

Università Politecnica delle Marche



Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

Relazione progetto per il corso Software Security & Blockchain



Implementazione di un software basato su blockchain per la cura di pazienti a distanza

Professore

Prof. Spalazzi luca

Componenti del gruppo

Bellante Luca
Coccia Giansimone
D'Anna Alessandra
Di Sabatino Walter
Ferretti Laura

Anno Accademico 2023-2024

Contents

1	Introduzione al lavoro svolto	6
1.1	Descrizione del progetto	6
1.2	Tecnologie adottate	6
1.3	Benefici del progetto	7
2	Requirement Engineering	9
2.1	Diagrammi i*	9
2.2	Requirement Analysis	13
2.3	Preliminary Risk Assessment	14
2.4	Risk Identification	15
2.4.1	Asset Identification	15
2.5	Risk Analysis	21
2.5.1	Asset Value Assessment & Policy Objectives	21
2.5.2	Threat Identification	22
2.6	Risk Decomposition	34
2.6.1	Attack Assessment	34
2.7	Risk Reduction	63
2.7.1	Control Identification	64
2.7.2	Feasibility Assessment	64
2.7.3	Security Requirements Definition	64
3	Software Design	114
3.1	Design assets	114
3.2	Architectural design	114
3.2.1	DB Design	116
4	Implementation	121
4.1	General overview	121
4.2	Local Database	122
4.2.1	Tabella Assistito	123
4.2.2	Tabella Autenticazione	123
4.2.3	Tabella cartellaClinica	124
4.2.4	Tabella Curato	124
4.2.5	Tabella Farmaci	124
4.2.6	Tabella medico	125
4.2.7	Tabella operatoreSanitario	125
4.2.8	Tabella patologie	126

4.2.9	Tabella paziente	126
4.2.10	Tabella visitaMedico	127
4.2.11	Tabella visitaOperatore	127
4.3	Interazioni Utente-Software	127
4.4	Authentication	129
4.5	Logging	132
4.6	Gestione delle criticità	133
4.6.1	Check Integrity	133
4.6.2	SQL Injection	135
4.7	Static analizer	136
4.7.1	Bandit	136
4.7.2	Slyther	137
4.8	Testing	137
4.8.1	Solidity contract test	140

List of Figures

1	Logo progetto	6
2	Ganache	7
3	Diagramma i* completo	10
4	Diagramma i* esecutore trattamento	11
5	Diagramma i* infermieri	12
6	Diagramma i* medico base	12
7	Diagramma i* paziente	13
8	Diagramma i* sistema	13
9	Tabella Asset-Value-Objectives	22
10	Tabella Dual-Stride	24
11	Tabella Dual-Stride	25
12	Tabella Dual-Stride	26
13	Tabella Dual-Stride	27
14	Tabella Dual-Stride	28
15	Tabella Dual-Stride	29
16	Tabella Dual-Stride	30
17	Tabella Dual-Stride	31
18	Tabella Dual-Stride	32
19	Tabella Dual-Stride	33
20	Tabella Dual-Stride	34

21	Accesso non autorizzato come amministratore	36
60	Alterazione gestione dei privilegi	41
61	Rendere il sistema poco resistente agli attacchi	42
62	Rendere il sistema poco affidabile	43
63	Accesso non autorizzato	44
64	Divulgare dati	45
65	Falsificazione identità utente	47
66	Falsificazione credenziali d'accesso	48
67	Alterazione dei dati immagazzinati	49
68	Limitazione capacità di accesso ai dat	50
69	Rendere il sistema poco resistente agli attacchi	51
70	Attack tree: dati attività attori.	53
71	Attack tree: policy autorizzazione.	54
72	Attack tree: policy autenticazione.	54
73	Attack tree: traccia degli acessi.	56
74	Attack tree: dati relativi ad un paziente.	57
75	Attack tree: dispositivi IoT per monitoraggio salute.	58
76	Attack tree: Monitorare accesso ai dati.	59
77	Attack tree: gestire processo di lettura, scrittura e salvataggio dei dati.	60
78	Attack tree: procedure di autenticazione.	61
79	Attack tree: controllo delle autorizzazioni.	62
80	Attack tree: accesso ai dati attività.	63
81	Schema STRIDE completo 1, 2 e 3 asset	66
82	Schema STRIDE completo 1, 2 e 3 asset	67
83	Schema STRIDE completo 4 asset	68
84	Schema STRIDE completo 5 asset	69
85	Schema STRIDE completo 6 asset	70
86	Schema STRIDE completo 7 asset	71
87	Schema STRIDE completo 8 asset	72
88	Schema STRIDE completo 9 asset	73
89	Schema STRIDE completo 10 asset	74
90	Schema STRIDE completo 11 asset	75
22	Rimozione o distruzione impropria di dati	76
23	Lettura dei dati confidenziali delle attività degli attori	77
24	Corruzione dei dati riferiti alle attività degli attori	78
25	Rendere non disponibili i dati riferiti alle attività	79
26	Modifiche alle policy di autorizzazione	80
27	Rendere non disponibili le policy di autorizzazione	81

28	Assenza servizi per atti malevoli	82
29	Manipolazione policy di autenticazione	83
30	Interruzione del servizio di tracciamento degli accessi	84
31	Divulgazione e/o lettura del file degli accessi	85
32	Corruzione del file degli accessi	86
33	Manipolazione dei registri relativi agli accessi	87
34	Accesso non autorizzato	88
35	Accesso non autorizzato ai dati clinici	89
36	Divulgazione o modifica di dati privati	90
37	Corruzione dei dati	91
38	Lettura non autorizzata e/o condivisione non autorizzata	92
39	Limitare accesso ai dati	93
40	Alterazione dell'ID di un dispositivo	94
41	Manipolazioni fonte dati IoT	95
42	Manomissione dispositivo	96
43	Rendere il sistema poco affidabile	97
44	Rendere il sistema poco resistente agli attacchi	98
45	Bypassare sistema di monitoraggio dei dati	99
46	Compiere azioni non autorizzate	100
47	Rendere il sistema poco affidabile	101
48	Rendere il sistema poco resistente agli attacchi	102
49	Intercettazione e manipolazione dei dati	103
50	Modifica apportata al processo	104
51	Compiere azioni non autorizzate	105
52	Limitare accesso al processo	106
53	Rendere il sistema poco affidabile	107
54	Rendere il sistema poco resistente agli attacchi	108
55	Falsificazione identità utente	109
56	Divulgazione informazioni riservate	110
57	Limitazione capacità di accesso alla procedura	111
58	Rendere il sistema poco affidabile	112
59	Rendere il sistema poco resistente agli attacchi	113
91	Interazione DB e Blockchain	115
92	Diagramma E-R	117
93	Tabella Assistito	123
94	Tabella autenticazione	123
95	Tabella cartellaClinica	124
96	Tabella curato	124

97	Tabella farmaci	124
98	Tabella medico	125
99	Tabella operatoreSanitario	125
100	Tabella patologie	126
101	Tabella paziente	126
102	Tabella visitaMedico	127
103	Tabella visitaOperatore	127
104	login e menù principale del <i>Medico</i>	129
105	login e menù principale <i>Operatore Sanitario</i>	130
106	login e menù principale del <i>Operatore Sanitario</i>	130
107	Display del tempo di attesa dopo 5 tentativi non andati a buon fine.	131
108	Display del tempo di attesa dopo 10 tentativi non andati a buon fine.	132
110	File di log: <i>app.log</i>	134
111	Check Integrity	134
112	Eccezione personalizzata in caso di <i>Integrity error</i>	135
113	SQL Injection	135
114	Eccezione personalizzata in caso di <i>SQL injection</i>	136
115	Classe di test database	138
116	Test retrieve data	139
117	Test modify data	139
118	Test eccezione	140
119	test visita medica	141

1 Introduzione al lavoro svolto

Nel contesto sempre più digitale della sanità, l'efficace gestione e l'accesso sicuro ai dati sanitari diventano fondamentali per garantire cure efficienti e personalizzate. Il nostro progetto si propone di affrontare questa sfida attraverso l'implementazione di un sistema innovativo basato su tecnologie avanzate come Python, Solidity e Ganache, che consentono l'interazione sicura e efficiente tra medici, pazienti e operatori sanitari, promuovendo la telemedicina e migliorando l'accesso alle cure sanitarie.

1.1 Descrizione del progetto

Il cuore del nostro progetto è un'applicazione software sviluppata in Python, che offre una piattaforma completa per la gestione delle visite mediche e specialistiche, l'aggiornamento delle cartelle cliniche, la tracciabilità dei trattamenti e dei vari farmaci. Utilizziamo Ganache, un ambiente di sviluppo blockchain locale, per garantire l'integrità e la sicurezza dei dati sanitari, salvandoli su blockchain e consentendo un accesso affidabile e sicuro da parte dei professionisti autorizzati.



Figure 1: Logo progetto

1.2 Tecnologie adottate

- *Python*: il linguaggio di programmazione principalmente utilizzato per lo sviluppo dell'applicazione, scelto per la sua flessibilità, facilità d'uso e vasta gamma di librerie disponibili per lo sviluppo di applicazioni web e desktop.

- *Ganache*: una blockchain locale che ci consente di simulare un ambiente blockchain in modo sicuro e privato. Utilizziamo Ganache per salvare e gestire i dati sanitari in modo trasparente e sicuro, sfruttando le caratteristiche di immutabilità e decentralizzazione della blockchain.
- *Solidity*: il linguaggio di programmazione utilizzato per la creazione dei contratti e l’interazione con la blockchain locale. La scelta di Solidity, rispetto ad altri linguaggi quali ad esempio: Scilla, Cadence ... Dipende principalmente da 3 fattori:
 - Solidity è il linguaggio utilizzato per programmare gli smart contract su Ethereum. Offre una ricca documentazione online e numerosi esempi applicativi disponibili sulla rete.
 - Solidity è noto per la sua facilità di apprendimento e utilizzo, grazie alla sua somiglianza con altri linguaggi procedurali come C++ e JavaScript, rendendolo accessibile anche ai programmatore con esperienza in quei linguaggi.
 - Solidity offre un supporto completo per tipi di dati avanzati, compresi quelli strutturati come array e altri tipi di aggregati, che consentono la creazione di costrutti complessi e sofisticati all’interno degli smart contract.



Figure 2: Ganache

1.3 Benefici del progetto

Il nostro progetto porta numerosi vantaggi per tutte le parti coinvolte nel settore sanitario:

- *Miglior accesso alle cure*: consentiamo ai pazienti di accedere alle cure e alle informazioni sanitarie da remoto, riducendo le barriere geografiche e temporali.
- *Integrità dei dati*: grazie all'utilizzo della blockchain, garantiamo l'integrità e l'inalterabilità dei dati sanitari, prevenendo frodi e manipolazioni.
- *Efficienza operativa*: semplifichiamo e ottimizziamo i processi di gestione dei dati sanitari, riducendo i tempi e i costi associati alla documentazione e alla conservazione dei dati.

Il nostro progetto vuole rappresentare un passo avanti verso una sanità digitale più efficiente, sicura e accessibile, migliorando l'esperienza dei pazienti e ottimizzando le risorse del sistema sanitario.

2 Requirement Engineering

2.1 Diagrammi i*

Di seguito elenchiamo i diagrammi i* da noi realizzati, partendo da quello completo mostrato in Figura 3, fino a quelli più specifici; in Figura 4 abbiamo l'i* riguardante l'esecutore del trattamento, in Figura 5 l'i* riguardante gli infermieri, in Figura 6 è possibile visualizzare l'i* sul medico base, in Figura 7 è possibile visionare l'i* sul paziente, mentre in Figura 8 è possibile visualizzare l'i* riguardante il sistema.

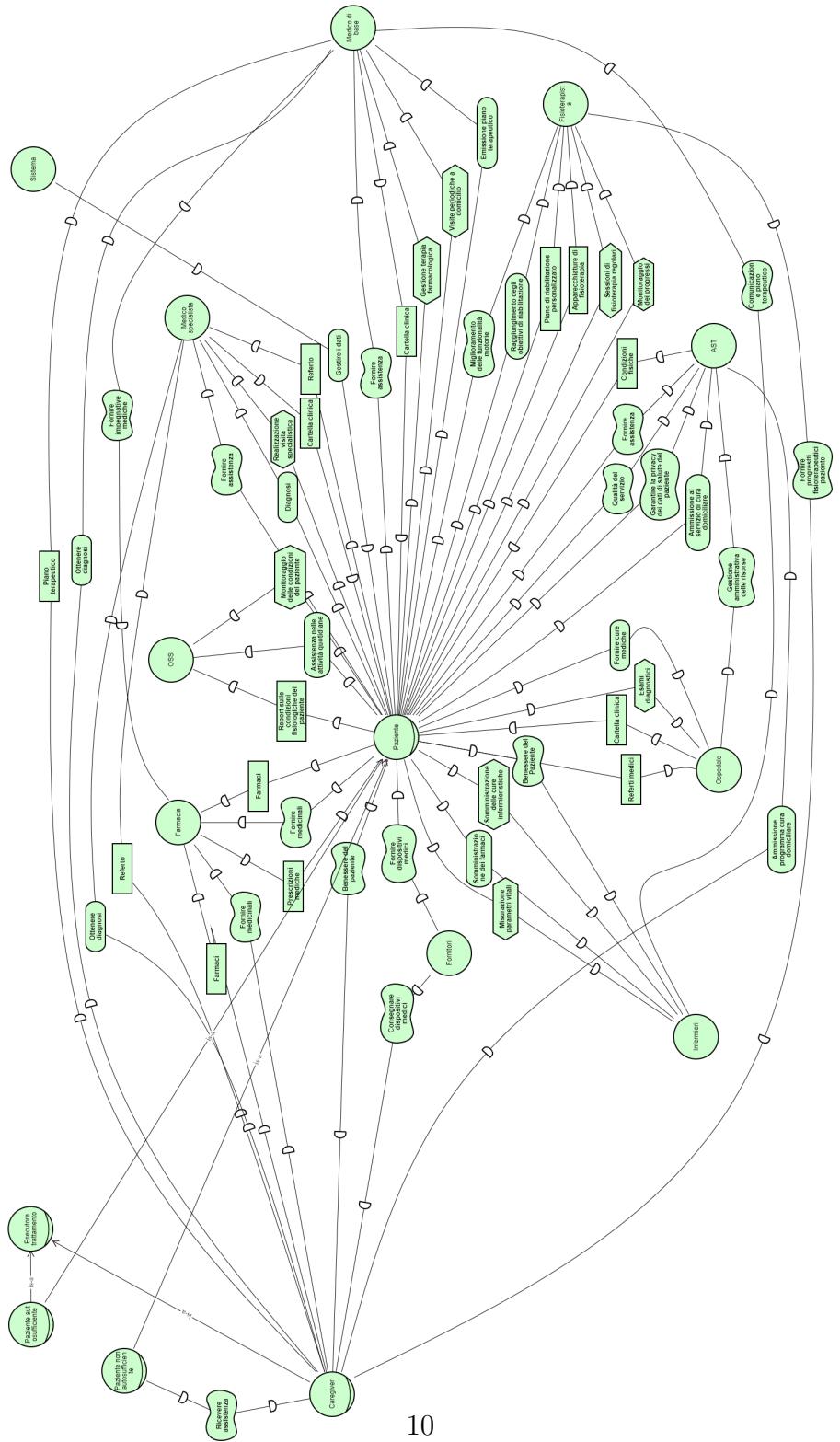


Figure 3: Diagramma i* completo

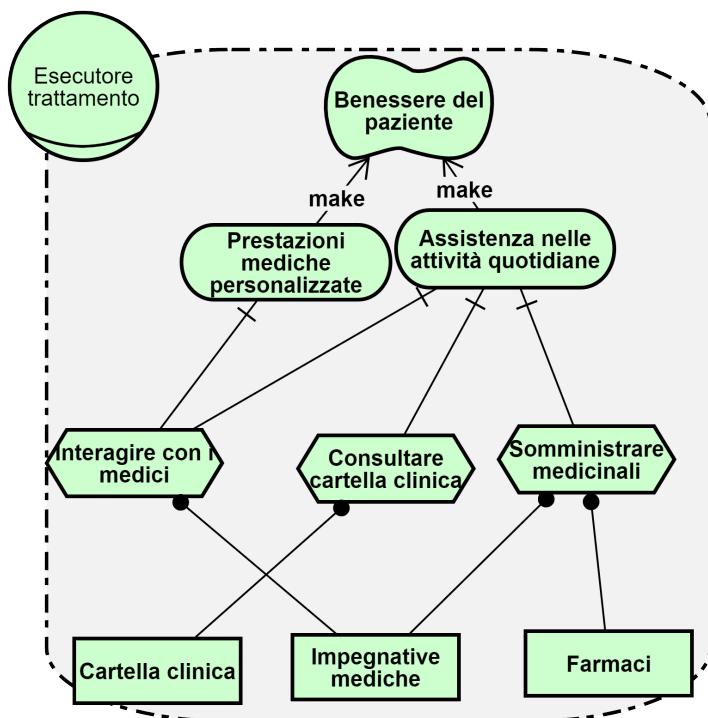


Figure 4: Diagramma i* esecutore trattamento

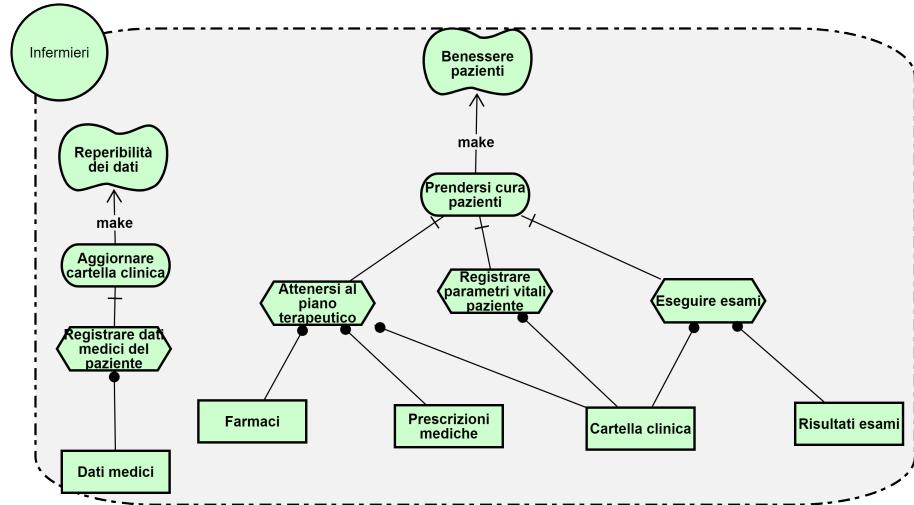


Figure 5: Diagramma i* infermieri

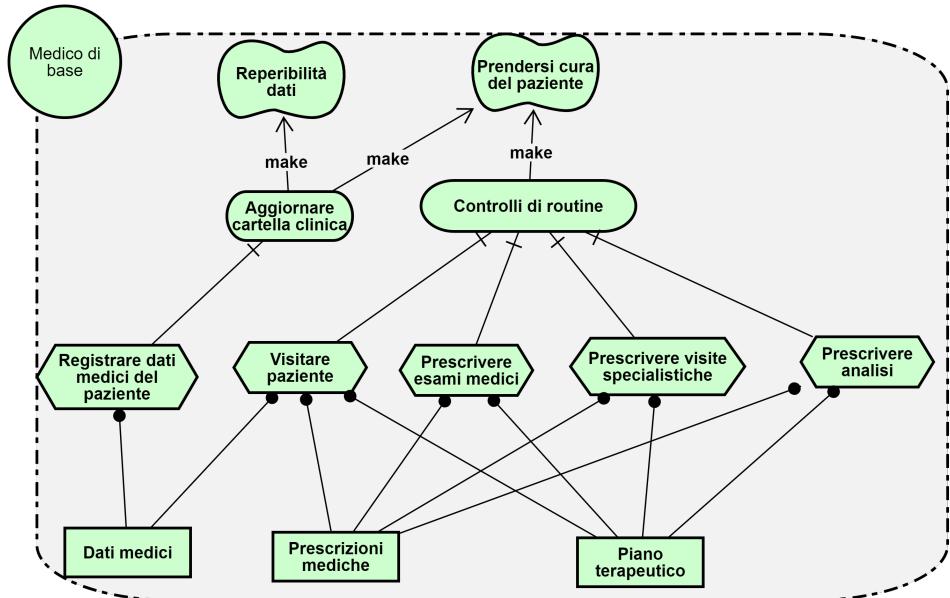


Figure 6: Diagramma i* medico base

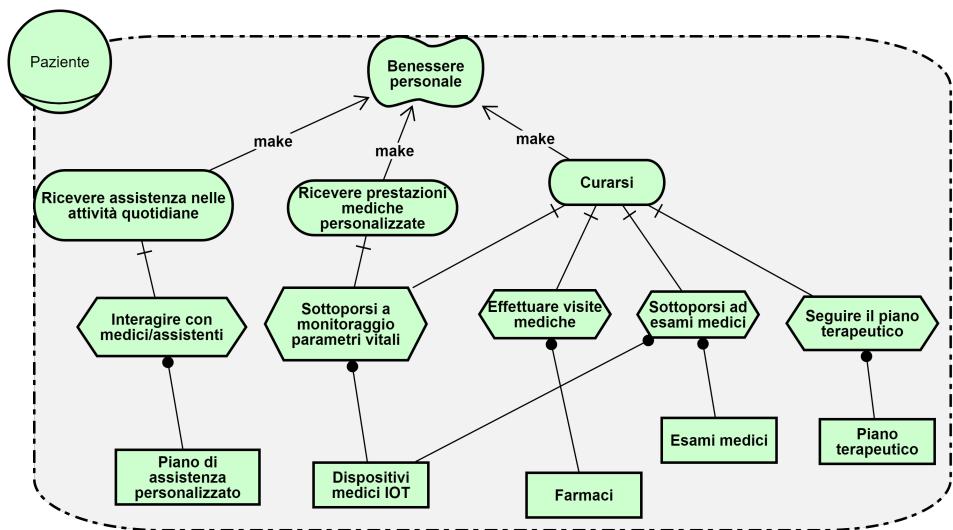


Figure 7: Diagramma i* paziente

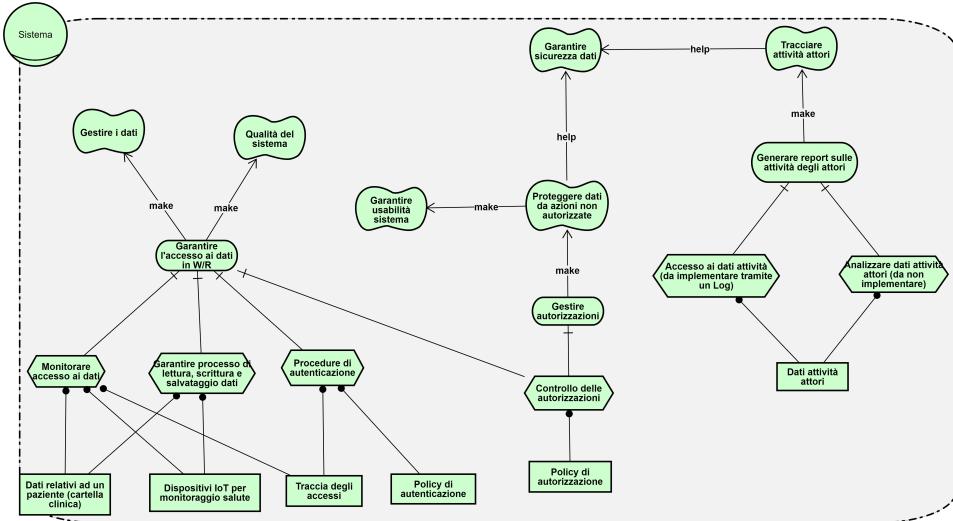


Figure 8: Diagramma i* sistema

2.2 Requirement Analysis

I requisiti di un sistema rappresentano la descrizione dei servizi che il sistema deve fornire e dei vincoli operativi che devono essere rispettati. Queste caratteristiche sono essen-

zialmente la manifestazione delle esigenze del cliente, poiché indicano la necessità di un sistema in grado di risolvere specifici problemi.

L'attività di individuazione, analisi, documentazione e verifica di tali servizi e vincoli è denominata Ingegneria dei Requisiti (RE). L'obiettivo principale di questa fase è identificare sia i requisiti funzionali, che descrivono le funzionalità e i servizi che il sistema deve fornire, sia i requisiti non funzionali, che rappresentano vincoli sulle funzioni e/o sui servizi offerti. Questi ultimi consentono di definire in che modo il sistema deve comportarsi e svolgere i suoi compiti.

Per la descrizione dei requisiti in questo progetto, è stata adottata la notazione **i***. Questo linguaggio di modellazione è particolarmente adatto alla fase iniziale dello sviluppo poiché consente di esaminare l'ambiente organizzativo, i sistemi informativi e gli attori eterogenei coinvolti, analizzando i loro obiettivi, spesso conflittuali, al fine di ottenere una comprensione approfondita del dominio del problema.

Gli attori principali sono:

- *Paziente*, che è in grado di registrarsi e autenticarsi per usufruire del sistema per la visualizzazione delle visite mediche effettuate, per visualizzare la sua cartella clinica e così via.
- *Medico*, che può autenticarsi ed effettuare attività quali ad esempio: inserimento di una visita medica per un paziente a cui è associato, visualizzare tutte le visite mediche effettuate, aggiornare una visita medica, aggiungere un nuovo paziente in cura e modificare la cartella clinica di un paziente (aggiunta di un farmaco, modifica del dosaggio di un farmaco, modifica dei trattamenti ecc...).
- *Operatore Sanitario*, che può autenticarsi per usufruire dei seguenti servizi: aggiungere un paziente come assistito, inserire una visita presso un proprio assistito, visualizzare le visite effettuate e modificare una visita inserita.

2.3 Preliminary Risk Assessment

La *Preliminary Risk Assessment*, o valutazione preliminare del rischio, è una fase critica nel ciclo di sviluppo dei progetti, poiché il suo obiettivo principale è identificare e valutare i potenziali rischi che potrebbero minacciare il successo del sistema in sviluppo. Questa fase è cruciale perché un'analisi accurata dei rischi permette di adottare misure preventive e di mitigazione in anticipo, riducendo così il rischio di costosi errori durante lo sviluppo.

La preliminary risk assessment si articola tipicamente in quattro passaggi fondamentali:

1. *Risk Identification*: identificazione dei rischi

2. *Risk Analysis*: analisi dei rischi individuati
3. *Risk Decomposition*: decomposizione del rischio
4. *Risk Reduction*: riduzione dei rischi

2.4 Risk Identification

Nella fase di *Risk Identification*, si procede con l'individuazione di tutte le potenziali minacce e vulnerabilità che potrebbero incidere sul sistema. Questo processo implica un'analisi approfondita delle minacce esterne, come gli attacchi informatici, così come delle vulnerabilità interne, come errori di programmazione o configurazioni non sicure.

Un punto cruciale in questa fase è l'identificazione degli asset, elementi di valore che richiedono particolare attenzione in quanto costituiscono risorse critiche per il sistema. Per ciascun asset individuato, vengono definiti gli obiettivi funzionali e di sicurezza, nonché le politiche necessarie per garantire la protezione e l'integrità delle risorse stesse.

2.4.1 Asset Identification

Questo processo mira a creare una mappa completa degli asset e delle relative minacce, fornendo così una base solida per la successiva valutazione e gestione dei rischi.

Di seguito riportiamo gli asset individuati nel nostro progetto e, per ciascuno di essi, gli obiettivi da rispettare e le politiche di utilizzo:

Dati attività attori - I dati attività attori sono dati che vengono sfruttati per monitorare le attività compiute dai diversi utenti che fanno uso del sistema. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Confidentiality
- SO2 - Authenticity
- SO3 - Accountability
- SO4 - Availability
- SO5 - Integrity

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - Solo gli utenti autorizzati possono accedere ai dati delle attività degli attori. L’accesso deve essere limitato solo gli utenti che necessitano di queste informazioni per svolgere le proprie mansioni
- OP2 - I dati attività attori sono dati che vengono sfruttati per monitorare le attività compiute dai diversi utenti che fanno uso del sistema.
- OP3 - È fondamentale garantire l’integrità dei dati delle attività degli attori. Ciò significa che i dati devono essere accurati, completi e aggiornati, e che devono essere protetti da modifiche non autorizzate o manipolazioni.

