Agenzia per l'Italia Digitale
Presidenza del Consiglio dei Ministri

LINEE GUIDA PER L'ADOZIONE DI UN CICLO DI SVILUPPO DI SOFTWARE SICURO

SOMMARIO

LISTA DELLE TABELLE

LISTA DELLE FIGURE

1 INTRODUZIONE

1.1 Scopo

Scopo del presente documento è fornire le linee guida per intraprendere un processo di sviluppo del software "sicuro", applicabile attraverso l'identificazione e l'implementazione di opportune azioni di sicurezza nel corso di tutte le fasi del ciclo di sviluppo software.

1.2 Ambito di Applicabilità

Il presente documento si applica al contesto tecnologico dell'Agenzia per l'Italia Digitale (in seguito AgID) nell'ambito dei servizi previsti dal Contratto Esecutivo in [DA-7].

1.3 Struttura del documento

Il presente documento è articolato come di seguito :

- **Esigenze e Ambiti di Applicazione**, come nasce l'esigenza dello sviluppo di software sicuro.
- **Analisi delle iniziative e degli standard**, analizza lo scenario nazionale e globale fornendo una vista delle iniziative, degli standard e dei risultati prodotti in termini di metodologie, raccomandazioni, modelli e Tool. L'analisi dello scenario ha consentito la creazione del *Catalogo dei Security Tools*.
- **Secure Software Development Life Cycle (SSDLC): Analisi delle metodologie e dei Processi.** Analizza i diversi metodi e modelli SDLC esistenti, con l'obiettivo di identificare le caratteristiche che rendono un ciclo di sviluppo software sicuro ed efficace.
- **La sicurezza in tutte le fasi del ciclo di sviluppo software**, è un approfondimento sulle fasi del SDLC dato che, tradizionalmente, gli aspetti di sicurezza non vengono considerati con sufficiente attenzione fin dall'inizio del SDLC.
- **Training e formazione**. Focalizza l'attenzione sul fatto che molti degli attuali problemi di sicurezza derivano da errori di progettazione o di implementazione, risolvibili solo disponendo di personale qualificato.
- **Certificazioni professionali**, è un elenco delle principali certificazioni riconosciute in ambito InfoSec.

2 RIFERIMENTI

2.1 Documenti di Riferimento

Rif.	Codice	Titolo
DR-1.	--	Reg. (UE) 679/2016 "Regolamento generale sulla protezione dei dati" del 27/04/2017
DR-2.	--	ISO/IEC 29100:2011 "Privacy Framework", 15/12/2011
DR-3.	--	ISO/IEC 12207:2008 "Software life cycle processes", 01/02/2008
DR-4.	--	ENISA "Privacy and Data Protection by Design – from policy to engineering", 12/2014
DR-5.	--	Ann Cavoukian "Privacy by Design – The 7 Foundational Principles", Information & Privacy Commissioner, Ontario, Canada, 01/2011
DR-6.	--	ISO/IEC 29134:2017 "Guidelines for privacy impact assessment", 01/06/2017
DR-7.	--	ISO/IEC 29151:2017 "Code of practice for personally identifiable information protection", 01/08/2017
DR-8.	--	NIST Special Publication 800-53A r4 Appendix J, "Privacy Control Catalog - Privacy controls, enhancements, and supplemental guidance"
DR-9.	--	MITRE Privacy Engineering Framework, MITRE Privacy Community of Practice (CoP), 18/07/2014
DR-10.	--	WP ART29 "Guidelines on Data Protection Impact Assessment (DPIA) and determining whether processing is "likely to result in a high risk" for the purposes of Regulation 2016/679", 04/10/2017
DR-11.	--	32nd International Conference of Data Protection and Privacy Commissioners. "Privacy by design Resolution", Jerusalem, Israele, 10/2010

Tabella 1 - Documenti di Riferimento

3 DEFINIZIONI E ACRONIMI

3.1 Definizioni

Vocabolo	Titolo
Applicazione Cloud	Applicazione sviluppata sfruttando la tecnologia Cloud Computing
Defence in Depth	Difesa a differenti livelli-Layered Defense.
Hardening	processo che mira attraverso operazioni di configurazione specifica di un dato sistema e dei suoi componenti, a minimizzare l'impatto di possibili vulnerabilità, migliorandone quindi la sicurezza complessiva
Fornitore	Vedi Raggruppamento
Raggruppamento	Raggruppamento Temporaneo di Impresa Leonardo Divisione Sistemi per la Sicurezza S.p.A. (nel seguito Leonardo), società mandataria, IBM S.p.A. (mandante), Sistemi Informativi S.p.A. (mandante) e Fastweb S.p.A. (mandante).

Tabella 2 - Definizioni

3.2 Acronimi

Codice	Titolo
Amministrazione	AgID
AgID	Agenzia per l'Italia Digitale
APP	Atom Publishing Protocol
AsmL	Abstract State Machine Language
BRP	Business Risk Profile
CAPEC	Common Attack Pattern Enumeration and Classification
CC	Common Criteria
CCRA	Common Criteria Recognition Agreement
CE	Contratto Esecutivo
CERT	Computer Emergency Response Team
CQ	Contratto Quadro
CWE	Common Weakness Enumeration
DAST	Dynamic Application Security Testing
DiDI	Defense-in-Depth Index
ESA	European Space Agency
IACS	Automation and Control Systems
IASP	Instrumented application security testing
IFSQ	Institute for Software Quality
IDS	Intrusion Detection System
IFSQ	Institute for Software Quality

Codice	Titolo
ISA	International Society of Automation
I&O	IT manager
MISRA	Motor Industry Software Reliability Association
MSAT	Microsoft Security Assessment Tool
NESSI	Networked European Software and Services Initiative
OWASP	Open Web Application Security Project
PII	Personal Identification Information
RASP	Runtime application security testing
RTI	Raggruppamento Temporaneo di Impresa
SAFE CODE	Software Assurance Forum for Excellence in Code
SAMATE	Software Assurance Metrics and Tool Evaluation
SAMML	Software Assurance Maturity Model
SANSI	SANS Software Security Institute
SAST	Static Application Security Testing
SCA	Software composition analysis
SDLC	Software Development Life Cycle
SSA	Software Security Assessment
SSE	Secure Software Engineering
SSDLC	Secure Software Development Life Cycle
S&R	Security & Risk
SW	Software
WASC	Web Application Security Consortium

Tabella 3 - Acronimi

4 ESIGENZE E AMBITI DI APPLICAZIONE

4.1 Il panorama delle vulnerabilità applicative

Il panorama delle minacce per la sicurezza delle applicazioni è in costante evoluzione.

Secondo la fonte Gartner[1], negli ultimi tempi ***OLTRE IL 75% DEGLI ATTACCHI SONO STATI INDIRIZZATI DIRETTAMENTE VERSO LE APPLICAZIONI***.

I fattori chiave di questa evoluzione sono i progressi fatti dagli attaccanti, il rilascio di nuove tecnologie, l'uso di sistemi sempre più complessi.

Gli obiettivi degli attacchi sono le vulnerabilità che si celano all'interno delle applicazioni software che forniscono un facile percorso d'ingresso per compromettere i sistemi o lanciare ulteriori attacchi e malware. Tra le cause c'è anche il fatto che fino ad ora si è seguito un approccio concentrato soprattutto sulla correzione delle difettosità funzionali e sulle performance delle logiche applicative, trascurando l'attuazione di pratiche di progettazione e programmazione che garantiscono la sicurezza del codice.

Da qui anche l'appello della comunità OWASP[2] che sottolinea la necessità di accrescere la consapevolezza sulla sicurezza delle applicazioni in quanto il SW non sicuro mette a repentaglio le infrastrutture anche più critiche (finanziarie, sanitarie e difensive).

Per rispondere in modo efficace alle sfide di sicurezza delle applicazioni, è necessario dotarsi di soluzioni adeguate per:

- Migliorare la gestione del programma di sicurezza delle applicazioni. Le componenti chiave di un programma di sicurezza devono includere:
  - Risk Management Integration,
  - Architect & Developer Guidance,
  - Process Improvement (SDLC),
  - Secure Development Activities,
  - Vulnerability Management Integration;
- Valutare il codice software e le applicazioni al fine di identificare le vulnerabilità;
- Automatizzare la correlazione dei risultati della verifica della sicurezza per applicazioni interattive, statiche e dinamiche.

4.2 Sviluppo applicazioni sicure

La sicurezza informatica è l'insieme delle tecniche che mirano a proteggere l'ambiente informatico che include gli utenti, le reti, le applicazioni, i processi, e i dati; è una sicurezza "integrata" che implica una visione della security a 360° e il cui obiettivo principale è di ridurre i rischi, compresa la prevenzione o mitigazione degli attacchi informatici.

Le applicazioni software dovrebbero avere caratteristiche di sicurezza base di default (***Secure By Default***) quali ad esempio, l'abilitazione automatica di meccanismi di costruzione di password complesse piuttosto che procedure di rinnovo delle stesse secondo una scadenza di natura temporale.

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

[2] A free and open software security community (https://www.owasp.org)

Allo scopo di proteggere un sistema informativo, è pertanto necessario che ogni sua componente disponga di un proprio meccanismo di protezione. La costruzione di strati multipli di controlli di sicurezza posti lungo un sistema è definita ***Defence in Depth***.

La Defense-in-Depth è l'approccio alla sicurezza delle informazioni che prevede il raggiungimento di una adeguato livello di sicurezza attraverso l'utilizzo coordinato e combinato di molteplici contromisure.

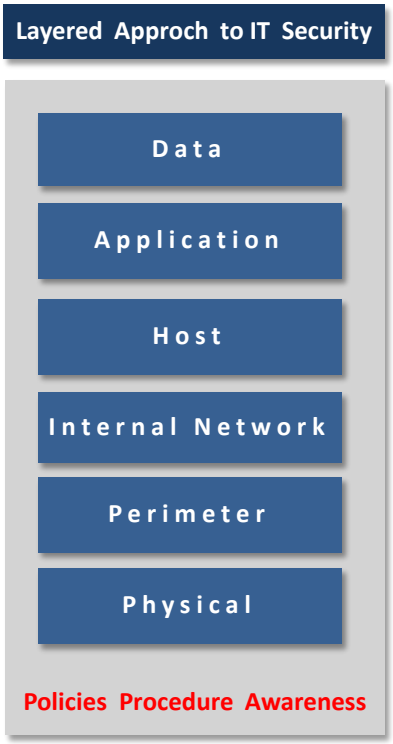


Figura 1 - Defence-in-Depth model for IT

Questa strategia difensiva si fonda sull'integrazione di differenti categorie di elementi: persone, tecnologie e modalità operative. La ridondanza e la distribuzione delle contromisure possono essere sintetizzate in una "difesa a differenti livelli" ("Layered Defenses"). Il concetto è di derivazione militare e si basa sull'assunto che nel caso in cui un attacco abbia successo, a causa del fallimento di un meccanismo di sicurezza, altri meccanismi di sicurezza possono intervenire per consentire un'adeguata protezione dell'intero Sistema.

Diverse sono le iniziative che si sono incentrate sulle problematiche Secure Development promuovendo azioni di sensibilizzazione (indirizzate ad aziende e community di sviluppatori) quali:

- la diffusione delle fondamentali best practices in materia di sicurezza applicativa (le prime tra tutte riconducibili ad una buona ingegnerizzazione del software);
- una piena comprensione delle minacce più comuni (compresi i difetti propri dei linguaggi di programmazione);
- ancora più importante, una considerazione della problematica fin dalle prime fasi del ciclo di sviluppo.

L'adozione di un Secure Software Development Life Cycle (SSDLC) atto a considerare ed implementare opportune attività di sicurezza nel corso di tutte le sue fasi del ciclo di vita del SW, dalla analisi alla progettazione, sviluppo, test fino alla manutenzione è una necessità inderogabile per rispondere alla domanda di sicurezza e per ridurre i costi che comporta trascurarla.

4.3 Security Tools

Nell'ambito della cybersecurity, **Forrester** nel suo report "**Five steps to reinforce and harden application security**"[3] sottolinea la necessità di cooperazione tra i team Security & Risk (S&R) e gli IT manager (I&O), ribadendo più volte come i primi non siano in grado, da soli, di coprire tutte le vulnerabilità scaturite dalle nuove esigenze in ambiti IT e digital business. Dal punto di vista dell'analista, infatti, l'IT team deve adottare, attraverso opportuni meccanismi di automazione e integrazione, le **security practices** all'interno di una '**continuous delivery pipeline**'. Questo garantisce una maggiore visibilità nelle interaizoni tra hardware, software, servizi web e customer data. I professionisti I&O hanno quindi l'obiettivo di creare un ambiente di sicurezza 'responsive'.

A tal fine, Forrester propone 5 step per costruire un **responsive security environment**:

[3] https://www.forrester.com/report/Five+Steps+To+Reinforce+And+Harden+Application+Security/-/E-RES127875

Step 1: rimuovere le 'inconsistenze' e creare un 'conto' dei materiali

Innanzitutto è necessario eliminare tutte le problematiche di sicurezza spesso derivanti da vulnerabilità riconducibili a servizi non più utilizzati e non più mantenuti o una cattiva gestione degli accessi e delle autorizzazioni. Tale attività deve essere svolta attraverso la collaborazione tra i team dedicati (I&O e S&R). In aggiunta, il censimento delle componenti applicative (attraverso un approccio 'application modeling') consente di ottenere ulteriori benefici in termini di: riduzione del mean-time-to-repair (attraverso l'impiego di strumenti di gestione della configurazione a sostegno del processo di monitoraggio delle modifiche applicative e dell'infrastruttura a supporto); utilizzo limitato di software per l'analisi delle vulnerabilità di terze parti (la visione completa dell'applicazione e di come interagisce con gli altri sistemi esistenti consente di limitare l'uso di ulteriori strumenti); rapida rimozione dei 'difetti' che possono generare vulnerabilità.

Step 2: limitare e rinforzare l'accesso ai sistemi e ai network device; monitorare i cambiamenti

Generalmente l'accesso non autorizzato ai dati consegue essenzialmente da vulnerabilità derivanti da un hardening non adeguato, da problematiche di sicurezza nel software/hardware e/o da una cattiva progettazione del sistema stesso; consentendo l'accesso intenzionale, non autorizzato, ai dati presenti all'interno della propria organizzazione. E' necessario lavorare a livello infrastrutturale per bloccare tutti gli accessi non autorizzati monitorando costantemente network e traffico sui sistemi. I team di infrastruttura e quelli della sicurezza dovrebbero cooperare nel processo di identificazione delle policy e dei tool per il monitoraggio, delle applicazioni in particolare, per verificare in tempo reale eventuali cambiamenti prima che questi si traducano in vulnerabilità.

Step 3: assistere i team di Security&Risk sul fronte intrusion detection & response

E' richiesto l'impiego di sistemi infrastrutturali e tool tecnologici a supporto delle politiche di sicurezza. Questi svolgono un ruolo determinante nella prevenzione (e nella risposta) delle intrusioni in quanto, a fronte di anomalie (legate ad esempio all'utilizzo delle Cpu o al numero delle transazioni di sistema), avendo il controllo di tutto lo stack tecnologico, riescono a fornire in tempo utile alert ed informazioni al team di sicurezza. Un sistema di controllo di questo tipo, accelera il mean-time-to-detection (il tempo di localizzazione di una vulnerabilità o di un attacco) e il tempo di risposta. Inoltre, cosa molto importante, riduce il range dei falsi allarmi di sicurezza (grazie ai controlli incrociati tra i team di infrastruttura e i team della sicurezza).

Step 4: 'loggare' quanto più possibile

E' estremamente importante l'attività di tracciamento e di monitoraggio in tutte le fasi del ciclo di vita di sviluppo dell'applicazione. L'obiettivo è di analizzare tutte le fonti dati nonché il materiale di ciascuna applicazione, e monitorarne ogni minimo cambiamento. A tal fine, dal punto di vista tecnologico, Forrester suggerisce:

i) l'integrazione degli Application Release Automation tool nei processi di auditing;

ii) ii) adottare sistemi di Automate Change Tracking e dashboard a supporto dei team di I&O e S&R.

Step 5: creare uno stack di application security tool

Le azioni precedenti concorrono alla creazione di un vero e proprio stack tecnologico incentrato sulla sicurezza applicativa. Al fine di indirizzare correttamente una protezione efficace delle applicazioni, è di fondamentale importanza individuare le vulnerabilità (e porvi rimedio) sin dalle prime fasi del ciclo di vita dello sviluppo, quando è ancora poco costoso e poco rischioso intervenire.

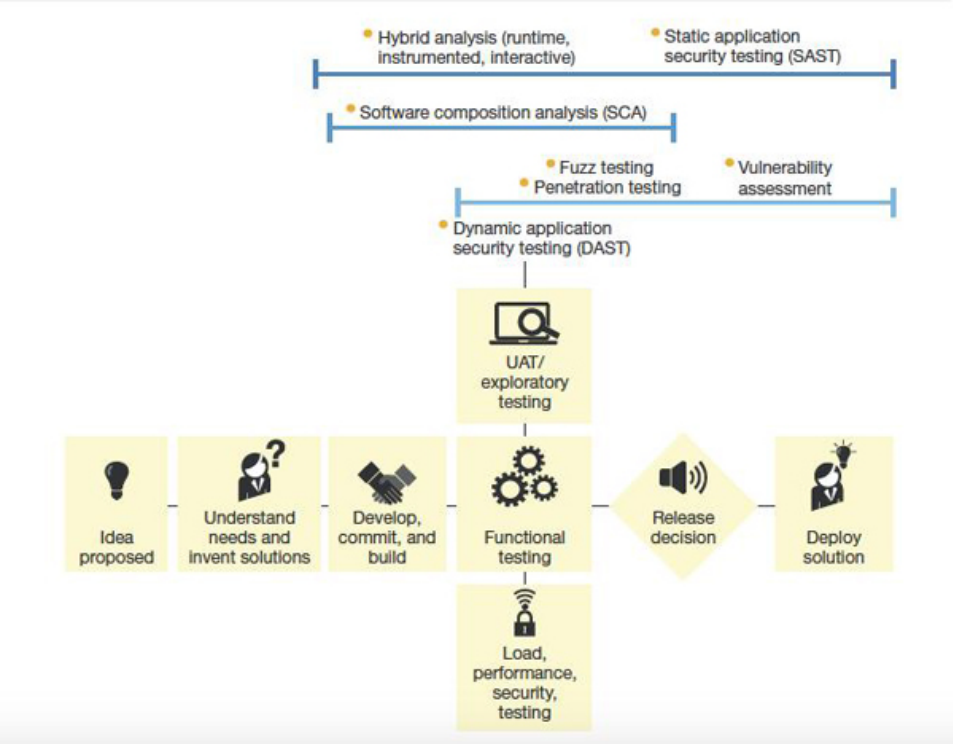


Figura 2 - Augment the life cycle with security tools

[Fonte: Forrester, Five Steps To Reinforce And Harden Application Security]

Per comporre lo stack, queste le tecnologie cui gli I&O professional dovrebbero porre attenzione:

- **Static Application Security Testing (SAST)**, tool che esaminano il codice binario e il codice di programmazione delle applicazioni senza 'mandare in esecuzione' l'applicazione (ossia senza la necessità di farla girare sui sistemi nei processi di testing);

- **Software composition analysis (SCA) tool**, tecnologie che consentono di analizzare le building block applicative per scovare vulnerabilità all'interno, per esempio, delle librerie, dei componenti open source o dei vari 'blocchi' di software che compongono l'applicazione.

