BC2WS] HTTPS Req (Credentials&Data) - Browser Client to Web Server, [WS2BC] HTTPS Req (Data) - Web Server to Browser Client, [WS2SQLDB] (Credentials&Data) Web Server to SQL Database e [WS2SQLDB] (Data) SQL Database to Web Server) dette anche interazioni.

6.1 Diagramma: Use Case

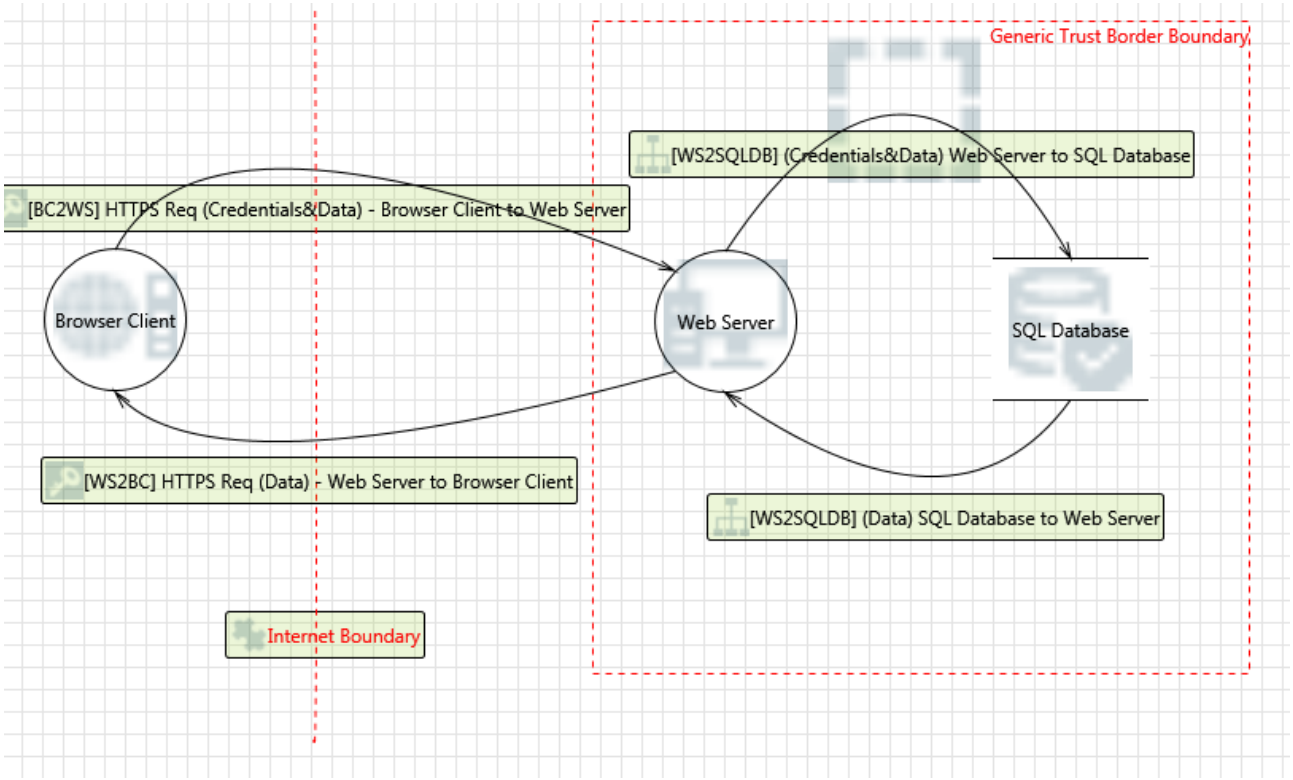


Figura 9 - Diagramma dello use case

A seguire, per ciascuna interazione/flusso dati, vengono individuate le possibili minacce sulla base dell'analisi STRIDE. Per ciascuna minaccia viene fornita la categoria STRIDE/Compliance di pertinenza a cui la minaccia appartiene, una breve descrizione e alcune contromisure da attuare nel processo di mitigazione. Viene inoltre indicato, attraverso l'analisi DREAD, un indice di priorità (ALTO, MEDIO e BASSO) da considerare nella risoluzione della minaccia stessa (DREAD Score).