Policy di autorizzazione - Le policy di autorizzazione permettono di definire le regole ed i criteri che determinano chi ha il permesso di accedere a specifiche risorse o eseguire determinate azioni nel sistema. Gli obiettivi che l’asset deve rispettare sono:

- SO1 - Availability
- SO2 - Integrity

Le regole di utilizzo dell’asset (policy) sono:

- OP1 - Le autorizzazioni devono essere assegnate in base ai ruoli e alle responsabilità degli utenti all’interno dell’organizzazione. Ciò assicura che gli utenti abbiano accesso solo alle informazioni pertinenti al loro lavoro.
- OP2 - Le autorizzazioni devono essere gestite utilizzando un sistema di controllo dell’accesso basato su criteri definiti, in questo caso sulla base dei ruoli.

Policy di autenticazione - Le policy di autenticazione permettono di definire le regole, le modalità di accesso e di identificazione degli utenti del sistema. Gli obiettivi che l’asset deve rispettare sono:

- SO1 - Availability
- SO2 - Integrity

Le regole di utilizzo dell’asset (policy) sono:

- OP1 - Tutti gli utenti devono essere soggetti a procedure di autenticazione prima di poter accedere ai dati delle attività degli attori.
- OP2 - Le credenziali di accesso devono essere gestite in modo sicuro e riservato.

- OP3 - Deve essere implementata una politica di blocco degli account che attivi un blocco temporaneo degli account dopo un numero specificato di tentativi di accesso falliti.

Traccia degli accessi - La traccia degli accessi è una rappresentazione dettagliata di tutte le attività di accesso che gli attori operano con il sistema. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Confidentiality
- SO2 - Authenticity
- SO3 - Accountability
- SO4 - Availability
- SO5 - Integrity

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - I registri degli accessi devono essere regolarmente revisionati e analizzati per identificare eventuali pattern anomali o comportamenti sospetti
- OP2 - Deve essere garantita l'integrità dei log degli accessi per assicurare che le informazioni registrate siano accurate e non siano state manipolate o alterate.

Dati relativi ad un paziente - I dati relativi ad un paziente possono essere identificati con la cartella clinica e rappresentano i dati relativi allo stato di salute di un particolare utente. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Confidentiality
- SO2 - Authenticity
- SO3 - Accountability
- SO4 - Availability
- SO5 - Integrity

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - Solo il personale sanitario autorizzato deve avere accesso ai dati della cartella clinica di un paziente
- OP2 - È fondamentale garantire l'integrità dei dati contenuti nella cartella clinica. Ciò significa che i dati devono essere accurati, completi e aggiornati.

Dispositivi IoT per monitoraggio salute - I dispositivi IoT sono strumenti fisici distribuiti ai pazienti da enti autorizzati, finalizzati al monitoraggio remoto di specifici parametri biometrici o di salute. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Safety
- SO2 - Authenticity
- SO3 - Accountability
- SO4 - Reliability
- SO5 - Resilience

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - L'accesso ai dati generati dai dispositivi IoT per il monitoraggio della salute deve essere limitato al personale sanitario autorizzato e ai pazienti stessi.
- OP2 - È essenziale garantire l'integrità dei dati raccolti dai dispositivi IoT.

Monitorare accesso ai dati - Il monitoraggio continuo dei dati è un'operazione essenziale che consente di garantire un livello di sicurezza adeguato. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Assurance
- SO2 - Accountability
- SO3 - Reliability
- SO4 - Resilience

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - Deve essere mantenuto un registro dettagliato di tutti gli accessi ai dati.
- OP2 - Il monitoraggio dell'accesso ai dati deve avvenire in tempo reale
- OP3 - Gli accessi ai dati sensibili dovrebbero essere limitati solo agli utenti autorizzati e necessari per svolgere le loro funzioni specifiche

Gestire procedure di lettura, scrittura e salvataggio dati - È importante gestire attentamente le operazioni di lettura, scrittura e salvataggio dei dati al fine di mantenere un registro completo delle informazioni relative ai pazienti, delle operazioni svolte su ciascun paziente e di altri dati rilevanti. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Integrity
- SO2 - Reliability
- SO3 - Resilience
- SO4 - Authenticity
- SO5 - Accountability
- SO6 - Assurance

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - Le autorizzazioni per la lettura, la scrittura e il salvataggio dei dati devono essere assegnate in base al principio del minimo privilegio. Ciò significa che gli utenti dovranno ricevere solo le autorizzazioni necessarie per svolgere le loro funzioni specifiche.

Procedure di autenticazione - Le procedure di autenticazione sono stabilite per prevenire l'accesso al sistema da parte di individui non autorizzati e fanno riferimento ai quei processi che verificano l'identità di un utente o di un'entità che tenta di accedere al sistema. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Availability
- SO2 - Reliability
- SO3 - Resilience
- SO4 - Confidentiality
- SO5 - Authentication

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - Dovrebbe essere implementato un sistema di monitoraggio degli accessi che registri e tracci le attività di accesso.
- OP2 - Le credenziali degli utenti devono essere gestite in modo centralizzato e protette da accessi non autorizzati

Controllo delle autorizzazioni - Il Controllo delle Autorizzazioni riguarda la gestione dei privilegi e delle autorizzazioni degli utenti una volta che sono stati autenticati con successo. Questo processo definisce quali azioni o risorse un utente è autorizzato ad accedere o manipolare all'interno del sistema. Le autorizzazioni, nel nostro caso, sono configurate in base a ruoli. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Confidentiality
- SO2 - Authorization
- SO3 - Resilience
- SO4 - Reliability
- SO5 - Integrity

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - I ruoli e le responsabilità degli utenti devono essere definite chiaramente.
- OP2 - è implementato un controllo degli accessi basato su ruolo.

Accesso ai dati attività - L'accesso ai dati attività rappresenta la capacità di ottenere informazioni relative ai movimenti compiuti dagli utenti, questi includono gli accessi e le varie operazioni svolte. Gli obiettivi che l'asset deve rispettare sono:

- SO1 - Accountability
- SO2 - Integrity
- SO3 - Availability
- SO4 - Authenticity
- SO5 - Resilience

Le regole di utilizzo dell'asset (policy) sono:

- OP1 - L'accesso ai dati delle attività dovrebbe essere concesso solo agli utenti autorizzati che necessitano di tali informazioni per svolgere le proprie mansioni.

2.5 Risk Analysis

L'analisi dei rischi è un processo che si pone l'obiettivo di valutare e gestire potenziali minacce e opportunità. Dopo aver identificato i potenziali rischi, essi vengono analizzati per determinare la loro gravità e la loro probabilità di occorrenza. Questo aiuta a prioritizzare i rischi in base alla loro importanza e adottare misure preventive proporzionate.

2.5.1 Asset Value Assessment & Policy Objectives

Nella valutazione del valore degli asset, nota come *Asset Value Assessment*, si analizza e quantifica l'importanza patrimoniale di ciascun asset all'interno del progetto. Tale processo consente di comprendere quanto sia cruciale ogni asset per il corretto funzionamento del sistema e quale impatto la sua perdita potrebbe avere sulle attività.

Parallelamente, la definizione degli obiettivi di politica, conosciuta come *Policy Objectives*, si concentra sulle mete e le priorità di un progetto in vari ambiti, offrendo una guida strategica per lo sviluppo e l'attuazione delle politiche stesse. Attraverso le Policy Objectives, si delineano le direzioni chiave su cui concentrare le risorse e le azioni necessarie per ottenere risultati desiderati.

Nel nostro caso, la valutazione si basa su una quantificazione in termini monetari. Di seguito riportiamo la tabella nella quale sono esplicitate queste valutazioni, composta da tre colonne:

- **Asset:** elenco degli asset individuati.
- **Value:** valore associato a ciascun asset.
- **Objective:** obiettivi che ciascun asset deve rispettare

Asset (solo sul sistema)	Value	Objective
Dati attività attori	Sono dati che servono per monitorare le attività compiute dai diversi utenti	Confidentiality, Authenticity, Accountability, Availability, Integrity
Policy di autorizzazione	Sono le policy che definiscono le regole e i criteri che determinano chi ha il permesso di accedere a specifiche risorse o eseguire determinate azioni nel sistema.	Integrity, Availability
Policy di autenticazione	Sono le policy che definiscono come avviene l'accesso e l'identificazione degli utenti in un sistema	Integrity, Availability
Traccia degli accessi	E' una registrazione dettagliata di tutte le attività di accesso che gli attori fanno con il sistema	Confidentiality, Authenticity, Integrity, Availability, Accountability
Dati relativi ad un paziente (cartella clinica)	Raccolta di dati relativi allo stato di salute un particolare utente (paziente)	Confidentiality, Integrity, Availability, Accountability, Authenticity
Dispositivi IoT per monitoraggio salute	Dispositivi fisici rilasciati al paziente da enti autorizzati, e che consentano il monitoraggio a distanza di determinati valori	Safety, Reliability, Resilience, Authenticity, Accountability
Monitorare accesso ai dati	Monitoring continuo dell'accesso ai dati, è un'azione richiesta per garantire un livello di sicurezza adeguato	Reliability, Resilience, Accountability, Assurance
Gestire processo di lettura, scrittura e salvataggio dei dati	E' opportuno gestire le fasi di lettura, scrittura e salvataggio dei dati così da avere a disposizione un registro completo di tutti i pazienti, le operazioni effettuate per un dato paziente ecc	Integrity , Reliability, Resilience, Authenticity, Accountability, Assurance
Procedure di autenticazione	Procedura definita per impedire che soggetti non autorizzati entrino nel sistema sviluppato	Availability ,Reliability, Resilience, Confidentiality, Authentication
Controllo delle autorizzazioni	Procedura che gestisce i privilegi degli utenti per accedere alle risorse o eseguire determinate azioni	Confidentiality, Authorization, Resilience, Reliability, Integrity
Accesso ai dati attività	Capacità di ottenere informazioni relative agli accessi effettuati e alle operazioni svolte	Accountability, Integrity, Availability, Authenticity, Resilience

Figure 9: Tabella Asset-Value-Objectives

2.5.2 Threat Identification

L'identificazione delle minacce rappresenta una fase cruciale che richiede un'analisi approfondita delle potenziali fonti di rischio che potrebbero impattare negativamente il progetto. Questo processo coincide con la possibile violazione degli obiettivi previamente identificati per ciascun asset. In questo contesto, ci avvaliamo dei modelli STRIDE e Dual STRIDE, utilizzati nel campo della sicurezza informatica per identificare e classificare le minacce al nostro sistema. STRIDE è un acronimo che comprende:

- S - Spoofing
- T - Tampering
- R - Repudiation
- I - Information Disclosure
- D - Denial of Service
- E - Elevation of Privilege

Il Dual STRIDE estende il concetto di STRIDE aggiungendo un'ulteriore dimensione nella valutazione delle minacce informatiche. Oltre alle minacce intenzionali considerate

da STRIDE, il Dual STRIDE include anche le minacce non intenzionali o accidentali, derivanti da errori umani, malfunzionamenti hardware o software, incidenti naturali e altre situazioni non maliziose. Queste minacce, integrate nel processo di valutazione del rischio, consentono di ottenere una visione più completa e realistica delle vulnerabilità del sistema, aggiungendo colonne come Danger, Unreliability e Absence of Resilience a quelle precedentemente elencate.

Come risultato di questa fase, si ottiene il Dual-STRIDE, rappresentato da 10 a 20. In particolare, per ogni asset viene inserita una "X" ogni volta che viene individuata una minaccia che comporta la violazione di un requisito di sicurezza. Vengono inoltre definiti i valori monetari per il "Value" (il valore dell'asset) e l'"Exposure" (il rischio associato a una gestione errata dell'asset). Infine, vengono identificati gli "Attack", cioè gli attacchi ai quali gli asset possono essere vulnerabili.

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		X										Accesso non autorizzato come amministratore
			X									Rimozione o distruzione impropria di dati
Dati attività attori	100.000 - 200.000			X							50.000 - 100.000	Lettura dei dati confidenziali delle attività degli attori
				X								Corruzione dei dati riferiti alle attività degli attori al fine di rendere incomprensibile quale attività è stata compiuta e da quale attore
												Rendere non disponibili i

Figure 10: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
						X						Rendere non disponibili dati riferiti alle attività compiute dagli attori
Policy di autorizzazione	50.000 - 60.000		X								10.000 - 20.000	Modifiche alle policy di autorizzazione
						X						Rendere non disponibili le policy di autorizzazione
Policy di autenticazione	50.000 - 60.000					X					10.000 - 20.000	Assenza servizi per atti malevoli dovuti a errori di autenticazione
			X									Manipolazione policy di autenticazione

Figure 11: Tabella Dual-Stride

Asset	Value	Spoofering	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
					X							Interruzione del servizio di tracciamento degli accessi
				X								Divulgazione e/o lettura dei file degli accessi
Traccia degli accessi	20.000 - 30.000		X								10.000 - 20.000	Corruzione dei file degli accessi al fine di rendere incomprensibile chi ha eseguito l'accesso e in quale momento
		X										Manipolazione dei registri relativi agli accessi degli attori

Figure 12: Tabella Dual-Stride

Asset	Value	Spoofering	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		X										Accesso non autorizzato (solo chi è autorizzato può accedere al file)
		X										Accesso non autorizzato ai dati clinici riferiti ad un paziente
			X									Divulgazione o modifica di dati privati
				X								Corruzione dei dati al fine di rendere incomprensibile chi ha prescritto quale farmaco/medicinale
Dati relativi ad un paziente (cartella clinica)	500.000 - 600.000			X							150.000 - 200.000	Lettura non autorizzata e/o condivisione non autorizzata dei dati clinici di un paziente
					X							Limitare accesso ai dati

Figure 13: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		X										
			X									Alterazione dell'ID di un dispositivo
												Manipolazioni fonte dati IoT
Dispositivi IoT per monitoraggio salute	300.000 - 400.000						X				50.000 - 100.000	Manomissione dispositivo
									X			Rendere il sistema poco affidabile

Figure 14: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
									X			Rendere il sistema poco resistente agli attacchi
				X								Bypassare sistema di monitoraggio dei dati
							X					Compiere azioni non autorizzate
Monitorare accesso ai dati	40.000 - 50.000								X		20.000 - 30.000	Rendere il sistema poco affidabile
										X		Rendere il sistema poco resistente agli attacchi

Figure 15: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		X										Intercettazione e manipolazione dei dati
			X									Modifica apportata al processo da parte di ignoti
					X							Compire azioni non autorizzate
Gestire processo di lettura, scrittura e salvataggio dei dati	100.000 - 200.000					X					50.000 - 60.000	Limitare accesso al processo
					X							Rendere il sistema poco affidabile
							X					Rendere il sistema poco resistente agli attacchi

Figure 16: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
Procedure di autenticazione	50.000 - 60.000	X										Falsificazione identità utente
				X								Divulgazione informazioni riguardanti le procedure
					X							Limitazione capacità di accesso alla procedura
							X					Rendere il sistema poco affidabile

Figure 17: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
										X		Rendere il sistema poco resistente agli attacchi
			X									Alterazione della gestione dei privilegi
										X		Rendere il sistema poco resistente agli attacchi
									X			Rendere il sistema poco affidabile
Controllo delle autorizzazioni	50.000 - 60.000										10.000 - 20.000	

Figure 18: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DOS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		X										Accesso non autorizzato
					X							Divulgare dati
												Falsificazione identità utente
Accesso ai dati attività	80.000 - 100.000			X							50.000 - 100.000	Falsificazione credenziali d'accesso

Figure 19: Tabella Dual-Stride

Asset	Value	Spoofing	Tampering	Repudiation	Information disclosure	DoS	Elevation of privilege	Danger	Unreliability	Absence of Resilience	Exposure	Attack
		x										Alterazione dei dati immagazzinati
					x							Limitazione capacità di accesso ai dati
									x			Rendere il sistema poco resistente agli attacchi

Figure 20: Tabella Dual-Stride

2.6 Risk Decomposition

La *Risk Decomposition*, o scomposizione del rischio, è un processo utilizzato nell'analisi del rischio per suddividere un rischio complesso in componenti più gestibili e comprensibili. Durante questa fase, il rischio viene analizzato in termini di variabili o fattori che contribuiscono alla sua manifestazione. Questi fattori possono includere eventi di base, cause radicate, processi o azioni che influenzano il verificarsi del rischio e le conseguenze che potrebbero derivarne. Questo approccio consente una valutazione più dettagliata e precisa dei rischi, nonché una migliore comprensione delle cause sottostanti.

2.6.1 Attack Assessment

L'*Attack Assessment*, o valutazione degli attacchi, si concentra sull'identificazione e sulla valutazione degli attacchi potenziali che potrebbero essere diretti verso il sistema informativo. Durante questa fase, vengono esaminati diversi scenari e tecniche utilizzate dagli attaccanti per compromettere la sicurezza del sistema, compresi attacchi informatici come hacking, malware, phishing, ingegneria sociale, denial of service (DoS) e altri.

L'obiettivo primario dell'attack assessment è individuare i potenziali punti di vulnerabilità nel sistema e valutarne la criticità e l'esposizione effettiva agli attacchi. Una volta identificati gli attacchi potenziali e valutata la loro gravità, è possibile sviluppare e implementare strategie di mitigazione e protezione per ridurre il rischio di compromissione della sicurezza del sistema.

Per rappresentare gli attacchi individuati, facciamo uso delle *Tabelle di Jacobson*. Di seguito riporteremo gli attacchi individuati per ciascun asset, consentendo una comprensione più dettagliata delle minacce specifiche che potrebbero influenzare ogni asset e guidare le strategie di difesa.

Dati attività attori Per tale asset abbiamo individuato 5 possibili rischi:

1. *Accesso non autorizzato come amministratore*: questo rischio si verifica quando un individuo o un attaccante ottiene accesso ai dati o ai sistemi del sistema informativo con privilegi amministrativi non autorizzati. La relativa tabella di Jacobson è 21
2. *Rimozione o distruzione impropria di dati*: questo rischio si riferisce alla possibilità che i dati delle attività degli attori vengano eliminati o danneggiati in modo improprio o intenzionale. La relativa tabella di Jacobson è 22
3. *Lettura dei dati confidenziali delle attività degli attori*: questo rischio riguarda la possibilità che un attaccante possa accedere e leggere i dati confidenziali relativi alle attività degli attori. La relativa tabella di Jacobson è 23
4. *Corruzione dei dati riferiti alle attività degli attori al fine di rendere incomprensibile quale attività è stata compiuta e da quale attore*: questo rischio si riferisce alla possibilità che i dati relativi alle attività degli attori vengano alterati o manipolati in modo tale da rendere difficile o impossibile comprendere quali azioni sono state compiute e da chi. La relativa tabella di Jacobson è 24
5. *Rendere non disponibili i dati riferiti alle attività compiute dagli attori*: questo rischio si verifica quando i dati relativi alle attività degli attori diventano inaccessibili o non disponibili a causa di malfunzionamenti del sistema, attacchi informatici o altre cause. La relativa tabella di Jacobson è 25

Policy di autorizzazione Per tale asset abbiamo individuato 2 possibili rischi:

1. *Modifiche alle policy di autorizzazione*: questo rischio si riferisce alla possibilità che le policy di autorizzazione, che determinano i livelli di accesso e le autorizzazioni degli utenti al sistema, vengano modificate in modo improprio o non autorizzato. La relativa tabella di Jacobson è 26
2. *Rendere non disponibili le policy di autorizzazione*: questo rischio si verifica quando le policy di autorizzazione diventano inaccessibili o non disponibili per un periodo di tempo prolungato. La relativa tabella di Jacobson è 27

Use case ID: UCA-01	
Use case Name: Accesso non autorizzato come amministratore	
Actors	Sistema, Attaccante
Description	Dopo un certo numero di tentativi l'attaccante riesce ad indovinare le credenziali d'accesso e ad effettuare l'accesso come amministratore
Data	Dati utente
Stimulus and preconditions	<ul style="list-style-type: none"> 1. Il sistema si basa su un sistema di autenticazione a singolo fattore. 2. L'attaccante deve conoscere il nome utente (username) relativo all'account dell'amministratore.
Basic Flow	<ul style="list-style-type: none"> 1. L'attaccante scrive un algoritmo che genera delle password a partire da un dizionario. 2. L'algoritmo prova in maniera iterativa tutte le password fin quando non indovina quella corretta.
Alternative Flow	-
Exception Flow	<ul style="list-style-type: none"> 1. A seguito delle troppe richieste di inserire la password, il sistema per gli accessi smette di funzionare.
Response and Postconditions	L'attaccante accede al sistema come se fosse l'amministratore
Non Functional Requirements	Authentication
Comments	-

Figure 21: Accesso non autorizzato come amministratore

Policy di autenticazione Per tale asset abbiamo individuato 2 possibili rischi:

1. *Assenza servizi per atti malevoli dovuti a errori di autenticazione*: questo rischio si riferisce alla possibilità che errori nell'autenticazione degli utenti possano consentire ad attaccanti di eseguire azioni malevole all'interno del sistema. La relativa tabella di Jacobson è 28
2. *Manipolazione policy di autenticazione*: questo rischio si verifica quando le policy di autenticazione, che definiscono i requisiti e i criteri per la verifica dell'identità degli utenti, vengono manipolate o alterate in modo improprio. La relativa tabella di Jacobson è 29

Traccia degli accessi Per tale asset abbiamo individuato 5 possibili rischi:

1. *Interruzione del servizio di tracciamento degli accessi*: questo rischio si verifica quando il servizio di registrazione e tracciamento degli accessi al sistema viene interrotto o compromesso. Le relative tabelle di Jacobson sono 30
2. *Divulgazione e/o lettura del file degli accessi*: questo rischio si riferisce alla possibilità che il file contenente i log degli accessi al sistema venga divulgato o letto da utenti non autorizzati. La relativa tabella di Jacobson è 31
3. *Corruzione del file degli accessi al fine di rendere incomprensibile chi ha eseguito l'accesso e in quale momento*: questo rischio si verifica quando il file di log degli accessi viene corrotto o alterato in modo tale da rendere difficile o impossibile determinare chi ha effettuato determinate azioni e quando. La relativa tabella di Jacobson è 32
4. *Manipolazione dei registri relativi agli accessi degli attori*: questo rischio si riferisce alla possibilità che i registri degli accessi vengano manipolati o alterati in modo improprio. La relativa tabella di Jacobson è 33
5. *Accesso non autorizzato (solo chi è autorizzato può accedere al file)*: questo rischio si verifica quando gli utenti non autorizzati ottengono accesso al file degli accessi. La relativa tabella di Jacobson è 34

Dati relativi ad un paziente Per tale asset abbiamo individuato 5 possibili rischi:

1. *Accesso non autorizzato ai dati clinici riferiti ad un paziente*: questo rischio si riferisce alla possibilità che individui non autorizzati ottengano accesso ai dati clinici di un paziente, violando la sua privacy e compromettendo la riservatezza delle informazioni mediche. La relativa tabella di Jacobson è 35
2. *Divulgazione o modifica di dati privati*: questo rischio si verifica quando i dati privati di un paziente vengono divulgati o modificati senza autorizzazione. La relativa tabella di Jacobson è 36
3. *Corruzione dei dati al fine di rendere incomprensibile chi ha prescritto quale terapia/medicinale al paziente*: questo rischio si riferisce alla possibilità che i dati relativi alle prescrizioni mediche vengano corrotti o alterati in modo tale da rendere difficile o impossibile determinare quale terapia o medicinale è stato prescritto a un paziente. La relativa tabella di Jacobson è 37

4. *Lettura non autorizzata e/o condivisione non autorizzata dei dati clinici di un paziente*: questo rischio si verifica quando i dati clinici di un paziente vengono letti o condivisi da individui non autorizzati. La relativa tabella di Jacobson è 38
5. *Limitare accesso ai dati*: questo rischio si riferisce alla possibilità che l'accesso ai dati clinici di un paziente venga limitato in modo improprio o non autorizzato. La relativa tabella di Jacobson è 39

Dispositivi IoT per il monitoraggio della salute Per tale asset abbiamo individuato 5 possibili rischi:

1. *Alterazione dell'ID di un dispositivo*: questo rischio si riferisce alla possibilità che l'identificatore di un dispositivo IoT venga alterato o manipolato, compromettendo l'integrità e l'affidabilità dei dati raccolti e trasferiti dal dispositivo. La relativa tabella di Jacobson è 40
2. *Manipolazioni fonte dati IoT*: questo rischio si verifica quando i dati provenienti dai dispositivi IoT vengono manipolati o alterati in modo improprio. La relativa tabella di Jacobson è 41
3. *Manomissione dispositivo*: questo rischio si riferisce alla possibilità che i dispositivi IoT per il monitoraggio della salute vengano manomessi o danneggiati, compromettendo la loro capacità di raccogliere e trasmettere dati accurati sulla salute del paziente. La relativa tabella di Jacobson è 42
4. *Rendere il sistema poco affidabile*: questo rischio si verifica quando il sistema di monitoraggio della salute basato su dispositivi IoT diventa poco affidabile a causa di malfunzionamenti, errori o problemi tecnici. La relativa tabella di Jacobson è 43
5. *Rendere il sistema poco resistente agli attacchi*: questo rischio si riferisce alla possibilità che il sistema di monitoraggio della salute basato su dispositivi IoT diventi vulnerabile agli attacchi informatici e alle intrusioni. La relativa tabella di Jacobson è 44

Monitorare accesso ai dati Per tale asset abbiamo individuato 4 possibili rischi:

1. *Bypassare sistema di monitoraggio dei dati*: questo rischio si verifica quando gli utenti o gli attaccanti riescono a eludere o aggirare il sistema di monitoraggio dei dati, impedendo al sistema di rilevare e registrare le attività di accesso non autorizzato o sospetto. La relativa tabella di Jacobson è 45

2. *Compiere azioni non autorizzate*: questo rischio si riferisce alla possibilità che gli utenti compiano azioni non autorizzate sui dati, come l'accesso, la modifica o la cancellazione di informazioni, senza le autorizzazioni appropriate. La relativa tabella di Jacobson è 46
3. *Rendere il sistema poco affidabile*: questo rischio si verifica quando il sistema di monitoraggio dell'accesso ai dati diventa poco affidabile a causa di malfunzionamenti, errori o problemi tecnici. La relativa tabella di Jacobson è 47
4. *Rendere il sistema poco resistente agli attacchi*: questo rischio si riferisce alla possibilità che il sistema di monitoraggio dell'accesso ai dati diventi vulnerabile agli attacchi informatici e alle intrusioni. La relativa tabella di Jacobson è 48

Gestire procedure di lettura, scrittura e salvataggio dati Per tale asset abbiamo individuato 6 possibili rischi:

1. *Intercettazione e manipolazione dei dati*: questo rischio si riferisce alla possibilità che i dati vengano intercettati durante la loro trasmissione e manipolati in modo improprio o dannoso. La relativa tabella di Jacobson è 49
2. *Modifica apportata al processo da parte di ignoti*: questo rischio si verifica quando persone non autorizzate modificano le procedure di lettura, scrittura e salvataggio dei dati in modo improprio o dannoso. La relativa tabella di Jacobson è 50
3. *Compiere azioni non autorizzate*: come nel caso precedente, questo rischio si riferisce alla possibilità che gli utenti compiano azioni non autorizzate sui dati, violando le policy e le regolamentazioni stabilite per la gestione dei dati. La relativa tabella di Jacobson è 51
4. *Limitare accesso al processo*: questo rischio si verifica quando l'accesso alle procedure di lettura, scrittura e salvataggio dei dati viene limitato in modo improprio o non autorizzato, ostacolando le operazioni del sistema e compromettendo la produttività. La relativa tabella di Jacobson è 52
5. *Rendere il sistema poco affidabile e poco resistente agli attacchi*: questi due rischi sono analoghi a quelli individuati nell'asset "Monitorare accesso ai dati" e si riferiscono alla possibilità che il sistema diventi poco affidabile o vulnerabile agli attacchi informatici, compromettendo la sicurezza e l'integrità dei dati. Le relative tabelle di Jacobson sono 53 e 54

Procedure autenticazione Per tale asset abbiamo individuato 5 possibili rischi:

1. *Falsificazione identità utente*: questo rischio si riferisce alla possibilità che un utente falsifichi la propria identità per ottenere accesso non autorizzato al sistema. La relativa tabella di Jacobson è 55
2. *Divulgazione informazioni riguardanti le procedure*: questo rischio si verifica quando informazioni sensibili sulle procedure di autenticazione vengono divulgati o resi disponibili a persone non autorizzate. La relativa tabella di Jacobson è 56
3. *Limitazione capacità di accesso alla procedura*: questo rischio si riferisce alla possibilità che l'accesso alle procedure di autenticazione venga limitato in modo improprio o non autorizzato, impedendo agli utenti autorizzati di accedere al sistema. La relativa tabella di Jacobson è 57
4. *Rendere il sistema poco affidabile e poco resistente agli attacchi*: questi due rischi sono analoghi a quelli individuati nei precedenti asset e si riferiscono alla possibilità che il sistema di autenticazione diventi poco affidabile o vulnerabile agli attacchi informatici, compromettendo la sicurezza e l'integrità del sistema. Le relative tabelle di Jacobson sono 58 e 59