- **Dynamic Application Security Testing (DAST)**, sistemi che permettono di osservare in dettaglio come si comporta l'applicazione quando è in funzione per scovarne imperfezioni o vulnerabilità prima che si prosegua con lo step di sviluppo successivo;

- **Fuzz testing tool**, sistemi che analizzano le vulnerabilità sul fronte di protocolli network, application data e input location (sempre durante i cicli di testing applicativo);

- **Hybrid analysis tool**, si tratta di tecnologie di testing per la sicurezza delle applicazioni che integrano funzionalità di Instrumented application security testing (Iast) e Runtime application security testing (Rasp) utili per ridurre i falsi positivi e i falsi negativi generalmente evidenziati dai sistemi Dast;

- **Vulnerability assessment tool**, sistemi utili a rendere visibili eventuali criticità a livello di sistema operativo, configurazione dei sistemi, micro-configurazioni dei server e delle altre architetture con cui l'applicazione in sviluppo dovrà interagire una volta messa in produzione;

- **Penetration testing tool**, tecnologie utili a 'validare' l'assessment delle vulnerabilità perché mostrano come potrebbero avvenire gli attacchi simulando la penetrazione nei sistemi e nelle applicazioni.

5 ANALISI DELLE INIZIATIVE E DEGLI STANDARD

5.1 Iniziative Internazionali

5.1.1 Open Web Application Security Project (OWASP)

L'Open Web Application Security Project (chiamato semplicemente OWASP) è un progetto open-source per la sicurezza delle applicazioni Web. L'OWASP offre anche guide con consigli sulla creazione di applicazioni Internet sicure, e indicazioni per i test a cui andrebbero sottoposte. È stato anche pubblicato un WebGoat, un progetto che insegna la sicurezza sulle applicazioni web. Nel 2004 fu istituita una fondazione no-profit che supporta l'OWASP, che persegue l'obiettivo di aumentare la sicurezza delle applicazioni consentendo di prendere le decisioni in base ai rischi. In Europa è un'organizzazione no-profit registrata a partire da giugno 2011 ed è presente anche in Italia.

URL	**http://www.owasp.org**
Country of HQ location	**US**
Geographic Scope	**International**
Type	**Various Industry (not for profit)**

L'iniziativa è organizzata come una comunità collaborativa che produce tools e documenti in tre aree principali:

- Protection,
- Detection,
- Life-cycle security.

Relativamente a queste 3 aree, OWASP ha prodotto un insieme di guide sulle buone pratiche quali: OWASP Testing Guide, OWASP Code Review, Software Assurance Maturity Model.

Tra le uscite anche il Report ' OWASP Top 10' sui rischi per le applicazioni web. Le altre attività OWASP.

Risultati più rilevanti:

Good Practice	[Protection Area] OWASP Secure Coding Practices Quick Reference Guide v2.0 - A technology-agnostic set of general software security coding practices, in a comprehensive checklist format, that can be integrated into the development life cycle. [Protection Area] OWASP Developers Guide v2.0 (2005) - An extensive document covering all aspects of web application and web service security. [Detection Area] OWASP Code Review Guide v1.1 A guide that captures best practice for reviewing code. [Detection Area] OWASP Testing Guide v3.0 - A guide on application security testing procedures and checklists.
Standards	[Detection Area] Application Security Verification Standard (ASVS) The ASVS defines an international standard for conducting application security assessments. It covers both automated and manual approaches for assessing (verifying) applications, using both security testing and code review techniques.
Tools	[Protection Area] AntiSamy - Java and .NET APIs validating rich HTML/CSS input from users to prevent cross-site scripting and phishing attacks. Enterprise Security API (ESAPI) Project - A collection of free and open source security libraries that can be used by developers to build secure web applications. [Detection Area] JBroFuzz Project - A web application fuzzer for performing testing

over HTTP and/or HTTPS. Its purpose is to provide a single, portable application that offers stable web protocol fuzzing capabilities.
[Detection Area] Live CD Project This CD collects some open source security projects in a single environment. Web developers, testers and security professionals can boot from this Live CD and have access to a full security testing suite.
[Detection Area] WebScarab Project.An intercepting proxy tool for performing all types of security testing on web applications and web services.
[Detection Area] Zed Attack Proxy Project - Another intercepting proxy tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range of security experience and, as such, is ideal for developers and functional testers who are new to penetration testing.
[Detection Area] DirBuster Project.An application designed to apply brute force to directories and file names on web/application servers.
[Detection Area] SWF Intruder Project.Is a tool for analysing and testing security of flash applications at run time.
[Life cycle security Area] WebGoat Project - An insecure web application for teaching web application security through interactive practical lessons.In the future, it is expected to become a security benchmarking platform.
[Protection Area] OWASP Back-End Security Project.This project aims to create a new guide that could allow developers, administrators and testers to understand any part of the security process of back-end components that communicate directly with web applications, databases, ldaps, payment gateways and much more. The project comprises three sections: security development, security hardening and security testing. Currently the document is not in guide format.
OWASP Backend Security OWASP O2 PlatformA highly-customisable platform for linking, managing and consuming IT security data, tools and applications.
ProjectInventory 2017:Tools [Health Check January 2017];OWASP Zed Attack Proxy;OWASP Web Testing Environment Project;OWASP OWTF;OWASP Dependency CheckOWASP Security Schepherd
Code [Health Check January 2017]:OWASP ModSecurity Core Rule Set Project;OWASP CSRFGuard Project;OWASP AppSensor
ProjectDocumentation[Health Check January 2017];OWASP Application Security Verification Standard Project;OWASP Software Assurance Maturity Model (SAMM);OWASP AppSensor Project;OWASP Top Ten Project;OWASP Testing Project

5.1.2 Common Criteria (CC)

I Common Criteria sono uno standard pubblicato dall'ISO (ISO/IEC 15408:2005) e sono costituiti da tre parti:

- Introduzione e modello generale.
- Requisiti di sicurezza funzionali.
- Requisiti di sicurezza di assurance.

Con i CC viene fornita anche una metodologia per la valutazione, la Common Criteria Evaluation Methodology (CEM), anch'essa standardizzata dall'ISO (ISO/IEC 18405:2005). Il processo di valutazione CC di un prodotto (software o hardware) riguarda diverse fasi SDLC:

- Requisiti (Protection Profile document - PP),
- Implementazione (Security Target document – ST),
- Test.

Le verifiche previste durante il processo di valutazione mirano ad accertare che siano stati soddisfatti, da parte del sistema/prodotto, del suo sviluppatore e del valutatore, opportuni requisiti di assurance che diventano sempre più severi al crescere del livello di valutazione. I CC definiscono una scala di 7 livelli di valutazione:

- EAL1. Functionally tested
- EAL2. Structurally tested
- EAL3. Methodically tested and checked
- EAL4. Methodically designed, tested and reviewed
- EAL5. Semi-formally designed and tested
- EAL6. Semi-formally verified design and tested
- EAL7. Formally verified design and tested.

I seguenti paesi hanno firmato l'accordo Common Criteria Recognition Agreement (CCRA) che si applica da EAL1 to EAL4:

- Paesi EU/EFTA: Austria, Repubblica Ceca, Danimarca, Finlandia, Francia, Germania, Grecia, Ungaria, Italia, Paesi Bassi, Norvegia, Spagna, Svezia e Regno Unito;
- Paesi Non-EU/EFTA: Australia, Canada, India, Israele, Giappone, Corea, Malesia, Nuova Zelanda, Pakistan, Singapore, Turchia e Stati Uniti.

L'European Mutual Recognition Agreement of IT Security Evaluation Certificates o 'SOGIS-agreement' è un accordo tra alcune nazioni europee con l'adesione dell'UE o dell'EFTA relativo al mutuo riconoscimento dei certificati di valutazione secondo gli standard CC per tutti i livelli di valutazione (EAL1 EAL7) .

URL	http://www.commoncriteriaportal.org/
Country of HQ location Geographic Scope	International
Type	Government

I criteri comuni per la valutazione della sicurezza informatica e la metodologia comune per la sicurezza delle tecnologie di valutazione sono stati pubblicati come standard ISO.

Risultati più rilevanti:

Standard	Common Methodology for Information Technology Security Evaluation and Common Criteria for Information Technology Security Evaluation These form the technical basis for an international agreement (the CCRA). Version 2.3 has also been published as ISO/IEC 15408:2005 and ISO/IEC 18045:2005
Future Related Standard	JTC 1/SC 27 ISO/IEC NP 20004 Information technology, Security techniques, Secure software development and evaluation under ISO/IEC 15408 and ISO/IEC 18405.

5.1.3 IEEE Computer Society (CS)

L'Iniziativa IEEE Computer Society è un'organizzazione senza fini di lucro ed i suoi principali progetti sono finalizzati alla pubblicazione di standard su tecnologie IT.

URL	http://www.computer.org /
Country of HQ location **Geographic Scope**	US International
Type	Academic (not for profit)

I principali risultati di questa iniziativa sono libri, conferenze, pubblicazioni su conferenze, riviste, corsi on-line, certificazioni di sviluppo software, standard e riviste tecniche.

Risultati più rilevanti:

Good Practice	Guide to the Software Engineering Body of Knowledge (SWEBOK) The SWEBOK Version 3, alpha version, will include Security as one of the proposed Supplemental Knowledge Areas.
Standard	Software & Systems Engineering Standards Committee (S2ESC) Formal Liaisons with ISO/IEC JTC1/SC7.

5.1.4 International Organisation for Standarditation (ISO)

ISO è il più grande sviluppatore ed editore al mondo di standard internazionali. Industrie ed esperti del settore generalmente contribuiscono come membri dei comitati tecnici ISO proponendo nuove normative che devono essere approvate almeno dal 70% dei membri ISO.

Il comitato tecnico che opera nell'ambito degli standard IT è il JTC 1. Questo comitato è a sua volta organizzato nei seguenti 3 sotto-comitati:

- JTC 1 / SC 7: software e ingegneria dei sistemi,
- JTC 1 / SC 22: linguaggi di programmazione, compresi ambienti e interfacce software di sistema
- JTC 1 / SC 27: tecniche di sicurezza IT.

Relativamente agli ambiti SSE le uscite principali ISO riguardano:

- pubblicazione di rapporti tecnici e standard -ISO / IEC TR 15026-1: 2010, ISO / IEC TR 24731-1: 2007, ISO / IEC TR 24772: 2010, ISO / IEC 15408 e ISO / IEC 18405
- 2 progetti in corso.

URL	http://www.iso.org
Geographic Scope	International
Type	Network of national standards institutes

Risultati più rilevanti:

JTC 1/SC 7	ISO/IEC 15026-1:2013 Systems and software engineering -- Systems and software assurance -- Part 1: Concepts and vocabulary ISO/IEC TR 15026-1:2010 Systems and software engineering - Systems and software assurance Part 1: Concepts and vocabulary.
JTC 1/SC 22	ISO/IEC TR 24731-1:2007 Information technology Programming languages, their environments and system software interfaces -Extensions to the C library - Part 1: Bounds-checking interfaces. Specifica una serie di estensioni del linguaggio di programmazione C, specificato dalla norma internazionale ISO/IEC 9899: 1999. Queste estensioni possono essere utili nella mitigazione delle vulnerabilità di sicurezza nei programmi. ISO/IEC TR 24772:2010 Information technology - Programming languages - Guidance on avoiding vulnerabilities in programming languages through language selection and use. Specifica le vulnerabilità del linguaggio di programmazione software da evitare nello sviluppo di sistemi in cui è richiesto un comportamento sicuro ai fini security/safety, mission critical e software business-critical. In generale, questa guida è applicabile al software sviluppato, rivisto, o mantenuto per qualsiasi applicazione. Le vulnerabilità sono descritte in modo generico, applicabili ad una vasta gamma di linguaggi di programmazione. Questa guida può essere anche utilizzata dagli sviluppatori per produrre o selezionare gli strumenti di valutazione del codice sorgente capaci di scoprire ed eliminare alcuni costrutti che potrebbero portare alla vulnerabilità del software o per selezionare un linguaggio di programmazione che consente di evitare i problemi attesi.

Progetti in corso:

JTC 1/SC 7	ISO/IEC FCD 15026-2 - Systems and software engineering - Systems and software assurance -- Part 2: Assurance case. Specifica i requisiti minimi per la struttura e il contenuto di un Assurance Case per migliorare la coerenza e la comparabilità degli Assurance Case e per facilitare le comunicazioni delle parti interessate, le decisioni di ingegneria e altri Assurance Case. Secondo questo documento ISO "*An assurance case includes a top-level claim for a property of a system or product (or set of claims), systematic argumentation regarding this claim, and the evidence and explicit assumptions that underly this argumentation. Arguing through multiple levels of subordinate claims, this structured argumentation connects the top-level claim to the evidence and assumptions*". ISO/IEC CD 15026-3 Systems and software engineering -- Systems and software assurance -- Part 3: Integrity levels. Si riferisce ai livelli di integrità dell'Assurance Case ed include i requisiti relativi al loro utilizzo con e senza un Assurance Case. Secondo questo documento ISO "*A software integrity level denotes a range of values of a software property necessary to maintain system risks within tolerable limits*".
JTC 1/SC 27	ISO/IEC 27021:2017 Preview Information technology -- Security techniques -- Competence requirements for information security management systems professionals ISO/IEC 15026-1:2013:Systems and software engineering -- Systems and software assurance -- Part 1: Concepts and vocabulary ISO/IEC NP 20004: Information technology - Security techniques - Secure software development and evaluation under ISO/IEC 15408 and ISO/IEC 18405. Si riferisce ad un problema differente e più urgente associato all'uso pratico dei Common Criteria, ossia la relazione tra i processi di sviluppo e di valutazione con l'analisi dei potenziali attacchi. E' legato all'iniziativa CAPEC.

5.1.5 International Society of Automation (ISA)

L'ISA è un'organizzazione globale no-profit che sviluppa standard per l'industria, certifica i professionisti di settore, offre istruzione e formazione, pubblica libri e articoli tecnici, e ospita convegni e fiere per i professionisti dell'automazione.

URL	http://www.isa.org /
Country of HQ location **Geographic Scope**	US International
Type	Industry (not for profit)

ISA99 standard "Manufacturing and Control Systems Security" ha alcune parti relative a SSE. Attualmente sono pubblicate solo le parti "99.01.01 Terminology, Concepts, and Models", "99.02.01 - Establishing an Industrial Automation and Control Systems Security Program" e "99.03.01 Security technologies for Industrial Automation and Control Systems". ISA e la Commissione Elettrotecnica Internazionale (IEC) hanno negoziato l'adozione degli standard ISA 99 e IEC 62443. I membri ISA pagano una tassa regolare (annuale o biennale), in base al loro tipo di appartenenza, al fine di ottenere i benefici ISA come l'accesso alle informazioni tecniche e alle risorse per lo sviluppo professionale.

Risultati più rilevanti:

Proposed Standards	ISA TR99.02.03 Patch Management in the IACS Environment. This technical report addresses the topic of patch management in an Industrial Automation and Control Systems (IACS) environment for asset owner and vendor communities. It is aimed at providing guidance in patch-testing and patch-management according to an acceptable level of risk. ISA 99.03.04 Product Development Requirements. This standard will address the security requirements for product development
Draft Standards	ISA 99.03.03 System Security Requirements and Security Assurance Levels This standard defines security requirements that are grouped into seven categories: 1) Access control, 2) Use control, 3) Data integrity, 4) Data confidentiality, 5) Restrict data flows, 6) Timely response to an event and 7) Network resource availability. Each category includes a mapping of security requirements to security assurance levels.

5.1.6 Software Assurance Forum for Excellence in Code (SAFECODE)

SAFECode è un'iniziativa privata creata da sviluppatori software e fornitori. Individuando e promuovendo le migliori pratiche in SSE, questa iniziativa sostiene che l'industria del software potrebbe rilasciare software, hardware e servizi più sicuri e affidabili. Tra le sue uscite principali, ci sono i documenti che raccolgono le migliori pratiche, tenendo conto del ciclo di vita di sviluppo del software.

URL	http://www.safecode.org
Country of HQ location **Geographic Scope**	US International
Type	Industry (not for profit)

SAFECode afferma che i suoi obiettivi futuri sono:

1. Identificare e condividere collaudate pratiche di garanzia del software
2. Promuovere una più ampia adozione di tali pratiche nell'ecosistema informatico,
3. Lavorare con istituzioni e fornitori di infrastrutture critiche per sfruttare le pratiche nella gestione dei rischi aziendali.