6.2 Interazione: da Browser Client a Web Server

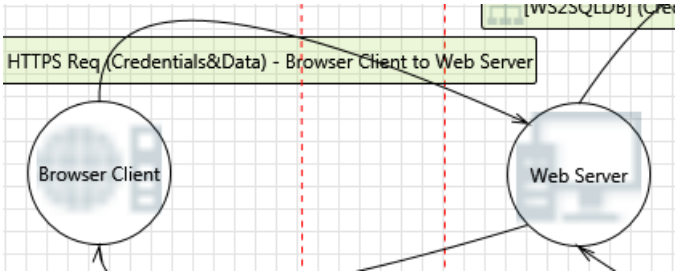


Figura 10 - Interazione tra Browser Client e Web Server

6.2.1 Assunzioni

Si assume che il Browser Client si autentica nei confronti del Web Server utilizzando una username e una password, ed esegue una post http per leggere e modificare i dati.

Si suppone inoltre, come già detto, che, l'utenza non autenticata (ovvero gli anonymous users) non possa accedere al sistema.

Il protocollo utilizzato è HTTPS, il quale garantisce:

- Autenticazione della Destinazione (Web Server);
- Confidenzialità;
- Integrità.

A seguire vengono riportate le minacce individuabili nell'interazione in oggetto.

6.2.2 Accesso a internet non valido

Categoria: Compliance

Descrizione: L'applicazione Web (qui "Server Web") non dovrebbe essere collegata direttamente a Internet.

Contromisure: Interconnettere il "Browser Client" con il "Server Web" tramite un gateway di protezione (firewall).

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	Se il Web Server è esposto direttamente sulla rete Internet, un attaccante può facilmente compromettere il sistema.	3
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	L'attacco non è banale: occorre una figura senior.	2
Affected Users	100%.	3
Discoverability	Occorre identificare un exploit (es. per mancanza di patching adeguato del Sistema Operativo) cui la macchina su cui il Web Server è installato risulta vulnerabile.	1

DREAD Score: 12/15 (ALTO)

6.2.3 Mancanza di convalida dell'input da parte del "Web Server"

Categoria: Tampering

Descrizione: Il 'Web Server' non verifica che i dati di input siano nel formato previsto. Questa è la causa principale di un numero molto elevato di problemi sfruttabili. Considerate tutti i percorsi e il modo in cui si gestiscono i dati di input. Verificare che tutti gli input siano verificati e, se in formato non previsto, scartati o sanitizzati.

Contromisure:

- La validazione dell'input deve essere eseguita prima che l'input entri nel 'Web Server' o venga elaborato da quest'ultimo.
- La convalida dei dati ricevuti deve comprendere almeno:
  - La verifica del tipo di dato;
  - Il controllo dell'intervallo di ammissibilità dei valori dei dati per garantire che i valori forniti siano entro i limiti prestabiliti (minimo e massimo / dimensione);
  - Il controllo sulla base delle regole di business previste.
- Definire il set consentito di caratteri da accettare. La convalida basata su White list è da preferire a quella basata su Black list. La White list prevede la definizione esatta di ciò che è consentito, e per definizione, tutto il resto non è ammesso. L'utilizzo di espressioni regolari può facilitare l'implementazione di schemi di validazione basati su White list. La Black List cerca di rilevare caratteri e modelli di attacco ed è un approccio sconsigliato, in quanto è possibile per un aggressore eludere tali filtri.
- Mettere in campo un meccanismo di controllo degli accessi efficace capace di garantire l'accettazione dei dati solo da parte di utenti autorizzati.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	La mancanza di validazione dell'input da parte del Web Server lo espone a un'ampia varietà di attacchi che possono anche arrivare a compromettere la macchina su cui il Web Server è installato come nel caso di "Command Injection". **NOTA BENE** Si tratta di una minaccia molto generica che vuole focalizzare l'attenzione sul principio "all input is evil". Nel prosieguo vengono esaminate minacce più specifiche legate alla mancanza di validazione dell'input (cross-site scripting, sql injection, ecc.).	3
Reproducibility	L'attacco può essere condotto in qualunque momento	3
Exploitability	L'attacco non è banale: occorre una figura senior.	2
Affected Users	100%	3
Discoverability	Occorre identificare un exploit, attraverso la manomissione dei dati di input, cui il Web Server risulta vulnerabile.	1

DREAD Score: 12/15 (ALTO)

6.2.4 Cross Site Scripting

Catogoria: Tampering

Descrizione: Il 'Web Server' potrebbe essere soggetto ad un attacco di Cross-Site Scripting in quanto non prevede la sanitizzazione dell'input non affidabile (untrusted) che potrebbe contenere script malevoli

Contromisure:

