Ad esempio, per valutare l'impatto in termini di riservatezza, integrità e disponibilità delle informazioni, ci si sofferma sugli aspetti economico/finanziario, operativo, reputazionale e legale (compliance).

Le fasi che costituiscono la gestione del rischio effettuato con Cyber Risk Management di AgID sono le seguenti:

1) Analisi del contesto. Vengono identificati i servizi erogati e i servizi trasversali utilizzati in ambito pubblica amministrazione. Di ogni servizio viene descritto un profilo di criticità.
2) Valutazione di ciascun servizio erogato e da ciascun servizio trasversale in termini d'impatto su riservatezza, integrità e disponibilità delle informazioni trattate.
3) Calcolo del rischio attuale, sulla base dei valori di probabilità di accadimento e d'impatto, per ogni minaccia identificata. La fase di Risk Assessment prevede anche l'identificazione delle contromisure da implementare per un'efficace mitigazione del rischio.
4) Applicazione delle contromisure previste dal piano di trattamento del rischio, volte a mitigare, accettare o trasferire i rischi individuati.
5) Analisi del rischio residuo, cioè la valutazione del rischio che permane, nonostante l'applicazione del piano di trattamento del rischio.
6) Fase di monitoraggio dell'intero processo, con eventuale adeguamento in seguito a modifiche del contesto o in presenza di nuove minacce alla sicurezza delle informazioni.

Il tool AGID di Risk Management è gratuito ed a completa disposizione di tutte le Pubbliche Amministrazioni: www.sicurezzait.gov.it

6.3 Requisiti

La fase di analisi e specifica dei requisiti è fondamentale nel ciclo di vita dello sviluppo software.

Di seguito si riportano i linguaggi e gli strumenti utili alla fase di definizione dei requisiti di sicurezza del software.

6.3.1 Linguaggi per la specifica dei requisiti

Un linguaggio di specifica in ambito sicurezza può essere considerato:

- un linguaggio di specifica software utilizzato per indicare gli attacchi (AsmL e UML state charts),
- l'estensione di un linguaggio di specifica software utilizzato per rappresentare gli attacchi (Misuse Cases , Abuse Cases, AsmLSec e UMLintr) e i requisiti di sicurezza (UMLsec, SecureUML, Secure Tropos e Misuse Cases),
- un linguaggio per la specifica degli attacchi (*attack specification language), per esempio* STATL e Snort Rules.

UMLsec[17] è un'estensione di UML per lo sviluppo di sistemi sicuri e usa stereotype, tag e constraint per specificare i requisiti di sicurezza. Gli stereotype servono come etichette per gli elementi del modello UML allo scopo di introdurre informazioni al modello e specificare i vincoli che devono essere soddisfatti da questo. I tag sono associati con gli stereotype e sono utilizzati per specificare in modo esplicito una

[17] https://en.wikipedia.org/wiki/UMLsec

semplice proprietà di un elemento del modello. UMLsec definisce 21 stereotype da utilizzare per rappresentare i seguenti requisiti di sicurezza:

- fair exchange (la necessità di uno scambio leale),
- non-repudiation (un'azione non si può negare),
- role-based access control,
- secure communication link,
- confidentiality,
- integrity,
- authenticity,
- freshness of a message (ad esempio nonce),
- secure information flow among components,
- guarded access (uso di protezioni per imporre il controllo di accesso).

Sette di questi stereotype hanno dei tag associati e nove hanno vincoli. Gli stereotype possono essere utilizzati per i diagrammi dei casi d'uso, i diagrammi delle classi, diagrammi di stato, diagrammi di attività, diagrammi di sequenza, i diagrammi e le implementazioni per specificare i requisiti di sicurezza in un modello UML (per le specifiche relative sia ai requisiti, sia al design). Un insieme di **tools** sono stati rilasciati per la modellazione attraverso l'impiego di UMLsec e per la verifica dei modelli così realizzati (utilizzando il model checking).

SecureUML[18] SecureUML è un'altra estensione di UML che si concentra sulle politiche di controllo degli accessi ad un modello basato sui ruoli. Queste politiche possono essere considerate come requisiti di sicurezza. SecureUML propone nove stereotype che possono essere utilizzati per annotare un diagramma delle classi, con informazioni di controllo di accesso basato sui ruoli. SecureUML utilizza l'oggetto Constraint Language (OCL) per specificare i vincoli, le azioni e le autorizzazioni per le risorse. Contrariamente a UMLsec, questi vincoli possono essere specificati in base alle esigenze del singolo componente software.

Snort Rules[19] è un network intrusion detection system (IDS) ampiamente utilizzato. Esso utilizza scenari di attacchi specificati come regole per rilevare gli attacchi attraverso la rete. Una snort rule specifica quale azione deve essere intrapresa se la regola è associata a un pacchetto di rete, gli indirizzi IP di origine e destinazione e le porte, il protocollo della rete osservato, e la direzione del pacchetto di rete. Un certo numero di opzioni possono anche essere specificate. Queste opzioni vanno dalla registrazione di un messaggio alla ricerca di una particolare stringa nel pacchetto.