Risultati più rilevanti:

Training	**Security Engineering Training** A framework for corporate training programs on the principles of secure software development.
Good Practice	**Software Integrity Controls** An assurance-based approach to minimizing risks in the software supply chain. Based on the practices of SAFECode members, the report provides software integrity controls for software sourcing, software development, software testing, software delivery and software resilience. **The Software Supply Chain Integrity Framework** This defines risks and responsibilities for making software secure in the global supply chain. Based on the experience of SAFECode members, it describes the software supply chain (staircase model of software suppliers) and the principles for designing software integrity controls. **Fundamental Practices for Secure Software Development** Based on the practices of SAFECode members, this outlines a set of practices for secure software development that can be applied in the different phases of the software development life cycle. **Software Assurance: An Overview of Current Industry Best Practices** This outlines the development methods and integrity controls used by SAFECode members to improve software assurance and security in the delivery.

5.1.7 SANS Software Security Institute (SAN SSI)

SANS SSI offre una libreria di iniziative di ricerca e di community per aiutare sviluppatori, architetti, programmatori e responsabili della sicurezza delle applicazioni a proteggere le loro applicazioni software/web.
Questa iniziativa raccoglie e fornisce informazioni tecniche aggiornate, come l'accesso gratuito alle risorse sui più recenti sui vettori di attacco e sulle vulnerabilità di sicurezza delle applicazioni, tra cui un blog aggiornato, news-letters settimanali, Webcast, articoli e documenti in materia di sicurezza del software.

URL	http://www.sans.org
Country of HQ location **Geographic Scope**	US International
Type	Academic

SANS pubblica relazioni annuali (Top 25 Software Errors) con l'analisi sugli errori di programmazione più pericolosi (vedi ad esempio http://www.sans.org/top25-software-errors/).

Risultati più rilevanti:

Resources	**Application Security Resources:** Application security whitepapers and application security webcasts **Security Laboratory:** The "Security Laboratory" is an informal set of articles and whitepapers about security, IT and the computer security industry. **Fundamental Practices for Secure Software Internet Storm Center (ISC)** The ISC provides a free analysis and warning service to Internet users and

	organisations. Volunteers donate their time to analyse defects and anomalies, and post a daily diary of their analysis and thoughts on the Storm Center website. **Application Security Procurement Language:** This is a draft software contract for buyers of custom software. Its objective is to make code developers responsible for checking the code and fixing security flaws before delivery of the software. **Top 25 Software Errors** These are listed in three categories: • Insecure Interaction Between Components • Risky Resource Management • Porous Defences. **Each error includes:** • The ranking of each Top 25 entry • Links to full Common Weakness Enumeration (CWE, see section 3.4) entry data • Data fields for weakness prevalence and consequences • Remediation cost • Ease of detection • Code examples • Detection Methods • Attack frequency and attacker awareness • Related CWE entries and related patterns of attack for this weakness. It also includes fairly extensive prevention and remediation steps that developers can take to mitigate or eliminate the weakness

5.1.8 Web Application Security Consortium (WASC)

WASC produce best practice per le applicazioni web. WASC riassume la sua missione così "*to develop, adopt, and advocate standards for web application security*".

URL	http://www.webappsec.org/
Country of HQ location **Geographic Scope**	US International
Type	Industry (not for profit)

Risultati più rilevanti:

Resources	**Web Application Security Scanner Evaluation Criteria** A set of criteria for evaluating web application security. **The Web Hacking Incidents Database** Database of web applications and related security incidents. **The Script Mapping Project** List of ways of executing script within a web page without using <script> tags. **Distributed Open Proxy Honeypots** Analysis of HTTP traffic through specially configured open proxies to categorise the requests into threat classifications**.**

	Web Security Glossary Index of terms and terminology relating to web applications security **Web Security Threat Classification** An attempt to develop and promote industry-standard terminology for describing threats to the security of a website. **Web Application Firewall Evaluation Criteria** Development of detailed criteria for evaluating a web application firewall (WAF). **Web Application Security Statistics** Collection of application vulnerability statistics for identifying and mapping application security issues on enterprise websites.

5.1.9 Institute for Software Quality (IFSQ)

L'Istituto per la Qualità del Software, con sede nei Paesi Bassi, è un gruppo di professionisti coinvolti nello sviluppo e nella distribuzione di software. IfSQ persegue un obiettivo comune: aumentare gli standard software (e dello sviluppo software) in tutto il mondo attraverso la promozione del Code Inspection, come prerequisito del Software Testing nel ciclo di produzione e rilascio del software.

URL	http://ifsq.nl/
Country of HQ location **Geographic Scope**	The Netherlands International
Type	Industry (non profit)

IfSQ ha analizzato, quantificato e migliorato lo stato dell'arte scientifico sulla qualità del software, e ha prodotto un insieme di indicatori (Defect Indicators) che sono stati raccolti in un insieme coordinato di tre standard, che sono pubblicati sul sito, in forma di opuscolo e sotto forma di corsi e workshop. La maggior parte dei criteri di valutazione, in particolare "major string", "parametri non controllati" e " unexpected state not trapped", sono rilevanti per migliorare la sicurezza del software.

Risultati più rilevanti:

Resources	**Software Quality Standards -** Levels 1, 2 and 3 are available.

5.2 Iniziative europee

Questa sezione ha l'obiettivo di fornire una vista delle iniziative in ambito Europeo. Le iniziative di seguito presentate sono state classificate sulla base dell'ambito geografico e della tipologia di appartenenza (accademiche, governative, industria).

Analizzando ambiti, obiettivi e risultati di ognuna, emerge che:

- un insieme di iniziative rappresentano per obiettivi e risultati una categoria isolata. Tra queste iniziative diciamo 'non raggruppabili' ci sono: NESSI, OWASP Local Chapters, MISRA e Serenity Forum.
- altre iniziative posso essere 'raggruppate' sulla base di alcuni elementi che li caratterizzano e li accomunano: Events and Periodicals, Certifications, Academic Education. Queste iniziative potrebbero essere classificate con più tag sulla base dei loro risultati rilevanti o attesi in SSE: standardisation, industry platform, vulnerability detection, vulnerability protection, information sharing, specialised workshop, certification and training.

5.2.1 Networked European Software and Services Initiative (NESSI)

NESSI è la piattaforma tecnologica europea dedicata al Software e ai Servizi. L'obiettivo principale di NESSI si indirizza sul potenziamento dei servizi Internet attraverso attività di ricerca, standard e policy, e contributi costruiti attraverso una community industria/università.

I partecipanti NESSI sono divisi in tre gruppi:

- partner NESSI: prevalentemente industriale, ma ci sono anche alcuni profili accademici - coordinano la piattaforma e forniscono il sostegno finanziario per le attività NESSI;
- I membri NESSI: industria, mondo accademico e gli utenti - rappresentano i principali stakeholders del dominio della fornitura di servizi ICT. Non è obbligatorio un contributo finanziario
- abbonati NESSI: usano diversi canali di informazione per tenersi aggiornati sulle attività di NESSI.

URL	http://www.nessi-europe.com
Country of HQ location **Geographic Scope**	Belgium Europe
Type	Industry

Piattaforme tecnologiche nazionali e regionali sono parte della rete NESSI: gestiscono obiettivi NESSI da un punto di vista locale.

I focus NESSI hanno alcune correlazioni SSE:

- Identificare le direzioni della ricerca futura sui servizi
- costruire contributi formali sui settori chiave
- investire sulla rete NESSI per migliorare il coordinamento tra i programmi di ricerca europei, nazionali e regionali.

Risultati più rilevanti:

Research Agenda	NESSI strategic research agenda (Lastest version) One of the Research Priorities for 2009-2010 (Volume 3.2 - Revision 2 - May 2009) is "End-to-end Trust, Security, Privacy and Resilience".
Working Group related to SSE	Trust, Security and Dependability NWG This NWG reports on the state of play regarding web services trust, security and dependability (reliability), as well as giving recommendations on future priorities, producing guidelines and identifying best practice. Task forces in security areas are expected to replace this NWG.

5.2.2 Piattaforme Nazionali NESSI

L'obiettivo generale delle Piattaforme NESSI è quello di promuovere lo sviluppo e l'applicazione di tecnologie e servizi ICT per affrontare le sfide future all'interno dell'industria europea e del governo.

Nella tabella che segue vengono sintetizzate le attività per ogni piattaforma nazionale che gestisce gli obiettivi NESSI da un punto di vista locale e pubblica la sua SRA nazionale.

URL	http://www.nessi-europe.com
NESSI - Norway	E' la filiale norvegese del NESSI. Il suo obiettivo principale è quello di creare un'arena norvegese per gli stakeholders del settore industria, ricerca/mondo accademico e pubblico e di influenzare la strategia di ricerca ICT del governo

	norvegese.

URL	http://www.nessi-europe.com
NESSI - Slovenia	Alla base di queste attività è che NESSI assumerà la responsabilità del contenuto e dell'attuazione del 7° programma quadro dell'UE per R&D. Essi invitano chiunque sia coinvolto in attività di R&D a partecipare a questo lavoro.

URL	http://www.iipsaas.nl
IIP SaaS-Netherlands	E' la piattaforma olandese di NESSI per il Software as a Service (SaaS). IIP SaaS lavora a stretto contatto con il programma di ricerca Jacquard [www.jacquard.nl/].

URL	http://www-it.fmi.uni-sofia.bg/nessibg
NESSI-Bulgaria	NESSI-Bulgaria è stata fondata nel 2005. Si tratta di un forum per lo scambio di conoscenze, lo sviluppo di strategie e la ricerca di nuove potenzialità a livello internazionale IT e servizi industriali. La visione centrale della piattaforma è di consentire nuovi modelli di business orientate ai servizi. I loro obiettivi sono: • Definire una Roadmap bulgara e l'SRA per l'evoluzione del programma di innovazione R&D bulgaro. • Supporto alle attività R&D nei settori del software e dei servizi. • Fornire formazione: nuovi corsi, programmi MSc, programmi PhD e formazione

URL	http://www.nessi-hungary.com http://www.nessi.hu/
NESSI- Hungary	NESSI-Ungheria è stata fondata nel 2007 con lo scopo di evolvere la direzione della ricerca e dello sviluppo strategico nel settore del software e dei servizi, sulla base di un approccio unificato. Gruppi di lavoro di questa piattaforma sono divisi in due sottogruppi: domain-oriented e technological-oriented. La piattaforma è aperta a qualsiasi altra organizzazione ungherese.

URL	http://www.bicc-net.de/
Germany Bicc-Net	BICC-NET, Piattaforma di NESSI tedesca, è il Polo ICT bavarese della Germania. Fondata nel 2007, intende stimolare selettivamente l'innovazione. BICC-NET comprende quanto segue: • sviluppo e distribuzione del software • lo sviluppo e la distribuzione di hardware • Telecomunicazioni • sistemi software e hardware embedded nei prodotti • Processi basati su software in fase di sviluppo, la produzione, i servizi e della pubblica amministrazione • Servizi nelle aree di cui sopra

	BICC-NET viene utilizzato per garantire la crescita ICT in Baviera. Essa è guidata dalla BICC sede ufficiale "cluster", che è stato direttamente commissionato dal Ministero bavarese per gli Affari economici, infrastrutture, trasporti e tecnologia. BICC-NET supporterà i profili di innovazione delle aziende ICT bavaresi e gli sviluppi in corso.

URL	http://www.fi-stockholm.eu/
NESSI- Sweden	NESSI svedese è stata fondata nel 2010. L'obiettivo generale di NESSI Svezia è quello di promuovere lo sviluppo e l'applicazione di tecnologie e servizi ICT per affrontare le sfide future all'interno dell'industria svedese e del governo

URL	http://www.nessi-europe.com/
NESSI- Romania	NESSI Romania è stata fondata nel 2010. Gli obiettivi a breve termine di NESSI-Romania sono: • istituire gruppi di lavoro nazionali su diversi argomenti definiti in NESSI SRA • Definire un SRA nazionale per l'evoluzione futura del programma nazionale R&D e innovazione relativamente a software e servizi • Diffondere i risultati NESSI dei progetti strategici e compatibili

5.2.3 OWASP Local Chapters

Questa sezione fornisce una vista dei gruppi di lavoro OWASP distribuiti sul territorio Europeo.

URL	http:// www.owasp.org/index.php/Belgium
OWASP Belgium Local Chapter	Le principali attività svolte riguardano l'organizzazione di incontri su come difendere le applicazioni web da attacchi.

URL	http://www.owasp.org/index.php/Denmark
OWASP Denmark Local Chapter	Le principali attività svolte riguardano l'organizzazione di incontri su diversi argomenti di sicurezza delle informazioni legate alle applicazioni web. Le presentazioni sono disponibili sul sito web

URL	http://www.owasp.org/index.php/France
OWASP France Local Chapter	Le principali attività svolte riguardano l'organizzazione di incontri e la traduzione della documentazione OWASP in francese. Questo Chapter fornisce anche la formazione su progetti e risorse OWASP attraverso il programma "OWASP projects and resources you can use today", che ha lo scopo di promuovere progetti OWASP, fornendo una selezione di progetti maturi ed enterprise-ready, insieme con esempi pratici di come usarli..

URL	http://www.owasp.org/index.php/Germany

OWASP Germany Local Chapter	Le principali attività riguardano l'organizzazione di incontri, conosciuti come AppSec Germany Conference, che si svolge ogni anno.

URL	http://www.owasp.org/index.php/Geneva
OWASP Geneva Local Chapter	Le principali attività svolte da questo capitolo riguardano l'organizzazione di incontri legati alle identità digitali e autenticazione nelle applicazioni web.

URL	http://www.owasp.org/index.php/Greece
OWASP Greece Local Chapter	Il gruppo di lavoro OWASP greco è stata fondato nel 2005 con l'obiettivo di informare la comunità greca sui rischi per la sicurezza nelle applicazioni web. Il motivo principale che ha spinto alla sua creazione è il sempre crescente numero di incidenti di sicurezza su Internet, come ad esempio i tentativi di phishing a banche greche. Oggi, il gruppo greco promuove localmente l'iniziativa OWASP attraverso il Software Libero/Open e la traduzione in greco della documentazione OWASP. Emettono una newsletter mensile, mantengono una mailing list per gli aggiornamenti e gestiscono dibattiti online su problemi di sicurezza di attualità. La comunità greca OWASP vuole riunire tutti coloro che sono interessati e preoccupati per la sicurezza delle applicazioni web. Allo stesso tempo, accoglie i volontari che sono disposti a lavorare su progetti coordinati dall'OWASP, utilizzando software libero/open source. Invitano a chiunque di condividere le proprie idee, pensieri e riflessioni sugli attacchi, la difesa, i metodi di risposta, strumenti e buone pratiche in materia di sicurezza di Internet.

URL	http://www.owasp.org/index.php/Ireland-Dublin http://www.owasp.org/index.php/Ireland-Limerick
OWASP Ireland Local Chapter	Questo paese ha due gruppi locali: Dublino e Limerick. Il gruppo più attivo è quello di Dublino le cui attività principali riguardano l'organizzazione di eventi e conferenze. Questo gruppo fornisce anche la formazione su progetti e risorse OWASP attraverso il programma " OWASP projects and resources you can use today". Questo ha lo scopo di promuovere i progetti OWASP, fornendo una selezione di progetti maturi ed enterprise-ready con esempi pratici di come usarli.

URL	http://www.owasp.org/index.php/Italy
OWASP Italy Local Chapter	Le attività riguardano l'organizzazione di eventi e lo sviluppo di tool. Il gruppo cerca di organizzare almeno 2 conferenze all'anno, uno in primavera e un altro in autunno. Recentemente, hanno lavorato sullo sviluppo di sqlmap, un *automatic SQL injection tool* sviluppato in Python. L'iniziativa è sostenuta da partner come IsecLab, CLUSIT e ISACA Roma.

URL	http://www.owasp.org/index.php/Latvia
OWASP Latvia Local Chapter	E' stata creata nell'ottobre 2007. Le attività principali riguardano l'organizzazione di eventi. Il gruppo non si è dimostrato molto attivo negli ultimi anni.

URL	http://www.owasp.org/index.php/Latvia
OWASP Leeds/Northern Local Chapter	Questo è gruppo nuovo e molto attivo. Ha tenuto riunioni in tutta l'Inghilterra settentrionale, tra cui a Leeds, Manchester e Newcastle-upon-Tyne.

URL	http://www.owasp.org/index.php/London
OWASP London Local Chapter	Le attività di OWASP Londra si concentrano sulla preparazione e l'organizzazione di eventi, conferenze e presentazioni. Il gruppo ha registrato elevata attività nel corso del 2010. Esso prevede anche la formazione su progetti e risorse OWASP attraverso il programma "OWASP projects and resources you can use today", che mira a promuovere progetti OWASP, fornendo una selezione di progetti maturi ed enterprise-ready con esempi pratici di come usarli.

URL	http://www.owasp.org/index.php/Luxembourg
OWASP Luxembourg Local Chapter	Le attività del gruppo riguardano la preparazione e l'organizzazione di eventi e conferenze come il Java User Group (YAJUG) o Chaos Computer Club Letzebuerg (C3L). Attualmente sembra che vi sia poca attività in questo gruppo.

URL	http://www.owasp.org/index.php/Norway
OWASP Norway Local Chapter	Le attività di OWASP Norvegia riguardano la preparazione e l'organizzazione di eventi e conferenze. Questo gruppo è stato molto attivo negli anni passati, quando ha organizzato 8 conferenze in Norvegia in un anno.

URL	http://www.owasp.org/index.php/Poland
OWASP Poland Local Chapter	L'attività principale che questo gruppo è quella di organizzare eventi. In questo gruppo sembra essere molto attivo, sono stati coinvolti in 11 conferenze nel corso del 2010. L'iniziativa è sostenuta da ISSA.

URL	http://www.owasp.org/index.php/Portuguese
OWASP Portugal Local Chapter	Le attività di questo gruppo riguardano l'organizzazione di conferenze e pubblicazioni. Ha organizzato uno dei più importanti eventi di OWASP: *Ibero-American Web Application Security Conference IBWAS'2010*.

URL	http://www.owasp.org/index.php/Scotland
OWASP Scotland Local Chapter	Le principali attività svolte da questo gruppo, secondo quanto riportato sul loro sito, sono finalizzate a fonrire risposte insieme ad altri gruppi britannici locali ai diversi uffici governativi del Regno Unito. Questo gruppo sembra che organizzi anche incontri annuali.