- Eseguire l'escape del testo HTML prima di inserire i dati non attendibili nel contenuto degli elementi HTML.
- Eseguire l'escape del valore di un attributo prima di inserire dati non attendibili in attributi HTML.
- Eseguire l'escape del testo JavaScript prima di inserire dati non attendibili nel codice JavaScript.
- Eseguire l'escape HTML di valori JSON prima di inserire i dati nel contenuto degli elementi HTML e leggere i dati con "JSON.parse".
- Eseguire l'escape CSS e attuare rigorose validazioni prima di inserire i dati non attendibili nei valori di proprietà di stile HTML.
- Eseguire l'escape dell'URL prima di inserire dati non attendibili nei valori dei parametri dell'URL.
- Sanitizzare i Markup HTML con una libreria progettata a tale scopo.
- Utilizzare il flag HTTPOnly per i cookie.
- Implementare la politica Content Security Policy.
- Utilizzare un sistema Auto-Escaping Template System.
- Utilizzare l'X-XSS-Protection Response Header.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	Lo script malevolo può accedere a qualsiasi cookie, token di sessione o altre informazioni sensibili conservate dal browser (qui Browser Client) e utilizzati esclusivamente nel dialogo con il sito d'origine (qui Web Server). Questi script possono anche riscrivere il contenuto della pagina HTML. In definitiva il Tampering dell'url produce Information Disclosure, tra cui la compromissione del token di sessione che abilita il "Session hijacking" (che è una forma di furto di identità – spoofed identity). Nel peggiore dei casi, l'attaccante potrebbe impersonare l'amministratore del Web Server.	2
Reproducibility	L'attacco funziona sempre. Tuttavia il token di sessione (che è il dato la cui compromissione è particolarmente grave: spoofed identity) è utilizzabile finché la sessione non scade.	2
Exploitability	L'attacco non è banale: occorre una figura senior.	2
Affected Users	100% (nel caso in cui l'attaccante arrivasse a impersonare l'amministratore.)	3
Discoverability	Occorre identificare un url in cui un input utente ritorna in output senza aver subito sanitizzazioni o che può modificare il "DOM" enviroment.	1

DREAD Score: 10/15 (MEDIO)

6.2.5 Ripudio di dati da parte del 'Browser Client'

Categoria: Repudiation

Descrizione: Il 'Client Browser' sostiene di non aver inviato i dati al 'Web Server'.

Contromisure:

- È consigliabile che l'applicazione ricevente (qui 'Web Server') utilizzi file di log o di audit per registrare l'origine, l'ora e il riepilogo dei dati ricevuti, affinché il mittente di informazioni non possa negare l'invio delle stesse.
- Si raccomanda inoltre che il destinatario autentichi il mittente per assicurare che la comunicazione avvenga con il mittente corretto.
- Implementare le protezioni contro la manomissione dei dati di log/audit poiché, dati di log/audit manomessi possono produrre potenziali repudiation.
- Considerare di far sì che l'applicazione ricevente richieda al mittente di firmare i dati trasmessi per garantire che il mittente di informazioni non possa negare l'invio delle stesse.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	A fronte della elaborazione dati o dell'esecuzione di transazioni disconosciute dalla sorgente, non si ha modo di attribuire il malfunzionamento alla parte che ne è responsabile.	1
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2
Affected Users	100% (qualunque utente potrebbe tentare la repudiation).	3
Discoverability	Il rilevamento della minaccia è contestualizzato nell'ambito di un'utenza autenticata.	2

DREAD Score: 11/15 (MEDIO)

6.2.6 Crash o fermo del processo 'Web Server'

Categoria: Denial Of Service

Descrizione: Il 'Web Server' va in crash, si ferma o risponde lentamente, in ogni caso violando una metrica di disponibilità.

Contromisure:

- Convalidare tutti i dati di input per assicurare che i valori non possano causare il crash del 'Web Server'.
- Gestire tutti i casi di errore (sia le exceptions del linguaggio di programmazione sia i casi di errore nelle condizioni logiche) in modo graceful, ossia in modo che non provochi crash o servizi degradati.
- I log devono indicare se è in atto o meno una corretta validazione degli input (per evitare crash) e come vengono trattati i casi eccezionali.
- Prevedere il ripristino del sistema: Si consideri l'utilizzo di un meccanismo di recupero (ad esempio il watchdog) per riavviare il 'Web Server' in caso di crash.

- Utilizzare le tecniche di throttling e rate-limiting per evitare che il 'Web Server'collassi.