Controllo delle autorizzazioni Per tale asset abbiamo individuato 5 possibili rischi:

1. *Alterazione della gestione dei privilegi*: questo rischio si riferisce alla possibilità che i privilegi di accesso vengano alterati o manipolati in modo improprio o non autorizzato. La relativa tabella di Jacobson è 60
2. *Rendere il sistema poco resistente agli attacchi*: questo rischio si riferisce alla possibilità che il sistema di controllo delle autorizzazioni diventi vulnerabile agli attacchi informatici e alle intrusioni. La relativa tabella di Jacobson è 61
3. *Rendere il sistema poco affidabile*: questo rischio si verifica quando il sistema di controllo delle autorizzazioni diventa poco affidabile a causa di malfunzionamenti, errori o problemi tecnici. La relativa tabella di Jacobson è 62
4. *Accesso non autorizzato*: questo rischio si riferisce alla possibilità che gli utenti ottengano accesso non autorizzato a risorse o dati sensibili a causa di errori nel sistema di controllo delle autorizzazioni. La relativa tabella di Jacobson è 63

5. *Divulgare dati*: questo rischio si verifica quando i dati sensibili vengono divulgati o resi disponibili a persone non autorizzate a causa di errori o violazioni nel sistema di controllo delle autorizzazioni. La relativa tabella di Jacobson è 64

Use case ID: UCA-40	
Use case Name: Alterazione della gestione dei privilegi	
Actors	Sistema, attaccante
Description	L'attaccante altera la gestione dei privilegi nel sistema di controllo delle autorizzazioni per ottenere accesso non autorizzato a risorse sensibili, minando l'integrità del controllo delle autorizzazioni.
Data	Controllo delle autorizzazioni
Stimulus and preconditions	L'attaccante ricerca e identifica le vulnerabilità nel sistema di controllo delle autorizzazioni che consentono l'alterazione dei privilegi.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante sfrutta le vulnerabilità identificate per alterare il proprio livello di privilegio, ottenendo accesso non autorizzato a risorse o funzionalità limitate. 2. Una volta sfruttate le vulnerabilità, l'attaccante modifica i propri privilegi nel sistema di controllo delle autorizzazioni, ottenendo un livello di accesso superiore a quello assegnato originariamente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	1. L'attaccante, grazie alla modifica dei privilegi, ottiene accesso non autorizzato a risorse o funzionalità riservate, andando a minare l'integrità del controllo delle autorizzazioni.
Non Functional Requirements	Integrity
Comments	-

Figure 60: Alterazione gestione dei privilegi

Use case ID: UCA-41	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Controllo delle autorizzazioni
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata 2. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target. 3. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target. 4. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema. 5. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target. 6. L'attaccante può cercare di mantenere l'attacco nel tempo, adattandone le eventuali contromisure adottate dalla difesa del target.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 61: Rendere il sistema poco resistente agli attacchi

Use case ID: UCA-42	
Use case Name: Rendere il sistema poco affidabile	
Actors	Sistema, Attaccante
Description	<p>Durante la fase di sviluppo di un sistema di monitoraggio della salute basato su dispositivi IoT, un attaccante sfrutta una vulnerabilità nel processo di sviluppo per introdurre codice dannoso. Questo codice dannoso è progettato per rendere il sistema poco affidabile, causando malfunzionamenti, falsi positivi/negativi e potenzialmente compromettendo la sicurezza e l'integrità delle informazioni di monitoraggio della salute.</p>
Data	Controllo delle autorizzazioni
Stimulus and preconditions	Il sistema di monitoraggio della salute è in fase di sviluppo. Gli sviluppatori stanno implementando nuove funzionalità o modificando il codice esistente.
Basic Flow	<ol style="list-style-type: none"> L'attaccante, approfittando di una vulnerabilità durante lo sviluppo, inserisce codice dannoso nel repository del progetto o durante la compilazione del codice. Il codice dannoso è progettato per compromettere l'affidabilità del sistema di monitoraggio della salute, causando errori casuali, falsi positivi/negativi, o il malfunzionamento del dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il codice dannoso è stato integrato nel sistema di monitoraggio della salute durante lo sviluppo.
Non Functional Requirements	Reliability
Comments	-

Figure 62: Rendere il sistema poco affidabile

Use case ID: UCA-43	
Use case Name: Accesso non autorizzato	
Actors	Sistema,attaccante
Description	Un'attaccante ottiene accesso non autorizzato a causa di una debolezza nel meccanismo di autenticazione o tramite lo sfruttamento di un difetto nell'implementazione di uno schema di autenticazione.
Data Stimulus and preconditions	Controllo delle autorizzazioni Un meccanismo o sottosistema di autenticazione che implementa una forma di autenticazione come password, autenticazione digest, certificati di sicurezza, ecc., ma che presenta qualche difetto.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza il funzionamento del meccanismo di autenticazione per individuare eventuali debolezze o vulnerabilità. 2. L'attaccante raccoglie informazioni sulle credenziali degli utenti, i dettagli del processo di autenticazione o eventuali punti deboli noti del sistema. 3. L'attaccante sperimenta diverse tecniche di attacco per tentare di indovinare la password. 4. Se il meccanismo di autenticazione presenta difetti, l'attaccante cerca di eludere le protezioni per ottenere l'accesso non autorizzato.
Alternative Flow Exception Flow	- -
Response and Postconditions Non Functional Requirements Comments	Accesso a dati sensibili Authentication -

Figure 63: Accesso non autorizzato

Use case ID: UCA-44	
Use case Name: Divulgare dati	
Actors	Sistema, Attaccante
Description	In questo scenario, un attaccante, autenticato nel sistema, sfrutta la vulnerabilità degli Overread Buffers per ottenere accesso a dati sensibili a cui normalmente non dovrebbe avere accesso. Manipolando i dati in ingresso, l'attaccante provoca una lettura oltre i limiti consentiti di un buffer nel sistema di gestione delle autorizzazioni. Successivamente, utilizza i dati ottenuti per tentare di manipolare le autorizzazioni, cercando di ottenere accesso a risorse o funzionalità a cui non ha diritto.
Data	Controllo delle autorizzazioni
Stimulus and preconditions	<ul style="list-style-type: none"> 1. Il sistema di gestione delle autorizzazioni gestisce i dati sensibili e le autorizzazioni degli utenti. 2. L'attaccante ha un accesso legittimo come utente autenticato ma intende ottenere accesso non autorizzato a dati sensibili.
Basic Flow	<ul style="list-style-type: none"> 5. L'attaccante sfrutta la vulnerabilità degli Overread Buffers, manipolando i dati in ingresso per provocare un'operazione di lettura oltre i limiti consentiti di un buffer nel sistema di gestione delle autorizzazioni. 6. Durante questa operazione, l'attaccante riesce a recuperare dati sensibili che non dovrebbe poter leggere normalmente. 7. Utilizzando i dati ottenuti, l'attaccante cerca di manipolare le autorizzazioni nel sistema, cercando di ottenere accesso a risorse o funzionalità a cui non è autorizzato.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante ha ottenuto accesso a dati sensibili non autorizzati e ha tentato di manipolare le autorizzazioni nel sistema di gestione delle autorizzazioni.
Non Functional Requirements	Confidentiality
Comments	-

Figure 64: Divulgare dati

Accesso ai dati attività Per tale asset abbiamo individuato 5 possibili rischi:

1. *Falsificazione identità utente*: questo rischio si riferisce alla possibilità che un utente

falsifichi la propria identità per ottenere accesso non autorizzato al sistema. Le relative tabelle di Jacobson sono 65

2. *Falsificazione credenziali d'accesso:* Questo rischio si verifica quando le credenziali d'accesso di un utente vengono falsificate o manipolate per ottenere accesso non autorizzato al sistema. Le relative tabelle di Jacobson sono 66
3. *Alterazione dei dati immagazzinati:* questo rischio si riferisce alla possibilità che i dati immagazzinati nel sistema vengano alterati o manipolati in modo improprio o dannoso. Le relative tabelle di Jacobson sono 67
4. *Limitazione capacità di accesso ai dati:* questo rischio si riferisce alla possibilità che l'accesso ai dati venga limitato in modo improprio o non autorizzato, impedendo agli utenti autorizzati di accedere alle informazioni di cui hanno bisogno per svolgere le proprie attività. Le relative tabelle di Jacobson sono 68
5. *Rendere il sistema poco resistente agli attacchi:* questo rischio si riferisce alla possibilità che il sistema di accesso ai dati diventi vulnerabile agli attacchi informatici e alle intrusioni. Le relative tabelle di Jacobson sono 69

Use case ID: UCA-45	
Use case Name: Falsificazione identità utente	
Actors	Sistema, Attaccante
Description	Un attaccante ottiene accesso non autorizzato a causa di una debolezza nel meccanismo di autenticazione o tramite lo sfruttamento di un difetto nell'implementazione di uno schema di autenticazione
Data	Accesso ai dati attività
Stimulus and preconditions	Un meccanismo o sottosistema di autenticazione che implementa una forma di autenticazione come password, autenticazione digest, certificati di sicurezza, ecc., ma che presenta qualche difetto.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza il funzionamento del meccanismo di autenticazione per individuare eventuali debolezze o vulnerabilità. 2. L'attaccante raccoglie informazioni sulle credenziali degli utenti, i dettagli del processo di autenticazione o eventuali punti deboli noti del sistema. 3. L'attaccante sperimenta diverse tecniche di attacco per tentare di indovinare la password. 4. Se il meccanismo di autenticazione presenta difetti, l'attaccante cerca di eludere le protezioni per ottenere l'accesso non autorizzato.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso a dati sensibili
Non Functional Requirements	Authentication
Comments	-

Figure 65: Falsificazione identità utente

Use case ID: UCA-46	
Use case Name: Falsificazione credenziali d'accesso	
Actors	Sistema, Attaccante
Description	Un attaccante crea delle credenziali di sessione false ma funzionali al fine di ottenere o usurpare l'accesso ad un servizio.
Data	Accesso ai dati attività
Stimulus and preconditions	L'applicazione presa di mira deve utilizzare credenziali di sessione per identificare gli utenti legittimi. Gli identificatori di sessione rimangono invariati quando cambiano i livelli di privilegio. Identificatori di sessione prevedibili.
Basic Flow	<ol style="list-style-type: none"> 1. Analizzare e Comprendere gli ID di Sessione: L'attaccante scopre che l'applicazione presa di mira utilizza credenziali di sessione per identificare gli utenti legittimi. 2. Un aggressore effettua molte connessioni anonime e registra gli ID di sessione. 3. L'attaccante manipola il messaggio di richiesta HTTP e aggiunge i propri ID di sessione contraffatti nelle richieste o nei cookie. 4. L'attaccante carica l'ID di sessione predefinito o previsto nel proprio browser e accede a dati o funzionalità protetti.
Alternative Flow	2. Un aggressore effettua connessioni autorizzate e registra i token o le credenziali di sessione.
Exception Flow	-
Response and Postconditions	L'attaccante può accedere a dati o funzionalità protette o sfruttare privilegi di un determinato account
Non Functional Requirements	Accountability
Comments	-

Figure 66: Falsificazione credenziali d'accesso

Use case ID: UCA-47	
Use case Name: Alterazione dei dati immagazzinati	
Actors	Sistema, Attaccante
Description	Un attaccante riesce a manipolare impropriamente i dati sulle attività degli attori
Data	Accesso ai dati attività
Stimulus and preconditions	<ol style="list-style-type: none"> 1. Il target sta registrando l'azione e i dati dell'utente. 2. Il target protegge in modo insufficiente l'accesso ai registri o ai meccanismi di registrazione.
Basic Flow	<ol style="list-style-type: none"> 1. Determinare le azioni e i dati che verranno registrati 2. Determinare il formato di registrazione del sistema 3. L'attaccante altera i dati registrati dal sistema nel log
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ul style="list-style-type: none"> • Controllare attentamente l'accesso ai file di log fisici. • Non permettere che dati contaminati siano scritti nel file di log senza una valida convalida dell'input. Potrebbe essere utilizzata una "allowlist" per convalidare correttamente i dati. • Utilizzare la sincronizzazione per controllare il flusso di esecuzione. • Utilizzare strumenti di analisi statica per identificare vulnerabilità di falsificazione dei log. • Evitare di visualizzare i log con strumenti che potrebbero interpretare caratteri di controllo nel file, come le shell della riga di comando.
Non Functional Requirements	Integrity
Comments	-

Figure 67: Alterazione dei dati immagazzinati

Use case ID: UCA-48	
Use case Name: Limitazione capacità di accesso ai dati	
Actors	Sistema, Attaccante
Description	<p>Un utente malintenzionato modifica il contenuto o gli attributi dei file (come le estensioni o i nomi) in modo da causare un'elaborazione errata da parte di un'applicazione. Gli aggressori utilizzano questa classe di attacchi per far entrare le applicazioni in stati instabili, sovrascrivere o esporre informazioni sensibili e persino eseguire codice arbitrario con i privilegi dell'applicazione. Questa classe di attacchi si differenzia dagli attacchi alle informazioni di configurazione (anche se</p> <p>basati su file) in quanto la manipolazione del file causa l'elaborazione di comportamenti non standard, come buffer overflow o l'uso di un interprete non corretto. Gli attacchi alla configurazione si basano sull'interpretazione corretta del file da parte dell'applicazione per inserire informazioni di configurazione dannose. Allo stesso modo, gli attacchi alla localizzazione delle risorse si basano sul controllo della capacità dell'applicazione di individuare i file, mentre gli attacchi alla manipolazione dei file non richiedono che l'applicazione cerchi in una posizione non predefinita, sebbene le due classi di attacchi siano spesso combinate.</p>
Data	Accesso ai dati attività
Stimulus and preconditions	La destinazione deve utilizzare il file interessato senza verificarmene l'integrità.
Basic Flow	<ol style="list-style-type: none"> Identificazione dell'applicazione che manipola file. Comprensione del modo in cui l'applicazione legge, scrive o elabora i file. Modifica dei nomi e degli attributi per ingannare l'applicazione. Inserimento di file manipolati nell'ambiente dell'applicazione. Sfruttamento delle manipolazioni per causare comportamenti indesiderati nell'applicazione. Utilizzo delle debolezze create per ottenere accesso non autorizzato o eseguire codice dannoso. Tentativo di nascondere le manipolazioni effettuate. Mantenimento di un accesso persistente all'ambiente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Limitato accesso alle procedure di autenticazione
Non Functional Requirements	Accountability
Comments	-

Figure 68: Limitazione capacità di accesso ai dati

Use case ID: UCA-49	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Accesso ai dati attività
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata 2. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target. 3. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target. 4. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema. 5. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target. 6. L'attaccante può cercare di mantenere l'attacco nel tempo, adattanAvailabilityi alle eventuali contromisure adottate dalla difesa del target
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 69: Rendere il sistema poco resistente agli attacchi

Gli *Attack Tree* sono uno strumento di analisi fondamentale nel campo della sicurezza

informatica, utilizzati per valutare e comprendere le vulnerabilità di un sistema. Essenzialmente, sono una rappresentazione grafica delle possibili sequenze di eventi che un attaccante potrebbe sfruttare per compromettere la sicurezza di un sistema.

Questi alberi di attacco sono costituiti da diversi elementi chiave. Innanzitutto, c'è la radice, che è il nodo principale e rappresenta l'obiettivo finale dell'attacco. I nodi figli rappresentano le fasi o le azioni necessarie per raggiungere questo obiettivo. Ogni nodo figlio può a sua volta avere ulteriori nodi figli, che dettagliano azioni o passaggi specifici richiesti per portare avanti l'attacco.

L'analisi degli attack tree consente di identificare le falte nella sicurezza nel sistema valutando i possibili scenari di attacco. Questo approccio consente di comprendere meglio le potenziali minacce e di sviluppare strategie di difesa più efficaci.

Dati attività attori: Nella figura 70 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Dati Attività Attori. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern di attacco scelto tra i vari pattern CAPEC.

1. *Dictionary-based Password Attack*: l'attaccante riesce ad accedere al sistema con delle credenziali di accesso di un amministratore. Egli si serve di un algoritmo che genera password a partire da un dizionario e le prova in maniera iterativa fino ad indovinare quella corretta.
2. *Log Injection-Tampering-Forging*: l'attaccante riesce a manipolare impropriamente i dati sulle attività degli attori. Si cerca di identificare le informazioni che vengono registrate dai file di log, compresi gli eventi critici e le azioni eseguite dagli utenti. Successivamente, esaminano il formato di registrazione utilizzato dal sistema per memorizzare queste informazioni. Una volta compreso il formato, gli attaccanti procedono con l'iniezione, la manipolazione o la falsificazione dei dati nei file di log al fine di nascondere le loro attività dannose o creare una falsa percezione degli eventi.
3. *Functionality Misuse*: l'attaccante individua una funzionalità difettosa e ne fa uso per accedere a dati sensibili.
4. *File Manipulation*: l'attaccante individua file o dati che contengono informazioni sulle attività degli attori e sfrutta le vulnerabilità del sistema per accedervi. Con l'accesso a questi dati egli riesce a ad alterare i dati sensibili per poi cercare di mascherare le proprie azioni.

5. *Exploitation of Trusted Identifiers*: l'attaccante modifica il contenuto o gli attributi dei file. A seguito dell'individuazione degli identificatori di fiducia questi vengono sfruttati per accedere ad informazioni sensibili. Tali informazioni vengono manipolate per ottenere accesso non autorizzato o per eseguire operazioni dannose.

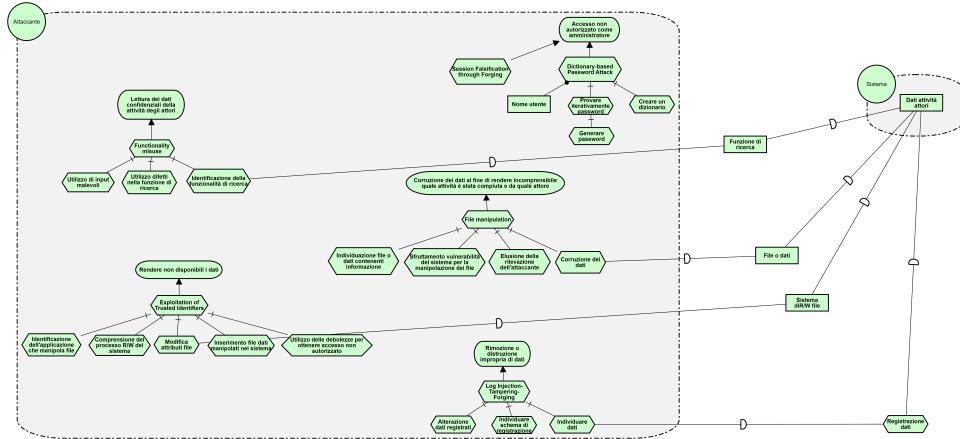


Figure 70: Attack tree: dati attività attori.

Policy di Autorizzazione: Nella figura 71 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Policy di Autorizzazione. Abbiamo già individuato e indicato le 2 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern di attacco scelto tra i vari pattern CAPEC.

1. *Removal of filters: Input filters, output filters, data masking*: l'attaccante individua i filtri di input e mira ad eluderli o disabilitarli con tecniche apposite. Una volta fatto ciò egli può inviare dati dannosi all'applicazione che, una volta elaborati, apre la possibilità allo sfruttamento di vulnerabilità.
2. *Forced Deadlock*: l'attaccante mira a generare una negazione del servizio, ovvero uno stato di deadlock, questo può essere fatto tramite l'innescamento di un azione che, ad effetto domino, ne innesca altre portando il sistema ad uno stato di "attesa" perenne.

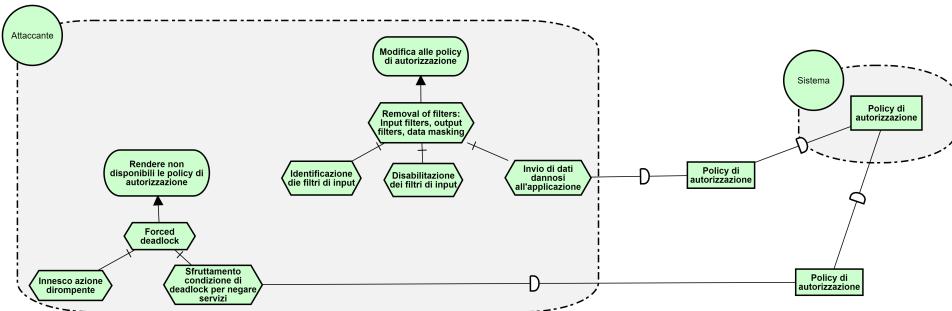


Figure 71: Attack tree: policy autorizzazione.

Policy di Autenticazione: Nella figura 72 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Policy di Autenticazione. Abbiamo già individuato e indicato le 2 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern CAPEC.

1. *Authentication Bypass*: l'attaccante sfrutta controlli di protezione inadeguati per eseguire azioni non autorizzate. Una volta "sondati" i punti deboli è possibile sviluppare delle metodologie per aggirare il controllo di accesso.
2. *Exploiting Incorrectly Configured Access Control Security Levels*: l'attaccante individua le regole di autenticazione del sistema per poi sfruttare vulnerabilità insite nelle politiche di autenticazione, questo porta a modifiche di parametri o regole per eludere i controlli di autenticazione. In questo modo l'attaccante ottiene accesso al sistema.

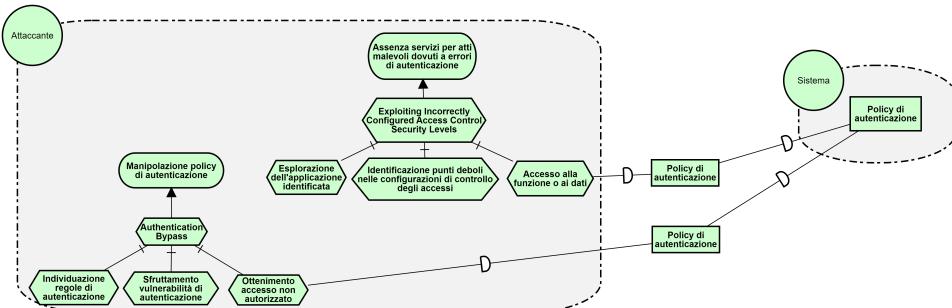


Figure 72: Attack tree: policy autenticazione.

Traccia degli Accessi: Nella figura 73 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Traccia degli Accessi. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern di attacco scelto tra i vari pattern CAPEC.

1. *Forced Deadlock*: l'attaccante mira a generare una negazione del servizio, ovvero uno stato di deadlock, questo può essere fatto tramite l'innescamento di un azione che, ad effetto domino, ne innesca altre portando il sistema ad uno stato di "attesa" perenne.
2. *Collect Data from Clipboard*: l'attaccante deve trovare, innanzitutto, un'applicazione che consenta di copiare e incollare informazioni sensibili. In seguito, egli prende di mira gli utenti per ottenere loro informazioni su basi periodiche e utilizzare le loro informazioni sensibili in un secondo momento.
3. *Subverting Environment Variable Values*: l'attaccante sfrutta uno strumento automatizzato per ispezionare l'applicazione e registrarne i punti di accesso e vulnerabilità. In seguito si usano i punti individuati come destinazioni per indiettare payload per determinarne esattamente il tipo di vulnerabilità e come sfruttarla.
4. *File Manipulation*: l'attaccante individua file o dati che contengono informazioni sulle attività degli attori e sfrutta le vulnerabilità del sistema per accedervi. Con l'accesso a questi dati egli riesce a ad alterare i dati sensibili per poi cercare di mascherare le proprie azioni.
5. *Authentication Abuse*: una volta analizzate debolezze nei meccanismi di autenticazione, l'attaccante raccoglie informazioni sulle credenziali e sperimenta tecniche per tentare di infovinare la password.

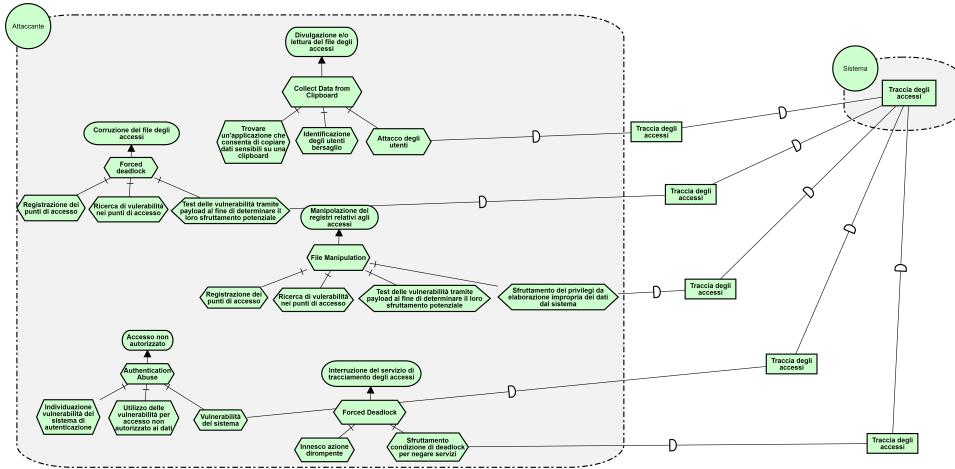


Figure 73: Attack tree: traccia degli accessi.

Dati relativi ad un paziente: Nella figura 74 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Dati relativi ad un paziente. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern di attacco scelto tra i vari pattern CAPEC.

1. *Authentication Abuse*: una volta analizzate debolezze nei meccanismi di autenticazione, l'attaccante raccoglie informazioni sulle credenziali e sperimenta tecniche per tentare di indovinare la password.
2. *Functionality Misuse*: L'avversario interagisce con l'applicazione, dopo averne identificato le debolezze, ed opera strategie di esfiliazione per trasferire i dati cercati.
3. *Input Data Manipulation*: l'attaccante ricerca vulnerabilità per riuscire a manipolare i dati di input o delle prescrizioni mediche.
4. *File Manipulation*: l'attaccante individua file o dati che contengono informazioni sulle attività degli attori e sfrutta le vulnerabilità del sistema per accedervi. Con l'accesso a questi dati egli riesce a ad alterare i dati sensibili per poi cercare di mascherare le proprie azioni.
5. *Forced Deadlock*: l'attaccante mira a generare una negazione del servizio, ovvero uno stato di deadlock, questo può essere fatto tramite l'innesto di un'azione che, ad effetto domino, ne innesca altre portando il sistema ad uno stato di "attesa" perenne.

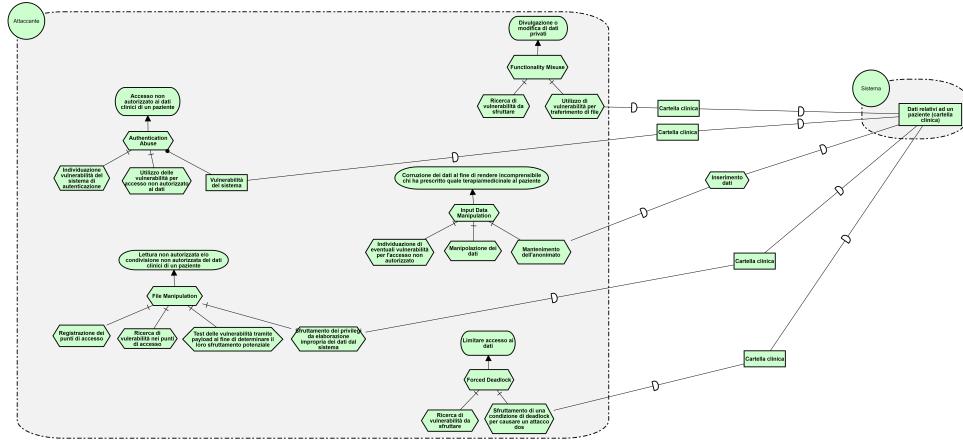


Figure 74: Attack tree: dati relativi ad un paziente.

Dispositivi IoT per monitoraggio salute: Nella figura 75 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Dispositivi IoT per monitoraggio salute. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati un pattern di attacco scelto tra i vari pattern CAPEC.