URL	http://www.owasp.org/index.php/Slovakia
OWASP Slovakia Local Chapter	L'attività principale di questo gruppo riguarda l'organizzazione di eventi.

URL	http://www.owasp.org/index.php/Spain
OWASP Spain Local Chapter	Questo gruppo svolge due attività principali. Da un lato collabora attivamente con OWASP su un progetto per fornire le specifiche e i requisiti legali per le applicazioni Web. D'altra parte, come la maggior parte degli altri gruppi locali di questa sezione, organizza eventi e conferenze annuali. Ha partecipato anche all'evento IBWAS'2010 [https://www.owasp.org/index.php/IBWAS10] in collaborazione con il gruppo portoghese.

URL	http://www.owasp.org/index.php/Sweden
OWASP Sweden Local Chapter	Questo gruppo si concentra sull'organizzazione di meeting ed eventi. Ha organizzato conferenze anche in collaborazione con altri gruppi del nord, come il norvegese e il finlandese.

URL	http://www.owasp.org/index.php/Switzerland
OWASP Switzerland Local Chapter	Questo gruppo organizza incontri su base periodica, soprattutto nella parte tedesca della Svizzera. I loro incontri e gli eventi sono principalmente su temi come test di sicurezza, lo sviluppo sicuro, hacking e architetture sicure. Sul loro sito Web sono fruibili diapositive di eventi e conferenze.

URL	http://www.owasp.org/index.php/Ukraine
OWASP Ukraine Local Chapter	E' un gruppo di recente formazione ancora in fase di organizzazione

5.2.4 Motor Industry Software Reliability Association (MISRA)

MISRA è una *Motor Companies Consortium* all'interno del Regno Unito. I suoi risultati (ricerca, i risultati della ricerca e standard de facto, le linee guida) sono finalizzati principalmente al software sicuro e affidabile per sistemi embedded nel settore automobilistico.

Nei primi anni 1990, il progetto MISRA era stato concepito col fine di sviluppare linee guida per la realizzazione di software embedded nei componenti elettronici dei veicoli stradali. Dopo la chiusura del il finanziamento ufficiale cessato, i membri Misra hanno deciso di continuare a lavorare insieme.

MISRA instaura quindi una collaborazione tra costruttori di veicoli, fornitori di componenti e di consulenza ingegneristica. Esso mira a promuovere le migliori pratiche nello sviluppo di sistemi elettronici legati alla sicurezza dei veicoli stradali e di altri sistemi embedded.

La sua documentazione non è accessibile al pubblico, ma può essere acquistata sul sito web del consorzio.

URL	http://www.misra.org.uk
Country of HQ location **Geographic Scope**	UK National
Type	Industry

I lavori in corso MISRA includono:

Model based development and autocode – Incoraggia alle buone pratiche ed evita le caratteristiche del linguaggio di modellazione mal definito.

- MISRA C++ (Produzione di una serie di linee guida per l'uso di C ++ in sistemi critici)
- MISRA C3 (3rd review of MISRA C)
- Mira a promuovere le migliori pratiche nello sviluppo di sistemi elettronici legati alla sicurezza nei veicoli stradali e di altri sistemi embedded (è stato adottato e utilizzato in una vasta gamma di settori e applicazioni, tra cui il settore ferroviario, aerospaziale, militare e medico)

MISRA Safety Analysis – Offre una guida sui requisiti della decomposizione. Esso descrive come il ciclo di vita della sicurezza dei sistemi automotive si inserisce nel ciclo di vita dello sviluppo dei veicoli.

Risultati più rilevanti:

Good Practice	Guidelines for the Use of the C Language in Vehicle Based Software, ISBN 978-0-9524156-6-5, April 1998, October 2002 Guidelines for the Use of the C Language in Critical Systems, ISBN 0 9524156 2 3 (paperback), ISBN 0 9524156 4 X (PDF), October 2004 Guidelines for safety analysis of vehicle based programmable systems, ISBN 978-0-9524156-5-7 (paperback), ISBN 978-0-9524156-7-1 (PDF), November 2007. Guidelines for the Use of the C++ Language in Critical Systems, ISBN 978-906400-03-3 (paperback), ISBN 978-906400-04-0 (PDF), June 2008.
Standard	MISRA AC GMG: Generic modelling design and style guidelines, ISBN 978-906400-06-4 (PDF), May 2009.

5.2.5 European Space Agency (ESA)

Dall'inizio degli anni '90 l'ESA si è occupata di definire la qualità dei prodotti software. La famiglia PSS[4] di standard (poi sostituito da standard ECSS) include un software engineering standard e una serie di guide.

URL	http://www.esa.int
Country of HQ location **Geographic Scope**	Paris European
Type	Collaboration of Several European Countries

Uno degli standard di software ampiamente utilizzato in quella serie, chiamato "Guide to applying the ESA Software Engineering Standards to small software projects" è disponibile all'indirizzo: ftp://ftp.estec.esa.nl/pub/wm/wme/bssc/Bssc962.pdf

Questo standard definisce una serie di criteri di qualità per i requisiti software e di design, che hanno una influenza diretta e indiretta sulla sicurezza del software. Per i *quality criteria requirements* i seguenti sono rilevanti:

- *Are the characteristics of users and of typical usage mentioned? (No user categories missing)*
- *Are all the external interfaces of the software explicitly mentioned? (No interfaces missing)*
- *Is each requirement prioritised? (Is the meaning of the priority levels clear?)*
- *Is each requirement verifiable (in a provisional acceptance test)? (Measurable: where possible, quantify; capacity, performance, accuracy)*
- *Are the requirements consistent? (Non-conflicting)*
- *Are the requirements sufficiently precise and unambiguous? (Which interfaces are involved, who has the initiative, who supplies what data, no passive voice.)*
- *Are the requirements complete? Can everything not explicitly constrained indeed be viewed as developer freedom? Is a product that satisfies every requirement indeed acceptable? (No requirements missing)*
- *Are the requirements understandable to those who will need to work with them later?*
- *Are the requirements realisable within budget?*
- *Most of the design quality criteria are relevant to software security.*

Risultati principali:

Good Practice	The PSS [http://www.esa.int/TEC/Software_engineering_and_standardisation/TECBUCUXBQE_2.html] family of standards for Software Quality. A guide to applying ESA Software Engineering Standards to small software projects is available at: ftp://ftp.estec.esa.nl/pub/wm/wme/bssc/Bssc962.pdf Eindhoven University of Technology provides further simplified requirements and design checklists. [http:// http://www.win.tue.nl/is/doku.php]

5.2.6 Serenity Forum

Questo forum è stato creato dai partner del progetto Serenity (un progetto R&D FP6 finanziato fino a giugno 2009) per dare seguito alla community istituita nel corso del progetto. SERENITY Forum è incaricato

[4] http://www.esa.int/TEC/Software_engineering_and_standardisation/TECBUCUXBQE_2.html

di fornire un approccio radicalmente nuovo alla Security Engineering attraverso una vasta gamma di modelli di sicurezza e schemi di integrazione. Si compone dei membri del progetto Serenity e di persone individuali. Tuttavia non si evidenziano a seguire molte attività derivanti da questo evento.

URL	www.serenity-forum.org
Country of HQ location **Geographic Scope**	European
Type	Academic

5.3 Iniziative US

In questa sezione viene fornita una panoramica delle iniziative SSE negli Stati Uniti. Questi sono stati classificati in base alla tipologia (accademiche o di governo).

5.3.1 CERT Secure Coding

Il CERT Secure Coding è un'iniziativa di sicurezza del programma Computer Emergency Response Team (CERT). Questo programma fa parte del Software Engineering Institute (SEI) alla Canergie Mellon University (Pennsylvania, USA). Alcuni dei suoi programmi sono finanziati dal governo degli Stati Uniti.

Nel novembre 1988, la Defense Advanced Research Projects Agency (DARPA)incaricò il SEI, con la creazione di un centro per coordinare la comunicazione tra gli esperti di sicurezza durante le emergenze e per aiutare a prevenire futuri incidenti. Nell'ambito di questo compito, CERT ha sviluppato il Software Initiative Assurance, che comprende: Secure Coding Standards, Source Code Analysis Lab, Vulnerability analysis, Function extraction for malicious code

Il SEI è un centro di ricerca e sviluppo finanziato dal governo federale, che conduce ricerche di ingegneria del software in acquisizione, architetture e linee di prodotto, miglioramento dei processi e misurazione delle performance, sicurezza e l'interoperabilità del sistema e l'affidabilità.

Il SEI lavora a stretto contatto con le organizzazioni di difesa e di governo, soprattutto l'Ufficio Secretary of Defense/Acquisition, Technology, and Logistics (OSD/AT&L)[5], l'industria e il mondo accademico, con l'obiettivo di migliorare continuamente i sistemi software-intensive.

URL	http://www.cert.org/secure-coding/
Country of HQ location **Geographic Scope**	US National
Type	Academic

Le aree di lavoro CERT Secure Coding sono:

- **Secure coding standards** [http://www.cert.org/secure-coding/research/secure-coding-standards.cfm] - Proposes standards for enhancing the security of programming languages.
- **International Standards Development** [http://www.cert.org/secure-coding/standards/index.cfm] - Development of International Standards.

[5] http://www.acq.osd.mil/

- **Source Code Analysis Laboratory (SCALe)** [http://www.cert.org/secure-coding/products-services/index.cfm] SCALe allows source code to be assessed against a set of secure coding standards. SCALe issues and certifies conformance testing when the test‟s findings have been addressed by the developers.
- **Development Tools and Libraries** [http://www.cert.org/secure-coding/devtools.html] - Tools and libraries for secure software development
- **TSP Secure** [http://www.cert.org/secure-coding/secure.html] - Secure Team Software Process methodology.

CERT Secure Coding vuole influenzare fornitori per migliorare la sicurezza base all'interno dei loro prodotti. Al fine di raggiungere questo obiettivo, CERT Secure Coding lavora con sviluppatori di software e organizzazioni di sviluppo software per ridurre le vulnerabilità derivanti da errori di codifica (C, C ++ o linguaggi di programmazione Java) prima di essere distribuiti. Inoltre, gli analisti CERT valutano le cause della vulnerabilità e identificano le pratiche di secure coding.

CERT collabora con ISO per la creazione di diversi standard su secure coding.

Risultati più rilevanti:

Training	**Secure Coding in C and C++** [http://www.sei.cmu.edu/training/p63.cfm] Course of secure coding in C and C++ based on Addison-Wesley's material: "Secure Coding in C and C++" and "The CERT C Secure Coding Standard"
Standards for Software Developers	**CERT C Secure Coding Standard, Version 2.0** **CERT C++ Secure Coding Standard** [https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=637] **CERT Oracle Secure Coding Standard for Java** [https://www.securecoding.cert.org/confluence/display/java/SEI+CERT+Oracle+Coding+Standard+for+Java]

5.3.2 Build Security In

E' un'iniziativa del governo degli Stati Uniti basata su un sito web online, in cui sono raccolte informazioni relative al Software Assurance. Si tratta del Strategic Initiatives Branch del National Cyber Security Division (NCSD) [https://www.us-cert.gov/] del US DHS [https://www.dhs.gov/]. Questa iniziativa mira a fornire agli sviluppatori di software una guida pratica su come produrre software sicuro.

Il programma Software Assurance US DHS cerca di ridurre le vulnerabilità del software, minimizzare lo sfruttamento e indirizzare su routine di sviluppo migliori per la distribuzione di prodotti software affidabili. Queste attività porteranno a software più sicuro e affidabile a supporto dei requisiti mission-critical da parte di imprese e infrastrutture critiche.

Build Security cerca di spostare il paradigma di sicurezza dalla gestione delle patch al Software Assurance. Questo cambiamento è stato progettato per incoraggiare gli sviluppatori di software ad aumentare la qualità e la sicurezza complessiva del software all'inizio, piuttosto che applicare le patch ai sistemi all'emergere delle vulnerabilità.

Questo progetto vuole essere un luogo in cui la community US Software Engineering (sviluppatori di software e organizzazioni di sviluppo software) possono trovare informazioni e consigli pratici su come produrre software sicuro e affidabile.

I contenuti del sito sono suddivisi in tre aree principali:

- Best Practice: current best thinking, available technology, industry practice
- Knowledge: conoscenza effettiva security-related che tutti gli ingegneri dovrebbero avere sui Tools. Informazioni su classi generali di tool, con riferimento ai tool specifici.

URL	https://buildsecurityin.us-cert.gov/bsi/home.html
Country of HQ location **Geographic Scope**	US National
Type	Government

Il contenuto di Build Security In si basa sul principio che la sicurezza del software è fondamentalmente un problema di ingegneria del software e deve essere affrontato in modo sistematico per tutto il ciclo di vita di sviluppo del software. Esso contiene, ed è collegato a una vasta gamma di informazioni provenienti da diverse fonti, informazioni sulle US best practices, tools, linee guida, regole, principi, e altre conoscenze per aiutare le aziende a costruire software sicuro e affidabile.

Il personale della SEI della Carnegie Mellon University contribuisce e revisiona gli articoli e mantiene il sito. Ai contenuti contribuiscono anche ricercatori e professionisti provenienti da Cigital, Inc. e altre organizzazioni (vedi Contributing Authors [https://buildsecurityin.us-cert.gov/about-us/authors]).

I membri della community software assurance sono invitati a caricare gli articoli per la pubblicazione sul sito web Build Security In o di rivedere gli articoli caricati.

Risultati più rilevanti pubblicati in articoli:

Best Practice	Acquisition [https://buildsecurityin.us-cert.gov/articles/best-practices/acquisition] The objective is to raise provider awareness. The articles describe an acquisition life-cycle framework for security activities, products, and reviews and for selected acquisition contexts and life-cycle phases. The authors provide additional guidance on methods and resources for identifying and managing security risks.
	Architectural Risk Analysis [https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/architecture.html] Presents best practice for reviewing, assessing, and validating the specification, architecture, and design of a software system with respect to software security, reliability and performance goals. It includes a discussion on the identification, assessment, prioritisation, mitigation and validation of the risks associated with architectural flaws.
	Code Analysis [https://buildsecurityin.us-cert.gov/articles/best-practices/code-analysis] - Presents best practice in performing code analysis to uncover errors in, and improve the quality of, source code. Methods include manual code auditing, walkthroughs, static analysis, dynamic analysis, metric analysis, testability analysis, crypto analysis, random number analysis and fault injection.
	Deployment and Operations [https://buildsecurityin.us-cert.gov/articles/best-practices/deployment-and-operations] - The objective is to describe, and provide pointers to, commonly accepted best practice and processes and the relevant characteristics of organisations that demonstrate competence in sustaining adequate security

	during deployment and operations.
	Governance and Management [https://buildsecurityin.us-cert.gov/articles/best-practices/governance-and-management] - These articles provide a recommended sequence of steps to take in order to govern and manage enterprise, information, and software security. Detailed "how-to" guidance is not provided. Security at the enterprise and organisational level is addressed
	Incident Management [https://buildsecurityin.us-cert.gov/articles/best-practices/incident-management] - Incident management is defined. Examples of best practice in building an incident management capability are presented. It also takes a look at one particular component of an incident management capability, a computer security incident response team (CSIRT), and discusses its role in the systems development life cycle.
	Legacy Systems [https://buildsecurityin.us-cert.gov/articles/best-practices/legacy-systems] - Describes the kinds of security risks that can be present in legacy systems, both in-house and commercially off-the-shelf, and offers guidance for assessing those risks and making sound decisions about addressing them.
	Measurement [https://buildsecurityin.us-cert.gov/articles/best-practices/measurement] - Best practice is described in relation to measurements for managing the quality of software systems during development. Several proposed measures for characterizing specific security-related features are discussed, and the current extent of the practice of software measurement with specific attention to the use of security-related measures is described.
	Penetration Testing [https://buildsecurityin.us-cert.gov/articles/best-practices/penetration-testing] The concepts and goals of traditional penetration testing are discussed and recommendations are made on how these can be adopted to better suit the needs of software developers. Additionally, the present state of the available tool base is described.
	Project Management [https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/project.html] Focuses on how security influences project management tasks and suggests refinements to existing practices. For example, project management can affect how well security requirements are satisfied, in terms of how the inputs from the technical, management, and operational communities are coordinated. Planning has to reflect the resources, effort, and risks associated with securing a new technology, such as Web Services. Design and implementation decisions may create new security threats, which should be represented in both project monitoring and planning.
	Requirements Engineering [https://buildsecurityin.us-cert.gov/articles/best-practices/requirements-engineering] Best practice for security requirements engineering is presented, including processes that are specific to eliciting, specifying, analysing and validating security requirements. Specific techniques that are relevant to security requirements, such as the development of misuse/abuse cases, attack trees and specification techniques are also discussed or referenced.
	Risk Management