Valutazione della priorità della minaccia (Ranking)

- Si considera lo scenario del DDOS, oggi disponibile "As-a-Service" sul Dark Web.

DREAD	Descrizione	Score
Damage Potential	L'attaccante può impedire agli utenti del sistema di interagire con esso (del tutto o in modo degradato).	2
Reproducibility	L'attacco funziona solo in certe finestre temporali: un attacco DDOS ha per sua natura una durata limitata nel tempo.	1
Exploitability	L'attacco richiede un figura senior capace di organizzare un DDOS.	1
Affected Users	100% (la piattaforma è resa indisponibile o comunque ne viene degradato il funzionamento).	3
Discoverability	L'attaccante dovrà impegnare parecchie risorse per sfruttare la vulnerabilità (deve probabilmente sapersi muoversi sul Dark Web e pagare il "servizio").	1

DREAD Score (DDOS): 8/15 (MEDIO)

Valutazione della priorità della minaccia (Ranking)

- Si considera lo scenario del crash, nell'ipotesi richieda lo sfruttamento di una vulnerabilità 0-day del 'Web Server'.

DREAD	Descrizione	Score
Damage Potential	L'attaccante può impedire agli utenti del sistema di interagire con esso.	2
Reproducibility	L'attacco funziona fino all'applicazione della patch (che richiede la scoperta dello 0-day, la produzione della patch, il testing della patch, l'installazione della patch).	2
Exploitability	L'attacco richiede una figura senior capace di trovare una vulnerabilità tale da provocare il crash di Web Server.	1
Affected Users	100% (la piattaforma è resa indisponibile o comunque ne viene degradato il funzionamento).	3
Discoverability	L'attaccante dovrà impegnare parecchie risorse per scoprire la vulnerabilità sfruttabile. Lo sforzo potrebbe non valere il risultato.	1

DREAD Score (Crash): 9/15 (MEDIO)

DREAD Score complessivo (il peggiore tra i due): 9/15 (MEDIO)

6.2.7 Interruzione del flusso dati HTTPS (o inaccessibilità da parte del 'Web Server')

Categoria: Denial Of Service

Descrizione: Un agente esterno interrompe i dati che fluiscono attraverso '[BC2WS] HTTPS Req (Credentials&Data) - Browser Client to Web Server' in entrambe le direzioni.

Contromisure:

- Se il "Client Browser" non è in grado di proseguire l'elaborazione, dovrebbe fornire risposte appropriate agli utenti in attesa.
- Rilevamento: Si consideri l'utilizzo di un meccanismo di allarme che, rilevando lo spostamento del comportamento del Server Web da metriche predefinite, consenta di segnalare la sospetta condizione di DOS (per attivare le risposte appropriate).
- 3) Ridondanza: considerare la possibilità di configurare un secondo "Web Server" per essere utilizzato come un backup di quello sotto attacco.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	L'attaccante può impedire agli utenti del sistema di interagire con esso.	2
Reproducibility	Non sembra verosimile che l'attaccante "a comando" / "a piacimento" possa interrompere il flusso dati in qualunque momento. Sembra ragionevole assumere che l'attacco funzioni solo in certe condizioni che si raggiungono raramente.	1
Exploitability	L'attacco, sia esso condotto sul piano dell'interruzione fisica della connessione o sul piano dell'interruzione logica del flusso dei dati, è complesso.	1
Affected Users	100% (la piattaforma è resa indisponibile).	3
Discoverability	L'attaccante dovrà impegnare parecchie risorse per organizzare l'attacco.	1

DREAD Score: 8/15 (MEDIO)

6.2.8 Elevazione di privilegi attraverso l'esecuzione remota di codice da parte del 'Web Server'

Categoria: Elevation Of Privilege

Descrizione: 'Client Browser' potrebbe essere in grado di eseguire codice in remoto sul sistema ' Web Server'.

Contromisure:

- Il processo non deve contenere percorsi che mandano in esecuzione dati presi dal flusso di input (es. il nome di un eseguibile).
- Se un processo manda in esecuzione dati presi dal flusso di input, questi devono essere convalidati in modo da escludere che venga eseguito codice arbitrario.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	L'attaccante potrebbe prendere il controllo dell'intero sistema, attraverso tecniche di "lateral moving" (il "lateral moving" di solito comporta attività legate alla ricognizione <<information gathering>>, furto di credenziali e spostamenti su altri computer).	3
Reproducibility	L'attacco può essere condotto in qualunque momento.	3

Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata. Si richiedono anche skill elevati.	1
Affected Users	100% (se l'esito finale fosse effettivamente il controllo del sistema).	3
Discoverability	L'attaccante dovrà impegnare parecchie risorse per scoprire la vulnerabilità sfruttabile (l'attacco di norma sfrutta una catena di debolezze).	1

DREAD Score: 11/15 (MEDIO)

6.2.9 Elevazione dei privilegi attraverso il cambiamento del flusso di esecuzione nel codice del 'Web Server'

Categoria: Elevation Of Privilege

Descrizione: Un attaccante può passare dati al 'Web Server' in modo da cambiare a suo vantaggio il flusso di esecuzione del programma all'interno del 'Web Server' stesso.