1. *Identity Spoofing*: una volta identificate delle vulnerabilità del dispositivo o nel suo sistema di monitoraggio, l'avversario le sfrutta per modificare l'ID del dispositivo stesso con uno arbitrario.
2. *Input Data Manipulation*: l'attaccante ricerca vulnerabilità per riuscire a manipolare i dati, aggiungendone di dannosi. I dati manipolati possono essere sfruttati per generare un comportamento indesiderato nell'applicazione.
3. *Malicious Hardware Component Replacement*: una volta identificato il dispositivo o il componente hardware target, l'avversario traccia la catena di fornitura per ottenere l'accesso fisico al sistema. Utilizzando questa vulnerabilità creata, l'avversario può sostituire il componente mirato con uno dannoso e ottenere accesso ai sistemi.
4. *Development Alteration*: l'attaccante, mediante lo sfruttamento di una vulnerabilità, inserisce codice dannoso nella repository del progetto o durante la compilazione del codice. Il tutto compromette l'affidabilità del sistema di monitoraggio.
5. *Flooding*: esistono diverse tipologie di flooding e l'avversario, studiando il sistema, deve identificare quale sia la più appropriata. Tramite strumenti automatizzati o script si genera un alto volume di richieste o traffico rete da inviare al target,

sovraffollando le risorse con una conseguente diminuzione di prestazioni. La saturazione può portare ad un'interruzione del servizio e alla possibilità per l'attaccante di accedere alle risorse target.

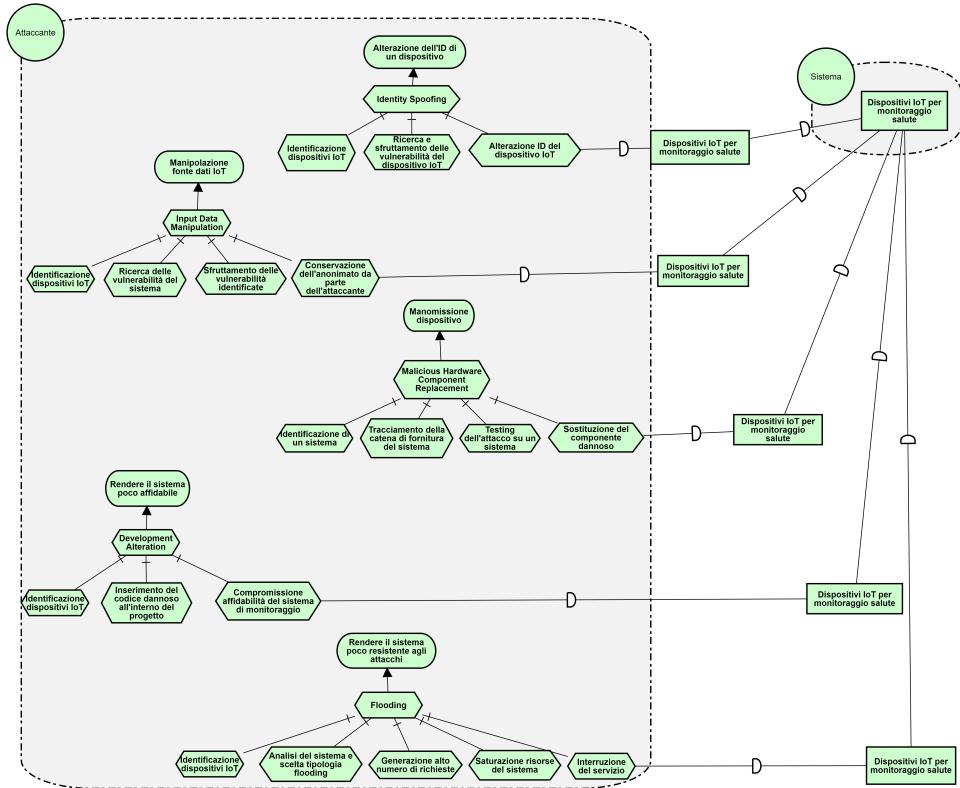


Figure 75: Attack tree: dispositivi IoT per monitoraggio salute.

Monitorare accesso ai dati: Nella figura 76 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Monitorare accesso ai dati. Abbiamo già individuato e indicato le 4 vulnerabilità riscontrate, a ciascuna di esse sono stati associati uno o più pattern di attacco scelti tra i vari pattern CAPEC.

1. *Functionality Bypass*: l'attaccante sfrutta controlli di protezione inadeguati per eseguire azioni non autorizzate. Una volta "sondati" i punti deboli è possibile sviluppare delle metodologie per aggirare il controllo di accesso.
2. *Exploitation of Trusted Identifiers*: l'attaccante modifica il contenuto o gli attributi dei file. A seguito dell'individuazione degli identificatori di fiducia questi vengono

sfruttati per accedere ad informazioni sensibili. Tali informazioni vengono manipolate per ottenere accesso non autorizzato o per eseguire operazioni dannose.

3. *Removing Important Client Functionality*: l'avversario sfrutta delle vulnerabilità per poter inserire nella repository, o durante la compilazione, del codice dannoso. Tale codice è, in genere, progettato per compromettere l'affidabilità del sistema di monitoraggio della salute o il malfunzionamento del dispositivo.
4. *Flooding*: esistono diverse tipologie di flooding e l'avversario, studiando il sistema, deve identificare quale sia la più appropriata. Tramite strumenti automatizzati o script si genera un alto volume di richieste o traffico rete da inviare al target, sovraccaricandone le risorse con una conseguente diminuzione di prestazioni. La saturazione può portare ad un'interruzione del servizio e alla possibilità per l'attaccante di accedere alle risorse target.

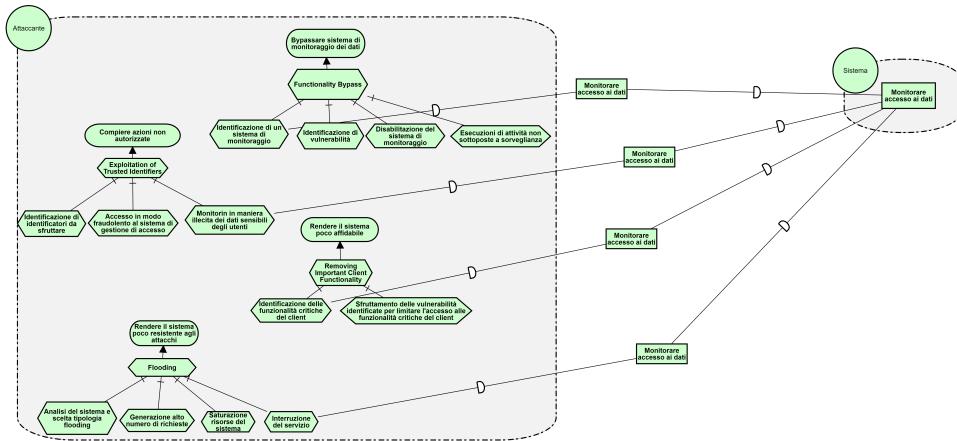


Figure 76: Attack tree: Monitorare accesso ai dati.

Gestire processo di lettura, scrittura e salvataggio dei dati: Nella figura 77 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset Gestire processo di lettura, scrittura e salvataggio dei dati. Abbiamo già individuato e indicato le 6 vulnerabilità riscontrate, a ciascuna di esse sono stati associati uno o più pattern di attacco scelti tra i vari pattern CAPEC.

1. *Adversary-In-The-Middle*: l'attaccante, si frappone tra il mittente ed il destinatario, riuscendo ad intercettare dati sensibili durante le operazioni di lettura, scrittura e salvataggio dei dati.

2. *Transaction or Event Tampering via Application API Manipulation* : l'attaccante, attraverso la manipolazione dei diversi e differenti dati di processo provenienti da chiamate API, riesce a modificare e salvare il contenuto di questi ultimi.
3. *Removing Important Client Functionality*: l'attaccante sfrutta delle vulnerabilità identificate al fine di rendere il sistema poco affidabile.
4. *Flooding*: attraverso l'interruzione di opportuni servizi all'interno dell'applicazione, l'attaccante, ha come obiettivo quello di diminuire la resistenza del sistema verso attacchi esterni.
5. *Exploitation of Trusted identifiers*: l'attaccante monitora in maniera illecita i dati sensibili degli utenti, compiendo dunque azioni non autorizzate, che impattano sulla privacy di tutti gli utenti del sistema: pazienti, medici ed operatori sanitari.

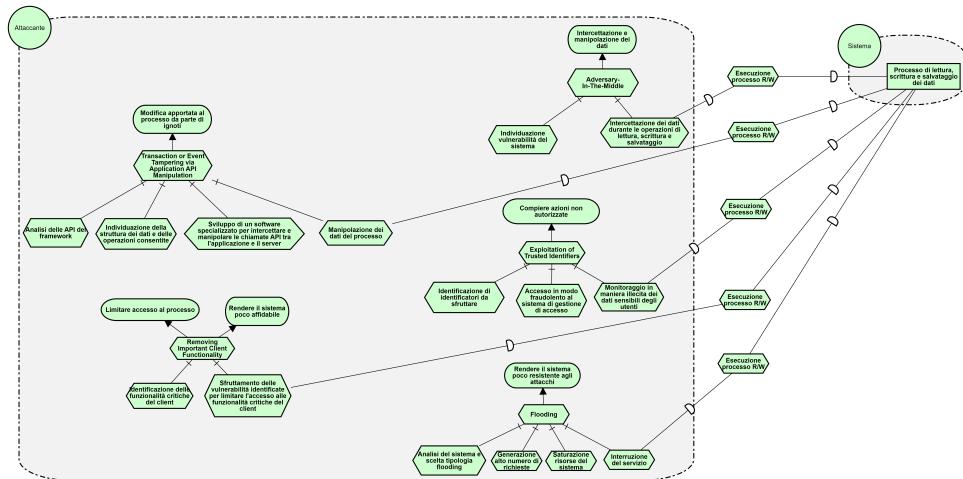


Figure 77: Attack tree: gestire processo di lettura, scrittura e salvataggio dei dati.

Procedure di autenticazione: L'attack tree in figura 78 è molto importante, in quanto modella delle tipologie di attacco su un asset essenziale per il corretto funzionamento del sistema. Più dettagliatamente, possiamo osservare modellate le seguenti tipologie di attacco:

1. *Exploitation of Trusted Identifiers*: l'attaccante, studiando il sistema ed osservando diversi identificatori, si pone come obiettivo quello di risucire a falsificare l'identità utente.

2. *Development Alteration*: l'attaccante cerca di introdurre del codice malevolo all'interno del sistema, al fine di compromettere l'affidabilità di tutte le procedure di autenticazione.
3. *Eavesdropping*: l'attaccante, dopo aver individuato tutte le comunicazioni all'interno del sistema che contengono informazioni sensibili sulle procedure di autenticazione, le rende pubbliche.
4. *Inducing Account lockout*: l'attaccante blocca le procedure di autenticazione per determinati account, impedendo loro di accedere correttamente al sistema.
5. *Flooding*: l'attaccante analizza la porzione del sistema che gestisce l'asset in questione, cercando di ridurre la sua resistenza agli attacchi esterni, al fine di prepararsi per eseguire ulteriori attacchi in seguito.

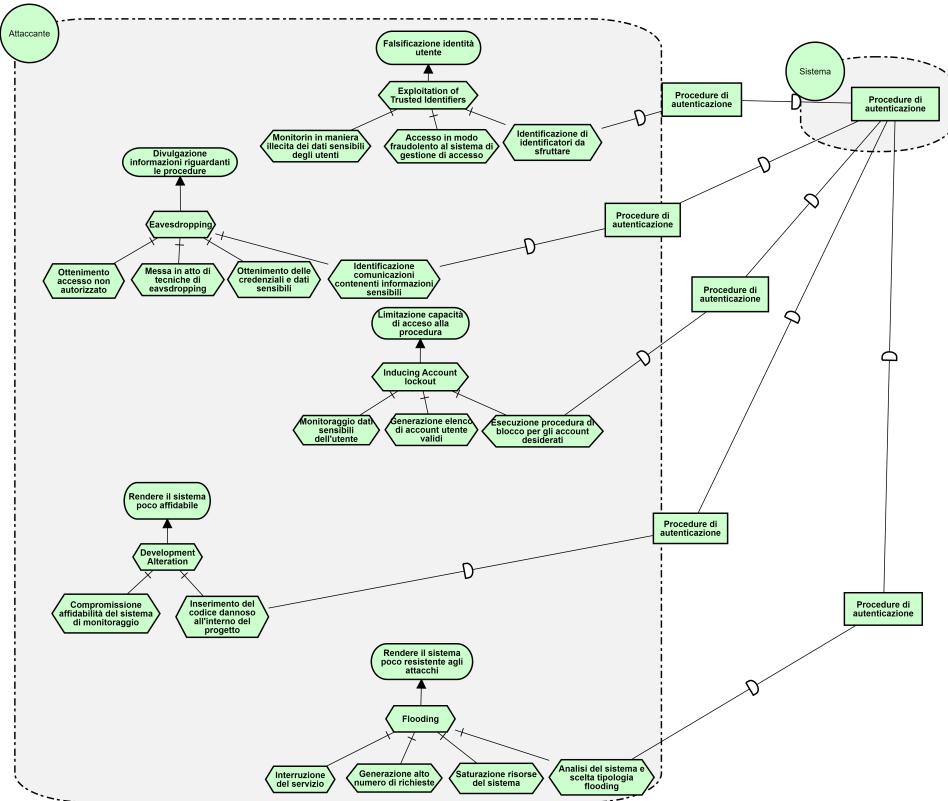


Figure 78: Attack tree: procedure di autenticazione.

Controllo delle autorizzazioni: Nella figura 79 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset inerente al controllo delle autorizzazioni. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati uno o più pattern di attacco scelti tra i vari pattern CAPEC. Di seguito sarà descritto l'attack tree:

1. *Privilege escalation*: l'attaccante sfrutta le vulnerabilità già presenti nel sistema al fine di aumentare i propri privilegi e compiere attività illecite.
2. *Authentication abuse*: l'attaccante sfrutta vulnerabilità precedentemente individuate per eseguire un accesso non autorizzato, con l'obiettivo finale di assumere il controllo dell'asset relativo alle autorizzazioni e manipolarlo secondo le proprie intenzioni.
3. *Removing important Client Functionality*: l'attaccante sfrutta vulnerabilità al fine di limitare l'accesso alle funzionalità critiche del client, compromettendo così l'affidabilità dell'implementazione di tale asset.
4. *Overread Buffers*: in questa modalità d'attacco, si osserva come l'attaccante compia azioni mirate a ottenere il controllo dell'asset riguardante le autorizzazioni, per poi divulgare pubblicamente.
5. *Flooding*: attraverso la compromissione dei servizi pertinenti, l'attaccante mira a indebolire la resistenza del sistema di controllo delle autorizzazioni agli attacchi.

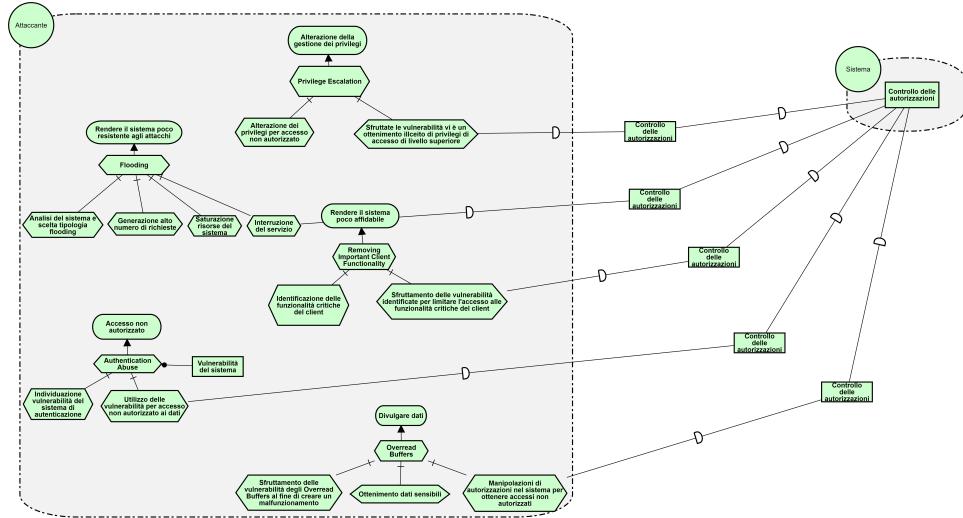


Figure 79: Attack tree: controllo delle autorizzazioni.

Accesso ai dati attività: Nella figura 80 osserviamo l'attack tree destinato allo studio delle vulnerabilità dell'asset inerente all'accesso ai dati attività. Abbiamo già individuato e indicato le 5 vulnerabilità riscontrate, a ciascuna di esse sono stati associati uno o più pattern di attacco scelti tra i vari pattern CAPEC. Di seguito sarà descritto l'attack tree:

1. *Inducing Account Lockout*: dopo che l'attaccante ha bloccato gli account che ha preso di mira, si limita di conseguenza l'accesso ai dati e alle attività correlate.
2. *Log-Injection-Tampering-Forging*: in questa strategia di attacco, l'utente malintenzionato ha come obiettivo primario l'alterazione dei dati delle attività, con l'intento di compromettere l'integrità del sistema o di ottenere benefici illeciti.
3. *Exploitation of trusted identifiers*: l'attaccante si appropria di identità verificate al fine di ottenere accesso ai dati delle attività associate a esse.
4. *Session credential falsification through forging*: in questa tipologia di attacco, l'utente malevolo manipola o falsifica le credenziali d'accesso al fine di accedere ai dati delle attività in modo non autorizzato.
5. *Flooding*: l'attaccante esegue una serie di interruzioni mirate ai servizi pertinenti, con l'obiettivo finale di rendere il sistema di logging dei dati delle attività meno resistente agli attacchi.

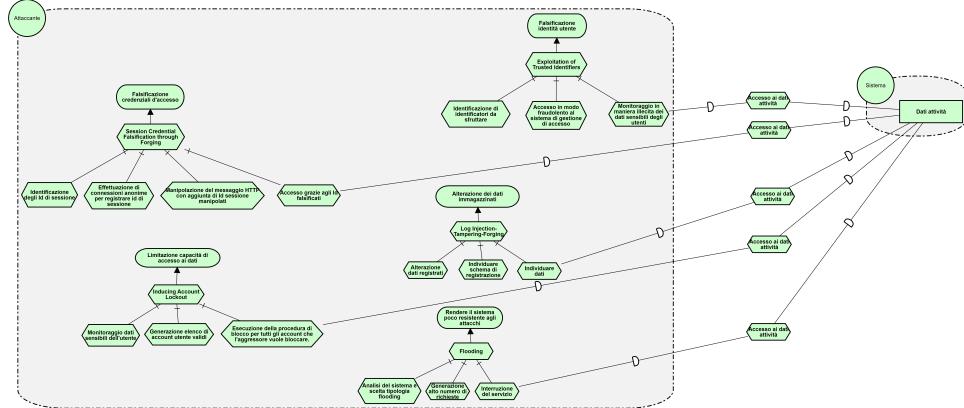


Figure 80: Attack tree: accesso ai dati attività.

2.7 Risk Reduction

La fase di riduzione del rischio è un tassello fondamentale nella gestione complessiva del rischio, durante la quale vengono sviluppate e implementate strategie mirate a ridurre

i rischi identificati a un livello accettabile. Questo processo implica l'adozione di varie misure preventive e correttive per proteggere il sistema da potenziali minacce. Questa viene portata avanti tramite l'implementazione di diverse sottofasi che verranno esplorate in dettaglio nei seguenti sottoparagrafi.

2.7.1 Control Identification

La *Control Identification* (identificazione dei controlli) è il processo di individuazione e catalogazione dei controlli di sicurezza necessari per proteggere un sistema informativo, un'organizzazione o un processo dalle potenziali minacce o vulnerabilità. Questo processo coinvolge un'analisi approfondita dei rischi e delle vulnerabilità esistenti, insieme all'identificazione delle misure di sicurezza appropriate per mitigare tali rischi. Nell'analisi delle soluzioni ammissibili è fondamentale considerare che la mera enumerazione delle singole soluzioni non è più sufficiente, ma è necessario ipotizzarne anche l'utilizzo combinato per affrontare le criticità in modo efficace.

2.7.2 Feasibility Assessment

La *Feasibility Assessment* (valutazione di fattibilità) è un'attività che mira a determinare se le misure di sicurezza individuate durante l'identificazione dei controlli sono praticabili e attuabili nell'ambito specifico del progetto. L'obiettivo è garantire che i controlli proposti siano realistici e adattabili alle esigenze e alle capacità del sistema. Questa valutazione si basa su uno studio dettagliato di ogni soluzione definita, considerando la loro fattibilità tecnica, operativa ed economica, nonché la loro compatibilità con l'ambiente esistente e le risorse disponibili.

Accanto alla valutazione della fattibilità, è stato condotto uno studio incentrato sulla riduzione del rischio per ciascun asset, valutando diverse opzioni. Non sempre la soluzione che riduce maggiormente il rischio è quella più conveniente a livello economico. Per questo motivo, si è deciso di utilizzare un'ulteriore misura, il *Return of Control*, che può essere espresso con la seguente formula:

$$\text{Return of Control} = \frac{\text{Riduzione del Rischio}}{\text{Costo Implementazione}} - 1$$

2.7.3 Security Requirements Definition

La *Security Requirements Definition* (Definizione dei Requisiti di Sicurezza) è il processo di identificazione e documentazione dei requisiti specifici di sicurezza che devono essere soddisfatti per garantire la protezione delle risorse informative. Questi requisiti derivano

dai risultati ottenuti nella fase di studio della fattibilità e consentono il completamento della tabella STRIDE, compilando le seguenti colonne:

- Capc Pattern: si riferisce ai modelli di attacco descritti nel Common Attack Pattern Enumeration and Classification (CAPEC), che forniscono un elenco strutturato di schemi di attacco comuni.
- Inherent Probability: indica la probabilità intrinseca che un determinato rischio si verifichi senza l'applicazione di contromisure o controlli di sicurezza.
- Inherent Risk: rappresenta il livello di rischio associato a un asset o a un processo prima dell'implementazione di controlli di sicurezza. È determinato dalla combinazione della probabilità intrinseca di un evento dannoso e dell'impatto potenziale di tale evento.
- Control: si riferisce alle misure di sicurezza o ai controlli implementati per mitigare o gestire i rischi identificati. Questi possono essere politiche, procedure, tecnologie o altre azioni volte a proteggere le risorse informative.
- Cost: indica il costo associato all'implementazione e alla gestione del controllo di sicurezza.
- Feasability: valuta la praticabilità e l'attuabilità del controllo proposto nell'ambito specifico dell'organizzazione o del progetto.
- Residual Probability: si riferisce alla probabilità che un rischio si verifichi dopo l'implementazione dei controlli di sicurezza.
- Residual Impact: rappresenta l'entità dell'impatto che un evento dannoso avrebbe sul sistema o sull'organizzazione dopo l'attuazione dei controlli di sicurezza.
- Residual Risk: è il livello di rischio residuo associato a un asset o a un processo dopo l'implementazione dei controlli di sicurezza.
- R.o.C (Return of Control): questo valore esprime la misura in cui l'implementazione dei controlli di sicurezza riduce il rischio originario. Può essere espresso come una percentuale o con un'altra metrica definita.
- Overall Cost: indica il costo totale associato all'implementazione e alla gestione dei controlli di sicurezza, compresi eventuali costi aggiuntivi dovuti all'impatto residuo dei rischi.

La tabella STRIDE con l'indicazione di questi valori associati a ciascun asset può essere visionata in 81, 82, 83, 84 e 85

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Dati attività attori	Session Falsification through Forging (CAPEC-196)	15%	7.500-15.000	Implementazione di sistemi di monitoraggio degli accessi e delle attività degli utenti, con particolare attenzione agli account con privilegi elevati	1450	I sistemi di monitoraggio degli accessi permettono di rilevare tempestivamente attività sospette e di analizzare i pattern comportamentali degli utenti, identificando anomalie	2%	1.000-2.000	200-300	5,72	1650 - 1750
	Dictionary-based Password Attack (CAPEC-16)	10%	5.000-10.000	Implementazione di politiche di complessità delle password, blocco degli account dopo un certo numero di tentativi falliti, e limitazione dei tentativi di accesso.	400	Questi sistemi consentono una gestione più sicura e puntuale delle password, evitando dispersioni di queste e, allo stesso tempo, evitando possibili attacchi riguardanti frodi di informazioni personali.	3%	1.500-3.0000	500-600	15,13	900 - 1000
	Log Injection-Tampering-Forging(CAPEC-93)	14%	7.000-14.000	Realizzazione di sistemi di backup e archiviazione di dati sicuri	1000	Questo strumento permette di recuperare i dati in caso di perdita accidentale o di attacco malevolo attraverso la garanzia di backup regolari e disciolti in più luoghi.	2%	1.000-2.000	200-300	8,75	1200 - 1300
	Functionality Misuse (CAPEC-212)	17%	8.500-17.000	Sistema di sicurezza basato su autorizzazioni e crittografia avanzata	1200	Questo sistema garantisce l'accesso esclusivamente agli utenti autorizzati per le informazioni sensibili specifiche. Inoltre, assicura una protezione robusta durante la trasmissione e l'archiviazione dei dati.	3%	1.500-3.000	300-400	9,54	1500 - 1600
	File Manipulation(CAPEC-165)	12%	6.000-12.000	Firme digitali per autenticare i dati, consentendo di verificare l'integrità del contenuto.	1500	La firma digitale è un metodo avanzato di autenticazione che impiega algoritmi crittografici per creare una firma unica associata a un documento o a un insieme di dati. Questa firma digitale viene generata utilizzando una chiave privata, e la sua autenticità può essere verificata da chiunque possieda la chiave pubblica corrispondente.	2%	1.000-2.000	200-300	4,17	1700 - 1800
	Inducing Account Lockout (CAPEC-2)	10%	4.000 - 8.000	Blocco temporaneo dell'account	2000	Implementare un meccanismo di blocco temporaneo dell'account dopo un numero specifico di tentativi di accesso falliti.	3%	1.000 - 3.000	200 - 400	1,85	2.200 - 2.400
	Exploitation of Trusted Identifiers (CAPEC-21)	9%	4.500 - 9.000	Criptografia degli identificatori importanti	1000	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	2%	1.000 - 2.000	100 - 200	5,6	1.100 - 1.200

Figure 81: Schema STRIDE completo 1, 2 e 3 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Policy di autorizzazione	Removal of filters: Input filters, output filters, data masking(CAPEC-200)	30%	3.000 - 6.000	Gestione delle autorizzazioni	3000	Limitare l'accesso ai filtri di input, filtri di output e tecniche di mascheramento dei dati solo al personale autorizzato. Applica rigorosi controlli di accesso per garantire che solo utenti autorizzati possano modificarli o rimuoverli.	10%	1.000 - 2.000	800 - 900	0,21	3.800 - 3.900
	Forced Deadlock(CAPEC-25)										
Policy di autenticazione	Exploiting Incorrectly Configured Access Control Security Levels (capec-180)	30%	3.000 - 6.000	Principio del minimo privilegio	2000	Applicare il principio del "minimo privilegio", assegnando agli utenti solo le autorizzazioni necessarie per svolgere le loro funzioni specifiche. Evita di concedere privilegi eccessivi.	10%	1.000 - 2.000	700 - 900	0,85	2.700 - 2.900
	Exploitation of Improperly Configured or Implemented Memory Protections (CAPEC-679)										
	Authentication Bypass(115)	40%	4.000 - 8.000	Autenticazione multifattore (MFA)	4000	Configurare e implementare protezioni di memoria robuste, come l'esecuzione di dati (DEP/NX), che impediscono l'esecuzione di codice da regioni di memoria designate come dati.	8%	800 - 1.600	200 - 300	0,65	2.200 - 2.300