	[https://buildsecurityin.us-cert.gov/articles/best-practices/risk-management] - A framework for identifying, tracking and managing software risks is provided. Best practices associated with software risk management are presented, together with content that discusses understanding software risks in a business context, identifying business and technical risks, prioritising business and technical risks, and defining risk mitigation strategies.
	Security Testing [https://buildsecurityin.us-cert.gov/articles/best-practices/security-testing] - The primary objective is to improve the understanding of some of the processes of security testing, such as test vector generation, test code generation, results analysis and reporting. This will help testers to improve the generation of test vectors and increase their confidence when testing security function behaviours.
	Software Assurance [https://buildsecurityin.us-cert.gov/swa/software-assurance-pocket-guide-series] A series of documents on software assurance in acquisition and outsourcing, software assurance in development, the software assurance life cycle and software assurance measurement and information needs.
	System Strategies [https://buildsecurityin.us-cert.gov/articles/best-practices/system-strategies] System complexity is an aggregate of technology, scale, scope, operational, and organisational issues. Business usage, the technologies applied, and the changing operational environment raise software risks that are typically not addressed in current practice. It discusses the effects of the changing operational environment on the development of secure systems. Vulnerability analysis has typically concentrated on errors in coding or in the interfaces among components; however, system interactions can also be a seed bed for vulnerabilities. One article in this content area includes discussions on the software assurance challenges inherent in networked systems development and proposes a structured approach, using scenarios, to analysing potential system stresses.
	Training and Awareness [https://buildsecurityin.us-cert.gov/articles/best-practices/training-and-awareness] This examines current practice in software security training and awareness offerings across the industry. It also briefly describes and compares the commercial sector's training offerings with current academic curricula in some of the US's top universities.
	White Box Testing [https://buildsecurityin.us-cert.gov/articles/best-practices/white-box-testing] This presents best practice in performing white box activities for testing code construction. The activities that provide the basis for white box dynamic analysis include: specifying the operational or expected usage or test profile; specifying key interfaces that feed data into the software; and compiling a list (or partial list) of undesirable output events, for which the software's behaviour should be monitored. Also discussed are strategies for examining the internal structure of a program, statement coverage, decision coverage, condition coverage, decision/condition coverage, and multiple-condition coverage.
Tools	Black Box Testing [https://buildsecurityin.us-cert.gov/articles/tools/black-box-testing] Information is provided about black box testing tools. This term is used to refer

	to tools that take a black box view of the system under test; they do not rely on the availability of software source code and, in general, take an outside-in view of the software, which means that they try to explore the software's behaviour from the outside. The focus is on black box testing technologies that are unique to software.
	Modelling Tools [https://buildsecurityin.us-cert.gov/articles/tools/modeling-tools]This provides an introduction to modelling in the context of security analysis and discusses how tools can support security analysis during development. A model is an abstract representation of an object. The decomposition of a system might be grouped into components and their dependencies. A model can demonstrate the consistency of the system specifications or be a predictor of system behaviour. The analysis of system performance in data throughput or computation efficiency, so as to meet critical real-time performance requirements, depends on how that aspect of system behaviour is modelled.
	Penetration Testing Tools [https://buildsecurityin.us-cert.gov/articles/tools/penetration-testing-tools] Information about penetration testing tools is provided.
	Source Code Analysis [https://buildsecurityin.us-cert.gov/articles/tools/source-code-analysis]- Outlines what automated security analysers can do, provides a business case for their use, and provides some criteria for evaluating individual tools. Code samples are provided for running tools against, in order to verify that the tools are able to detect known problems in the code.
Knowledge	Assurance Cases [https://buildsecurityin.us-cert.gov/articles/knowledge/assurance-cases] This introduces the concepts and benefits of creating and maintaining assurance cases for security. A security assurance case uses a structured set of arguments and a corresponding body of evidence to demonstrate that a system satisfies specific claims with respect to its security properties.
	Attack Patterns [https://buildsecurityin.us-cert.gov/articles/knowledge/attack-patterns] These articles discuss the concept of attack patterns as a mechanism for capturing and communicating the attacker‟s perspective. Attack patterns are descriptions of common methods of exploiting software.
	Business Case Models [https://buildsecurityin.us-cert.gov/articles/knowledge/business-case-models] - This presents a conceptual framework for quantifying the cost and benefits of investments in secure coding techniques. Guidance on implementing the framework will include the variables and data elements to focus on and the means of measuring and quantifying them. With these measurements, one can calculate the economic benefits (cost) of these investments. Details are also provided on current practice and current research on the case for secure coding techniques.
	Coding Practices [https://buildsecurityin.us-cert.gov/articles/knowledge/coding-practices] Describes methods, techniques, processes, tools and runtime libraries that can prevent or limit exploits against vulnerabilities. Each document describes the development and technology context in which the coding practice is applied, as well as the risk of not following the practice and the type of attacks that could result.

	Coding Rules [https://buildsecurityin.us-cert.gov/bsi/76-BSI.html].Coding rules are representations of knowledge gained from real-world experience of potential vulnerabilities that exist in programming languages like C and C++. Creating and using software with a given coding environment enables the discovery of, and learning about, vulnerabilities that exist in this environment, how to recognise whether they crop up in our code and how to fix them. Coding Rules are the codification of this knowledge. They help software developers, whether manually or in conjunction with tools, to discover, explore, remove and eventually prevent security vulnerabilities in their code.
	Guidelines [https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/guidelines.html] This provides information and data for educating software development professionals on the concept, applicability and value of design guidelines. In addition, this section collects, and makes available, a set of Design Guidelines to assist software development professionals with identifying and removing potential vulnerabilities in software systems. They are building, as well as developing, more mature and security-knowledge-aware design practices for future software systems.
	Lessons Learned [https://buildsecurityin.us-cert.gov/articles/knowledge/lessons-learned] This describes the lessons learned as a result of actual project experience. Lessons learned can be both positive and negative, providing both the opportunity to learn about techniques and approaches that can be followed on future projects and about the pitfalls to avoid.
	Principles [https://buildsecurityin.us-cert.gov/articles/knowledge/principles].This provides information and data for educating software development professionals on the concept, applicability and value of software security principles. It also contains a set of key secure software principles that will help software development professionals analyse and create their software architectures from a security perspective and gain a greater understanding of the key underlying concepts and patterns that, depending on how they are addressed, can make software either more, or less, secure.
	SDLC Process [https://buildsecurityin.us-cert.gov/articles/knowledge/sdlc-process] This discusses the application of software assurance best practice in the context of various SDLC methodologies.

5.3.3 Software Assurance Metrics and Tool Evaluation (SAMATE)

SAMATE è un'iniziativa US Government software assurance, un progetto inter-agenzie tra gli Stati Uniti e il DHS National Institute of Standards and Technology (NIST). Il suo obiettivo è quello di migliorare la garanzia software: (i) sviluppando metriche e metodologie per valutare i tool di sicurezza del software; (ii) identificando le vulnerabilità relative alla pratiche di codifica e dei metodi di ingegneria del software.

Il progetto di riferimento di SAMATE sviluppa casi di test al fine di esaminare il codice sorgente di strumenti e applicazioni. Rileva e segnala le debolezze, in modo da fornire agli utenti finali e sviluppatori tool di garanzia del software con una serie di flaws noti attraverso i quali valutare i propri tool.

L'uscita principale di questa iniziativa è il SAMATE Reference Dataset (SRD), che è un database online alimentato regolarmente da SAMATE. Questa banca dati online, a disposizione del pubblico, fornisce casi di test per gli sviluppatori e utenti finali, attraverso i quali è possibile effettuari valutazioni di tool di sicurezza.

URL	http://samate.nist.gov
Country of HQ location **Geographic Scope**	US National
Type	Governement

SAMATE è finalizzato al miglioramento del software assurance attraverso lo sviluppo di metodologie che consentano la valutazione software dei tool, misurare l'efficacia dei tool e delle tecniche, individuare le lacune negli strumenti e nei metodi. Il progetto sostiene Tools Software Assurance della US DHS e R&D Requirements Identification Program (in particolare, la Parte 3, tecnologia -strumenti e requisiti-), che affronta l'individuazione, la valorizzazione e lo sviluppo di software assurance tools.

Il progetto SAMATE compone di due parti:

- sviluppo di metriche per l'efficacia dei software security assessment (SSA) tools
- valutazione di metodi e strumenti SSA attuali al fine di individuare le carenze che possono portare a guasti dei prodotti software e vulnerabilità

Infine, SAMATE sta sviluppando anche alcune specifiche rivolte agli sviluppatori di strumenti di garanzia del software, che gli consentano di classificare e valutare questa tipologia di tool.

Risultati più significativi:

Specifications	Source Code Security Analysis [https://samate.nist.gov/index.php/Source_Code_Security_Analysis.html] Specifications and test plans for source code security analyser tools. This type of tool examines source code in order to detect and report weaknesses that can lead to security vulnerabilities.
	Web Application Scanner Specification [https://samate.nist.gov/index.php/Web_Application_Scanner.html] "Web Application Scanner Functional Specification Version 1.0". These specifications are brought together in NIST Special Publication 500-269 [https://samate.nist.gov/docs/webapp_scanner_spec_sp500-269.pdf].
Test Cases	SAMATE reference datasheet [https://samate.nist.gov/SRD/] Provides users, researchers, and software security assurance tool developers with a set of known security flaws. These will allow end users to evaluate tools, and tool developers to test their methods.
	SRD database [https://samate.nist.gov/SRD/view.php] A collection of test cases aimed at detecting code weaknesses.

5.3.4 Common Weakness Enumeration (CWE)

CWE è un'iniziativa sostenuta e co-sponsorizzata dalla NCSD della US DHS e il NIST. Attualmente è mantenuta e guidata da MITRE Corporation.

Il CWE è una lista formale o tassonomia, che classifica le tipologie più comuni di vulnerabilità del software. Gli obiettivi principali di CWE sono:

- Gestire la *common taxonomy* per la classificazione delle vulnerabilità comuni del software relativamente ad architettura, progettazionem e codice;
- Fornire una classificazione standard per tool di protezione del software
- Fornire una linea di base da cui partire per aiutare la community SSE ad identificare, attenuare e prevenire questo tipo di debolezza software.

URL	http://cwe.mitre.org/ http://nvd.nist.gov/cwe.cfm
Country of HQ location **Geographic Scope**	US National
Type	Government

Questo progetto utilizza i risultati del progetto SAMATE per creare l'elenco CWE delle vulnerabilità e la sua tassonomia associata e l'albero di classificazione (vedi figura sotto tratta dal NIST).

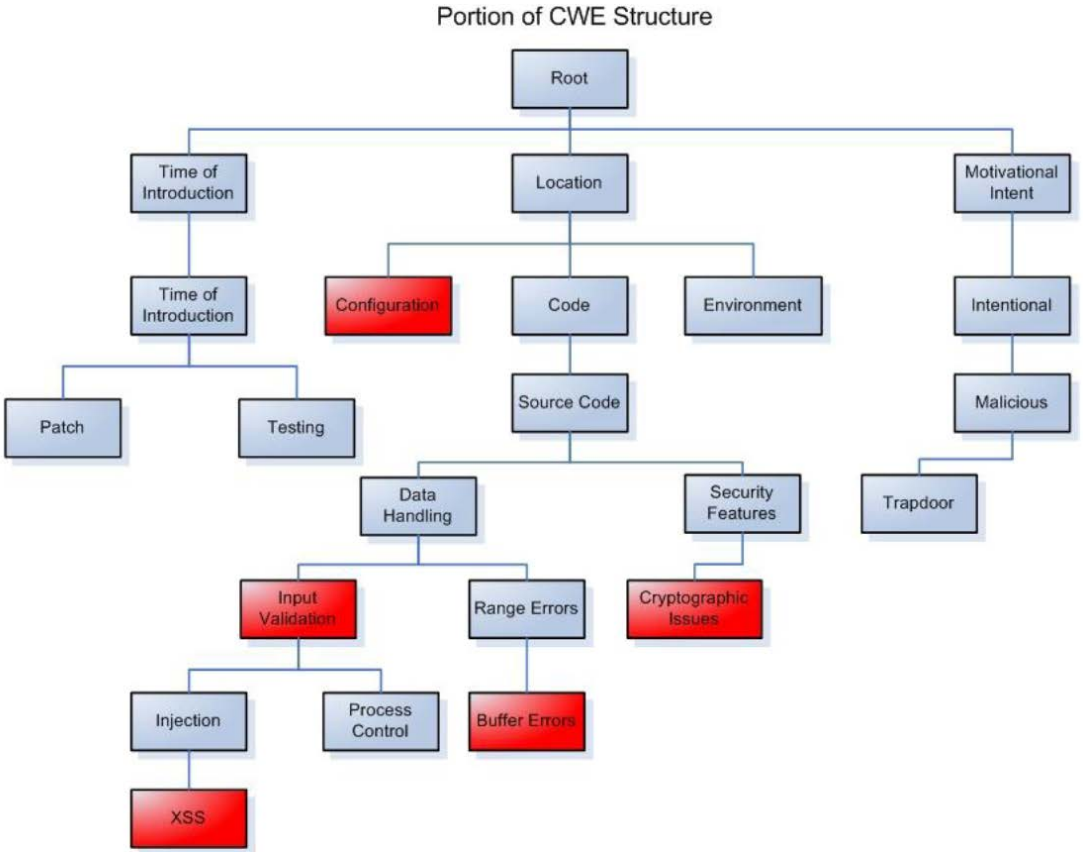


Figura 3: Una porzione dell'albero di classificazione CWE

Va inoltre sottolineato che CWE è una community-developed, l'elenco formale delle vulnerabilità comuni del software coinvolgono il mondo accademico, il settore commerciale e il governo degli Stati Uniti.

Risultati più rilevanti:

CWE List [http://cwe.mitre.org/data/index.html]

Le definizioni e le descrizioni di CWE supportano la scoperta delle tipologie di flaw di sicurezza software nel codice, prima di rilasciarlo. Ciò significa che sia gli utilizzatori che gli sviluppatori dei tool e dei servizi di sicurezza software possono utilizzare CWE come un meccanismo per descrivere i flaw di sicurezza del software.

L'elenco CWE è disponibile in tre diversi formati:

- dizionario ad alto livello delle vulnerabilità individuate
- visualizzazione dell'albero di classificazione, che può fornire l'accesso alle vulnerabilità attraverso i livelli di classificazione
- la visualizzazione grafica della struttura di cui al punto precedente per comprendere meglio le vulnerabilità.

5.3.5 Common Attack Pattern Enumeration and Classification (CAPEC)

CAPEC è un'iniziativa co-sponsorizzata dal NCSD dell'US DHS e guidata dalla Cigital[6]. Costruttori di software sicuro devono proteggersi da importanti vulnerabilità potenziali. Per identificare e mitigare le vulnerabilità relative al software, la community di sviluppo ha bisogno di capire la prospettiva dell'attaccante e gli approcci utilizzati per sfruttare il software.

Gli schemi di attacco sono le descrizioni di metodi comuni per lo sfruttamento del software, fornendo sia la prospettiva che la guida dell'attaccante sui modi per mitigare il loro effetto. Essi derivano dal concetto di pattern design applicato in un distruttivo, piuttosto che costruttivo, contesto e sono generati da un'analisi approfondita di specifici esempi di casi del mondo reale.

Questa iniziativa mira a fornire un catalogo a disposizione del pubblico di schemi di attacco, insieme ad uno schema di classificazione e tassonomia completo. La filosofia è quella di evolvere il catalogo con la partecipazione e i contributi pubblici e così consolidare un meccanismo standard per l'identificazione, la raccolta, la raffinazione, e la condivisione di modelli di attacco nella community software.

URL	http://capec.mitre.org
Country of HQ location **Geographic Scope**	US National
Type	Government

Secondo questa iniziativa, le informazioni sugli schemi di attacco, se catturati in modo formale, possono portare un notevole valore per considerazioni di sicurezza del software attraverso tutte le fasi del SDLC e le altre attività relative alla sicurezza, tra cui:

- Requirements gathering *Identification of relevant security requirements, misuse and abuse cases*
- Architecture and design *Provide context for architectural risk analysis and guidance for security architecture*
- Implementation and coding *Prioritise and guide activities of secure code review*
- Software testing and quality assurance *Provide context for appropriate risk-based and penetration testing*
- Systems operation *Leverage lessons learned from security incidents into preventative guidance*
- Policy and standard generation - *Guide the identification of appropriate prescriptive organisational policies and standards"*

[6] www.cigital.com

6 LA SICUREZZA IN TUTTE LE FASI DEL CICLO DI SVILUPPO DEL SOFTWARE

6.1 Secure SDLC

Generalmente gli aspetti di sicurezza sono sottovalutati fin dalle prime fasi del ciclo di vita dello sviluppo software e di conseguenza sono molte le vulnerabilità che vengono introdotte e trasmesse negli stadi successivi. È stato stimato, ad esempio, che un errore introdotto nella fase di specifica dei requisiti, se non rimosso immediatamente, può costare fino a 200 volte in più correggerlo nelle successive fasi di sviluppo. L'attuazione corretta e completa delle **attività di sicurezza** nelle prime fasi di fatto consente di incrementare sensibilmente il livello di sicurezza di ogni singola fase successiva con un ritorno non indifferente:

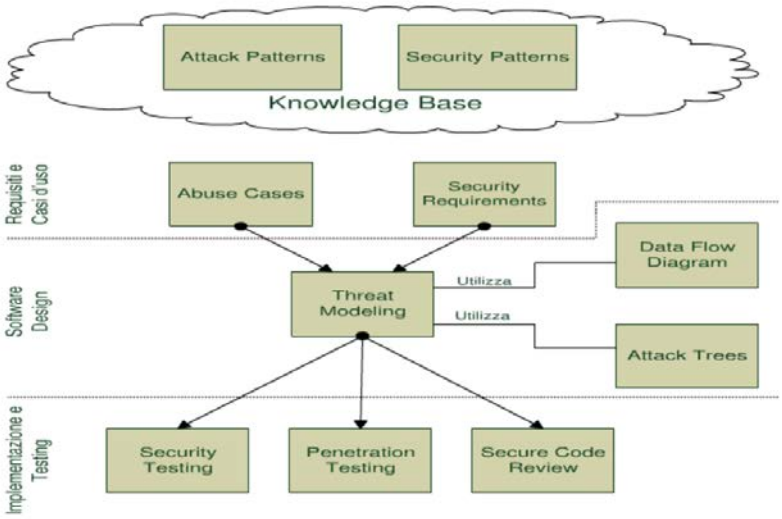


Figura 4 – Secure development activities

Un **Secure Software Development Life Cycle (SSDLC)** considera e implementa opportune attività di sicurezza nel corso di tutte le fasi del processo SDLC, come illustrato nella figura che segue:

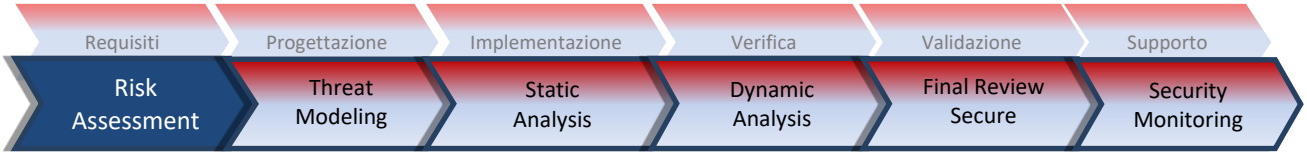


Figura 5: Modello fasi SSDLC

Requisiti: in questa fase vengono effettuate le analisi dei requisiti di sicurezza, dei rischi, delle probabilità di impatto delle minacce, dei casi di abuso tramite rappresentazione UML. In questa fase inoltre si considerano le best practices di carattere generale nella definizione dei requisiti di sicurezza;

Progettazione: in questa fase si esamina il sistema in divenire con l'ausilio di tecniche di analisi e modellazione delle minacce, producendo requisiti di sicurezza di dettaglio che si aggiungono a quelli prodotti dalla precedente fase;

Implementazione: in questa fase si realizza il sistema attraverso la programmazione di codice sicuro, test di sicurezza basati sull'analisi delle minacce ed analisi statica del codice sorgente. Quest'ultima può produrre nuovi requisiti di sicurezza che indirizzano la revisione del codice;

Verifica: in questa fase si analizzano gli aspetti di sicurezza del sistema in esecuzione in un ambiente controllato impiegando tecniche e strumenti di analisi dinamica;

Validazione: è la fase immediatamente prima del rilascio, viene effettuta una final security review per al verifica del rispetto dei requisiti dati;

Supporto: in questa fase si esamina il sistema in essere con l'ausilio di tecniche di: analisi e modellazione delle minacce e/o verifica statica/dinamica del codice applicativo, al fine di produrre nuovi requisiti di sicurezza di dettaglio che indirizzano una eventuale fase di reingegnerizzazione e/o di patching del sistema in oggetto.