Contromisure:

- Convalidare in modo appropriato gli input e gestire le eccezioni per evitare percorsi di esecuzione imprevisti.
- Impiegare meccanismi di protezione contro il buffer overflow e altri problemi di gestione della memoria.
- Applicare principio del minimo privilegio.
- Implementare le protezioni contro la manomissione dei percorsi di autenticazione e di autorizzazione (ad esempio, se l'autenticazione e l'autorizzazione dipendono dai dati del database, i percorsi di codice che interagiscono con il database devono essere protetti per garantire l'integrità di tali dati).
- Utilizzare implementazioni dell'Address Space Layout Randomization (ASLR) per rendere più difficile l'esecuzione di istruzioni privilegiate agli indirizzi noti in memoria tramite buffer overruns.
- Utilizzare compilatori che bloccano il buffer overruns.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	L'attaccante potrebbe prendere il controllo del Web Server, se riuscisse a elevare i propri privilegi fino a livello appunto di amministratore del Web Server.	2
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2
Affected Users	100% (se l'esito finale fosse effettivamente il controllo del sistema).	3
Discoverability	L'attaccante dovrà impegnare parecchie risorse per scoprire la vulnerabilità sfruttabile.	1

DREAD Score: 11/15 (MEDIO)

6.2.10 Cross Site Request Forgery

Categoria: Elevation Of Privilege

Descrizione: Il Cross Site Request Forgery (CSRF o XSRF) è un tipo di attacco in cui un attaccante fa in modo che un utente vittima (qui un Autheticated User del Web Server) invii involontariamente una richiesta HTTPS dal suo browser (qui Client Browser) al sistema web (qui Web Server) dove è attualmente autenticato.

L'attaccante deve: a) trovare un difetto (flaw) lato server (qui Web Server) tale per cui il sito web processa una richiesta di cambio stato a fronte della sola presenza di una sessione valida (che attesta una precedente autenticazione); b) indurre un utente ignaro (qui un Autheticated User del Web Server) ad esercitare un url che sfrutta il difetto di cui sopra mentre quell'utente ha una sessione aperta sul server (qui Web Server).

Il sistema, vulnerabile al CSRF, riceve dal browser dell'utente la richiesta contraffatta (dietro cui, cioè, si cela un'azione studiata dall'attaccante) con un cookie di sessione valido (dal momento che la vittima è stata precedentemente autenticata e la sessione è ancora attiva) e la elabora.

Contromisure: Fare in modo che tutte le richieste di cambiamento di stato oltre ad essere autenticate includano un ulteriore elemento di payload segreto (canary o CSRF token) conosciuto solo dal sito legittimo e dal browser (parti che comunicano tra loro in modo protetto tramite HTTPS).

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	L'attaccante può eseguire richieste di cambio stato al sistema che sono prerogativa di Autheticated User o, peggio, dell'Administrator (es. la modifica di configurazioni o il furto/modifica di dati privati).	2
Reproducibility	L'attacco funziona finché la sessione della vittima non scade.	1
Exploitability	L'attacco non è banale: occorre una figura senior.	2
Affected Users	In genere l'attacco è condotto con tecniche di social engineering: gli utenti coinvolti sono una quota parte del totale.	2
Discoverability	Occorre identificare un url che presenti la vulnerabilità XSRF.	1

DREAD Score: 8/15 (MEDIO)

6.3 Interazione: da Web Server a Browser Client

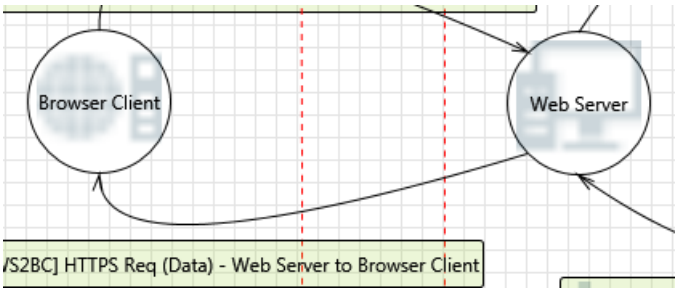


Figura 11 - Interazione tra Web Server e Browser Client

6.3.1 Assunzioni

Si assume che il Browser Client si autentica nei confronti del Web Server utilizzando una username e una password, ed esegue una post http per leggere e modificare i dati.