Figure 82: Schema STRIDE completo 1, 2 e 3 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Traccia degli accessi	Forced Deadlock(CAPEC-25)	28%	2.800 - 5.600	Gestione transazioni	1000	Utilizzare meccanismi di gestione transazionale per garantire che le transazioni siano completate o annullate in modo coerente, evitando situazioni di deadlock prolungate.	8%	800 - 1.600	300 - 400	2,85	1.300 - 1.400
	Collect Data from Registries (capec-647)	20%	2.000 - 4.000	Limitare l'accesso ai registri	900	Ridurre al minimo i privilegi di accesso ai registri solo agli utenti e ai processi autorizzati. Utilizza le impostazioni di controllo degli accessi per garantire che solo persone autorizzate possano accedere ai registri.	7%	700 - 1.400	200 - 300	2,06	1.100 - 1.200
	Collect Data from Clipboard (CAPEC-637)	18%	1.800 - 3.600	Monitoraggio del Clipboard	500	Implementare meccanismi di monitoraggio del clipboard per rilevare attività sospette o accessi non autorizzati. Includi avvisi o registrazioni degli eventi quando vengono rilevati accessi impropri.	5%	500 - 1.000	300 - 400	5,1	2.100-4.000
	Leverage Alternate Encoding (CAPEC-267)	30%	3.000 - 6.000	Filtraggio degli input	1500	Implementare filtri di input che rilevano e bloccano caratteri o sequenze di caratteri non validi o sospetti nelle richieste dei clienti o nelle viste degli utenti.	8%	800 - 1.600	500 - 600	2,8	3.500-6.600
	Subverting Environment Variable Values (CAPEC-13)	25%	2.500 - 5.000	Validazione e sanitizzazione degli input	1000	Applicare una rigorosa validazione e sanitizzazione degli input per impedire l'inserimento di valori dannosi o manipolati nelle variabili di ambiente.	10%	1.000 - 2.000	700 - 800	3	3.200-5.800
	File Manipulation(CAPEC-165)	20%	2.000 - 4.000	Firme digitali per autenticare i dati, consentendo di verificare l'integrità del contenuto.	800	La firma digitale è un metodo avanzato di autenticazione che impiega algoritmi critografici per creare una firma unica associata a un documento o a un insieme di dati. Questa firma digitale viene generata utilizzando una chiave privata, e la sua autenticità può essere verificata da chiunque possieda la chiave pubblica corrispondente.	4%	400 - 800	100 - 200	2,93	2.100-4.200
	Authentication Abuse(CAPEC-114)	35%	3.500 - 7.000	Autenticazione multifattore (MFA)	1800	Implementare l'autenticazione multifattore per aggiungere uno strato aggiuntivo di sicurezza, richiedendo almeno due metodi di verifica dell'identità.	10%	1.000 - 2.000	300 - 400	2,12	2.100 - 2.200

Figure 83: Schema STRIDE completo 4 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Dati relativi ad un paziente (cartella clinica)	Authentication Abuse (CAPEC-114)	20%	30.000 - 40.000	Autenticazione multifattore (MFA)	20000	Implementare l'autenticazione multifattore per aggiungere uno strato aggiuntivo di sicurezza, richiedendo almeno due metodi di verifica dell'identità.	5%	7.500 - 10.000	2.000 - 4.000	0,9	22.000-44.000
	Functionality Misuse (CAPEC-212)	18%	27.000 - 36.000	Sistema di sicurezza basato su autorizzazioni e crittografia avanzata	12000	Questo sistema garantisce l'accesso esclusivamente agli utenti autorizzati per le informazioni sensibili specifiche. Inoltre, assicura una protezione robusta durante la trasmissione e l'archiviazione dei dati.	3%	4.500 - 6.000	1.500 - 2.000	1,77	31.500-42.000
	Input Data Manipulation (CAPEC-153)	10%	15.000 - 20.000	Validazione severa degli input	8000	Implementare una validazione rigorosa degli input dell'utente, accettando solo dati conformi a formati specifici e rifiutando input non validi o sospetti.	2%	3.000 - 4.000	1.000 - 2.000	1,37	16.000-22.000
	File Manipulation (CAPEC-165)	10%	15.000 - 20.000	Firme digitali per autenticare i dati, consentendo di verificare l'integrità del contenuto.	8000	La firma digitale è un metodo avanzato di autenticazione che impiega algoritmi crittografici per creare una firma unica associata a un documento o a un insieme di dati. Questa firma digitale viene generata utilizzando una chiave privata, e la sua autenticità può essere verificata da chiunque possieda la chiave pubblica corrispondente.	2%	3.000 - 4.000	1.000 - 2.000	1,37	16.000-22.000
	Inducing Account Lockout (CAPEC-2)	12%	18.000 - 24.000	Politiche di blocco degli account	3500	Configurare politiche di blocco degli account che definiscono regole per il numero massimo di tentativi di accesso falliti prima di bloccare un account. Assicurarsi che queste politiche siano bilanciate per impedire attacchi di forza bruta senza causare inconvenienti agli utenti legittimi.	4%	6.000 - 8.000	2.000 - 3.000	5,71	20.000-27.000
	Forced Deadlock(CAPEC-25)	12%	18.000 - 24.000	Gestione transazioni	3500	Utilizzare meccanismi di gestione transazionale per garantire che le transazioni siano completate o annullate in modo coerente, evitando situazioni di deadlock prolungate.	4%	6.000 - 8.000	2.000 - 3.000	4,29	5500 - 6500

Figure 84: Schema STRIDE completo 5 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Dispositivi IoT per monitoraggio salute	Identity Spoofing (CAPEC-51)	10%	5.000 - 10.000	Autenticazione forte e multifattore (MFA)	1500	Implementare un sistema di autenticazione forte che richieda più di una forma di verifica dell'identità. L'autenticazione multifattore (MFA) aggiunge un livello aggiuntivo di sicurezza.	2%	1.000-2.000	200-300	3,84	1700-1800
	Modify Shared File(562)	12%	6.000-12.000	Controllo degli accessi	1800	Implementare controlli rigorosi degli accessi per assicurarti che solo gli utenti autorizzati possano accedere e modificare i file condivisi. Usare meccanismi di controllo degli accessi basati su ruoli per limitare i privilegi.	3%	1.500 - 3.000	300 - 400	2,75	2100 - 2200
	Input Data Manipulation (CAPEC-153)	15%	7.500 - 15.000	Validazione severa degli input	2000	Implementare una validazione rigorosa degli input dell'utente, accettando solo dati conformi a formati specifici e rifiutando input non validi o sospetti.	4%	2.000 - 4.000	500 - 600	4,35	2.500 - 2.600
	Malicious Hardware Component Replacement (CAPEC-522)	10%	5.000 - 10.000	Fornitori affidabili	1500	Acquistare hardware solo da fornitori affidabili e verificati. Effettuare una rigorosa valutazione della sicurezza del fornitore e richiedi informazioni sulla catena di approvvigionamento.	2%	1.000-2.000	200-300	3,84	1700-1800
	Removing Important Client Functionality (CAPEC-207)	15%	7.500 - 15.000	Controllo degli accessi	2000	Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	4%	2.000 - 4.000	500 - 600	4,35	2.500 - 2.600
	Developement Alteration (CAPEC-444)	12%	6.000-12.000	Controllo degli accessi	1800	Limitare l'accesso ai repository di codice sorgente e agli artefatti di sviluppo solo a personale autorizzato. Utilizzare controlli di accesso basati su ruoli per assegnare privilegi appropriati.	4%	2.000 - 4.000	500 - 600	4,35	2.500 - 2.600
	Flooding (CAPEC-125)	14%	7.000-14.000	Limitazione delle richieste	1000	Applicare limiti alle richieste provenienti da singoli utenti o indirizzi IP per prevenire il flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	2%	1.000-2.000	200-300	8,75	1200 - 1300

Figure 85: Schema STRIDE completo 6 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	RoC	Overall Cost
Monitorare accesso ai dati	Functionality Bypass (CAPEC-554)	17%	3.400 - 5.100	Controllo degli accessi	500	Implementare un robusto sistema di controllo degli accessi per garantire che gli utenti possano accedere solo alle funzionalità per le quali sono autorizzati. Usare principi come il modello di autorizzazione basato su ruoli.	3%	600 - 900	100 - 200	7,2	600 - 700
	Exploitation of Trusted Identifiers (CAPEC-21)	10%	2.000 - 3.000	Criptografia degli identificatori importanti	700	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	2%	400 - 600	200 - 250	2,25	900 - 950
	Removing Important Client Functionality (CAPEC-207)	12%	2.400 - 3.600	Controllo degli accessi	800	Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	3%	600 - 300	300 - 400	2,31	1100 - 1200
	Developement Alteration (CAPEC-444)	17%	3.400 - 5.100	Controllo degli accessi	500	Limitare l'accesso ai repository di codice sorgente e agli artefatti di sviluppo solo a personale autorizzato. Utilizzare controlli di accesso basati su ruoli per assegnare privilegi appropriati.	3%	600 - 900	100 - 200	7,2	600 - 700
	Flooding (CAPEC-125)	10%	2.000 - 3.000	Limitazione delle richieste	700	Applicare limiti alle richieste provenienti da singoli utenti o indirizzi IP per prevenire il flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	2%	400 - 600	200 - 250	2,25	900 - 950

Figure 86: Schema STRIDE completo 7 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Gestire processo di lettura, scrittura e salvataggio dei dati	Adversary-In-The-Middle (CAPEC-94)	14%	7.000 - 8.400	Crittografia delle comunicazioni	900	Utilizzare protocolli di crittografia sicuri (come TLS/SSL) per proteggere le comunicazioni tra client e server. Ciò renderà più difficile per un attaccante intercettare o manipolare i dati trasmessi.	2%	1.000 - 1.200	300 - 400	7,17	1200 - 1300
	Transaction or Event Tampering via Application API Manipulation (CAPEC-385)	16%	8.000 - 9.600	Validazione e sanitizzazione degli input	1200	Condurre una rigorosa validazione e sanitizzazione degli input ricevuti dalle API per garantire che solo dati validi e sicuri vengano elaborati. Evitare l'esecuzione di comandi o query direttamente basati sugli input dell'utente.	3%	1.500 - 1.800	400 - 500	5,96	1600 - 1700
	Exploitation of Trusted Identifiers (CAPEC-21)	14%	7.000 - 8.400	Criptografia degli identificatori importanti	900	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	2%	1.000 - 1.200	300 - 400	7,17	1200 - 1300
	Removing Important Client Functionality (CAPEC-207)	10%	5.000 - 6.000	Controllo degli accessi	700	Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	4%	2.000 - 2.400	500 - 600	6,07	1200 - 1300
	Removing Important Client Functionality (CAPEC-207)	16%	8.000 - 9.600	Controllo degli accessi	1200	Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	3%	1.500 - 1.800	400 - 500	5,96	1600 - 1700
	Developement Alteration (CAPEC-444)	10%	5.000 - 6.000	Controllo degli accessi	700	Limitare l'accesso ai repository di codice sorgente e agli artefatti di sviluppo solo a personale autorizzato. Utilizzare controlli di accesso basati su ruoli per assegnare privilegi appropriati.	4%	2.000 - 2.400	500 - 600	6,07	1200 - 1300
	Flooding (CAPEC-125)	14%	7.000 - 8.400	Limitazione delle richieste	900	Applicare limiti alle richieste provenienti da singoli utenti o indirizzi IP per prevenire il flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	2%	1.000 - 1.200	300 - 400	7,17	1200 - 1300

Figure 87: Schema STRIDE completo 8 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Procedure di autenticazione	Exploitation of Trusted Identifiers (CAPEC-21)	12%	1.200-2.400	Criptografia degli identificatori importanti	900	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	3%	300-600	100-250	0,5	1.000-1.150
	Identity Spoofing (CAPEC-151)	16%	1.600-3.200	Autenticazione multifattore (MFA)	450	Implementare l'autenticazione multifattore per richiedere più di un metodo di verifica dell'identità. Ciò rende più difficile per un attaccante impersonare un utente legittimo.	4%	400-800	250-400	4,72	700-1250
	Use of Known Domain Credentials (CAPEC-560)	11%	1.100-2.200	Gestione delle password robusta	600	Implementare politiche di password robuste, con requisiti di complessità e scadenze periodiche. Utilizza strumenti per la gestione delle password che consentano la creazione, la rotazione e la memorizzazione sicura delle stesse.	4%	400-800	250-400	0,75	850-1000
	Eavesdropping(CAPEC-651)	18%	1.800-3.600	Crittografia delle comunicazioni	1150	Utilizzare protocolli di crittografia sicuri, come TLS/SSL, per proteggere le comunicazioni tra le parti. Ciò renderà difficile per un attaccante intercettare o interpretare il contenuto delle comunicazioni.	2%	200-400	50-175	1,44	1350-1550
	Inducing Account Lockout (CAPEC-2)	30%	3.000 - 6.000	Politiche di blocco degli account		Configurare politiche di blocco degli account che definiscono regole per il numero massimo di tentativi di accesso falliti prima di bloccare un account. Assicurarsi che queste politiche siano bilanciate per impedire attacchi di forza bruta senza causare inconvenienti agli utenti legittimi.	8%	800 - 1.600	500 - 600	2,8	3.500-6.600
	Removing Important Client Functionality (CAPEC-207)	25%	2.500 - 5.000	Controllo degli accessi		Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	10%	1.000 - 2.000	700 - 800	3	3.200-5.800
	Developement Alteration (CAPEC-444)	20%	2.000 - 4.000	Controllo degli accessi		Limitare l'accesso ai repository di codice sorgente e agli artefatti di sviluppo solo a personale autorizzato. Utilizzare controlli di accesso basati su ruoli per assegnare privilegi appropriati.	4%	400 - 800	100 - 200	2,93	2.100-4.200
	Flooding (CAPEC-125)	35%	3.500 - 7.000	Limitazione delle richieste		Applicare limiti alle richieste provenienti da singoli clienti o indirizzi IP per prevenire il Flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	10%	1.000 - 2.000	300 - 400	0,97	2.100 - 2.200

Figure 88: Schema STRIDE completo 9 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Controllo delle autorizzazioni	Privilege Escalation (CAPEC-233)	12%	1.200-2.400	Modello di autorizzazione basato su ruoli	900	Implementa un modello di autorizzazione basato su ruoli che assegna privilegi in modo granulare in base al ruolo dell'utente. Garantisce che gli utenti abbiano solo i privilegi necessari per svolgere le loro attività.	3%	300-600	100-250	0,5	1.000-1.150
	Flooding (CAPEC-125)	28%	2.800 - 5.600	Limitazione delle richieste	1000	Applicare limiti alle richieste provenienti da singoli utenti o indirizzi IP per prevenire il flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	8%	800 - 1.600	300 - 400	2	1300 - 1400
	Removing Important Client Functionality (CAPEC-207)	16%	1.600-3.200	Controllo degli accessi	450	Applicare rigorosi controlli degli accessi per garantire che solo utenti autorizzati abbiano la capacità di modificare o rimuovere funzionalità critiche dal client.	4%	400-800	250-400	4,72	700-1250
	Developement Alteration (CAPEC-444)	11%	1.100-2.200	Controllo degli accessi	600	Limitare l'accesso ai repository di codice sorgente e agli artefatti di sviluppo solo a personale autorizzato. Utilizzare controlli di accesso basati su ruoli per assegnare privilegi appropriati.	4%	400-800	250-400	0,75	850-1000
	Authentication Abuse (CAPEC-114)	16%	1.600-3.200	Autenticazione multifattore (MFA)	450	Implementare l'autenticazione multifattore per aggiungere uno strato aggiuntivo di sicurezza, richiedendo almeno due metodi di verifica dell'identità.	4%	400-800	250-400	4,72	700-1250
	Exploitation of Trusted Identifiers (CAPEC-21)	11%	1.100-2.200	Criptografia degli identificatori importanti	600	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	4%	400-800	250-400	0,75	850-1000
	Information Elicitation (CAPEC-410)	12%	1.200-2.400	Formazione sulla consapevolezza della sicurezza	900	Fornire formazione regolare agli utenti sulla consapevolezza della sicurezza, educandoli sui rischi associati all'information elicitation e sull'importanza di non divulgare informazioni sensibili.	3%	300-600	100-250	0,5	1.000-1.150
	Overread Buffers (CAPEC-540)	28%	2.800 - 5.600	Validazione degli input	1000	Implementare una rigorosa validazione degli input per assicurare che i dati ricevuti rispettino i limiti predefiniti e siano conformi alle aspettative dell'applicazione.	8%	800 - 1.600	300 - 400	2	1300 - 1400

Figure 89: Schema STRIDE completo 10 asset

Asset	Capec Pattern	Inherent Probability	Inherent Risk	Control	Cost	Feasibility	Residual Probability	Residual Impact	Residual Risk	R.o.C	Overall Cost
Accesso ai dati attività	Exploitation of Trusted Identifiers (CAPEC-21)	15%	7.500-15.000	Criptografia degli identificatori importanti	1450	Utilizzare la crittografia per proteggere gli identificatori fidati durante la trasmissione e l'archiviazione. Ciò rende più difficile per un attaccante utilizzare o manipolare gli identificatori in modo non autorizzato.	2%	1.000-2.000	200-300	5,72	1650 - 1750
	Identity Spoofing (CAPEC-151)	14%	7.000-14.000	Autenticazione multifattore (MFA)	1000	Implementare l'autenticazione multifattore per richiedere più di un metodo di verifica dell'identità. Ciò rende più difficile per un attaccante impersonare un utente legittimo.	2%	1.000-2.000	200-300	8,75	1200 - 1300
	Use of Known Domain Credentials (CAPEC-560)	12%	6.000-12.000	Gestione delle password robusta	1500	Implementa politiche di password robuste, con requisiti di complessità e scadenze periodiche. Utilizza strumenti per la gestione delle password che consentano la creazione, la rotazione e la memorizzazione sicura delle stesse.	2%	1.000-2.000	200-300	4,17	1700 - 1800
	Session Credential Falsification through Forging (CAPEC-196)	10%	5.000-10.000	Autenticazione multifattore (MFA)	400	Implementare l'autenticazione multifattore per richiedere più di un metodo di verifica dell'identità durante il processo di accesso. Questo rende più difficile la falsificazione delle credenziali di sessione.	3%	1.500-3.000	500-600	15,13	900 - 1000
	Log Injection-Tampering-Forging(CAPEC-93)	10%	5.000-10.000	Realizzazione di sistemi di backup e archiviazione di dati sicuri	400	Questo strumento permette di recuperare i dati in caso di perdita accidentale o di attacco malevolo attraverso la garanzia di backup regolari e dislocati in più luoghi.	3%	1.500-3.000	500-600	15,13	900 - 1000
	Inducing Account Lockout (CAPEC-2)	17%	8.500-17.000	Politiche di blocco degli account	1200	Configurare le politiche di blocco degli account che definiscono regole per il numero massimo di tentativi di accesso falliti prima di bloccare un account. Assicurarsi che queste politiche siano bilanciate per impedire attacchi di forza bruta senza causare inconvenienti agli utenti legittimi.	3%	1.500-3.000	300-400	9,54	1500 - 1600
	Flooding (CAPEC-125)	12%	6.000-12.000	Limitazione delle richieste	1500	Applicare limiti alle richieste provenienti da singoli utenti o indirizzi IP per prevenire il flooding. Questo può includere limiti di frequenza, limiti di velocità o altre politiche di controllo delle richieste.	2%	1.000-2.000	200-300	4,17	1700 - 1800

Figure 90: Schema STRIDE completo 11 asset

Use case ID: UCA-02	
Use case Name: Rimozione o distruzione impropria di dati	
Actors	Sistema, attaccante
Description	Un attaccante riesce a manipolare impropriamente i dati sulle attività degli attori
Data	Dati attività attori
Stimulus and preconditions	<ol style="list-style-type: none"> 1. Il target sta registrando l'azione e i dati dell'utente. 2. Il target protegge in modo insufficiente l'accesso ai registri o ai meccanismi di registrazione.
Basic Flow	<ol style="list-style-type: none"> 1. Determinare le azioni e i dati che verranno registrati 2. Determinare il formato di registrazione del sistema 3. L'attaccante altera i dati registrati dal sistema nel log
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ul style="list-style-type: none"> • Controllare attentamente l'accesso ai file di log fisici. • Non permettere che dati contaminati siano scritti nel file di log senza una valida convalida dell'input. Potrebbe essere utilizzata una "allowlist" per convalidare correttamente i dati. • Utilizzare la sincronizzazione per controllare il flusso di esecuzione. • Utilizzare strumenti di analisi statica per identificare vulnerabilità di falsificazione del log. • Evitare di visualizzare i log con strumenti che potrebbero interpretare caratteri di controllo nel file, come le shell della riga di comando.
Non Functional Requirements	Integrity
Comments	

Figure 22: Rimozione o distruzione impropria di dati

Use case ID: UCA-03	
Use case Name: Lettura dei dati confidenziali delle attività degli attori	
Actors	Sistema, attaccante
Description	Un attaccante attraverso l'uso eccessivo di una specifica funzionalità o sfruttando una funzionalità con difetti di progettazione accede a dati sensibili.
Data	Dati attività attori
Stimulus and preconditions	Il sistema target non implementa adeguatamente le misure di sicurezza per prevenire l'uso improprio di azioni/processi autorizzati.
Basic Flow	<ol style="list-style-type: none"> 1. <i>Identificazione della Funzionalità.</i> L'attaccante individua la funzione di ricerca nell'applicazione. 2. L'attaccante scopre e sfrutta difetti nella funzione di ricerca. 3. <i>Usando input malevoli,</i> l'attaccante ottiene accesso non autorizzato a dati sensibili.
Alternative Flow	-
Exception Flow	-
Response and	La riservatezza delle informazioni degli utenti è compromessa.
Postconditions	
Non Functional Requirements	Confidentiality
Comments	-

Figure 23: Lettura dei dati confidenziali delle attività degli attori

Use case ID: UCA-04	
Use case Name: Corruzione dei dati riferiti alle attività degli attori al fine di rendere incomprensibile quale attività è stata compiuta e da quale attore	
Actors	Attaccante, sistema.
Description	L'attaccante corrompe i dati relativi alle attività degli attori al fine di rendere incomprensibile quale attività è stata compiuta e da quale attore.
Data	File attività attori
Stimulus and preconditions	-
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante individua i file o i dati che contengono informazioni sulle attività degli attori all'interno del sistema. 2. L'attaccante ricerca e sfrutta vulnerabilità nel sistema che consentono l'accesso non autorizzato o la manipolazione dei file identificati. 3. Sfruttando le vulnerabilità scoperte, l'attaccante corrompe intenzionalmente i dati relativi alle attività degli attori, alterando informazioni critiche o rendendo illeggibili i dettagli delle azioni compiute. 4. L'attaccante cerca di conservare l'anonimato o di mascherare le proprie azioni per eludere la rilevazione, aumentando l'efficacia dell'attacco
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Le attività degli attori sono incomprensibili.
Non Functional Requirements	Accountability
Comments	-

Figure 24: Corruzione dei dati riferiti alle attività degli attori

Use case ID: UCA-05	
Use case Name: Rendere non disponibili i dati riferiti alle attività compiute dagli attori	
Actors	Sistema, Attaccante
Description	<p>Un utente malintenzionato modifica il contenuto o gli attributi dei file (come le estensioni o i nomi) in modo da causare un'elaborazione errata da parte di un'applicazione. Gli aggressori utilizzano questa classe di attacchi per far entrare le applicazioni in stati instabili, sovrascrivere o esporre informazioni sensibili e persino eseguire codice arbitrario con i privilegi dell'applicazione. Questa classe di attacchi si differenzia dagli attacchi alle informazioni di configurazione (anche se basati su file) in quanto la manipolazione del file causa l'elaborazione di comportamenti non standard, come buffer overflow o l'uso di un interprete non corretto. Gli attacchi alla configurazione si basano sull'interpretazione corretta dei file da parte dell'applicazione per inserire informazioni di configurazione dannose. Allo stesso modo, gli attacchi alla localizzazione delle risorse si basano sul controllo della capacità dell'applicazione di individuare i file, mentre gli attacchi alla manipolazione dei file non richiedono che l'applicazione cerchi in una posizione non predefinita, sebbene le due classi di attacchi siano spesso combinate.</p>
Data	Dati attività attori
Stimulus and preconditions	La destinazione deve utilizzare il file interessato senza verificarne l'integrità.
Basic Flow	<ol style="list-style-type: none"> 1. Identificazione dell'applicazione che manipola file. 2. Comprensione del modo in cui l'applicazione legge, scrive o elabora i file. 3. Modifica dei nomi e degli attributi per ingannare l'applicazione. 4. Inserimento di file manipolati nell'ambiente dell'applicazione. 5. Sfruttamento delle manipolazioni per causare comportamenti indesiderati nell'applicazione. 6. Utilizzo delle debolezze create per ottenere accesso non autorizzato o eseguire codice dannoso. 7. Tentativo di nascondere le manipolazioni effettuate. 8. Mantenimento di un accesso persistente all'ambiente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Dati attività attori non disponibili
Non Functional Requirements	Availability
Comments	-

Figure 25: Rendere non disponibili i dati riferiti alle attività ⁷⁹

Use case ID: UCA-06	
Use case Name: Modifiche alle policy di autorizzazione	
Actors	Sistema, Attaccante
Description	L'attaccante mira a eludere i meccanismi di filtraggio dell'input sull'applicazione di destinazione per inviare dati dannosi che potrebbero portare a vulnerabilità o compromettere l'integrità del sistema.
Data	Policy di autorizzazione
Stimulus and preconditions	L'applicazione di destinazione deve utilizzare una sorta di meccanismo di filtraggio (ingresso, uscita o mascheramento dei dati).
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza l'applicazione di destinazione e identifica i filtri di input. 2. L'attaccante utilizza tecniche per eludere o disabilitare i filtri di input. 3. Senza filtri, l'attaccante invia dati dannosi all'applicazione. 4. L'applicazione elabora i dati dannosi, aprendo la possibilità di sfruttare vulnerabilità.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso non autorizzato, esecuzione di codice dannoso, danni o interruzioni del servizio.
Non Functional Requirements	Integrity
Comments	-

Figure 26: Modifiche alle policy di autorizzazione

Use case ID: UCA-07	
Use case Name: Rendere non disponibili le policy di autorizzazione	
Actors	Sistema, Attaccante
Description	L'avversario mira a innescare e sfruttare una condizione di deadlock nel software di destinazione per causare un attacco di negazione del servizio (Availability).
Data	Policy di autorizzazione
Stimulus and preconditions	L'host di destinazione espone un'API all'utente.
Basic Flow	<ol style="list-style-type: none"> 1. L'avversario avvia una fase esplorativa per familiarizzare con il sistema. 2. L'avversario innesca una prima azione (ad esempio, il possesso di una risorsa) e ne avvia una seconda che attende il completamento della prima. 3. Se il programma bersaglio presenta una condizione di deadlock, il programma attende indefinitamente, causando una negazione del servizio.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Viene intaccata la disponibilità del servizio.
Non Functional Requirements	Availability
Comments	-

Figure 27: Rendere non disponibili le policy di autorizzazione

Use case ID: UCA-08	
Use case Name: Assenza servizi per atti malevoli dovuti a errori di autenticazione	
Actors	Sistema, Attaccante
Description	L'attaccante cerca di sfruttare controlli di protezione inadeguati nel sistema di destinazione per eseguire azioni non autorizzate approfittando di livelli di protezione insufficienti o di politiche di accesso troppo restrittive.
Data	Policy di autenticazione
Stimulus and preconditions	L'obiettivo deve applicare i controlli di accesso, ma configurarli in modo errato. Tuttavia, non tutte le configurazioni errate possono essere sfruttate da un aggressore. Se la configurazione errata applica un livello di sicurezza troppo basso ad alcune funzionalità, l'aggressore può essere in grado di sfruttarlo se il controllo degli accessi è l'unica cosa che impedisce l'accesso di un aggressore e non lo è più. Se la configurazione errata applica troppa sicurezza, deve impedire un'attività legittima e l'attaccante deve essere in grado di costringere altri a richiedere questa attività.
Basic Flow	<ol style="list-style-type: none"> <i>Esplora:</i> L'attaccante esamina l'applicazione di destinazione, eventualmente come utente valido e autenticato. <i>Espérimento:</i> Identificare i punti deboli nelle configurazioni di controllo degli accessi: L'attaccante sonda il controllo degli accessi per le funzioni e i dati identificati nella fase di esplorazione per identificare potenziali punti deboli nella configurazione del controllo degli accessi. <i>Sfruttamento:</i> Accesso alla funzione o ai dati aggirando il controllo di accesso: L'aggressore esegue la funzione o accede ai dati identificati nella fase di esplorazione aggirando il controllo di accesso.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Vengono limitate le funzionalità
Non Functional Requirements	Availability
Comments	-

Figure 28: Assenza servizi per atti malevoli

Use case ID: UCA-09	
Use case Name: Manipolazione policy di autenticazione	
Actors	Sistema, Attaccante
Description	L'attaccante cerca di sfruttare controlli di protezione inadeguati nel sistema di destinazione per eseguire azioni non autorizzate approfittando di livelli di protezione insufficienti o di politiche di accesso troppo restrittive.
Data	Policy di autenticazione
Stimulus and preconditions	L'attaccante deve avere un'idea generale delle regole di autenticazione del sistema.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante individua le regole di autenticazione del sistema. 2. L'attaccante sfrutta vulnerabilità nelle politiche di autenticazione. 3. Modificando parametri o regole, l'attaccante elude i controlli di autenticazione. 4. L'attaccante ottiene accesso non autorizzato al sistema. 5. L'attaccante cerca di nascondere le proprie azioni.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante esegue azioni dannose compromettendo la sicurezza del sistema.
Non Functional Requirements	Integrity
Comments	-