6.2 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.

6.2.1 Linguaggi per la specifica dei requisiti

Un linguaggio di specifica in ambito security 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, and UMLintr) e i requisiti di sicurezza (UMLsec, SecureUML, Secure Tropos, and Misuse Cases),
- un *attack specification language* (e.g., STATL and Snort Rules).

UMLsecè un'estensione di UML per lo sviluppo di sistemi sicuri e usa stereotypes, tags e constraints per specificare i requisiti di sicurezza. Gli stereotipi servono come etichette per gli elementi del modello UML per introdurre informazioni al modello e specificare i vincoli che devono essere soddisfatti dal modello. I tag sono associati con gli stereotipi e sono utilizzati per specificare in modo esplicito una semplice proprietà di un elemento del modello. UMLsec definisce 21 stereotipi da utilizzare per rappresentare i seguenti requisiti di sicurezza:

- fair exchange (non barare tra le parti),
- 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 stereotipi hanno associati tag e nove stereotipi hanno vincoli. Questi stereotipi possono essere utilizzati per i diagrammi dei casi d'uso, i diagrammi delle classi, diagrammi di stato, diagrammi di attività, diagrammi di sequenza, diagrammi e implementazioni per specificare i requisiti di sicurezza in un modello UML (per entrambe le specifiche relative ai requisiti e al design). Un insieme di **tools** sono stati

realizzati per consentire agli sviluppatori la modellazione attraverso l'impiego di UMLsec e quindi di verificare questi modelli (utilizzando il model checking).

SecureUML SecureUML è un'altra estensione di UML che si concentra sulla specifica delle politiche di controllo degli accessi basati sui ruoli (queste politiche possono essere considerati come requisiti di sicurezza) in un modello. SecureUML propone nove stereotipi 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 per le risorse, le azioni e le autorizzazioni. Contrariamente a UMLsec, questi vincoli possono essere specificati in base alle esigenze del singolo componente software.

Snort Rules 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 ad 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 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 ad un'altra *secure resource*). Un *actor* può dipendere da un altro *actor* per raggiungere un *goal/soft goal*, per svolgere un *task*, o rilaasciare 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 è un 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 maliziosi). *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 Un altro modo per specificare il comportamento indesiderato di un pezzo di software utilizzando i diagrammi UML è quello 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 è un'estensione di UML che utilizza stereotipi 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 stereotipato. Ci sono tre stereotipi definiti per le classi e dodici per lo use case diagram. Gli stereotipi per le classi hanno anche i tag.

Abstract State Machine Language (AsmL)ASML è un linguaggio a stati finiti machine-based eseguibile utilizzato anche per specificare scenari di attacco. In generale, attacchi con step multipli possono essere specificati in ASML. Tali scenari di attacco possono essere tradotti automaticamente in *Snort rules* che possono poi essere utilizzati con un'estensione di IDS Snort. Tali scenari di attacco sono 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 è un'estensione di ASML sviluppata per specificare scenari di attacco. AsmLSec utilizza stati, eventi e transizioni per rappresentare gli attacchi. Ogni transizione ha una 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.

UML State Charts for Security i diagrammi di stato UML (senza alcuna estensione) sono stati utilizzati per specificare gli attacchi. Gli attacchi specificati nei diagrammi di stato possono essere collegati alle Snort rules. Questi diagrammi di stato possono essere tradotti manualmente nelle Snort rules e quindi possono essere poi utilizzati con un'estensione di IDS Snort. Attraverso l'impiego dei diagrammi di stato, è possibile rappresentare attacchi complessi con step multipli che normalmente non possono essere rappresentati con ordinarie regole di Snort rules.

STATL è un linguaggio di specifica a stati finiti machine-based eseguibile. STATL utilizza due costrutti principali per specificare un attacco: Stato e transizione. Ogni transizione deve avere un evento associato che, quando si verifica, avvia la transizione. Le transizioni hanno anche azioni facoltative che vengono eseguite una volta che una transizione è avviata. Stato e transizione specifiche possono anche avere il codice eseguibile al loro interno. Un ambiente di sviluppo per STATL è inoltre disponibile e può essere utilizzato, tra le altre cose, per visualizzare scenario di attacco specificati come macchina a stati.

6.2.2 Tool per la specifica dei requisiti

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Requirements Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Analyst Pro	Requirements management	Requirements		http://www.analysttool.com
aNimble	Requirements management	Requirements	Free	http://sourceforge.net/projects/nmble/
CaseComplete	Requirements management	Requirements		http://casecomplete.com
Code Assure Solo	Requirements management	Requirements		
GMARC	Requirements management	Requirements		
IBM DOORS Next Generation	Requirements management	Requirements		http://ibm.com
IBM Rational RequisitePro solution	Requirements management	Requirements		http://ibm.com
IrqA	Requirements management	Requirements		
Objectives	Requirements management	Requirements	Available by	http://www.objectiver.com

			Request	
Open Source Requirements Management Tool (OSRMT)	Requirements management	Requirements	Open Source	http://sourceforge.net/projects/osrmt/
Optimaltrace	Requirements management	Requirements		www.compuware.com/products/optimaltrace
Polarion	Application Lifecycle Management (ALM)	Requirements		http://www.emerasoft.com/agile-application-lifecycle-management/polarion-alm/
Reqtify	Requirements management	Requirements		http://users.reqtify.tni-software.com/?p=home
rmtoo	Requirements management	requirements	Free	http://sourceforge.net/projects/rmtoo/
RTD	Requirements management	Requirements		http://www.igatech.com/rdt
RTM	Requirements management	Requirements		http://www.serena.com/Products/rtm/home.asp
SeaMonster	Requirements management	Requirements		https://sourceforge.net/projects/seamonster/
TcSE (Teamcenter Systems Engineering)	Requirements management	Requirements		
Telelogic DOORS	Requirements Management	Requirements	Free	http://telelogic-doors.software.informer.com/

6.2.2.1 Tool per l'analisi del rischio

Microsoft Security Assessment Tool (MSAT). E´ uno strumento Microsoft gratuito progettato per aiutare le organizzazioni a valutare i rischi di sicurezza, individuare un elenco di problemi in ordine di priorità e a fornire raccomandazioni specifiche per ridurre al minimo tali rischi.
Lo strumento si basa su un approccio olistico per valutare le condizioni di sicurezza generali e copre aspetti che riguardano gli utenti, i processi e la tecnologia.

MSAT risponde ad una gmma di 200 domande che riguardano l'infrastruttura, le applicazioni, le attività e gli utenti. Le relative risposte e le raccomandazioni si basano su procedure consigliate e comunemente accettate da standard quali ISO 17799 e NIST-800.x, oltre che da raccomandazioni e indicazioni del Microsoft Trustworthy Computing Group e di altre fonti di protezione esterne.

Lo strumento genera un profilo del rischio aziendale **(BRP, Business Risk Profile)**, misurando il rischio dell'attività in base al modello aziendale e di settore, ed un indice delle capacità difensive, dato dalla sovrapposizione delle diverse misure di protezione, detto **Indice di Difesa in Profondità (DiDI, Defense-in-Depth Index)**. I valori BRP e DiDI vengono quindi confrontati per misurare la distribuzione dei rischi nelle varie aree di analisi: infrastruttura, applicazioni, attività e utenti.

6.3 Progettazione

La fase di progettazione identifica i requisiti generali e la struttura per il software. In questa fase viene definita l'architettura di sicurezza e le linee guida di progettazione; vengono documentati gli elementi della superficie d'attacco; vengono modellate le minacce.

6.3.1 Secure Design Languages

Molti dei linguaggi per specificare i requisiti di sicurezza sono utilizzati anche per le specifiche di design. Ciò è dovuto al fatto che i requisiti di basso livello sono davvero vicini alla progettazione statica e dinamica.

Questi linguaggi (ad esempio, UMLsec, SecureUML, e SecureTropos) sono già stati discussi nella Sezione precedente. Ci sono due principali punti che dovrebbero essere considerati nella scelta di un linguaggio di design sicuro; essi sono:

- la varietà di schemi disponibili per rappresentare un disegno da vari aspetti e livelli di astrazione
- la disponibilità degli strumenti.

UMLsec fornisce una varietà di schemi e ha strumenti disponibili.
SecureUML può essere utilizzato anche per la progettazione di software sicuro; tuttavia, si limita a rappresentare solo nozioni di controllo degli accessi basati sui ruoli in un diagramma delle classi UML.
Sicure Tropos propone di utilizzare gli Agent UML capability diagrams. Questi schemi sono simili ai diagrammi di attività UML (piano e capacità) e diagrammi di sequenza (interazione agente).

6.3.2 Software Design Tools

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Design Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Coras	Threat Modeling tool/practies	Design	Open Source	coras.sourceforge.net
Microsoft Threat Modeling Tool	Threat Modeling tool	Design	Free	https://www.microsoft.com
MyAppSecurity ThreatModeler	Threat Modeling tool	Design	Available by Request	myappsecurity.com
TRIKE	Threat Modeling tool/practies	Design	Open Source	http://octotrike.org/tools.shtml

6.4 Implementazione

Durante questa fase il team di sviluppatori mette in atto le contromisure secondo le specifiche della fase precedente ed effettua dei test sul codice sorgente per verificare l'assenza di security flaws.

6.4.1 Software Implementation Tools

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Implementation Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
BRAKEMAN	SAST	Implementation	Open Source	https:// brakemanscanner.org
Burp Suite by PortSwigger	SAST, DAST, Penetration Testing	Implementation / Verification	Free Tier	https:// portswigger.net

Checkmarx	SAST, DAST, RASP	Implementation / Verification	Available by Request	https://www. checkmarx.com
Cigital	SAST, DAST	Implementation / Verification	N/A	https://www.cigital.com
CodeDx	SAST, DAST	Implementation / Verification	Available by Request	https://codedx.com
CodeProfiler by Virtual Forge	SAST	Implementation	Available by Request	https://www.virtualforge.com
Contrast Enterprise	IAST, RASP	Implementation / Verification	Available by Request	https://www.contrastsecurity.com
CppCheck	SAST	Implementation	Open Source	cppcheck.sourceforge.net
Dependency Checker	Library Inspection	Implementation	Open Source	https://www.owasp.org
FindBugs	SAST	Implementation	Open Source	findbugs.sourceforge.net
FxCop	SAST	Implementation	Open Source	www. microsoft.com
HP Fortify Static Code Analyzer	SAST, DAST, IAST, RASP	Implementation / Verification	Available by Request	www.hp.com
IBM Security AppScan	SAST, DAST, IAST	Implementation / Verification	Available by Request	https://www.ibm.com
JSHint	SAST	Implementation	Open Source	jshint.com
Gendarme	SAST	Implementation	Open Source	www.mono-project.com/Gendarme
MetaFlows	Cloud Security Scanning	Implementation	14 Day Free Trial	https://www.metaflows.com
Metascan by OPSWAT	SAST	Implementation	Available by Request	https://www opswat.com
Microsoft BinScope	SAST	Implementation	Free	www.microsoft.com
Microsoft Code Analysis Tool	SAST	Implementation	Free	www.microsoft.com
Microsoft FxCop	Library Inspection	Implementation	Free	www.microsoft.com
Microsoft SDL Regex Fuzzer	SAST	Implementation	Free	www.microsoft.com
Microsoft SDL MiniFuzz File Fuzzer	SAST	Implementation	Free	www.microsoft.com
ModSecurity	WAF	Implementation / Verification	Open Source	https://www.modsecurity.org

N-Stalker Cloud Web Scan	SAST, DAST	Implementation / Verification	Free Tier Available	https://www.nstalker.com
OWASP Dependency Check	SAST	Implementation	Open Source	www.owasp.org
PMD	SAST	Implementation	Open Source	https://pmd.github.io
PYLINT	SAST	Implementation	Open Source	https://www.pylint.org
Risk Fabric by Bay Dynamics	Predictive Security Analytics	Implementation / Verification / Response	Available by Request	https://baydynamics.com
RSA ECAT by EMC	DAST	Implementation / Verification	Available by Request	https://www.emc.com
Security AppScan by IBM	SAST, DAST, IAST	Implementation / Verification	Available by Request	https://www.ibm.com
SiteLock TrueCode SAST	SAST, DAST	Implementation / Verification	Available by Request	https://www.sitelock.com
SonarLint	SAST	Implementation	Open Source	https://www.sonarlint.org
SonarQube	SAST	Implementation	Open Source	https://www.sonarqube.org
Symantec Advanced Threat Protection	IAST, RASP	Implementation / Verification	60 Day Free Trial	https://www symantec.com
Tanium Endpoint Platform	Endpoint Security, App Security Scanning	Implementation / Verification	Available by Request	https://www tanium.com
Trend Micro Deep Security Platform	SAST, DAST	Implementation / Verification	N/A	https://www.trendmicro.com
Tripwire Enterprise	IAST, RASP	Implementation / Verification	Available by Request	https://www.tripwire.com
Veracode Cloud Platform	SAST, DAST, Mobile AST, Penetration Testing	Implementation / Verification	Available by Request	https://www.veracode.com
WhiteHat Sentinel	SAST, DAST	Implementation / Verification	30 Day Free Trial	https://www.whitehatsec.com

6.5 Verifica

Prima della fase di rilascio definitiva del software i team che lavorano in sicurezza effettuano un ulteriore verifica del codice elaborato mediante test di sicurezza. I test di sicurezza mirano a controllare la vulnerabilità delle superficie di attacco, in modo da agire in via preventiva alla correzione di eventuali problemi che potrebbero verificarsi in fase di rilascio.

6.5.1 Software Verification Tools

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Verification Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Acunetix Web Vulnerability Scanner	DAST, IAST	Verification	14 Day Free Trial	www.acunetix.com
Adallom	Cloud Access Security Broker	Verification	Available by Request	adallom.com
AppSpider Pro by Rapid7	DAST	Verification	Available by Request	https://www.rapid7.com
Appthority	Mobile AST	Verification	Available by Request	https://www.appthority.com
AuditMyApps by Pradeo	Mobile AST	Verification	Available by Request	https://auditmyapps.com
Backtrack-linux	Penetration Testing	Verification	Open Source	www.backtrack-linux.org
BeEF	Penetration Testing	Verification	Open Source	beefproject.com
Bit9 + Carbon Black	Endpoint Security	Verification / Response	Available by Request	www.bit9.com
Black Duck Hub	Open Source Scanning	Verification	Available by Request	https://www.blackducksoftware.com
BrightCloud Threat Intelligence by Webroot	DAST	Verification	N/A	https://www.brightcloud.com
Checkmarx	SAST, DAST, RASP	Implementation / Verification	Available by Request	www.checkmarx.com
CloudSOC by Elastica	Cloud Security Testing/Scanning	Verification	Free Risk Assessment	https://www.elastica.net
CodeDx	SAST, DAST	Implementation / Verification	Available by Request	https://codedx.com
ContextIntelligence by Yottaa	CDN, DDoS Protection, WAF	Verification	N/A	www.yottaa.com
Defendpoint by Avecto	Endpoint Security	Verification / Response	Available by Request	https://www.avecto.com
Falcon Host by CrowdStrike	Endpoint Security	Verification / Response	Available by Request	https://www.crowdstrike.com
Hillstone Networks		Verification	Available by Request	hillstonenet.com
Kali Linux	Penetration Testing	Verification	Open Source	kali.org
Security AppScan by IBM	SAST, DAST, IAST	Implementation / Verification	Available by Request	https://www.ibm.com

LogRhythm Security Intelligence Platform	Predictive Security Analytics	Verification / Response	Available by Request	www.logrhythm.com
Malwarebytes Endpoint Security	Endpoint Security	Verification	N/A	www.malwarebytes.org
Metasploit by Rapid7	Penetration Testing	Verification	Open Source	www.metasploit.com
Microsoft Application Verifier	DAST	Verification	Free	www.microsoft.com
Microsoft Attack Surface Analyzer	Intrusion Prevention	Verification	Free	www.microsoft.com
NetScaler AppFirewall by Citrix	WAF	Verification	N/A	citrix.com
Nevis Security and Compliance Suite by	WAF, Authentication, Identity mngt	Verification	Available by Request	www.adnovum.ch
AdNovum	Management	Verification		
Nikto2	Web Server Scanner	Verification	Open Source	cirt.net
Nmap	Penetration Testing and Network Mapping	Verification / Response	Open Source	www.nmap.org
NSFOCUS Web Application Firewall	DAST, WAF	Verification	N/A	www.nsfocus.com
OWASP Zed Attack Proxy (ZAP)	SAST, DAST/ Penetration Testing	Verification / Response	Open Source	www.owasp.org
PA-7000 Series Firewall by Palo Alto	WAF	Verification	N/A	https://www.paloaltonetworks.com
Networks		Verification		
Peach Fuzzer	Penetration Testing	Verification / Response	Available by Request	www.peachfuzzer.com
Prevoty	RASP	Verification / Response	Available by Request	www.prevoty.com
ProtectWise Cloud Network DVR	CDN, App Security Scanning	Verification	Available by Request	www.protectwise.com
Qualys Security & Compliance Suite	DAST, WAF	Verification / Response	Available by Request	https://www.qualys.com
Samurai Web Testing Framework	DAST, Penetration testing	Verification	Open Source	https://www.samurai-wtf.org
SRX Series Firewall by Juniper	WAF	Verification	N/A	www.juniper.net

Networks				
Sucuri	WAF	Verification	N/A	www.sucuri.net
Thunder TPS by A10 Networks	DDoS Protection	Verification / Response	N/A	https://www.at10networks.com
Trustwave Secure Web Gateway	CDN, DAST	Verification	N/A	www.trustwave.com
Trustwave Web Application Firewall	WAF, Penetration Testing	Verification	N/A	www.trustwave.com
Veracode	DAST	Verification	Available by Request	www.veracode.com
vSentry by Bromium	Endpoint Security	Verification / Response	Available by Request	www.bromium.com
vThreat Platform	Penetration Testing, App Security Scanning	Verification	Available by Request	www.vthreat.com
Wireshark	Penetration Testing and Packet-level Monitoring	Verification	Open Source	www.wireshark.org

6.6 Validazione

Durante questa fase il software è oggetto di una Final Security Review finalizzato a stabilire se il software soddisfa tutti i requisiti di sicurezza individuati nella fase iniziale del progetto.