Si suppone inoltre, come già detto, che, l'utenza non autenticata (ovvero gli anonymous users) non possa accedere al sistema.

Il protocollo utilizzato è HTTPS, il quale garantisce:

- Autenticazione della Destinazione (Web Server);
- Confidenzialità;
- Integrità.

6.3.2 Analisi delle minacce e mitigazioni

Valgono le raccomandazioni già proposte in "Interazione: da Browser Client a Web Server".

6.4 Interazione: da Web Server a SQL Database

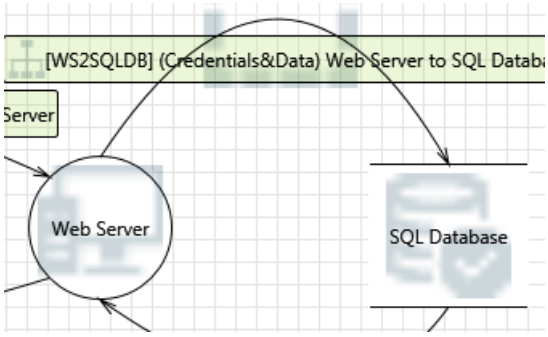


Figura 12 - Interazione tra Web Server e SQL Database

6.4.1 Assunzioni

Si assume che il Web Server si autentica nei confronti del SQL Server utilizzando una username e una password, e può inserire, leggere, modificare, cancellare dati.

Si suppone che l'interazione avvenga all'interno di un Trusted Boundary.

A seguire vengono riportate le minacce individuabili nell'interazione in oggetto.

6.4.2 Vulnerabilità di SQL Injection nel 'SQL Database'

Categoria: Tampering

Descrizione: La SQL Injection è una tecnica di attacco di tipo "code injection", usata per attaccare un

database, nella quale viene inserito del codice SQL malevolo all'interno di parametri di input in modo che vada in esecuzione sul database sotto attacco (es. per inviare all'attaccante /modificare/distruggere il contenuto del database e, in certi casi, "saltare dal DB al Sistema Operativo" e prendere il controllo della macchina). A patto che la query risultante da questo tipo di attacco sia sintatticamente corretta, il database la eseguirà.

Contromisure:

- Usare i Prepared Statements con query parametrizzate.
- Usare le Stored Procedures.
- Eseguire la validazione di tipo White List di ogni input esterno usato per costruire statements SQL.
- 4) Eseguire l'escape di ogni input utente.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	La possibilità di eseguire codice malevolo sul database dell'applicazione la espone potenzialmente alla totale compromissione.	3
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2
Affected Users	100%.	3
Discoverability	Occorre identificare un exploit, attraverso la manomissione dei dati di input, cui il Web Server risulta vulnerabile.	1

DREAD Score: 12/15 (MEDIO)

6.4.3 Possibile corruzione del 'SQL Database'

Categoria: Tampering

Descrizione: Assicurare l'integrità dei dati all'interno dell'archivio 'SQL Database'.

Contromisure:

- Autenticare tutti gli utenti.
- Mettere in pratica un efficace meccanismo di controllo degli accessi che garantisca che i dati possono essere scritti o modificati solo da utenti autorizzati.
- Rispettare il principio del privilegio minimo.
- Attuare controlli che seguano e registrino correttamente le azioni degli utenti.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	La possibilità, senza averne diritto, di modificare i dati all'interno del database dell'applicazione la espone potenzialmente alla totale compromissione.	3
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2

Affected Users	100%.	3
Discoverability	Occorre identificare un exploit, attraverso la manomissione dei dati di input, cui il Web Server risulta vulnerabile.	1

DREAD Score: 12/15 (MEDIO)

6.4.4 Consumo eccessivo di risorse da parte del 'Web Server' o del 'SQL Database'

Categoria: Denial Of Service

Descrizione: Il "Web Server" o il "SQL Database" adottano passi espliciti per controllare il consumo di risorse? Fare attenzione che le richieste di risorse non producano deadlock e che, nel caso peggiore, vadano in timeout.