Figure 29: Manipolazione policy di autenticazione

Use case ID: UCA-10	
Use case Name: Interruzione del servizio di tracciamento degli accessi	
Actors	Sistema, Attaccante
Description	L'avversario mira a innescare e sfruttare una condizione di deadlock nel software di destinazione per causare un attacco di negazione del servizio (Availability).
Data	Policy di autorizzazione
Stimulus and preconditions	L'host di destinazione espone un'API all'utente.
Basic Flow	<ul style="list-style-type: none"> 4. L'avversario avvia una fase esplorativa per familiarizzare con il sistema. 5. L'avversario innesca una prima azione (ad esempio, il possesso di una risorsa) e ne avvia una seconda che attende il completamento della prima. 6. Se il programma bersaglio presenta una condizione di deadlock, il programma attende indefinitamente, causando una negazione del servizio.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Viene intaccata la disponibilità del servizio.
Non Functional Requirements	Availability
Comments	-

Figure 30: Interruzione del servizio di tracciamento degli accessi

Use case ID: UCA-11	
Use case Name: Divulgazione e/o lettura del file degli accessi	
Actors	Sistema, Attaccante
Description	L'avversario sfrutta un'applicazione che consente di copiare dati o informazioni sensibili raccogliendo le informazioni copiate negli appunti. I dati copiati negli appunti possono essere consultati da altre applicazioni, come ad esempio malware costruiti per esfiltrare o registrare il contenuto degli appunti su base periodica. In questo modo, l'avversario mira a raccogliere informazioni a cui non è autorizzato.
Data	Traccia degli accessi
Stimulus and preconditions	L'avversario deve disporre di un mezzo per raccogliere i dati dagli appunti e memorizzarli. In altre parole, quando l'obiettivo copia i dati negli appunti, l'avversario deve disporre di un mezzo per catturare tali dati in una terza posizione.
Basic Flow	<ol style="list-style-type: none"> 1. Trovare un'applicazione che consenta di copiare dati sensibili su una clipboard: Un avversario deve innanzitutto trovare un'applicazione che consenta di copiare e incollare informazioni sensibili. Potrebbe trattarsi di un'applicazione che stampa sullo schermo password temporanee, indirizzi e-mail privati o qualsiasi altra informazione o dato sensibile. 2. Puntare agli utenti dell'applicazione: Un avversario prenderà di mira gli utenti dell'applicazione per ottenere le informazioni contenute nei loro appunti su base periodica. 3. Attacco successivo: Utilizzare tutte le informazioni sensibili trovate per effettuare un attacco successivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso a dati sensibili
Non Functional Requirements	Confidentiality
Comments	-

Figure 31: Divulgazione e/o lettura del file degli accessi

Use case ID: UCA-12	
Use case Name: Corruzione del file degli accessi al fine di rendere incomprensibile chi ha eseguito l'accesso ed in quale momento	
Actors	Sistema, Attaccante
Description	L'attaccante sfrutta la possibilità di codificare in modo dannoso i file in modo da non rendere possibile l'utilizzo delle informazioni
Data	Traccia degli accessi
Stimulus and preconditions	Il decodificatore dell'applicazione accetta e interpreta caratteri codificati. La canonizzazione dei dati, il filtraggio degli input e la convalida non vengono eseguiti correttamente, lasciando aperta la porta a caratteri dannosi per l'host di destinazione.
Basic Flow	<ol style="list-style-type: none"> 1. Utilizzando un browser, uno strumento automatizzato o ispezionando direttamente l'applicazione, un avversario registra tutti i punti di accesso all'applicazione. 2. Sondare i punti di accesso per individuare vulnerabilità 3. L'avversario usa i punti di accesso come elenco di destinazione e inietta vari payload utilizzando diversi tipi di codifiche per determinare se un punto di accesso rappresenta effettivamente una vulnerabilità con una logica di convalida insufficiente e per caratterizzare l'entità con cui la vulnerabilità può essere sfruttata.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso ai dati sensibili.
Non Functional Requirements	Accountability
Comments	-

Figure 32: Corruzione del file degli accessi

Use case ID: UCA-13	
Use case Name: Manipolazione dei registri relativi agli accessi degli attori	
Actors	Sistema, Attaccante
Description	Un'attaccante modifica il contenuto o gli attributi in modo da causare un elaborazione errata.
Data	Traccia degli accessi
Stimulus and preconditions	Il target deve utilizzare i file modificati senza verificare la sua integrità.
Basic Flow	<p>I primi tre punti sono gli stessi del caso d'uso precedente</p> <p>4. L'attaccante osserva gli effetti della manipolazione</p> <p>5. Se l'attaccante riesce con successo a manipolare i file in modo</p> <p>che l'applicazione li elabori in modo improprio, può sfruttare questa per ottenere privilegi</p>
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso ai dati sensibili
Non Functional Requirements	Integrity
Comments	-

Figure 33: Manipolazione dei registri relativi agli accessi

Use case ID: UCA-14	
Use case Name: Accesso non autorizzato	
Actors	Sistema, Attaccante
Description	Un'attaccante ottiene accesso non autorizzato a causadi una debolezza nel meccanismo di autenticazione o tramite lo sfruttamento di un difetto nell'implementazione di uno schema di autenticazione.
Data	Traccia degli accessi
Stimulus and preconditions	Un meccanismo o sottosistema di autenticazione che implementa una forma di autenticazione come password, autenticazione digest, certificati di sicurezza, ecc., ma che presenta qualche difetto.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza il funzionamento del meccanismo di autenticazione per individuare eventuali debolezze o vulnerabilità. 2. L'attaccante raccoglie informazioni sulle credenziali degli utenti, i dettagli del processo di autenticazione o eventuali punti deboli noti del sistema. 3. L'attaccante sperimenta diverse tecniche di attacco per tentare di indovinare la password. 4. Se il meccanismo di autenticazione presenta difetti, l'attaccante cerca di eluderne le protezioni per ottenere l'accesso non autorizzato.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso a dati sensibili
Non Functional Requirements	Authentication
Comments	

Figure 34: Accesso non autorizzato

Use case ID: UCA-15	
Use case Name: Accesso non autorizzato ai dati clinici riferiti ad un paziente	
Actors	Sistema, Attaccante
Description	Un'attaccante ottiene accesso non autorizzato a causa di una debolezza nel meccanismo di autenticazione o tramite lo sfruttamento di un difetto nell'implementazione di uno schema di autenticazione.
Data	Dati relativi ad un paziente
Stimulus and preconditions	Un meccanismo o sottosistema di autenticazione che implementa una forma di autenticazione come password, autenticazione digest, certificati di sicurezza, ecc., ma che presenta qualche difetto.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza il funzionamento del meccanismo di autenticazione per individuare eventuali debolezze o vulnerabilità. 2. L'attaccante raccoglie informazioni sulle credenziali degli utenti, i dettagli del processo di autenticazione o eventuali punti deboli noti del sistema. 3. L'attaccante sperimenta diverse tecniche di attacco per tentare di indovinare la password. 4. Se il meccanismo di autenticazione presenta difetti, l'attaccante cerca di eluderne le protezioni per ottenere l'accesso non autorizzato.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso a dati sensibili
Non Functional Requirements	Authentication
Comments	-

Figure 35: Accesso non autorizzato ai dati clinici

Use case ID: UCA-16	
Use case Name: Divulgazione o modifica di dati privati	
Actors	Sistema, Attaccante
Description	Un avversario ottiene accesso a dati sensibili che possono, così, essere diffusi o modificati, provocando un impatto negativo
Data	Dati relativi ad un paziente
Stimulus and preconditions	L'avversario ha la capacità di interagire direttamente con l'applicazione. Il sistema di destinazione non implementa adeguatamente le misure di sicurezza per prevenire un uso improprio delle azioni/processi autorizzati.
Basic Flow	<ol style="list-style-type: none"> 1. L'avversario cerca dei punti di debolezza 2. L'avversario interagisce con l'applicazione sfruttando le debolezze 3. L'avversario opera strategie di esfiliazione per trasferire i dati cercati
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Accesso ai dati sensibili
Non Functional Requirements	Integrity
Comments	-

Figure 36: Divulgazione o modifica di dati privati

Use case ID: UCA-17	
Use case Name: Corruzione dei dati al fine di rendere incomprensibile chi ha prescritto quale terapia/medicinale al paziente	
Actors	Sistema, Attaccante
Description	L'attaccante corrompe i dati di input relativi alle prescrizioni mediche per rendere incomprensibile chi ha prescritto quale terapia o medicinale a un paziente.
Data	Cartella clinica
Stimulus and preconditions	L'attaccante individua i dati relativi alle prescrizioni mediche nel sistema, compresi i dettagli sul medico e la terapia prescritta.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante ricerca e sfrutta vulnerabilità nel sistema che consentono l'accesso non autorizzato o la manipolazione dei dati di input delle prescrizioni mediche. 2. Sfruttando le vulnerabilità identificate, l'attaccante corrompe intenzionalmente i dati relativi alle prescrizioni mediche, alterando informazioni critiche come il nome del medico o il medicinale prescritto. 3. L'attaccante cerca di conservare l'anonimato o di mascherare le proprie azioni per eludere la rilevazione.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ol style="list-style-type: none"> 1. La corruzione dei dati genera confusione nel riconoscere chi ha prescritto quale terapia o medicinale a un paziente, compromettendo la sicurezza e l'integrità delle informazioni mediche.
Non Functional Requirements	Accountability
Comments	-

Figure 37: Corruzione dei dati

Use case ID: UCA-18	
Use case Name: Lettura non autorizzata e/o condivisione non autorizzata dei dati clinici di un paziente	
Actors	Sistema, Attaccante
Description	Un utente malintenzionato accede a file, modifica il contenuto o gli attributi in modo da causare un' elaborazione errata da parte di un'applicazione.
Data	Dati relativi ad un paziente
Stimulus and preconditions	Il target deve utilizzare il file interessato senza verificarne l'integrità.
Basic Flow	<ol style="list-style-type: none"> 1. Utilizzando un browser, uno strumento automatizzato o ispezionando direttamente l'applicazione, un avversario registra tutti i punti di accesso all'applicazione. 2. Sondare i punti di accesso per individuare vulnerabilità 3. L'avversario usa i punti di accesso come elenco di destinazione e inietta vari payload utilizzando diversi tipi di codifiche per determinare se un punto di accesso rappresenta effettivamente una vulnerabilità con una logica di convalida insufficiente e per caratterizzare l'entità con cui la vulnerabilità può essere sfruttata. 4. L'attaccante osserva gli effetti della manipolazione <p>5. Se l'attaccante riesce con successo a manipolare i file in modo che l'applicazione li elabori in modo improprio, può sfruttare questa per ottenere privilegi</p>
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Le modifiche apportate al file sono state correttamente salvate nel sistema.
Non Functional Requirements	Confidentiality
Comments	-

Figure 38: Lettura non autorizzata e/o condivisione non autorizzata

Use case ID: UCA-19	
Use case Name: Limitare accesso ai dati	
Actors	Sistema, Attaccante
Description	L'avversario mira a innescare e sfruttare una condizione di deadlock nel software di destinazione per causare un attacco di negazione del servizio (Availability).
Data	Dati relativi ad un paziente
Stimulus and preconditions	Il target deve utilizzare il file interessato senza verificarne l'integrità.
Basic Flow	<ol style="list-style-type: none"> 1. L'avversario mira a innescare e sfruttare una condizione di deadlock nel software di destinazione per causare un attacco di negazione del servizio (Availability). L'avversario avvia una fase esplorativa per familiarizzare con il sistema. 2. L'avversario innesca una prima azione (ad esempio, il possesso di una risorsa) e ne avvia una seconda che attende il completamento della prima. 3. Se il programma bersaglio presenta una condizione di deadlock, il programma attende indefinitamente, causando una negazione del servizio.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Viene intaccata la disponibilità del servizio.
Non Functional Requirements	Availability
Comments	-

Figure 39: Limitare accesso ai dati

Use case ID: UCA-20	
Use case Name: Alterazione dell'ID di un dispositivo	
Actors	Sistema,attaccante
Description	L'attaccante altera l'ID di un dispositivo IoT per monitoraggio della salute per impersonare un'altra identità o eludere i controlli di autenticazione nel sistema di monitoraggio della salute.
Data	Dispositivi IOT per il monitoraggio della salute
Stimulus and preconditions	L'attaccante individua l'ID di un dispositivo IoT specifico per il monitoraggio della salute all'interno del sistema in modo tale che la sua modifica risulti essere "non rilevabile".
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante ricerca e sfrutta le vulnerabilità nel dispositivo IoT o nel sistema di monitoraggio della salute che consentono la modifica dell'ID del dispositivo. 2. Sfruttando le vulnerabilità identificate, l'attaccante altera l'ID del dispositivo IoT, sostituendolo con un valore arbitrario o con l'ID di un altro dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ol style="list-style-type: none"> 1. L'alterazione dell'ID consente all'attaccante di impersonare un'altra identità o di eludere i controlli di autenticazione basati sull'ID del dispositivo, manipolando i dati di monitoraggio della salute.
Non Functional Requirements	Authentication
Comments	-

Figure 40: Alterazione dell'ID di un dispositivo

Use case ID: UCA-21	
Use case Name: Manipolazione fonte dati IoT	
Actors	Sistema, Attaccante
Description	<p>Un aggressore sfrutta una debolezza nella convalida dell'input controllando il formato, la struttura e la composizione dei dati a</p> <p>un'interfaccia di elaborazione dell'input. Fornendo input di forma non standard o inaspettata, un aggressore può influire negativamente sulla sicurezza dell'obiettivo.</p>
Data	Dispositivi IOT per il monitoraggio della salute
Stimulus and preconditions	L'obiettivo deve accettare i dati dell'utente per l'elaborazione e il modo in cui questi dati vengono elaborati deve dipendere da qualche aspetto del formato o dei flag che l'attaccante può controllare.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante identifica un'applicazione o un sistema che accetta input utente o dati esterni. 2. L'attaccante analizza i punti di ingresso dell'applicazione, individuando i campi in cui è possibile inserire dati. 3. L'attaccante inserisce dati manipolati o dannosi attraverso i punti di ingresso identificati. 4. L'attaccante sfrutta i dati manipolati per ottenere un comportamento indesiderato dall'applicazione, come l'esecuzione di codice arbitrario, l'accesso non autorizzato o altri risultati dannosi. 5. L'attaccante cerca di nascondere le proprie tracce, modificando i dati manipolati o sfruttando eventuali debolezze nella registrazione delle attività.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ol style="list-style-type: none"> 1. Dopo aver rilevato e risolto l'attacco, le misure di sicurezza devono essere ripristinate per evitare futuri casi di manipolazione dei dati di input. 2. I dati manipolati devono essere ripuliti o riparati in modo che l'applicazione possa continuare a operare correttamente.
Non Functional Requirements	Authenticity
Comments	-

Figure 41: Manipolazioni fonte dati IoT

Use case ID: UCA-22	
Use case Name: Manomissione dispositivo	
Actors	Sistema, Attaccante
Description	Un avversario sostituisce l'hardware legittimo del sistema con hardware difettoso, contraffatto o manomesso nel canale di distribuzione della catena di fornitura, con l'obiettivo di provocare un'interruzione dolosa o di consentire un'ulteriore compromissione quando il sistema viene implementato.
Data	Dispositivi IoT per monitoraggio salute
Stimulus and preconditions	Accesso fisico al sistema dopo che ha lasciato il produttore, ma prima che venga installato nel luogo in cui si trova la vittima.
Basic Flow	<ol style="list-style-type: none"> 1. L'avversario deve innanzitutto identificare un sistema che desidera colpire e un componente hardware specifico che può sostituire con un componente maligno. 2. L'avversario traccia la catena di fornitura del sistema preso di mira. Cerca le opportunità per ottenere l'accesso fisico al sistema dopo che questo ha lasciato il produttore, ma prima che venga distribuito alla vittima. 3. Prima di eseguire l'attacco in natura, un avversario lo testerà su un sistema che ha acquistato per assicurarsi che il risultato desiderato venga raggiunto. 4. Utilizzando la vulnerabilità nella catena di fornitura del sistema scoperta nella fase di esplorazione, l'avversario sostituisce il componente dannoso con quello mirato. In questo modo l'avversario ottiene un accesso non intenzionale ai sistemi una volta raggiunta la vittima e può portare a una serie di attacchi successivi.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'hardware legittimo del sistema viene sostituito con hardware difettoso
Non Functional Requirements	Safety
Comments	-

Figure 42: Manomissione dispositivo

Use case ID: UCA-23	
Use case Name: Rendere il sistema poco affidabile	
Actors	Sistema, Attaccante
Description	Durante la fase di sviluppo di un sistema di monitoraggio della salute basato su dispositivi IoT, un attaccante sfrutta una vulnerabilità nel processo di sviluppo per introdurre codice dannoso. Questo codice dannoso è progettato per rendere il sistema poco affidabile, causando malfunzionamenti, falsi positivi/negativi e potenzialmente compromettendo la sicurezza e l'integrità delle informazioni di monitoraggio della salute.
Data	Dispositivi IoT per monitoraggio salute
Stimulus and preconditions	Il sistema di monitoraggio della salute è in fase di sviluppo. Gli sviluppatori stanno implementando nuove funzionalità o modificando il codice esistente.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante, approfittando di una vulnerabilità durante lo sviluppo, inserisce codice dannoso nel repository del progetto o durante la compilazione del codice. 2. Il codice dannoso è progettato per compromettere l'affidabilità del sistema di monitoraggio della salute, causando errori casuali, falsi positivi/negativi, o il malfunzionamento del dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il codice dannoso è stato integrato nel sistema di monitoraggio della salute durante lo sviluppo.
Non Functional Requirements	Reliability
Comments	-

Figure 43: Rendere il sistema poco affidabile

Use case ID: UCA-24	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Dispositivi IoT per monitoraggio salute
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata 2. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target. 3. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target. 4. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema. 5. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target. 6. L'attaccante può cercare di mantenere l'attacco nel tempo, adattandone le eventuali contromisure adottate dalla difesa del target
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 44: Rendere il sistema poco resistente agli attacchi

Use case ID: UCA-25	
Use case Name: Bypassare sistema di monitoraggio dei dati	
Actors	Sistema,attaccante
Description	Un attaccante cerca di bypassare con successo il sistema di monitoraggio dei dati per nascondere o modificare le proprie attività.
Data	Monitorare accesso ai dati
Stimulus and preconditions	<ul style="list-style-type: none"> 1. Il sistema di monitoraggio dei dati è operativo. 2. L'attaccante ha accesso al sistema target.
Basic Flow	<ul style="list-style-type: none"> 1. L'attaccante identifica la presenza di un sistema di monitoraggio dei dati all'interno dell'ambiente target. 2. L'attaccante analizza il sistema di monitoraggio per identificare eventuali vulnerabilità o debolezze. 3. L'attaccante sfrutta le vulnerabilità identificate per eludere o disabilitare temporaneamente il sistema di monitoraggio. 4. Una volta bypassato il sistema di monitoraggio, l'attaccante esegue attività nel sistema che non sono più soggette alla sorveglianza o al rilevamento.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema di monitoraggio dei dati è bypassato con successo, consentendo all'attaccante di eseguire attività non monitorate nel sistema target.
Non Functional Requirements	Accountability
Comments	-

Figure 45: Bypassare sistema di monitoraggio dei dati

Use case ID: UCA-26	
Use case Name: Compire azioni non autorizzate	
Actors	Sistema,attaccante
Description	Un avversario indovina, ottiene o "cavalca" un identificatore di fiducia (ad esempio, ID di sessione, ID di risorsa, cookie, ecc.) per eseguire azioni autorizzate sotto le sembianze di un utente o di un servizio autenticato.
Data	Monitorare accesso ai dati
Stimulus and preconditions	<ol style="list-style-type: none"> 1. Il software del server deve basarsi su schemi di prova e/o verifica degli identificatori deboli. 2. Gli identificatori devono avere una lunga durata e un potenziale di riutilizzo. 3. Il software del server deve consentire l'esistenza di sessioni simultanee.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante identifica un modo per sfruttare un identificatore di fiducia all'interno del sistema, come un token di accesso o una chiave di sessione. 2. Utilizzando l'identificatore di fiducia compromesso, l'attaccante accede in modo fraudolento al sistema di gestione dell'accesso. 3. Una volta all'interno del sistema, l'attaccante compie azioni non autorizzate per monitorare l'accesso ai dati sensibili degli utenti autorizzati.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante ha ottenuto accesso non autorizzato ai dati sensibili e ha monitorato le attività degli utenti autorizzati.
Non Functional Requirements	Authorization
Comments	-

Figure 46: Compire azioni non autorizzate

Use case ID: UCA-27	
Use case Name: Rendere il sistema poco affidabile	
Actors	Sistema, Attaccante
Description	Durante la fase di sviluppo di un sistema di monitoraggio della salute basato su dispositivi IoT, un attaccante sfrutta una vulnerabilità nel processo di sviluppo per introdurre codice dannoso. Questo codice dannoso è progettato per rendere il sistema poco affidabile, causando malfunzionamenti, falsi positivi/negativi e potenzialmente compromettendo la sicurezza e l'integrità delle informazioni di monitoraggio della salute.
Data	Monitorare accesso ai dati
Stimulus and preconditions	Il sistema di monitoraggio della salute è in fase di sviluppo. Gli sviluppatori stanno implementando nuove funzionalità o modificando il codice esistente.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante, approfittando di una vulnerabilità durante lo sviluppo, inserisce codice dannoso nel repository del progetto o durante la compilazione del codice. 2. Il codice dannoso è progettato per compromettere l'affidabilità del sistema di monitoraggio della salute, causando errori casuali, falsi positivi/negativi, o il malfunzionamento del dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il codice dannoso è stato integrato nel sistema di monitoraggio della salute durante lo sviluppo.
Non Functional Requirements	Reliability
Comments	-

Figure 47: Rendere il sistema poco affidabile

Use case ID: UCA-28	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Monitorare accesso ai dati
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<p>7. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata</p> <p>8. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target.</p> <p>9. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target.</p> <p>10. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema.</p> <p>11. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target.</p> <p>12. L'attaccante può cercare di mantenere l'attacco nel tempo, adattanAvailability alle eventuali contromisure adottate dalla difesa del target</p>
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 48: Rendere il sistema poco resistente agli attacchi

Use case ID: UCA-29	
Use case Name: Intercettazione e manipolazione dei dati	
Actors	Sistema,attaccante
Description	L'attaccante intercetta e manipola i dati durante il processo di lettura, scrittura e salvataggio all'interno del sistema di gestione dati.
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	L'attaccante identifica i dati specifici che desidera intercettare e manipolare durante il processo di lettura, scrittura e salvataggio.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante cerca e sfrutta le vulnerabilità nel sistema di gestione dati che consentono l'accesso non autorizzato o l'intercettazione dei dati. 2. Sfruttando le vulnerabilità identificate, l'attaccante intercetta i dati durante le operazioni di lettura, scrittura e salvataggio, ottenendo così un accesso non autorizzato alle informazioni.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	
Non Functional Requirements	Authentication
Comments	-

Figure 49: Intercettazione e manipolazione dei dati

Use case ID: UCA-30	
Use case Name: Modifica apportata al processo da parte di ignoti	
Actors	Sistema,attaccante
Description	L'attaccante può manipolare il contenuto in modo tale da produrre messaggi o contenuti che sembrano autentici, ma possono sostituire un elemento con un altro o contraffare un elemento esistente
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	Il software mirato sta utilizzando le API del framework dell'applicazione.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante analizza le API del framework dell'applicazione per identificare punti di accesso e funzionalità che gestiscono transazioni o eventi. 2. L'attaccante esamina il flusso di lavoro delle transazioni o eventi per comprendere come sono strutturati i dati, quali sono le operazioni consentite e come avvengono le interazioni. 3. L'attaccante sviluppa un software specializzato che intercetta e manipola le chiamate API tra l'applicazione e il server, consentendo la modifica dei dati di transazioni o eventi in transito. 4. Utilizzando il software, l'attaccante manipola i dati delle transazioni o eventi in transito, sostituendo o modificando elementi specifici per ottenere il risultato desiderato. 5. L'applicazione riceve i dati manipolati e, a causa della manipolazione delle API, mostra una risposta falsa che potrebbe sembrare autentica ma è stata alterata dall'attaccante.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante è riuscito ad accedere al processo ed apportare modifiche non autorizzate ai dati.
Non Functional Requirements	Accountability
Comments	-

Figure 50: Modifica apportata al processo

Use case ID: UCA-31	
Use case Name: Compiere azioni non autorizzate	
Actors	Sistema, Attaccante
Description	<p>Un utente malintenzionato modifica il contenuto o gli attributi dei file (come le estensioni o i nomi) in modo da causare un'elaborazione errata da parte di un'applicazione. Gli aggressori utilizzano questa classe di attacchi per far entrare le applicazioni in stati instabili, sovrascrivere o esporre informazioni sensibili e persino eseguire codice arbitrario con i privilegi dell'applicazione. Questa classe di attacchi si differenzia dagli attacchi alle informazioni di configurazione (anche se basati su file) in quanto la manipolazione del file causa l'elaborazione di comportamenti non standard, come buffer overflow o l'uso di un interprete non corretto. Gli attacchi alla configurazione si basano sull'interpretazione corretta dei file da parte dell'applicazione per inserire informazioni di configurazione dannose. Allo stesso modo, gli attacchi alla localizzazione delle risorse si basano sul controllo della capacità dell'applicazione di individuare i file, mentre gli attacchi alla manipolazione dei file non richiedono che l'applicazione cerchi in una posizione non predefinita, sebbene le due classi di attacchi siano spesso combinate.</p>
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	La destinazione deve utilizzare il file interessato senza verificarne l'integrità.
Basic Flow	<ul style="list-style-type: none"> 9. Identificazione dell'applicazione che manipola file. 10. Comprensione del modo in cui l'applicazione legge, scrive o elabora i file. 11. Modifica dei nomi e degli attributi per ingannare l'applicazione. 12. Inserimento di file manipolati nell'ambiente dell'applicazione. 13. Sfruttamento delle manipolazioni per causare comportamenti indesiderati nell'applicazione. 14. Utilizzo delle debolezze create per ottenere accesso non autorizzato o eseguire codice dannoso. 15. Tentativo di nascondere le manipolazioni effettuate. 16. Mantenimento di un accesso persistente all'ambiente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante è riuscito ad accedere al processo ed apportare modifiche non autorizzate ai dati.
Non Functional Requirements	Accountability
Comments	-

Figure 51: Compierd~~0~~zioni non autorizzate

Use case ID: UCA-32	
Use case Name: Limitare accesso al processo	
Actors	Sistema,attaccante
Description	Processo in cui un attaccante cerca di limitare l'accesso al processo di lettura, scrittura e salvataggio dati nel sistema target, riducendo così la funzionalità critica del client.
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	<ul style="list-style-type: none"> 1. Il sistema di gestione dati è operativo. 2. L'attaccante ha accesso al sistema target.
Basic Flow	<ul style="list-style-type: none"> 1. L'attaccante identifica le funzionalità critiche del client relative alla lettura, scrittura e salvataggio dei dati all'interno del sistema di gestione dati. 2. L'attaccante analizza le vulnerabilità o debolezze nelle <p>funzionalità identificate, cercando punti deboli che possono essere sfruttati per limitare o disabilitare l'accesso.</p> <ul style="list-style-type: none"> 3. L'attaccante sfrutta le vulnerabilità identificate per limitare l'accesso alle funzionalità critiche del client, impedendo così il corretto processo di lettura, scrittura e salvataggio dei dati. 4. Una volta completato con successo l'attacco, l'accesso alle funzionalità critiche è limitato, riducendo la capacità del client di gestire dati in modo completo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'accesso al processo di lettura, scrittura e salvataggio dati è stato limitato con successo, riducendo la funzionalità critica del client nel sistema di gestione dati.
Non Functional Requirements	Availability
Comments	-