In questa fase si verifica, inoltre, che i bug di sicurezza precedentemente identificati siano stati risolti e che il SW sia sufficientemente robusto di fronte a nuove vulnerabilità.

Le azioni di sicurezza di questa fase possono essere così sintetizzate:

- **Software Remediation dopo un'analisi statica (SAST)**
  - Analisi della reportistica e classificazione degli errori, rilevati nella fase di analisi statica del codice;
  - Rimozione degli errori di sicurezza legati all'uso di librerie esterne a rischio di vulnerabilità, sostituendole con le versioni sanate;
  - Ristrutturazione delle classi e funzioni identificate come vulnerabili alle varie injection, al cross site scripting, etc.
  - Applicazione delle modifiche nei costrutti sintattici che rendono il software vulnerabile;
  - Considerare i «warning» sulla qualità del codice e procedere con le modifiche;
- **Software Remediation dopo un'analisi dinamica (DAST)**
  - Analisi della reportistica e classificazione degli errori, rilevati nella fase di analisi dinamica del codice.
  - Rimozione degli errori messi in evidenza dal fuzzy testing, ad esempio aumentando i controlli applicativi.

- Correzioni degli errori, eventualmente tramite implementazione di nuove funzioni, per esempio aggiungendo meccanismi di autenticazione o rivedendo la struttura delle classi e funzioni.

- Definizione di un **Incident Responce Plan** cioè produrre la documentazione contenente le istruzioni per rispondere e limitare gli effetti di un incidente di sicurezza.
- Security Review: configurazione finale, aggiornamento delle procedure di sicurezza, certificazione del rilascio software, testing e archiviazione.

Figura 6: Input ed Output della fase Final Review - Secure Release

6.6.1 Software Release Tools

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Release Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Armor Complete	Cloud Security Platform	Release	Available by Request	https://www.armor.com

6.7 Supporto

La fase di Supporto è tutto ciò che concerne un assistenza successiva alla fase di rilascio. Questa fase nasce per supportare tutte le evoluzioni in materia di sicurezza che il mercato dinamico informatico introduce per combattere le sempre nuove vulnerabilità software.

Le azioni di sicurezza di questa fase possono essere così sintetizzate:

- **Vulnerability assessment**:
  - esecuzione di test che consentano di individuare vulnerabilità, assegnazione della priorità/severità, definizione del Remediation Plan;
  - produzione di reportistica di sintesi e di dettaglio;
- **Data Loss/Leak Prevention**:
  - rilevazione dei dati che transitano nell'organizzazione, ovunque siano archiviati, analisi e classificazione;
  - creazione di regole predefinite per la protezione dei dati, per assicurarsi che siano usati in conformità con le politiche di privacy e sicurezza;
  - generazione automatica di alert nel caso in cui vengano violate le policy di sicurezza definite, visibilità e controllo sui dati in movimento, sia che si trovino in messaggi e-mail, nella mail sul Web, nell'instant messaging, e nei protocolli di rete;
- **Database Security**:

  - analisi dei database e valutazione dei rischi mediante controlli di vulnerabilità;
  - individuazione delle alterazioni dei dati, degli utenti e dei profili di accesso;
  - arresto in tempo reale delle sessioni che violano le policy, evitando che i dati vengano compromessi;
- **Web Application Firewall Management e Secure Web Gateway**:
  - funzionalità standard firewall (policy enforcement, stateful inspection, packet filtering, NAT, VPN client-to-site e site-to-site);
  - anti-malware e anti-spam;
  - Intrusion Prevention (IPS) per il blocco delle minacce;
- **Patching Update:** notifica, installazione e test di nuovi security improvement packages.

6.7.1 Software Response Tools

Il CATALOGO SECURITY TOOLS 6.8 raccoglie i tool disponibili, divisi per fase del processo SSDLC, che offrono funzionalità applicabili in ambito secure application development.

Si riporta di seguito la tabella 'Software Response Tools':

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Airlock Suite by Ergon Informatik	WAF, Authentication, Identity	Response	Available by Request	https://www.airlock.com
Airlock Suite by Ergon Informatik	Management	Response	Available by Request	
Akamai	CDN, DDoS Protection, WAF	Response	N/A	www.akamai.com
Alert Logic Security-as-a-Service	Intrusion Prevention System, Cloud Access Security Broker, WAF	Response	Available by Request	www.alertlogic.com
Amazon WAF	WAF	Response	N/A	https://www.aws.amazon.com
AppMobi Security Kit	Apache Cordova App Encryption and Authentication	Response	Available by Request	www.appmobi.com
AppWall by Radware	WAF, DDoS Protection	Response	Available by Request	www.radware.com
Arbor Networks APS	DDoS Protection	Response	N/A	https://www.arbornetworks.com
Arxan Application Protection	Anti-Tamper Software	Response	Available by Request	www.arxan.com
Barracuda Firewal	WAF	Response	N/A	www.barracuda.com
Blue Coat Cloud	Cloud Access Security Broker,	Response	Available by	https://www.bluecoat.com

	WAF		Request	
Bluebox	Mobile Access Security Broker	Response	Available by Request	https://www.bluebox.com
CD Protection by CD Networks	CDN, WAF, DDoS Protection	Response	N/A	https://www.cdnetworks.com
CipherCloud	Cloud Access Security Broker	Response	Available by Request	https://www.ciphercloud.com
Cisco ACE WAF	WAF	Response	N/A	www.cisco.com
CloudFlare	CDN, DDoS Protection, WAF	Response	N/A	www.cloudflare.com
CloudFront by Amazon	CDN, DDoS Protection	Response	N/A	https://www.aws.amazon.com
CloudLock Security Fabric	Cloud Access Security Broker	Response	Available by Request	https://www.cloudlock.com
CloudPassage Halo	Cloud Access Security Broker	Response	Available by Request	https://www.cloudpassage.com
DDoS Strike by Security Compass	DDoS Protection	Response	Available by Request	https://www.securitycompass.com
DenyAll WAF	WAF	Response	N/A	www.denyall.com
F5 Big-IP ADC platform	WAF, DDoS Protection	Response	N/A	https://f5.com
FireEye NX	Web Server Scanner, WAF	Response	N/A	https://www.fireeye.com
Fortigate Firewall Platform by Fortinet	WAF	Response	Available by Request	https://www.fortinet.com
FortiWeb by Fortinet	WAF	Response	Available by Request	https://www.fortinet.com
Imperva Incapsula	WAF, DDoS Protection	Response	N/A	www.imperva.com
InfoBlox DNS Firewall	WAF	Response	60 Day Free Trial	www.infoblox.com
Intelligent Next-Gen T-Series Firewall by	WAF	Response	N/A	https://www.hillstonenet.com
Klocwork by Rogue Wave Software	Code Quality Scanning	Response	Available by Request	https://www.klocwork.com
Kona Site Defender by Akamai	WAF, DDoS Protection	Response	N/A	www.akamai.com
Level 3 Content Delivery Network	CDN, DDoS Protection	Response	N/A	www.level3.com

Netsparker Web Application Security Scanner	DAST	Response	Available by Request	www.netsparker.com
Neustar	DDoS Protection	Response	N/A	www.neustar.biz
Palo Alto Enterprise Security Platform	RASP WAF	Response	Available by Request	https://www.paloaltonetworks.com
ProAccel by Bricata	Intrusion Prevention System	Response	Available by Request	www.bricata.com
Sophos Next-Gen Firewall	WAF	Response	30 Day Free Trial	www.sophos.com
Sucuri Website Firewall	WAF, DDoS Protection, App Security Scanning	Response	Available by Request	www.sucuri.net

6.8 Catalogo Security Tools

Il CATALOGO SECURITY TOOLS raccoglie i tool disponibili che offrono funzionalità applicabili in ambito secure application development.

In Appendice 1 viene riportato il Catalogo Security Tools con il seguente formato:

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
nome commerciale del tool	*indica la macro-funzione: per es. DAST, SAST, WAF ecc.*	*la fase del sw life-cycle coperta dal tool*	*tipo licenza*	*indirizzo web per approfondimenti*

Tabella 4 – Struttura del Catalogo Security Tool

6.9 Training e formazione

Le organizzazioni inoltre dovrebbero investire di più anche nello sviluppo di competenze interne sulla base anche del fatto che molti degli attuali problemi di sicurezza derivano da errori di progettazione o di implementazione, risolvibili solo disponendo di personale qualificato. Alcuni analisti affermano che il 64% degli sviluppatori non sono confidenti di poter scrivere applicazioni sicure [fonte: Microsoft Developer Research].

Questa sezione fornisce l'elenco dei corsi più accreditati disponibili a livello internazionale in ambito secure software development.

6.9.1 Secure Coding in C and C++

Il corso è basato su material di Addison-Wesley: "Secure Coding in C and C++" and "The CERT C Secure Coding Standard". Il training SEI può essere offerto anche fuori dall'area statunitense.

URL	http://www.sei.cmu.edu/training/p63.cfm
Country of HQ location	US
Geographic Scope	International
Type	Academic (SEI)

Questo corso fornisce una spiegazione dettagliata di errori di programmazione comuni in C e C ++ e descrive come questi errori possono portare a codice vulnerabile. Il corso si concentra sulle questioni di sicurezza intrinseche dei linguaggi di programmazione C e C ++ e delle librerie associate.

I partecipanti acquisiscono conoscenza sugli errori comuni di programmazione che portano a vulnerabilità del software, come questi errori possono essere sfruttati, e le strategie di mitigazione efficaci per impedire l'introduzione di tali errori. In particolare, i partecipanti acquisiscono competenze in merito a:

- migliorare la sicurezza complessiva di ogni tipo applicazione C o C ++
- contrastare attacchi buffer overflow e stack-smashing che sfruttano la manipolazione logica di stringhe insicure
- evitare vulnerabilità e security flaws derivanti dal non corretto utilizzo delle funzioni di gestione della memoria dinamica
- eliminare i problemi integer-related: integer overflows, sign errors, truncation errors
- usare correttamente le funzioni di output formattato senza introdurre vulnerabilità format-string
- evitare le vulnerabilità di I/O, tra cui condizioni *race conditions*
- evitare I/O vulnerabilities, including race conditions

6.9.2 Writing Secure Code - C++

Questo corso di formazione computer-based spiega quali sono le funzioni di sicurezza principali del linguaggio C ++, come evitare che gli sviluppatori cadano nelle trappole di sicurezza comuni e come costruire applicazioni aziendali sicure e affidabili utilizzando C ++. Gli studenti sono guidati attraverso esempi di codice hands-on che evidenziano i problemi e le soluzioni prescritte.

Il corso ha i seguenti moduli:

- Introduction to Software Security
- Data Protection – in Storage and in Transit
- Authentication
- Authorisation
- Data Validation
- Process Handling
- Error Handling and Exception Management
- Logging and Auditing
- Memory Management

6.9.3 Writing Secure Code - Java (J2EE)

Questo corso di formazione computer-based illustra le caratteristiche chiave di sicurezza della piattaforma J2EE, come evitare che gli sviluppatori cadano nelle trappole di sicurezza comuni e come creare applicazioni web sicure e affidabili utilizzando Java. Gli studenti sono guidati attraverso esempi di codice hands-on che evidenziano i problemi e le soluzioni prescritte.

Il corso ha i seguenti moduli:

- Introduction to Software Security
- Data Protection – in Storage and in Transit
- Authentication
- Authorisation
- Data Validation
- Process Handling
- Error Handling and Exception Management
- Logging and Auditing
- Memory Management

6.9.4 Foundstone (Mcafee) Courses

Foundstone offre un programma di formazione di sicurezza di rete per la creazione di professionisti della sicurezza qualificati.

URL	http://www.foundstone.com
Contact Method	http://www.mcafee.com/us/about/contact-us.aspx Email, web form, phone and address
Country of HQ location	US
Geographic Scope	International
Type	Industry (McAfee)

6.9.5 Threat Modelling

Questo corso di formazione computer-based spiega i processi e i concetti di creazione di software sicuro al fine di designare un quadro di sicurezza, identificando quindi minacce e contromisure. Gli studenti possono apprendere come utilizzare la modellazione delle minacce per migliorare il SDLC.

Il corso ha i seguenti moduli:

- Introduction to Threat Modelling and Hacme Books
- Identify Security Requirements
- Understand the System and the Application
- Identify Threats and Countermeasures
- Post-Threat Modelling Activities

6.9.6 Writing Secure Code - ASP.NET (C#)

Questo corso di formazione computer-based spiega le caratteristiche chiave di sicurezza della piattaforma .NET, come evitare che gli sviluppatori web cadano nelle trappole di sicurezza comuni e quindi come creare applicazioni web sicure e affidabili utilizzando ASP.NET. Gli studenti sono guidati attraverso esempi di codice hands-on che evidenziano i problemi e le soluzioni più idonee.

Il corso ha i seguenti moduli:

- Introduction to Software Security
- Data Protection – in Storage and in Transit
- Authentication
- Authorization
- Data Validation
- Process Handling
- Error Handling and Exception Management
- Logging and Auditing
- Memory Management

6.9.7 Oracle Courses

Oracle University è il principale fornitore di formazione per le tecnologie e i prodotti Oracle. Offre corsi class-based, on-site, virtuali e su CD-ROM, molti dei quali si concentrano sulla programmazione Java o sui prodotti Oracle.

URL	http://education.oracle.com
Contact Method	Education Contact Email and phone
Country of HQ location	US
Geographic Scope	International
Type	Industry (Oracle)

6.9.8 Developing Secure Java Web Services, Java EE 6

Il corso Developing Secure Java Web Services fornisce le informazioni necessarie per progettare, implementare, distribuire e gestire secure web services e web service client utilizzando componenti di tecnologia Java e Java Platform, Enterprise Edition 6 (Java EE 6 della piattaforma).

Gli studenti vengono guidati sulla necessità di garantire servizi web sicuri e sulle sfide associate alla sicurezza dei servizi Web. Gli studenti vengono formati anche sui principali standard di settore e sulle iniziative sviluppate per fornire soluzioni di sicurezza complete per i servizi web; nonchè come applicarli per garantire servizi web sicuri. In particolare, gli studenti imparano come proteggere i servizi Web utilizzando tecnologie application-layer security, transport-layer security e message-layer security, come ad esempio come quelle specificate dalle estensioni di sicurezza WS- *.

Questo corso introduce anche i concetti di gestione delle identità, i driver che stanno dietro le soluzioni di gestione delle identità e le funzioni di Sun Java System Access Manager.

Gli obiettivi del corso sono i seguenti:

- Identify the need to secure web services
- List and explain the primary elements and concepts of application security
- Outline the factors that must be considered when designing a web service security solution
- Describe the issues and concerns related to securing web service interactions
- Analyse the security requirements of web services
- Identify the security challenges and threats in a web service application
- Evaluate the tools and technologies available for securing a Java web service
- Secure web services by using application-layer security, transport-layer security and message-layer security
- Describe the concept of identity and the drivers behind identity management solutions
- Explain the role of Sun Java System Access Manager in securing web services
- Secure web services by using UserName token profile
- Secure web services by relying on Sun Java System Access Manager

Il corso tratta i seguenti argomenti:

- Encapsulating the Basics of Security
- Examining Web Services Security Threats and Countermeasures
- Securing Java Web Services Using JavaEE
- Introduction to Web Services Security
- Web Services Security with JAX-WS and Project Metro

- Authentication in JAX-WS
- Identity Management and OpenSSO

6.9.9 MySQL and PHP - Developing Dynamic Web Applications

Il corso MySQL and PHP - Developing Dynamic Web Applications spiega come sviluppare applicazioni in PHP e come usare MySQL in modo efficiente per le applicazioni. Con un approccio hands-on, questo corso con istruttore migliorerà le capacità di PHP e di come combinarle con collaudate tecniche di gestione di database per creare applicazioni web best-of-breed che siano efficienti, solide e sicure.