Contromisure:

- Non bloccare (deadlock) le richieste di risorse.
- Impostare i timeout per le richieste di risorse, se è applicabile.
- Validare i dati di input che si riferiscono al consumo di risorse.
- Limitare la dimensione dei dati elaborati dall'applicazione.
- Eseguire il rilascio delle risorse quando non sono più necessarie.
- Gli audit devono dare indicazioni sul consumo eccessivo di risorse da parte dell'applicazione.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	L'attaccante può degradare le prestazioni del sistema fino a renderlo potenzialmente indisponibile.	2
Reproducibility	L'attacco funziona sempre.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2
Affected Users	100% (la piattaforma è resa indisponibile o comunque ne viene degradato il funzionamento).	3
Discoverability	Il rilevamento della minaccia è contestualizzato nell'ambito di un'utenza autenticata.	2

DREAD Score (Crash): 12/15 (ALTO)

6.5 Interazione: da SQL Database a Web Server

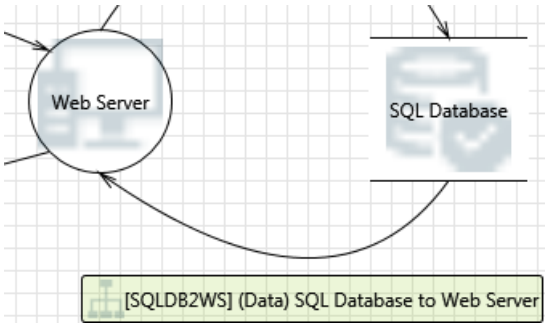


Figura 13 - Interazione tra SQL Database e Web Server

6.5.1 Assunzioni

Il Web Server si autentica nei confronti del SQL Database utilizzando una username e una password ed inserisce, legge, modifica e cancella dati.

Si suppone che l'interazione avvenga all'interno di un Trusted Boundary.

A seguire vengono riportate le minacce individuabili nell'interazione in oggetto.

6.5.2 Persistent Cross Site Scripting

Categoria: Tampering

Descrizione: Il 'Server Web' potrebbe essere soggetto ad un attacco di cross-site scripting di tipo persistente in quanto non sanitizza i dati di input al 'SQL Database' in fase di scrittura (che potrebbero contenere uno script malevolo) e non esegue l'escape dei dati di output dal 'SQL Database' in fase di lettura (ciò che si traduce nel mandare in esecuzione su 'Browser Client' lo script malevolo).

Contromisure: Applicare le tecniche di sanitizzazione e di escaping come nel caso di Cross Site Scripting.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	Lo script dannoso può accedere a qualsiasi cookie, token di sessione o altre informazioni sensibili conservate dal browser (qui Browser Client) e utilizzati esclusivamente nel dialogo con il sito d'origine (qui Web Server). Questi script possono anche riscrivere il contenuto della pagina HTML. In definitiva il Tampering dell'url produce Information Disclosure, tra cui la compromissione del token di sessione che abilita il "Session hijacking" (che è una forma di furto di identità – spoofed identity). Nel caso pessimo, l'attaccante potrebbe impersonare l'amministratore del Web Server.	2
Reproducibility	L'attacco funziona sempre. Tuttavia il token di sessione (che è il dato la cui compromissione è particolarmente grave: spoofed identity) è utilizzabile finché la sessione non scade.	2
Exploitability	L'attacco non è banale: occorre una figura senior.	2
Affected Users	100% (nel caso in cui l'attaccante arrivasse a impersonare l'amministratore).	3

Discoverability	Occorre identificare un url che ritorni in output, senza aver subito alcun encoding, un input utente malevolo, precedentemente persistito senza sanitizzazioni sul database.	1

DREAD Score: 10/15 (MEDIO)

6.5.3 Controllo accesso debole per una risorsa

Categoria: Information Disclosure

Descrizione: Una inadeguata protezione dei dati a livello di " SQL Database" può consentire a un attaccante di leggere informazioni non destinate alla divulgazione. Esaminare le impostazioni di autorizzazione.

Contromisure:

- Autenticare tutti gli utenti.
- Mettere in pratica un efficace meccanismo di controllo degli accessi che garantisca che i dati possono essere letti solo da utenti autorizzati.
- Rispettare il principio del minimo privilegio.
- Attuare controlli che seguano correttamente e registrino le azioni degli utenti
- Crittografare i dati.
- Assicurarsi che le utilità o le tecniche di riservatezza dei data store vengano appropriatamente utilizzate in modo che la riservatezza dei dati sia mantenuta e gestita in base alle esigenze aziendali/regole.