Secure Tropos[20] può essere utilizzato per lo sviluppo di software sicuro ed è un'estensione della metodologia di sviluppo Tropos. Secure Tropos utilizza le nozioni di *actor* (person(s), organization(s), software), *goal* (obiettivi che gli attori vogliono ottenere), *soft goal* (un obiettivo la cui realizzazione non può essere determinata in modo esplicito), *task* (un compito per raggiungere un obiettivo), *resource* (fisica o dati), *security constraint* (specificato come le dichiarazioni di alto livello), *secure goal* (utilizzato per soddisfare un vincolo di sicurezza), *secure task* (un compito per raggiungere un obiettivo di sicurezza), *secure resource* (una risorsa che è connessa a *security constraints*, *secure goal*, *secure task*, oppure a un'altra *secure resource*). Un *actor* può dipendere da un altro *actor* per raggiungere un *goal/soft goal*, per

[18] https://ieeexplore.ieee.org/document/6997358

[19] https://www.snort.org/downloads

[20] http://www.troposproject.eu/node/301

svolgere un *task*, o rilasciare una risorsa. La notazione SecureTropos può essere utilizzato per rappresentare vincoli di sicurezza (requisiti) sulle interazioni tra gli attori durante la fase di specifica dei requisiti.

Misuse Cases[21] è una tipologia di Use Case UML utilizzata per descrivere comportamenti indesiderati del software. Un *misuse case* è avviato da un particolare tipo di attore chiamato *mis-actor* (ad esempio, l'attore con intenti malevoli). *Misuse cases* e *mis-actors* possono essere utilizzati per suscitare più casi d'uso per neutralizzare le minacce poste dai casi di uso improprio. *Misuse cases* e *mis-actors* sono rappresentati in colore nero pieno per distinguerli dai casi d'uso e dagli attori UML. Due relazioni speciali chiamati "prevents" e " detects" mettono in relazione *use cases* e *misuse cases*. Il processo può essere utilizzato in modo graduale per sviluppare un diagramma dei casi d'uso (compresi i *misuse cases*) oppure, se necessario, può essere utilizzato anche in modo iterativo. Secondo tale processo, dovrebbero essere specificati prima gli *use cases* e poi i *misuse cases*. Dopo di che, devono essere identificate le relazioni potenziali tra gli *use cases* e i *misuse cases* perché spesso la funzionalità del software viene utilizzata per attaccarlo. Infine, i nuovi *use case* devono essere specificati per individuare o prevenire i *misuse cases*. Questi nuovi use case costituiscono i requisiti di sicurezza di alto livello del software e sono chiamati come " security use cases".

Abuse Cases[22] Un altro modo per specificare il comportamento indesiderato di un pezzo di software utilizzando i diagrammi UML è di sviluppare un *abuse case model*. Un *abuse case model* specifica le interazioni pericolose usando attori e *abuse case*. Non c'è differenza di notazione tra i componenti di un *UML use case diagram* e un *abuse case model*. Si raccomanda l'utilizzo di una struttura ad albero per gli approcci multipli. Questo aggiunge ulteriori dettagli al modello e permette di identificare tutte le possibili misure di sicurezza. Dettagli sugli attori come le loro risorse, le competenze, e l'obiettivo dovrebbero essere inclusi come testo. Gli *abuse case model* possono essere utilizzati nelle fasi di progettazione e collaudo.

UMLintr[23] è un'estensione di UML che utilizza stereotype e tag per specificare intrusioni (attacchi) utilizzando use case diagrams, class diagrams, state charts, package diagrams. Gli attacchi vengono divisi in quattro tipologie diverse. Ogni tipo è rappresentato come un pacchetto fornito di stereotype. Ci sono tre stereotype definiti per le classi e dodici per lo use case diagram. Gli stereotype per le classi hanno anche i tag.

Abstract State Machine Language (AsmL)[24] ASML è un linguaggio a stati finiti machine-based eseguibile utilizzato anche per specificare scenari di attacco. In generale, in ASML possono essere specificati attacchi con step multipli. Tali scenari di attacco possono essere tradotti automaticamente in *Snort rules* che possono poi essere utilizzati con un'estensione di IDS Snort; sono altresì in grado di catturare più attacchi con step multipli, utilizzando le informazioni di contesto. Le Snort rules, l'input standard di Snort, non possono rappresentare attacchi con step multipli.

AsmLSec[25] è un'estensione di ASML sviluppata per specificare scenari di attacco. AsmLSec utilizza stati, eventi e transizioni per rappresentare gli attacchi. Ogni transizione ha un'origine e uno stato di destinazione, una serie di condizioni da soddisfare e le azioni da compiere. Gli scenari di attacco rappresentati in AsmLSec possono essere tradotti automaticamente in ASML attraverso un compilatore appositamente sviluppato. E' stato sviluppato un IDS che prende in input gli scenari di attacco tradotti.

---

[21] https://en.wikipedia.org/wiki/Misuse_case

[22] https://en.wikipedia.org/wiki/Abuse_case

[23] https://ieeexplore.ieee.org/document/1607377

[24] https://www.microsoft.com/en-us/research/project/asml-abstract-state-machine-language/

[25] https://ieeexplore.ieee.org/document/4159874