Figure 52: Limitare accesso al processo

Use case ID: UCA-33	
Use case Name: Rendere il sistema poco affidabile	
Actors	Sistema, Attaccante
Description	Durante la fase di sviluppo di un sistema di monitoraggio della salute basato su dispositivi IoT, un attaccante sfrutta una vulnerabilità nel processo di sviluppo per introdurre codice dannoso. Questo codice dannoso è progettato per rendere il sistema poco affidabile, causando malfunzionamenti, falsi positivi/negativi e potenzialmente compromettendo la sicurezza e l'integrità delle informazioni di monitoraggio della salute.
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	Il sistema di monitoraggio della salute è in fase di sviluppo. Gli sviluppatori stanno implementando nuove funzionalità o modificando il codice esistente.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante, approfittando di una vulnerabilità durante lo sviluppo, inserisce codice dannoso nel repository del progetto o durante la compilazione del codice. 2. Il codice dannoso è progettato per compromettere l'affidabilità del sistema di monitoraggio della salute, causando errori casuali, falsi positivi/negativi, o il malfunzionamento del dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il codice dannoso è stato integrato nel sistema di monitoraggio della salute durante lo sviluppo.
Non Functional Requirements	Reliability
Comments	-

Figure 53: Rendere il sistema poco affidabile

Use case ID: UCA-34	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Processo di lettura, scrittura e salvataggio dati
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata 2. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target. 3. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target. 4. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema. 5. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target. 6. L'attaccante può cercare di mantenere l'attacco nel tempo, adattanAvailabilityi alle eventuali contromisure adottate dalla difesa del target
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 54: Rendere il sistema poco resistente agli attacchi

Use case ID: UCA-35	
Use case Name: Falsificazione identità utente	
Actors	Sistema, Attaccante
Description	<p>Un utente malintenzionato modifica il contenuto o gli attributi dei file (come le estensioni o i nomi) in modo da causare un'elaborazione errata da parte di un'applicazione. Gli aggressori utilizzano questa classe di attacchi per far entrare le applicazioni in stati instabili, sovrascrivere o esporre informazioni sensibili e persino eseguire codice arbitrario con i privilegi dell'applicazione. Questa classe di attacchi si differenzia dagli attacchi alle informazioni di configurazione (anche se basati su file) in quanto la manipolazione del file causa l'elaborazione di comportamenti non standard, come buffer overflow o l'uso di un interprete non corretto. Gli attacchi alla configurazione si basano sull'interpretazione corretta dei file da parte dell'applicazione per inserire informazioni di configurazione dannose. Allo stesso modo, gli attacchi alla localizzazione delle risorse si basano sul controllo della capacità dell'applicazione di individuare i file, mentre gli attacchi alla manipolazione dei file non richiedono che l'applicazione cerchi in una posizione non predefinita, sebbene le due classi di attacchi siano spesso combinate.</p>
Data	Procedure di autenticazione
Stimulus and preconditions	La destinazione deve utilizzare il file interessato senza verificarne l'integrità.
Basic Flow	<ol style="list-style-type: none"> 1. Identificazione dell'applicazione che manipola file. 2. Comprensione del modo in cui l'applicazione legge, scrive o elabora i file. 3. Modifica dei nomi e degli attributi per ingannare l'applicazione. 4. Inserimento di file manipolati nell'ambiente dell'applicazione. 5. Sfruttamento delle manipolazioni per causare comportamenti indesiderati nell'applicazione. 6. Utilizzo delle debolezze create per ottenere accesso non autorizzato o eseguire codice dannoso. 7. Tentativo di nascondere le manipolazioni effettuate. 8. Mantenimento di un accesso persistente all'ambiente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	<ol style="list-style-type: none"> 1. L'attaccante potrebbe aver compromesso l'identità associata agli identificatori fidati, ottenendo accesso non autorizzato o privilegi elevati. 2. L'attaccante può aver ottenuto accesso illegittimo a sistemi, risorse o dati in cui l'identificatore fidato è utilizzato come meccanismo di autenticazione o autorizzazione.
Non Functional Requirements	109 Authentication
Comments	-

Figure 55: Falsificazione identità utente

Use case ID: UCA-36	
Use case Name: Divulgazione informazioni riguardanti le procedure	
Actors	Sistema,attaccante
Description	Processo in cui un attaccante cerca di ottenere informazioni sensibili riguardanti le procedure di autenticazione attraverso l'eavesdropping.
Data	Procedure di autenticazione
Stimulus and preconditions	<ul style="list-style-type: none"> 1. Il sistema utilizza un meccanismo di autenticazione per garantire l'accesso autorizzato. 2. L'attaccante ha la capacità di eavesdrop sulle comunicazioni tra l'utente legittimo e il sistema di autenticazione.
Basic Flow	<ul style="list-style-type: none"> 1. L'attaccante identifica le comunicazioni contenenti informazioni sensibili relative alle procedure di autenticazione. Ciò potrebbe includere scambi di credenziali o token di autenticazione. 2. L'attaccante attua tecniche di eavesdropping per intercettare e monitorare le comunicazioni tra l'utente legittimo e il sistema di autenticazione. Ciò potrebbe avvenire su reti non protette o attraverso la compromissione di dispositivi di comunicazione. 3. L'attaccante raccoglie le credenziali o altri dati sensibili scambiati durante il processo di autenticazione. 4. L'attaccante analizza le informazioni intercettate per identificare le procedure di autenticazione, comprese eventuali debolezze o vulnerabilità. 5. L'attaccante utilizza le credenziali ottenute per ottenere accesso non autorizzato al sistema o per altri scopi malevoli.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	L'attaccante ha ottenuto informazioni sensibili sulle procedure di autenticazione attraverso l'eavesdropping.
Non Functional Requirements	Confidentiality
Comments	-

Figure 56: Divulgazione informazioni riservate

Use case ID: UCA-37	
Use case Name: Limitazione capacità di accesso alle procedure	
Actors	Sistema,attaccante
Description	Un utente malintenzionato modifica il contenuto o gli attributi dei file (come le estensioni o i nomi) in modo da causare un'elaborazione errata da parte di un'applicazione. Gli aggressori utilizzano questa classe di attacchi per far entrare le applicazioni in stati instabili, sovrascrivere o esporre informazioni sensibili e persino eseguire codice arbitrario con i privilegi dell'applicazione. Questa classe di attacchi si differenzia dagli attacchi alle informazioni di configurazione (anche se basati su file) in quanto la manipolazione dei file causa l'elaborazione di comportamenti non standard, come buffer overflow o l'uso di un interprete non corretto. Gli attacchi alla configurazione si basano sull'interpretazione corretta dei file da parte dell'applicazione per inserire informazioni di configurazione dannose. Allo stesso modo, gli attacchi alla localizzazione delle risorse si basano sul controllo della capacità dell'applicazione di individuare i file, mentre gli attacchi alla manipolazione dei file non richiedono che l'applicazione cerchi in una posizione non predefinita, sebbene le due classi di attacchi siano spesso combinate.
Data	Procedure di autenticazione
Stimulus and preconditions	La destinazione deve utilizzare il file interessato senza verificare l'integrità.
Basic Flow	<ol style="list-style-type: none"> 1. Identificazione dell'applicazione che manipola file. 2. Comprensione del modo in cui l'applicazione legge, scrive o elabora i file. 3. Modifica dei nomi e degli attributi per ingannare l'applicazione. 4. Inserimento di file manipolati nell'ambiente dell'applicazione. 5. Sfruttamento delle manipolazioni per causare comportamenti indesiderati nell'applicazione. 6. Utilizzo delle debolezze create per ottenere accesso non autorizzato o eseguire codice dannoso. 7. Tentativo di nascondere le manipolazioni effettuate. 8. Mantenimento di un accesso persistente all'ambiente.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Limitato accesso alle procedure di autenticazione
Non Functional Requirements	Availability
Comments	-

Figure 57: Limitazione capacità di accesso alla procedura

Use case ID: UCA-38	
Use case Name: Rendere il sistema poco affidabile	
Actors	Sistema, Attaccante
Description	Durante la fase di sviluppo di un sistema di monitoraggio della salute basato su dispositivi IoT, un attaccante sfrutta una vulnerabilità nel processo di sviluppo per introdurre codice dannoso. Questo codice dannoso è progettato per rendere il sistema poco affidabile, causando malfunzionamenti, falsi positivi/negativi e potenzialmente compromettendo la sicurezza e l'integrità delle informazioni di monitoraggio della salute.
Data	Procedure di autenticazione
Stimulus and preconditions	Il sistema di monitoraggio della salute è in fase di sviluppo. Gli sviluppatori stanno implementando nuove funzionalità o modificando il codice esistente.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante, approfittando di una vulnerabilità durante lo sviluppo, inserisce codice dannoso nel repository del progetto o durante la compilazione del codice. 2. Il codice dannoso è progettato per compromettere l'affidabilità del sistema di monitoraggio della salute, causando errori casuali, falsi positivi/negativi, o il malfunzionamento del dispositivo.
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il codice dannoso è stato integrato nel sistema di monitoraggio della salute durante lo sviluppo.
Non Functional Requirements	Reliability
Comments	-

Figure 58: Rendere il sistema poco affidabile

Use case ID: UCA-39	
Use case Name: Rendere il sistema poco resistente agli attacchi	
Actors	Sistema,attaccante
Description	Un attaccante potrebbe inondare il sistema di interazioni o richieste. In questo modo si andrebbero a consumare le risorse disponibili nel sistema di destinazione (come la banda di rete o la capacità di elaborazione) rendendolo maggiormente vulnerabile ad altri attacchi
Data	Procedure di autenticazione
Stimulus and preconditions	Ogni obiettivo che fornisce servizi tramite richieste è vulnerabile a questo attacco su qualche livello di scala.
Basic Flow	<ol style="list-style-type: none"> 1. L'attaccante studia il sistema e decide quale tipologia di flooding è appropriata 2. L'attaccante utilizza strumenti automatizzati o script per generare un alto volume di richieste o traffico di rete da inviare al target. 3. L'attaccante invia in modo massiccio le richieste al target, sfruttando la vulnerabilità nelle difese del sistema o sovraccaricando le risorse del target. 4. A causa del grande volume di richieste, le risorse del target vengono gradualmente saturate, portando a una diminuzione delle prestazioni del sistema. 5. Man mano che le risorse del target si saturano ulteriormente, si verifica un'interruzione del servizio per gli utenti legittimi che cercano di accedere alle risorse del target. 6. L'attaccante può cercare di mantenere l'attacco nel tempo, adattanAvailability alle eventuali contromisure adottate dalla difesa del target
Alternative Flow	-
Exception Flow	-
Response and Postconditions	Il sistema è reso vulnerabile
Non Functional Requirements	Resilience
Comments	-

Figure 59: Rendere il sistema poco resistente agli attacchi

3 Software Design

Durante la fase di analisi dei requisiti, vengono delineate specifiche che spesso non sono esaustive per la completa definizione di un software. Perciò diventa essenziale stabilire l'architettura del software e le interazioni tra i suoi componenti.

Le scelte di architettura e tecnologiche influenzano direttamente la sicurezza e il raggiungimento degli obiettivi di affidabilità. È quindi cruciale valutare attentamente le opzioni disponibili per assicurare il rispetto degli obiettivi di sicurezza prestabiliti. Ogni decisione presa in fase di progettazione deve conformarsi alle politiche di sicurezza identificate durante l'analisi dei rischi, seguendo il principio di "*Base decisions on an explicit security policy*", come indicato nel *Catalogo di Sommerville*.

Inoltre, si presta particolare attenzione al principio di "*Open Design*" del *Catalogo di Saltzer e Schroeder*, che corrisponde al principio "*Avoid Security by Obscurity*" nel *Catalogo OWASP*. Questi principi sottolineano l'importanza di sviluppare il sistema utilizzando informazioni progettuali condivise pubblicamente.

3.1 Design assets

Nel corso degli anni, sono emerse linee guida e buone pratiche fondamentali per una corretta fase di progettazione. Queste direttive mirano a sensibilizzare sui potenziali rischi di sicurezza associati alle decisioni di progettazione. Diversi cataloghi sono stati proposti nel tempo, molti dei quali integrano e ampliano le linee guida originarie delineate da Saltzer e Schroeder nel 1975, per adattarsi alle nuove sfide e innovazioni tecnologiche.

In particolar modo, durante la fase di progettazione abbiamo scelto di seguire le linee guida di Sommerville in quanto complete e aggiornate rispetto alle altre.

3.2 Architectural design

La ridondanza svolge un ruolo chiave nella progettazione di software sicuro, implicando la duplicazione di componenti o funzionalità critiche per assicurare la continuità del servizio. In caso di malfunzionamento o attacco che comprometta una copia, le altre copie rimangono operative, garantendo così la prestazione ininterrotta del servizio.

Nella scelta dell'architettura da adottare, la struttura della Blockchain ha avuto un impatto significativo. Questa tecnologia consente la creazione di un registro distribuito per la gestione di transazioni condivise tra più nodi di una rete. Ogni nodo possiede una copia identica della Blockchain e svolge le stesse operazioni: verifica e propone l'aggiunta di transazioni al registro. Un aspetto fondamentale è che i dati registrati in un blocco non possono essere modificati retroattivamente senza il consenso della maggioranza della rete,

poiché ciò richiederebbe la modifica di tutti i blocchi successivi. Questa caratteristica rende le transazioni sulla Blockchain immutabili, conferendo al registro un'inalterabile affidabilità e distribuendo le copie su vari nodi, garantendo così la ridondanza e la resilienza del sistema.



Figure 91: Interazione DB e Blockchain

La nostra struttura si basa su un'architettura che comprende un *Database* remoto, accessibile tramite il seguente link: <https://www.db4free.net/phpMyAdmin/>, utilizzando le nostre credenziali:

- Nome Utente: progettoss
- Password: progettoss

Nel database vengono memorizzati tutti i dati, inclusi quelli di autenticazione (*User-name* e *Password*), i quali vengono criptati utilizzando una chiave da noi scelta e poi recuperati in fase di autenticazione.

Successivamente, i dati vengono sottoposti a un processo di *hashing* e caricati sulla Blockchain. Ogni volta che si ha necessità di modificare o visualizzare un dato, prima di eseguire qualsiasi operazione, è possibile verificare l'integrità di tali dati confrontando l'hash restituito dalla Blockchain con i dati in nostro possesso.

Inoltre, è importante notare che la blockchain Ganache utilizzata nello sviluppo del progetto non offre la persistenza dei dati. Ciò significa che ogni volta che Ganache viene chiusa e riaperta, i dati memorizzati sulla blockchain vengono persi, a differenza dei dati nel database, che rimangono intatti anche dopo la chiusura del programma. Per affrontare questa limitazione, abbiamo implementato dei metodi di ripristino. Ogni volta che il programma viene riavviato, tutti i dati vengono nuovamente sottoposti a hashing e caricati sulla blockchain. Questo processo aiuta a garantire una forma di persistenza dei dati.

Gli utenti possono interagire con il sistema tramite la linea di comando, ma solo dopo essersi autenticati avranno accesso alle operazioni e quindi alle richieste di transazioni. Abbiamo deliberatamente scelto di limitare al minimo le informazioni disponibili agli utenti, in linea con il principio del *Least Privilege* definito nei cataloghi di Saltzer e Schroeder e OWASP.

Sarà poi la parte off-chain a gestire il deploying degli *Smart Contract* e ad attivare a sua volta la parte on-chain quando richiesto.

3.2.1 DB Design

In Figura 92 riportiamo il diagramma E-R relativo al nostro progetto, utilizzato in fase di implementazione per impostare correttamente il database locale da noi utilizzato.

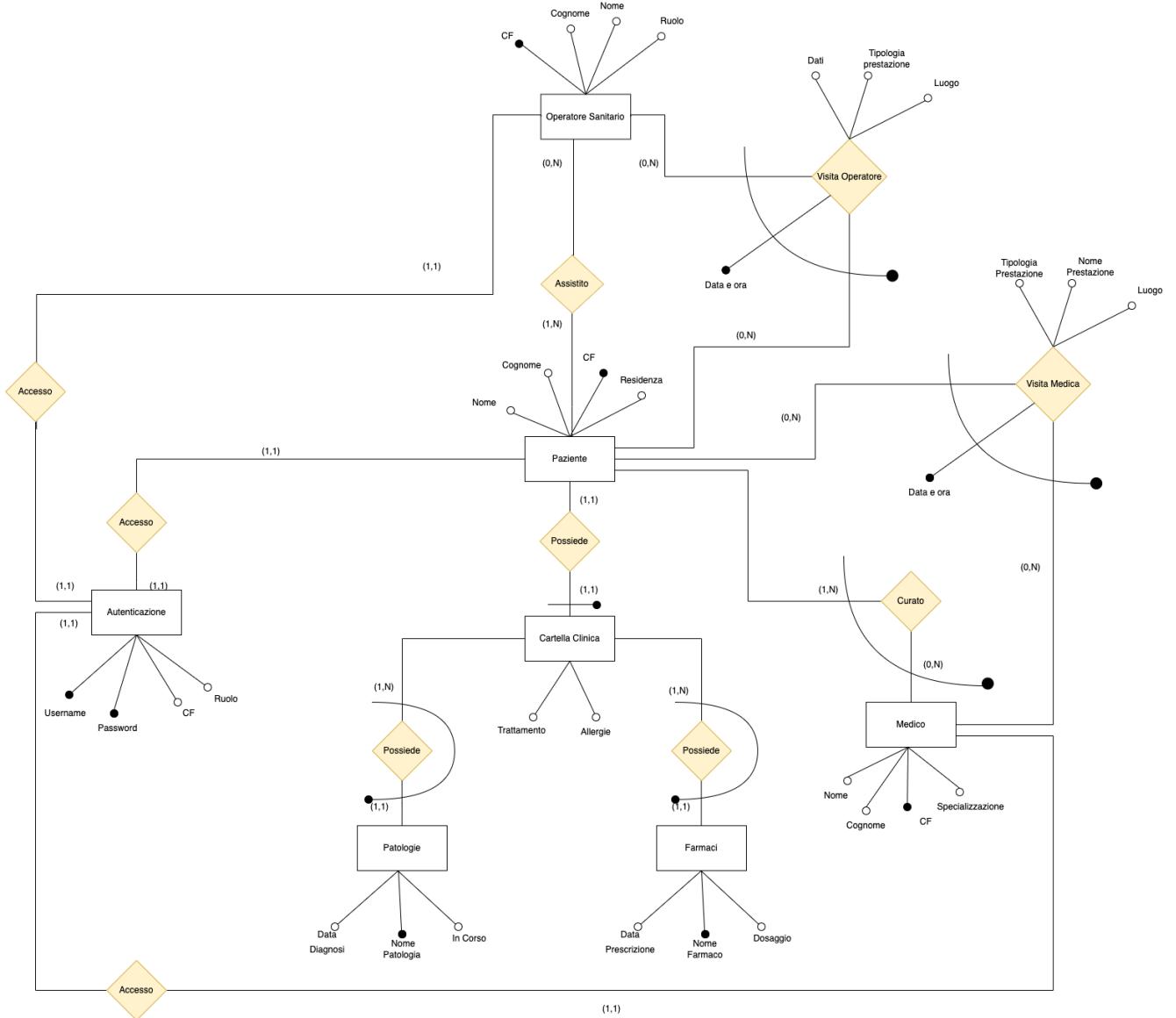


Figure 92: Diagramma E-R

Come si può notare in figura, sono state definite 7 entità:

- **Autenticazione**: rappresenta il sistema di accesso all'applicativo. Gli attributi definiti per tale entità sono:

- **CF**: Codice fiscale dell'utente

- **Username:** Il nome utente scelto per l’accesso.
 - **Password:** La password associata all’account. Insieme all’username costituiscono l’identificatore univoco dell’entità.
 - **Ruolo:** Il ruolo attribuito ad ogni utente in fase di registrazione. Può assumere tre valori, corrispondenti alle tipologie di utenti che possono operare nel sistema: Paziente, Medico, OperatoreSanitario.
- **Paziente:** rappresenta un utente sottoposto a trattamento sanitario. Gli attributi associati a questa entità sono:
 - **Nome:** Nome del paziente
 - **Cognome:** Cognome del paziente
 - **CF:** Identificatore univoco della entità
 - **Residenza:** Residenza del paziente
 - **Operatore Sanitario:** Rappresenta una figura professionale coinvolta nella gestione sanitaria. Gli attributi associati sono:
 - **CF:** Identificatore univoco dell’entità
 - **Nome:** Nome dell’operatore sanitario
 - **Cognome:** Cognome dell’operatore sanitario
 - **Ruolo:** Ruolo dell’operatore sanitario
 - **Medico:** Rappresenta un professionista medico coinvolto nel sistema. Gli attributi associati sono:
 - **CF:** Identificatore univoco dell’entità
 - **Nome:** nome del medico
 - **Cognome:** Cognome del medico
 - **Specializzazione:** Tipologia di medico.
 - **Cartella Clinica:** Rappresenta il raccoglitore principale delle informazioni mediche dei pazienti. Gli attributi associati sono:
 - **Trattamento:** Prescrizioni mediche o sanitarie.
 - **Allergie:** Allergie identificate.

L'identificatore univoco di questa entità è il CF del paziente a cui è associata.

- **Patologie:** Rappresenta una patologia diagnosticata ad un utente e associata ad una cartella clinica. Gli attributi definiti per questa entità sono:

- **Nome Patologia:** Nome della patologia
- **Data Diagnosi:** Data in cui è stata accertata
- **In Corso:** Stato della malattia

L'identificatore univo di questa entità è costituito dal CF del paziente e dal nome della patologia.

- **Farmaci:** Rappresenta un farmaco prescritto ad un utente e associato ad una cartella clinica. Gli attributi definiti per questa entità sono:

- **Nome Farmaco:** Nome del farmaco
- **Data Prescrizione:** Data in cui è stato prescritto il farmaco.
- **Dosaggio:** Dosaggio di assunzione del farmaco.

L'identificatore univo di questa entità è costituito dal CF del paziente e dal nome del farmaco.

Le entità sono associate tra loro mediante le seguenti relazioni:

- **Relazione tra Autenticazione e Paziente/Medico/Operatore Sanitario ”Accesso”:** Un paziente/medico/operatore sanitario per poter accedere al sistema devo autenticarmi mediante un username e una password.
- **Relazione tra Paziente e Cartella Clinica ”Possiede”:** Ogni paziente possiede una sola cartella clinica contenente le principali informazioni sullo stato di salute.
- **Relazione tra Cartella Clinica e Farmaci/Patologie:** Ogni cartella clinica è in relazione uno a n con l'entità farmaci e con l'entità patologie.
- **Relazione tra Paziente e Medico ”Curato”:** Ogni Paziente può essere preso in cura da uno o più medici.
- **Relazione tra Paziente e Medico ”Visita Medica”:** Paziente e Medico sono inoltre legati dalla relazione ”Visita Medica” che è caratterizzata da vari attributi:
 - Data e Ora

- Tipologia Prestazione
 - Luogo
 - Nome Prestazione
- **Relazione tra Paziente e Operatore Sanitario ”Assistito”:** Ogni paziente può essere assistito da uno o più operatori sanitari.
 - **Relazione tra Paziente e Operatore Sanitario ”Visita Operatore”:** Paziente e operatore sanitario sono inoltre legati dalla relazione ”Visita Operatore” che è caratterizzata da vari attributi:
 - Data e Ora
 - Tipologia Prestazione
 - Luogo
 - Dati

4 Implementation

Durante la fase di implementazione del software, sono state seguite e rispettate le linee guida di Sommerville. Tali linee guida saranno introdotte nelle sezioni successive, al fine di mostrare i motivi delle scelte implementative.

4.1 General overview

Per l'organizzazione della logica del nostro programma abbiamo deciso di adottare quanto più possibile il pattern *MVC* (*Model - View - Controller*), adattandolo alle nostre necessità e così impostato:

- *Model-View*: in quest'area rientrano i file come *medico.py*, *paziente.py*, *operatore-Sanitario.py* in cui, abbiamo la struttura del modello ma, allo stesso tempo, sono state implementate anche le logiche di view, come i menù e le varie interazioni, il tutto gestito tramite linea di comando quindi senza alcuna vista implementata.
- *Controller*: in quest'area, invece, abbiamo racchiuso tutti i controller suddivisi per operatori, quindi *controllerMedico.py*, *controllerPaziente.py*, *controllerOS.py*, ecc. Questi si occupano quindi di fare da intermediario tra i due precedenti.

In pratica, come già affermato, il nostro non rispecchia al massimo il pattern MVC, ma abbiamo cercato di adattarlo quanto più possibile alle nostre necessità per ribadire il concetto di linearità e indipendenza.

Altra caratteristica rilevante del nostro codice è l'utilizzo di un ulteriore pattern, ossia il pattern *Singleton*. Il pattern Singleton è un design pattern creazionale che viene utilizzato per garantire che una classe abbia una sola istanza e fornire un punto di accesso globale a tale istanza.

Le caratteristiche principali del pattern Singleton sono:

- *Una sola istanza*: Assicura che una classe abbia solo un'istanza nell'intero programma.
- *Accesso globale*: Fornisce un modo per accedere a quell'istanza da qualsiasi parte del programma.
- *Controllo dell'istanza*: Il pattern Singleton fornisce un meccanismo per controllare come e quando viene creata l'istanza unica della classe.

Questo pattern è stato più volte da noi utilizzato per garantire il rispetto dei tre punti precedentemente citati; nei controller, infatti, abbiamo deciso di usufruire delle caratteristiche di questo pattern.

Specificato ciò, concludiamo dando una visione generale su altri aspetti della struttura del programma. La parte di interazione con il database è rinchiusa nella classe *db.py* nella quale andiamo a sviluppare tutti i metodi che verranno richiamati per l'interazione con il database, inserimenti, modifiche, visualizzazioni ecc.

Per la parte on-chain, invece, abbiamo racchiuso il tutto in una cartella *solidityContracts* in cui abbiamo salvato i nostri tre principali smart contract:

- *MedicoContract.sol*
- *OSContract.sol*
- *PazienteContract.sol*

In ciascuno di questi file sono stati sviluppati mapping e metodi utili per l'interazione degli smart contract con la blockchain.

Per quanto riguarda la parte di autenticazione, gran parte della logica è gestita nel file *session.py* che, a sua volta, potrebbe richiedere l'utilizzo di ulteriori classi sparse nel codice.

Le variabili importanti per il nostro codice, come ad esempio l'indirizzo e la chiave da utilizzare su blockchain, oppure la chiave da utilizzare per la criptazione dei dati sul database, sono state correttamente salvate come *variabili d'ambiente* nel file *Chiavi.env*, così da garantirci un punto unico di accesso e un aspetto di sicurezza notevolmente migliorato.

4.2 Local Database

Il database implementato tramite PHPMyAdmin rappresenta il fondamento su cui si basa l'architettura di gestione dati del progetto. Questo database è stato sviluppato seguendo il modello concettuale preliminare delineato nel diagramma ER (Entity-Relationship), il quale ha guidato la strutturazione delle entità e delle relazioni fondamentali. L'implementazione del database in PHPMyAdmin si basa quindi strettamente sulla realizzazione del modello concettuale ER, garantendo coerenza e aderenza alle specifiche definite in fase di progettazione. Il database comprende una serie di tabelle correlate, ciascuna delle quali rappresenta un'entità specifica nel contesto del progetto. Le relazioni tra le tabelle sono state attentamente progettate per riflettere fedelmente le connessioni e le dipendenze definite nel diagramma ER, assicurando la coerenza e l'integrità dei dati. Inoltre, sono state

implementate funzionalità di sicurezza per proteggere l'accesso non autorizzato e garantire la riservatezza delle informazioni sensibili.

In particolare, il database è costituito da 11 tabelle: assistito, autenticazione, cartellaClinica, curato, farmaci, medico, operatoreSanitario, patologie, paziente, visitaMedico e visitaOperatore.