Valutazione della priorità della minaccia (Ranking)

DREAD	Descrizione	Score
Damage Potential	La possibilità, senza averne diritto, di leggere i dati all'interno del database dell'applicazione espone potenzialmente l'owner del sistema a violazioni di normative di legge (es. Privacy) o a danno reputazionale o a divulgazione di informazioni di business riservate.	2
Reproducibility	L'attacco può essere condotto in qualunque momento.	3
Exploitability	Per la natura del servizio, l'attacco richiede un'utenza autenticata.	2
Affected Users	100%.	3
Discoverability	Occorre identificare un exploit, attraverso la manomissione dei dati di input, cui il Web Server risulta vulnerabile.	1

DREAD Score: 11/15 (MEDIO)

7 BIBLIOGRAFIA

[1] H. E. a. P. D. Shafiq Hussain, Threat modeling using Formal Methods: A New Approach to Develop Secure Web Applications.

[2] G. Zannone, Towards the development of privacy-aware systems, 2009.

[3] A. Cavoukian, Privacy by Design.

[4] S. F. M. H. a. R. M. Kristian Beckers, «A Problem-based Approach for Computer Aided Privacy Threat Identification. In Privacy Technologies and Policy, volume 8319 of LNCS,» Springer, 2014, p. pages 1–16...

[5] E. K. a. S. G. Christos Kalloniatis, Addressing privacy requirements in system design: the PriS method. Requirements Engineering.

[6] S. S. a. L. F. Cranor, «Engineering privacy. IEEE Transactions on Software Engineering,» 2009, p. 35(1):67–82 .

[7] F. S. Gürses, Multilateral privacy requirements analysis in online social network services., 2010.

[8] S. I. C. K. a. S. G. Haralambos Mouratidis, « A framework to support selection of cloud providers based on security and privacy requirements. Journal of Systems and Software,» 2013, p. pages 2276–2293.

[9] L. C. M. S. L. P. a. B. N. Inah Omoronyia, « Engineering adaptive privacy: on the role of privacy awareness requirements. In Proceedings of the 2013 International Conference on Software Engineering,» 2013, pp. 632-641.

[10] C. Jensen, «Designing for privacy in interactive systems. PhD thesis,» Georgia Institute of Technology, 2005.

[11] « Privacy guidelines for developing software products and services, version 3.1. Technical report, Microsoft Coorporation, .,» Sept 2008.

[12] N. M. a. J. Z. Seiya Miyazaki, «Computer-aided privacy requirements elicitation technique,» Asia-Pacific Conference on Services Computing (APSCC'08), 2008, p. pages 367–372.

[13] A.I. Antón, J.B. Earp, and A. Reese. Analyzing website privacy requirements using a privacy goal taxonomy. In Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on, pages 23–31, 2002.

[14] Fahriye Seda Gürses. Multilateral privacy requirements analysis in online social network services. PhD thesis, Department of Computer Science, KU Leuven, 2010.

[15] Mina Deng, Kim Wuyts, Riccardo Scandariato, Bart Preneel, and Wouter Joosen. A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. Requirements Engineering Journal, 16(1):3–32, 2011.

[16] Jason I. Hong, Jennifer D. Ng, and Scott Lederer. Privacy risk models for designing privacy-sensitive ubiquitous computing systems. In Designing Interactive Systems (DIS2004), pages 91– 100. ACM Press, 2004.

[17] Andrew Patrick and Steve Kenny. From privacy legislation to interface design: Implementing information privacy in humancomputer interactions. In Privacy Enhancing Technologies, volume 2760 of LNCS, pages 107–124. Springer, 2003.

[18] Eric Yu and Luiz Marcio Cysneiros. Designing for privacy and other competing requirements. In Proceedings of the 2nd Symposium on Requirements Engineering for Information Security (SREIS), pages 15–16, 2002.

[19] Marc Langheinrich. Privacy by design - principles of privacyaware ubiquitous systems. In Proceedings of the 3rd International Conference on Ubiquitous Computing, UbiComp '01, pages 273–291, 2001.

[20] Victoria Bellotti and Abigail Sellen. Design for privacy in ubiquitous computing environments. In Proceedings of the Third Conference on European Conference on Computer-Supported Cooperative Work, pages 77–92, 1993.

[21] Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In Proceedings of the SIGCHI conference

on Human factors in computing systems: Empowering people, CHI '90, pages 249–256, 1990.