4.2.1 Tabella Assistito

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CFOperatore 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	CFPaziente  	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno

Figure 93: Tabella Assistito

4.2.2 Tabella Autenticazione

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CF	varbinary(255)			No	Nessuno
2	Username 	varbinary(255)			No	Nessuno
3	Password 	varbinary(255)			No	Nessuno
4	Ruolo	varbinary(255)			Sì	NULL

Figure 94: Tabella autenticazione

4.2.3 Tabella cartellaClinica

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CFPaziente 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	Trattamento	text	utf8mb4_0900_ai_ci		Sì	NULL
3	Allergie	text	utf8mb4_0900_ai_ci		Sì	NULL

Figure 95: Tabella cartellaClinica

4.2.4 Tabella Curato

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CFMedico 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	CFPaziente  	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno

Figure 96: Tabella curato

4.2.5 Tabella Farmaci

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	IdCartellaClinica 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	NomeFarmaco 	varchar(50)	utf8mb4_0900_ai_ci		No	Nessuno
3	DataPrescrizione	datetime			Sì	NULL
4	Dosaggio	text	utf8mb4_0900_ai_ci		Sì	NULL

Figure 97: Tabella farmaci

4.2.6 Tabella medico

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CF 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	Nome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
3	Cognome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
4	Specializzazione	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL

Figure 98: Tabella medico

4.2.7 Tabella operatoreSanitario

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CF 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	Nome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
3	Cognome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
4	Ruolo	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL

Figure 99: Tabella operatoreSanitario

4.2.8 Tabella patologie

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	IdCartellaClinica 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	NomePatologia 	varchar(50)	utf8mb4_0900_ai_ci		No	Nessuno
3	DataDiagnosi	datetime			Sì	NULL
4	InCorso	tinyint(1)			Sì	NULL

Figure 100: Tabella patologie

4.2.9 Tabella paziente

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CF 	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	Nome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
3	Cognome	varchar(50)	utf8mb4_0900_ai_ci		Sì	NULL
4	Residenza	varchar(100)	utf8mb4_0900_ai_ci		Sì	NULL

Figure 101: Tabella paziente

4.2.10 Tabella visitaMedico

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CFPaziente	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	CFMedico	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
3	Dati	text	utf8mb4_0900_ai_ci		Sì	NULL
4	DataOra	datetime			No	Nessuno
5	TipoPrestazione	varchar(100)	utf8mb4_0900_ai_ci		Sì	NULL
6	Luogo	varchar(100)	utf8mb4_0900_ai_ci		Sì	NULL

Figure 102: Tabella visitaMedico

4.2.11 Tabella visitaOperatore

#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito
1	CFPaziente	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
2	CFOperatoreSanitario	varchar(20)	utf8mb4_0900_ai_ci		No	Nessuno
3	Dati	text	utf8mb4_0900_ai_ci		Sì	NULL
4	DataOra	datetime			No	Nessuno
5	TipoPrestazione	varchar(100)	utf8mb4_0900_ai_ci		Sì	NULL
6	Luogo	varchar(100)	utf8mb4_0900_ai_ci		Sì	NULL

Figure 103: Tabella visitaOperatore

4.3 Interazioni Utente-Software

L’interfaccia del software è progettata per supportare le diverse azioni degli utenti in base al loro ruolo all’interno del sistema. Di seguito sono elencate le azioni disponibili per ciascun tipo di utente:

- **Medico**

- Inserimento dei Dati di una Nuova Visita Medica: Il medico può inserire i dettagli relativi a una nuova visita medica effettuata.
- Visualizzazione delle Visite Mediche Effettuate: Accesso alla lista completa delle visite mediche da lui effettuate.
- Aggiornamento delle Visite Mediche: Possibilità di aggiornare i dati di una visita medica precedentemente registrata.
- Aggiunta di un Nuovo Paziente in Cura: Il medico può aggiungere un paziente alla propria lista di assistiti.
- Aggiornamento della Cartella Clinica di un Paziente: Modifica e aggiornamento dei dati presenti nella cartella clinica di un paziente in cura.

- **Paziente**

- Visione della Propria Cartella Clinica: Accesso alla propria cartella clinica contenente informazioni cliniche e anamnesi.
- Visualizzazione dei Farmaci Prescritti: Consultazione dei farmaci prescritti dal medico.
- Visualizzazione delle Visite Effettuate: Accesso alla lista delle visite mediche effettuate, sia da un medico che da un operatore sanitario.

- **Operatore Sanitario**

- Aggiunta di un Nuovo Paziente come Assistito: Inserimento di un nuovo paziente nella lista degli assistiti.
- Inserimento di una Visita presso un Paziente: Registrazione dei dettagli relativi a una visita effettuata presso un paziente.
- Visualizzazione e Aggiornamento delle Visite Effettuate: Accesso alla lista delle visite mediche effettuate presso i pazienti e possibilità di aggiornarne i dati.

Queste funzionalità mirano a garantire un’esperienza utente fluida e personalizzata, adattandosi alle esigenze specifiche di ciascun ruolo all’interno del contesto sanitario.

4.4 Authentication

L'autenticazione, dato il compito delicato che il software dovrà svolgere, riveste un ruolo importante. L'autenticazione richiede l'inserimento libero di due parametri:

- Username
- Password

L'autenticazione è progettata per consentire un massimo di 15 tentativi non consecutivi. Dopo i primi 5 tentativi, l'utente è obbligato ad attendere un minuto (60 secondi), come illustrato nella Figura 107. Arrivati al 10° tentativo, l'attesa si estende a due minuti, come mostrato nella Figura 108. Infine, dopo 15 tentativi, l'applicazione termina l'esecuzione. Durante l'attesa, la tastiera viene temporaneamente disabilitata, ma l'utente può comunque utilizzare il mouse.

L'autenticazione ha lo scopo di *compartimentare* il funzionamento del software, offrendo menù specifici per ogni figura, correddati dei giusti permessi in base alla figura autenticata. Ciò è stato possibile, grazie a una attenta progettazione basata su design asset autorevoli e grazie ad un altrettanta implementazione "guidata" da check-list di sicurezza. Da una progettazione basata sul principio di "*Compartimentarizzazione degli asset*", si è passati ad un implementazione dell'autenticazione basato sulla regola di: "*Limitare la visibilità di informazioni in un programma*".

```
Inserisci username: Mario
Inserisci la password: mario1234
ACCESSO EFFETTUATO !
Menù per Medico
0. Per uscire dal programma
1. Per inserire una nuova visita medica
2. Per visualizzare le visite mediche effettuate
3. Per aggiornare una visita medica
4. Per aggiungere un nuovo paziente in cura
5. Per aggiornare la Cartella Clinica di un paziente
Digitare la scelta: |
```

Figure 104: login e menù principale del *Medico*.

```
Inserisci username: Alessandro
Inserisci la password: alessandro1234
ACCESSO EFFETTUATO !
Menù per OperatoreSanitario
0. Per uscire dal programma
1. Per inserire una visita presso un paziente
2. Per visualizzare le visite mediche effettuate
3. Per modificare una visita inserita
4. Per aggiungere un paziente come assistito
Digitare la scelta: █
```

Figure 105: login e menù principale *Operatore Sanitario*.

```
Inserisci username: Luigi
Inserisci la password: luigi1234
ACCESSO EFFETTUATO !
Menù per Paziente
0. Per uscire dal programma
1. Per visualizzare le visite mediche
2. Per visualizzare le visite fatte da un operatore sanitario
3. Per visionare la propria cartella clinica
4. Per visionare i farmaci prescritti
Digitare la scelta: █
```

Figure 106: login e menù principale del *Operatore Sanitario*.

```
ACCESSO NEGATO !
Inserisci username: ff
Inserisci la password: ff
ACCESSO NEGATO !
Inserisci username: fff
Inserisci la password: fff
ACCESSO NEGATO !
Inserisci username: fff
Inserisci la password: fff
ACCESSO NEGATO !
Hai eseguito troppi tentativi, aspetta:
Tempo rimasto: 0:50
```

Figure 107: Display del tempo di attesa dopo 5 tentativi non andati a buon fine.

```
Inserisci username: prova1
Inserisci la password: prova2
ACCESSO NEGATO !
Inserisci username: prova1
Inserisci la password: prova2
ACCESSO NEGATO !
Inserisci username: prova1
Inserisci la password: prova2
ACCESSO NEGATO !
Inserisci username: prova
Inserisci la password: prova2
ACCESSO NEGATO !
Inserisci username: prova1
Inserisci la password: prova2
ACCESSO NEGATO !
Hai eseguito troppi tentativi, aspetta:
Tempo rimasto: 1:47
```

Figure 108: Display del tempo di attesa dopo 10 tentativi non andati a buon fine.

4.5 Logging

Già in fase di progettazione (si faccia riferimento alla Figura 8), seguendo il principio: "Log user action" presente nel design Asset di Sommerville, abbiamo preso in considerazione la presenza di un sofisticato sistema di logging che aiutasse a tenere traccia di ogni

attività compiuta dall’utente. Per poter identificare in maniera puntuale e precisa ogni azione compiuta dall’utente all’interno del software, abbiamo realizzato, per ogni classe, un *metodo decorator* con lo scopo di ritornare una nuova funzione: *wrapper* che include la stampa su file di log del metodo ”decorato”.

Per una chiarezza e consapevolezza maggiore, ciascuna classe che è abilitata alle funzionalità di logging, deve implementare l’interfaccia *Ilog*(Figura: 109b) , come mostra la figura 109a.

La figura 110 mostra un’esempio di log, in grado di visualizzare ogni evento sia *off-chain*: chiamate ai metodi, errori di integrità dei dati, detection di attacchi del tipo *SQL injection* che *off-chain*: transazioni effettuate da ciascun smart contract.

```
class session(Ilog):
    def __init__(self, status = None):
        self.status = status
        self.email = None
        self.password = None
        self.utente = None
        self._emailTmp = None
        self._tentativi = self._getMaxAccessi()

    def log_actions(func):
        """Implementazione di un decoratore per il logger"""
        def wrapper(self, *args, **kwargs):
            logging.info(f'{self.__class__.__name__}: Chiamato {func.__name__}')
            return func(self, *args, **kwargs)
        return wrapper
```

(a) Metodo *decorator* per fare il logging di tutti i metodi presenti nella classe.

```
from abc import ABC, abstractmethod

class Ilog(ABC):

    @abstractmethod
    def log_actions(func):
        pass
```

(b) Interfaccia per abilitare il logging.

4.6 Gestione delle criticità

In questo paragrafo andremo ad analizzare le principali criticità incontrate, e come abbiamo deciso di risolverle.

4.6.1 Check Integrity

Il metodo `check_integrity` viene da noi utilizzato ogni volta per garantire l’integrità dei dati, quindi andare a confrontare l’hash ritornato dalla blockchain con l’hash ottenuto dal database, per accertarci che non ci siano stati problemi durante le operazioni. In Figura 111 è riportato il metodo da noi utilizzato.

```

3 2024-04-20 18:40:54,451 - INFO - solc 0.6.0 already installed at: C:\Users\lucab\.solcx\solc-v0.6.0
4 2024-04-20 18:41:00,114 - INFO - solc 0.6.0 already installed at: C:\Users\lucab\.solcx\solc-v0.6.0
5 2024-04-20 18:41:00,803 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
6 2024-04-20 18:41:01,388 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
7 2024-04-20 18:41:01,543 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
8 2024-04-20 18:41:01,779 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
9 2024-04-20 18:41:01,991 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
10 2024-04-20 18:41:02,168 - INFO - db: Chiamato ottieniCartelle , Porta: 3306
11 2024-04-20 18:41:02,559 - INFO - db: Chiamato ottieniCartelle , Porta: 3306
12 2024-04-20 18:41:02,704 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
13 2024-04-20 18:41:03,024 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
14 2024-04-20 18:41:10,464 - INFO - db: Chiamato ottieniDatiAuth , Porta: 3306
15 2024-04-20 18:41:10,828 - INFO - session: Chiamato eseguiAccesso
16 2024-04-20 18:41:15,899 - INFO - db: Chiamato gestisciAccesso , Porta: 3306
17 2024-04-20 18:41:15,900 - INFO - db: Chiamato ottieniDatiAuth , Porta: 3306
18 2024-04-20 18:41:15,984 - INFO - db: Chiamato ottieniProfessione , Porta: 3306
19 2024-04-20 18:41:15,984 - INFO - db: Chiamato ottieniDatiAuth , Porta: 3306
20 2024-04-20 18:41:16,069 - INFO - db: Chiamato ottieniCF , Porta: 3306
21 2024-04-20 18:41:16,069 - INFO - db: Chiamato ottieniDatiAuth , Porta: 3306
22 2024-04-20 18:41:16,154 - INFO - db: Chiamato ottieniDatiUtente , Porta: 3306
23 2024-04-20 18:41:16,238 - INFO - Medico: Chiamato menuMedico , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
24 2024-04-20 18:41:21,552 - INFO - Medico: Chiamato _selectPaziente , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
25 2024-04-20 18:41:21,552 - INFO - ControllerMedico: Chiamato datiPazientiCurati , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
26 2024-04-20 18:41:21,552 - INFO - ControllerMedico: Chiamato pazientiCurati , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
27 2024-04-20 18:41:21,552 - INFO - db: Chiamato ottieniCurati , Porta: 3306
28 2024-04-20 18:41:21,640 - INFO - db: Chiamato ottieniDatiUtente , Porta: 3306
29 2024-04-20 18:41:23,782 - INFO - ControllerMedico: Chiamato visualizzaRecordVisite , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
30 2024-04-20 18:41:23,782 - INFO - db: Chiamato ottieniDatiUtente , Porta: 3306
31 2024-04-20 18:41:23,904 - INFO - db: Chiamato ottieniVisitePaziente , Porta: 3306
32 2024-04-20 18:41:34,535 - INFO - Medico: Chiamato _selectPaziente , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
33 2024-04-20 18:41:34,536 - INFO - ControllerMedico: Chiamato datiPazientiCurati , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
34 2024-04-20 18:41:34,536 - INFO - ControllerMedico: Chiamato pazientiCurati , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
35 2024-04-20 18:41:34,536 - INFO - db: Chiamato ottieniCurati , Porta: 3306
36 2024-04-20 18:41:34,629 - INFO - db: Chiamato ottieniDatiUtente , Porta: 3306
37 2024-04-20 18:41:36,742 - INFO - Medico: Chiamato _updateCartellaInClinica , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
38 2024-04-20 18:41:36,745 - INFO - Medico: Chiamato _verificaCartella , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
39 2024-04-20 18:41:36,745 - INFO - ControllerMedico: Chiamato pazienteHaCartella , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
40 2024-04-20 18:41:36,745 - INFO - db: Chiamato retrieve_all_rows , Porta: 3306
41 2024-04-20 18:41:49,429 - INFO - ControllerMedico: Chiamato addFarmaco , Operatore: ('CFMedicol', 'Mario', 'Rossi', 'Oculista')
42 2024-04-20 18:41:49,429 - INFO - db: Chiamato ottieniFarmaci , Porta: 3306
43 2024-04-20 18:41:49,517 - INFO - db: Chiamato addTupla , Porta: 3306
44 2024-04-20 18:41:49,813 - INFO - db: Chiamato ottieniFarmaco , Porta: 3306
45 2024-04-20 18:41:50,000 - INFO - EVENTO BLOCKCHAIN -----> AttributeDict({'_sender': '0x174Ad9aed0018AFFE1dF0cbB4E6e4D86D102c5', '_message': 'Hash farmaco correttamente salvato'})
46 2024-04-20 18:42:10,108 - ERROR - IntegrityCheckError: Errore: Integrita' dati violata !
47 2024-04-20 18:42:10,108 - ERROR - SQLInjectionError: Errore: rilevata SQL INJECTION !

```

Figure 110: File di log: *app.log*

```

def check_integrity(self, blockchain_hash, to_check):
    hash_to_check = self.hash_row(to_check)
    return blockchain_hash == hash_to_check

```

Figure 111: Check Integrity

Il metodo `check_integrity` inizia calcolando l'hash dei dati nella tupla `to_check` utilizzando un altro metodo chiamato `hash_row`, che effettua l'hash di una tupla. Successivamente, confronta questo hash appena calcolato con l'hash fornito dalla blockchain (`blockchain_hash`). Se i due hash corrispondono, la funzione restituirà True, indicando che i dati nella tupla `to_check` sono integri rispetto all'hash della blockchain. In caso contrario, restituirà False, indicando una discrepanza o una modifica nei dati rispetto alla versione memorizzata sulla blockchain.

In ogni istanza del codice in cui viene utilizzato questo metodo, se il valore di ritorno è False, viene sollevata un'eccezione appropriata: `IntegrityCheckError`. La Figura 112 mostra la classe di errore personalizzata; si noti che ha un livello di log più elevato

rispetto a tutti gli altri log presenti nei diversi punti del codice.

```
import logging

logging.basicConfig(filename='app.log', level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

class IntegrityCheckError(Exception):
    """Eccezione personalizzata per gestire gli errori di integrità."""
    def __init__(self, messaggio):
        """Inizializzazione dell'eccezione con un messaggio specifico."""
        self.messaggio = messaggio
        logging.error(f"IntegrityCheckError: {messaggio}")
        super().__init__(self.messaggio)
```

Figure 112: Eccezione personalizzata in caso di *Integrity error*.

4.6.2 SQL Injection

Il metodo `_sqlInjectionCheck` verifica la presenza di potenziali attacchi di SQL injection nei parametri passati come argomenti. In Figura 113 è possibile visualizzare il metodo implementato.

```
def _sqlInjectionCheck(*params):
    """Controlla la presenza di potenziali SQL injection nei parametri."""
    sql_pattern = re.compile(r'\b(SELECT|INSERT|UPDATE|DELETE|FROM|WHERE|OR|AND|UNION|DROP|ALTER|EXEC)\b', re.IGNORECASE)

    for param in params:
        if re.search(sql_pattern, str(param)):
            raise SQLInjectionError("Potenziale SQL injection rilevata.")
```

Figure 113: SQL Injection

Esso controlla ogni parametro passato per la presenza di parole chiave comuni associate agli attacchi di SQL injection. Se una di queste parole chiave viene trovata in uno dei parametri, viene sollevata un'eccezione. Questo aiuta a prevenire potenziali attacchi di SQL injection proteggendo il sistema da possibili vulnerabilità. Il metodo funge quindi da *Run-time verification monitor*.

Quando viene rilevato un attacco di tipo SQL injection, viene sollevata un'eccezione appropriata, come illustrato nella Figura 114. Analogamente al caso dell'errore di integrità (si faccia riferimento al punto 4.6.1), il livello di log è impostato su *Error*, in contrasto con gli altri log presenti nel codice.

```

import logging

logging.basicConfig(filename='app.log', level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

class SQLInjectionError(Exception):
    """Eccezione personalizzata per gestire SQL INJECTION."""

    def __init__(self, messaggio):
        """Inizializzazione dell'eccezione con un messaggio specifico."""
        self.messaggio = messaggio
        logging.error("SQLInjectionError: {messaggio}")
        super().__init__(self.messaggio)

```

Figure 114: Eccezione personalizzata in caso di *SQL injection*.

4.7 Static analyzer

Nel contesto del nostro progetto, l’impiego di analizzatori statici riveste un ruolo cruciale nel garantire la qualità e l’affidabilità del codice sorgente. In particolare, ci concentreremo sull’uso di due analizzatori di rilievo: Bandit e Slyther. Questi strumenti sono stati selezionati per la loro capacità di individuare e risolvere una vasta gamma di problemi nel codice, dalla rilevazione di bug e vulnerabilità di sicurezza alla conformità alle best practice di codifica e all’ottimizzazione delle prestazioni.

4.7.1 Bandit

Bandit è uno strumento di analisi statica del codice progettato specificamente per rilevare potenziali vulnerabilità di sicurezza nel codice Python. Il suo obiettivo principale è identificare e segnalare pratiche di codifica rischiose che potrebbero essere sfruttate da attaccanti per compromettere la sicurezza di un’applicazione.

Funzionalità Principali:

- **Rilevamento di Vulnerabilità:** Bandit utilizza un insieme di regole predefinite per individuare potenziali problemi di sicurezza nel codice sorgente Python. Questi problemi possono includere vulnerabilità comuni come iniezioni di SQL, cross-site scripting (XSS), gestione inadeguata delle sessioni e molti altri.
- **Analisi Statica:** A differenza degli strumenti di analisi dinamica che richiedono l’esecuzione del codice, Bandit esamina il codice sorgente senza eseguirlo. Ciò consente di identificare le vulnerabilità potenziali durante il processo di sviluppo, riducendo il rischio di errori di sicurezza nella fase di produzione.

L'esecuzione di *Bandit* avviene automaticamente tramite uno specifico script Python che, utilizzando la libreria *subprocess*, avvia l'analisi di tutti i file .py presenti nel software, escludendo i file di test e gli script degli analizzatori.

La maggior parte delle vulnerabilità rilevate dal sistema rientra nell'identificativo 89 del catalogo *CWE (Common Weakness Enumeration)*: "Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')". Per risolvere tali problematiche, è stato inserito un *Run-time verification monitor* prima di ogni query. Per maggiori dettagli, si rimanda al punto 4.6.2.

4.7.2 Slyther

Slyther è uno strumento di analisi statica del codice progettato per l'ambiente di sviluppo della piattaforma Ethereum e Solidity. Il suo obiettivo principale è individuare potenziali problemi di sicurezza e vulnerabilità nel codice sorgente di contratti intelligenti scritti in Solidity, il linguaggio di programmazione predominante per lo sviluppo di contratti sulla blockchain Ethereum.

Funzionalità Principali:

- **Analisi dei Contratti:** Slyther si concentra sull'analisi approfondita dei contratti scritti in Solidity. Questi contratti rappresentano componenti critici all'interno di applicazioni decentralizzate (DApps) e devono essere esaminati attentamente per garantire la sicurezza e l'affidabilità della piattaforma Ethereum.
- **Rilevamento di Vulnerabilità Specifiche di Ethereum:** Slyther include un insieme di regole specifiche progettate per identificare vulnerabilità uniche riscontrate nell'ecosistema Ethereum. Queste vulnerabilità possono includere problemi come l'overflow/underflow di interi, la gestione inadeguata delle autorizzazioni e le minacce alla sicurezza dei fondi degli utenti.

4.8 Testing

Seppur tante volte ignorati dai programmati, i test sono una parte fondamentale del proprio operato per comprendere se le istruzioni utilizzate sono corrette o meno.

All'interno del nostro progetto abbiamo implementato una classe di test, *dbTest.py*, utilizzata per testare le chiamate al database.

In Figura 115 è riportata la definizione della classe di testing, con i due metodi principali:

- *setUp*: Questo metodo viene chiamato prima dell'esecuzione di ogni test all'interno della classe di test. Il suo scopo principale è inizializzare lo stato del test, preparare i

dati necessari per l'esecuzione del test e configurare l'ambiente di test. Ad esempio, è comune inizializzare oggetti o variabili, connettersi a database di test, creare oggetti simulati o impostare le condizioni iniziali necessarie per eseguire il test in modo corretto. Al suo interno viene creata un'istanza di un oggetto chiamato `db()` e lo si assegna all'attributo `database` dell'istanza della classe di test; inoltre viene anche creato un oggetto `MagicMock`, un tipo di oggetto fittizio che può simulare il comportamento di un oggetto reale e può essere utilizzato per sostituire oggetti reali durante i test.

- `tearDown`: Questo metodo viene chiamato dopo l'esecuzione di ogni test all'interno della classe di test. Il suo compito è pulire e ripristinare lo stato dell'ambiente di test dopo l'esecuzione del test. Questo può includere la chiusura di connessioni di database, il rilascio di risorse allocate durante l'esecuzione del test, la rimozione di oggetti creati durante il test o il ripristino dello stato pre-test. L'obiettivo è garantire che ogni test si esegua in un ambiente pulito e prevedibile, indipendentemente dall'esito del test precedente.

```
from faker import Faker

# Inizializza l'istanza di Faker
faker = Faker()

import unittest
from unittest.mock import MagicMock # Per evitare l'effettiva connessione al db
from database.db import db

Alessandra D'Anna, 5 giorni fa | 1 author (Alessandra D'Anna)
class TestDatabase(unittest.TestCase):

    def setUp(self):
        self.database = db()
        self.db.conn = MagicMock()

    def tearDown(self):
        self.db.conn.close.assert_called_once()
```

Figure 115: Classe di test database

```

def test_retrieve_data(self):
    # Verifica se i metodi di recupero dei dati restituiscono risultati corretti
    mock_data = [
        {'CF': '123456789', 'Username': 'user1', 'Password': 'password1', 'Ruolo': 'ruolo1'},
        {'CF': '987654321', 'Username': 'user2', 'Password': 'password2', 'Ruolo': 'ruolo2'}
    ]

    self.db.ottieniDatiAuth = MagicMock(return_value=mock_data)

    result = self.db.ottieniDatiAuth()
    self.assertEqual(result, mock_data)

```

Figure 116: Test retrieve data

Il metodo in Figura 117 verifica se i metodi *addNuovoPaziente*, *modificaDosaggiofarmaco* ed *eliminaVisitaOS* dell’oggetto db() funzionano correttamente, controllando se le operazioni di aggiunta, modifica ed eliminazione dei dati vengono eseguite come previsto.

```

def test_modify_data(self):
    # Verifica se i metodi di aggiunta/modifica/eliminazione dei dati funzionano correttamente

    # Aggiunta di un nuovo paziente di mock
    num_rows_inserted = self.db.addNuovoPaziente('CF123', 'Nome', 'Cognome', 'Residenza')
    self.assertEqual(num_rows_inserted, 1) # Verifica se è stata inserita una nuova riga

    # Modifica del dosaggio di un farmaco
    success = self.db.modificaDosaggiofarmaco('CF123', 'NomeFarmaco', 'NuovoDosaggio')
    self.assertTrue(success) # Verifica se la modifica è avvenuta con successo

    # Eliminazione di una visita
    visita = ('CFPaziente', 'CFOperatoreSanitario', 'Altro', 'DataOra')
    self.db.eliminaVisitaOS = MagicMock()
    self.db.eliminaVisitaOS(visita)
    self.db.eliminaVisitaOS.assert_called_once_with(visita)

```

Figure 117: Test modify data

Analizziamo ora il test riportato in Figura 118. Il test simula un errore di connessione al database chiudendo la connessione al database utilizzando `self.db.conn.close()`. Questo crea un’eccezione poiché il metodo `ottieniDatiAuth` presumibilmente dipende da una connessione attiva al database.

Successivamente, viene utilizzata una struttura `with` insieme al metodo `assertRaises` per verificare se viene sollevata un’eccezione durante l’esecuzione del metodo `ottiendiDatiAuth`. In questo caso, si verifica se viene sollevata qualsiasi eccezione di tipo `Exception`. Se il metodo `ottiendiDatiAuth` gestisce correttamente l’eccezione di connessione chiusa, il test passerà. Se il metodo non gestisce correttamente l’eccezione o non la solleva, il test fallirà. In entrambi i casi, ciò indica che il comportamento di gestione delle eccezioni del metodo viene verificato correttamente.

```
def test_exception_handling(self):
    # Verifica se le eccezioni vengono gestite correttamente

    # Simula un errore di connessione al database
    self.db.conn.close() # Chiudi la connessione per simulare un errore di connessione
    with self.assertRaises(Exception): # Verifica se viene sollevata un'eccezione
        self.db.ottiendiDatiAuth()

if __name__ == '__main__':
    unittest.main()
```

Figure 118: Test eccezione

4.8.1 Solidity contract test

Abbiamo eseguito una serie di test utilizzando il framework *Truffle* per verificare il corretto funzionamento dei nostri contratti Solidity. I test sono stati sviluppati per garantire che le operazioni di memorizzazione e recupero degli hash nelle nostre classi di contratto avvengano correttamente e che i dati memorizzati siano consistenti con quelli recuperati.

Per quanto riguarda la classe `MedicoContract`, abbiamo testato quattro funzionalità principali:

- Memorizzazione e recupero dell’hash della visita medica.
- Memorizzazione e recupero dell’hash della cartella clinica del paziente.
- Memorizzazione e recupero dell’hash del farmaco prescritto.
- Memorizzazione e recupero dell’hash delle patologie del paziente.

Ogni test è stato progettato per eseguire le seguenti operazioni:

```

    ][H]
it("should store and retrieve visita hash correctly", async () => {
  const medico = accounts[0];
  const paziente = accounts[1];
  const hashDati = "hashDatiVisita00000000000000000000";

  await medicoContractInstance.storeHashVisita(medico, paziente, hashDati);
  const retrievedHashes = await medicoContractInstance.retrieveHashVisita(medico, paziente);

  assert.equal(retrievedHashes.length, 1, "Il numero di hash memorizzati non è corretto");
  assert.equal(retrievedHashes[0], hashDati, "L'hash recuperato non corrisponde all'hash memorizzato");
});

```

Figure 119: test visita medica

- Memorizzazione di un hash utilizzando una specifica chiave di ricerca.
- Recupero dell'hash memorizzato e confronto con l'hash originale.

Per esempio, nel test per la memorizzazione e il recupero dell'hash della visita medica 119, abbiamo simulato il processo di memorizzazione dell'hash della visita medica tra un medico e un paziente. Successivamente, abbiamo verificato che l'hash memorizzato corrispondesse all'hash originale e che il numero di hash memorizzati fosse corretto. Analogamente, abbiamo eseguito test simili per le classi OSContract e PazienteContract, garantendo che le operazioni di memorizzazione e recupero degli hash avvengano correttamente anche in questi contesti.

I test sono stati eseguiti con successo, confermando il corretto funzionamento dei nostri contratti Solidity e garantendo che le operazioni di memorizzazione e recupero degli hash siano affidabili e consistenti.