Gli obiettivi del corso sono:

- Design web-based applications
- Design schemas based on MySQL
- Use „include files" to make code easier to maintain
- Use PHP 5 and take advantage of its advanced features
- Build applications, following a precise flow
- Authenticate users in a secure way against a database
- Handle errors in your PHP applications efficiently and elegantly
- Write composite queries using JOINs and subqueries
- Use indexing in order to manipulate large amounts of data efficiently
- Use JOINS to extract data from multiple tables
- Use GROUP BY clauses and aggregate functions
- Write applications whose components can be scaled to meet increased demand
- Build a complete application that includes authentication and session management
- Understand how PHP, MySQL and the Apache web server work together to deliver dynamic web content

Il corso tratta i seguenti argomenti:

- PHP Foundations
- MySQL Foundations
- Manage Databases
- Manage Tables
- SQL SELECT Commands
- SQL Expressions
- SQL DML Commands
- SQL JOINS
- MySQL Database-Driven Web-Based Forms
- Session Handling
- Object-Oriented Programming
- Authentication
- Securing PHP and MySQL

6.9.10 Google Gruyere

Google Code University fornisce un ambiente di laboratorio gratuito chiamato Gruyère[7], dove gli studenti possono provare ad hackerare applicazioni web. Gli studenti hanno l'opportunità di fare qualche prova reale di penetrazione, sfruttando esempi reali con complessità crescente. In particolare, gli studenti possono imparare:

[7] http://google-gruyere.appspot.com/

- come un'applicazione web può essere attaccata utilizzando vulnerabilità di sicurezza comune, come le vulnerabilità cross-site scripting (XSS) e cross-site request forgery (XSRF)
- come trovare, correggere ed evitare queste vulnerabilità comuni, e altri bug che hanno impattano sulla sicurezza, come ad esempio denial-of-service, la divulgazione di informazioni o l'esecuzione di codice remoto.

6.9.11 Other Training Courses

OWASP offre materiali di formazione gratuiti, video e presentazioni, e fornisce opportunità di formazione presso le sue conferenze sulla sicurezza delle applicazioni. Si impegna anche con i fornitori di istruzione di terze parti per sviluppare le competenze specialistiche/laurea.

7 CERTIFICAZIONI PROFESSIONALI

7.1 GIAC Secure Software Programmer (GSSP) Certification

GSSP Certification Exam coinvolge l'Istituto SANS, CERT CC, diverse agenzie governative statunitensi e aziende leader negli Stati Uniti, Giappone, India e Germania. SANS è il certificatore.

URL	http://www.sans-ssi.org/certification/

Questa certificazione si concentra sulle questioni reali che stanno dietro le vulnerabilità più comuni e i problemi di sicurezza applicativi. Gli esami riguardano tecniche e il linguaggi specifici (Java o C #) e molte delle domande usano esempi di codice reale. Gli esami aiutano le organizzazioni a soddisfare quattro obiettivi, che sono:

- identificare carenze nella conoscenza della sicurezza dei programmatori in-house e aiutare gli individui a colmare il divario;
- assicurarsi che i programmatori in outsourcing abbiano adeguate competenze Secure-coding;
- nominare nuovi dipendenti che non hanno bisogno di formazione correttiva in programmazione sicura;
- assicurarsi che ogni grande progetto di sviluppo abbia almeno una persona con avanzate capacità di programmazione sicura.

Dopo l'acquisizione di questa certificazione, i programmatori saranno a conoscenza dei difetti più comuni di sicurezza che si trovano in ambienti di programmazione specifici (Java o .NET), e sapranno come evitare questi problemi dovuti principalmente alla vulnerabilità delle applicazioni.

La certificazione GSSP rimane valida per quattro anni.

7.2 International Council of E-Commerce Consulatants (EC-Council) Certifications

L'EC-Council è un'organizzazione member-based che certifica gli individui in varie competenze e-business e di sicurezza delle informazioni.

URL	http://www.eccouncil.org
Contact Method	http://www.eccouncil.org/contact_us.aspx Email, web form, phone and address
Country of HQ	US

location	
Geographic Scope	International
Type	Industry

I diversi tipi di certificazione offerti dal EC-Council nelle aree SSE-correlate sono descritti nelle sezioni che seguono.

7.3 Certified Ethical Hacker (CEH)

La brochure CEH afferma che "Il Programma CEH certifica gli individui nella specifica disciplina di protezione della rete di Ethical Hacking dal punto di vista vendor-neutral" e "Un CEH è un professionista qualificato che capisce e sa guardare le debolezze e le vulnerabilità nei sistemi di destinazione e utilizza le stesse conoscenze e gli strumenti di un hacker malintenzionato ".

CEH dispone di 26 moduli, di cui i seguenti sono collegati a SSE:

- Module 17: Web Application Vulnerabilities
- Module 19: SQL Injection
- Module 24: Buffer Overflows
- Module 26: Penetration Testing Methodologies

7.4 Certified Security Analyst (ECSA)

La brochure ECSA afferma che questa certificazione completa la certificazione CEH (vedi sopra) "esplorando la fase analitica di hacking etico" e "ECSA prende un ulteriore passo avanti, esplorando come analizzare l'esito di questi strumenti e tecnologie. Attraverso metodie tecniche di test di penetrazione della rete ground-breaking, la classe ECSA aiuta gli studenti a effettuare le valutazioni necessarie per identificare e mitigare i rischi per la sicurezza delle infrastrutture ".

L'obiettivo di ECSA è quello di "aggiungere valore ai professionisti della sicurezza, aiutandoli ad analizzare i risultati dei loro test".

ECSA ha 47 moduli, di cui i seguenti sono collegati a SSE:

- Module 10: Advanced Exploits and Tools
- Module 11: Penetration Testing Methodologies
- Module 27: Stolen Laptop, PDAs and Cellphones Penetration Testing
- Module 28: Application Penetration Testing
- Module 40: Security Patches Penetration Testing
- Module 41: Data Leakage Penetration Testing
- Module 42: Penetration Testing Deliverables and Conclusion
- Module 43: Penetration Testing Report and Documentation Writing
- Module 44: Penetration Testing Report Analysis
- Module 45: Post-Testing Actions

7.5 Certified Secure Programmer (ECSP)

La certificazione ECSP è destinata ai programmatori che sono responsabili per la progettazione e la costruzione di applicazioni sicure basate su Web Windows / con .NET / Java Framework. È progettato per gli sviluppatori che hanno competenze nello sviluppo C #, C ++, Java, PHP, ASP, .NET e SQL.

ECSP dispone di 33 moduli, di cui i seguenti sono collegati a SSE:

- Module 01: Introduction to Secure Coding
- Module 02: Designing Secure Architecture
- Module 03: Cryptography

- Module 04: Buffer Overflows
- Module 05: Secure C and C++ Programming
- Module 06: Secure Java and JSP Programming
- Module 07: Secure Java Script and VBScript Programming
- Module 08: Secure Microsoft.NET Programming
- Module 09: Secure PHP Programming
- Module 10: Securing Applications from Bots
- Module 11: Secure SQL Server Programming
- Module 12: SQL Rootkits
- Module 13: Secure Application Testing
- Module 14: VMware Remote Recording and Debugging
- Module 15: Writing Secure Documentation and Error Messages
- Module 16: Secure ASP Programming
- Module 17: Secure PERL Programming
- Module 18: Secure XML, Web Services and AJAX Programming
- Module 19: Secure RPC, ActiveX and DCOM Programming
- Module 20: Secure Linux Programming
- Module 21: Secure Linux Kernel Programming
- Module 22: Secure Xcode Programming
- Module 23: Secure Oracle PL/SQL Programming
- Module 24: Secure Network Programming
- Module 25: Windows Socket Programming
- Module 26: Writing Shellcodes
- Module 27: Writing Exploits
- Module 28: Programming Port Scanners and Hacking Tools
- Module 29: Secure Mobile Phone and PDA Programming
- Module 30: Secure Game Designing
- Module 31: Securing E-Commerce Applications
- Module 32: Software Activation, Piracy Blocking and Automatic Updates
- Module 33: PCI Compliance and Secure Programming

7.6 Microsoft Certified Systems Engineer (MCSE) Security on Windows Server 2003

La certificazione MCSE copre competenze nella progettazione, implementazione e gestione di infrastrutture per soluzioni aziendali basate su Windows Server 2003 e Microsoft Windows 2000 Server. Microsoft rilascia la certificazione. Le competeze di implementazione includono installazione, configurazione e risoluzione dei problemi dei sistemi di rete.

Le specializzazioni MCSE forniscono programmi più mirati rispetto alla certificazione MCSE. MCSE Security in Windows Server 2003 è la specializzazione che si concentra sulla sicurezza.

URL	http://www.microsoft.com/learning/en/us/certification/mcse.aspx
Contact Method	http://support.microsoft.com/contactus/?ws=learning#tab0 Email, chat, phone and address
Country of HQ location	US
Geographic Scope	International
Type	Industry (Microsoft)

Per qualificarsi per la sicurezza MCSE su Windows Server certificazione 2003 è necessario superare otto esami, in qualsiasi ordine. I seguenti quattro esami sui sistemi di rete:

- Managing and Maintaining a Windows Server 2003 Environment.
- Implementing, Managing, and Maintaining a Windows Server 2003 Network Infrastructure.
- Planning and Maintaining a Windows Server 2003 Network Infrastructure.
- Planning, Implementing, and Maintaining a Windows Server 2003 Active Directory Infrastructure.

Un esame su sistemi operativi client, scelto tra i seguenti:

- TS: Configuring Windows Vista Client.
- Installing, Configuring, and Administering Windows XP Professional.

Un esame sul design:

- Designing Security for a Windows Server 2003 Network.

Due esami sulla specializzazione di sicurezza, scelto tra i seguenti:

- Implementing and Administering Security in a Windows Server 2003 Network.
- Implementing Microsoft Internet Security and Acceleration (ISA) Server 2004.
- TS: Microsoft Internet Security and Acceleration (ISA) Server 2006, Configuring.
- Third-party certifications, that could be:
    - CompTIA Security+
    - Systems Security Certified Practitioner (SSCP) or Certified Information
    - Systems Security Professional (CISSP) from (ISC)²
    - Certified Information Security Auditor (CISA) or Certified Information Security Manager (CISM) from ISACA.

Molti esami di questa certificazione sono stati ritirati. Se un esame richiesto è stato superato prima del suo ritiro, può essere utilizzato per la certificazione. La certificazione non ha scadenza.

7.7 Certified Software Security Lifecycle Professional (CSSLP) and Certified Information Systems Security Professional (CISSP)

Il CSSLP ha lo scopo di convalidare le conoscenze di sviluppo software sicuro e di buone pratiche. Il CSSLP è un codice in lingua neutrale e applicabile a chiunque sia coinvolto nel SDLC.

La certificazione è rilasciata dal Consorzio di Certificazione Internazionale Information Systems Security, (ISC)², un'organizzazione globale no-profit specializzata nella formazione e certificazione di professionisti della sicurezza informatica. Esso fornisce prodotti di formazione vendor-neutral.

URL	https://www.isc2.org/csslp/default.aspx
Contact Method	CSSLP Contact [https://www.isc2.org/csslp/default.aspx] Web form CISSP Contact [https://www.isc2.org/cissp/default.aspx] Web form General Contact [https://www.isc2.org/contactus/default.aspx] Web form, phone and address
Country of HQ location	US
Geographic Scope	International
Type	Industry (no profit)

In accordo al (ISC)², il CSSLP è progettato per:

- Stabilire le migliori pratiche, al fine di limitare la proliferazione delle vulnerabilità di sicurezza che derivano da processi di sviluppo insufficienti

- attestare la capacità professionista di mitigare i problemi di sicurezza e dei rischi che circondano lo sviluppo di applicazioni in tutto il SDLC, dalla specifica e progettazione alla realizzazione e manutenzione

I seguenti domini compongono il CSSLP Common Body of Knowledge (CBK), che si concentra sulla necessità di integrare la sicurezza nel SDLC:

- Secure Software Concepts: implicazioni di sicurezza nello sviluppo di software.
- Secure Software Requirements: catturare i requisiti di sicurezza nei raccolta dei requisiti di fase
- Secure Software Design: tradurre i requisiti di sicurezza in elementi di design di applicazioni
- Secure Software Implementation/Coding: unit testing per la funzionalità sicurezza e la resilienza contro gli attacchi, e lo sviluppo di codice sicuro e sfruttare la mitigazione
- Secure Software Testing: test integrati di quality assurance per la funzionalità sicurezza e la resilienza contro gli attacchi
- Software Acceptance: implicazioni per la sicurezza in fase di accettazione del software
- Software Deployment, Operations, Maintenance and Disposal: problemi di sicurezza intorno operazioni di steady-state e la gestione del software.

La qualificazione CSSLP è valida per tre anni, dopo di che deve essere rinnovata. Può essere rinnovata rifacendo l'esame o, più comune, con l'acquisizione di crediti formativi professionali (CPE).

Il CISSP, un altro programma di certificazione da (ISC)² con regole simili, è destinato ai professionisti che sviluppano politiche e procedure in materia di sicurezza delle informazioni.

7.8 Certified Information Security Auditor (CISA) and Certified Information Security Manager (CISM)

Il CISA e CISM da ISACA sono certificazioni destinati al management IT per convalidare le loro conoscenze in settori che vanno dalla governance IT per la protezione del patrimonio informativo e il processo di sviluppo. La sicurezza IT forma una gran parte di queste certificazioni, ma non molta enfasi viene data all'Ingegneria Secure Software.

URL	https://www.isaca.org/
Contact Method	General Contact [http://www.isaca.org/About-ISACA/Contact-Us/Pages/default.aspx] Web form, phone and address
Country of HQ location	US
Geographic Scope	International
Type	Industry (no profit)

In accordo con ISACA, CISA è designata a ricoprire le seguenti aree:

- The Process of Auditing Information Systems
- IT Governance and Management
- Information Systems Acquisition, Development and Implementation
- Information Systems Operations, Maintenance and Support.
- Protection of Information Assets and CISM is designed to cover the following areas:
- Information Security Governance
- Information Risk Management
- Information Security Program Development
- Information Security Program Management
- Incident Management and Response

I certificati CISA e CISM devono essere mantenuti attraverso l'acquisizione continua di crediti formativi professionali (CPE).

7.9 International Secure Software Engineering Council (ISSECO)

ISSECO promuove corsi di formazione sul SSE per ingegneri del software in modo che possano ottenere uno standard di certificazione (ISSECO Certified Professional per l'Ingegneria Secure Software). La certificazione è fornita dall'Istituto Internazionale Software Quality (iSQI)[8].

Secondo questa iniziativa, l'attenzione di ISSECO è sulla produzione di software sicuro e il suo obiettivo è quello di creare un ambiente informatico sicuro per tutti. Non è focalizzata su specifici linguaggi di programmazione.

URL	http://www.isseco.org
Contact Method	ISSECO Contact: http://www.isseco.org/index.php?p=contact Email ISQUI Contact: https://www.isqi.org/ Email, phone and address
Country of HQ location	Germany
Geographic Scope	National
Type	Industry (not for profit)

I temi principali della certificazione sono:

- Viewpoints of attackers and customers
- Trust and threat models
- Methodologies
- Requirements engineering with respect to security
- Secure design
- Secure coding
- Security testing
- Secure deployment
- Security response
- Security metrics
- Code and resource protection

Le attività di questa iniziativa sono supportati da partner diversi:

- Supporters (financial aid)
- Training providers (training material and classes)
- Certifiers (certification and certificate quality)

Le discussioni sono in corso di pubblicazione del materiale didattico ISSECO l'etichetta OWASP. Ciò potrebbe indurre un cambiamento nel business case.

[8] https://www.isqi.org/

8 SECURE SOFTWARE DEVELOPMENT LIFE CYCLE (SSDLC): ANALISI DELLE METODOLOGIE E DEI PROCESSI

8.1 Life Cycle & Maturity Models

8.1.1 Software Assurance Maturity Model (SAMML)

SAMM è un framework aperto per aiutare le organizzazioni a formulare e attuare una strategia di sicurezza software, che più si adatti ai rischi specifici della particolare organizzazione. Il progetto OpenSAMM, un'attività di OWASP, mantiene e aggiorna la documentazione SAMM.

References	**www.owasp.org/index.php/Category:Software_Assurance_Maturity_Model** **www.opensamm.org**

Le risorse fornite da SAMM attraverso il sito web aiutano a:

- Valutare le pratiche di sicurezza software esistenti di un'organizzazione
- Costruire un programma software security assurance in iterazioni ben definite
- Dimostrare miglioramenti concreti al programma di security assurance
- Definire e misurare le attività relative alla sicurezza in tutta l'organizzazione

Essendo un progetto Open, i contenuti SAMM sono liberamente fruibili. Il modello si basa su 4 funzioni aziendali di sviluppo software e di 12 procedure di sicurezza (vedi figura sotto tratta dal sito web del progetto SAMM):

Figura 7: SAMM Structure

Per ogni security practice, tre Maturity Levels sono definiti in termini di specifiche attività e metriche che un'organizzazione potrebbe adottare al fine di ridurre i rischi per la sicurezza e aumentare la garanzia software.

Risultati più rilevanti:

Maturity Model: SAMM version 1.0	**The model is available in XML and has been translated into other languages:** **http://www.opensamm.org/download/** **This page also lists supporting tools.**

8.1.2 Systems Security Engineering Capability Maturity Model (SEE-CMM)

Il modello SSE-CMM si indirizza sui requisiti per l'implementazione della sicurezza in un sistema.

URL	**http://www.sse-cmm.org**
Country of HQ location	**US**
Geographic Scope	**International**
Type	**Industry (not for profit)**

Questo modello ha undici aree di processo di sicurezza dove ogni area comprende un insieme di pratiche di base. Queste aree si concentrano sui controlli, sulle minacce, sulla scoperta e eliminazione delle vulnerabilità:

- Administer Security Controls
- Assess Impact
- Assess Security Risk
- Assess Threat
- Assess Vulnerability
- Build Assurance Argument
- Coordinate Security
- Monitor Security Posture
- Provide Security Input
- Specify Security Needs
- Verify and Validate Security

Le migliori pratiche di sicurezza potrebbero essere applicate nell'ingegneria del software.

Risultati più significativi:

Maturity Model	Capability Maturity Model - Model Description Document - http://all.net/books/standards/ssecmmv3final.pdf
Standard	ISO/IEC 21827 - http://www.iso.org/iso/catalogue_detail.htm?csnumber=44716

8.1.3 Building Security In Maturity Model (BSIMM)

BSIMM non è una guida completa 'how to' di sicurezza software, ma piuttosto una raccolta di idee e attività che sono oggi in uso all'interno delle aziende di sviluppo software.

Il BSIMM è stato creato attraverso un processo di comprensione e analisi dei dati del mondo reale provenienti dalle esperienze di nove imprese nell'ambito sicurezza software, che sono stati a seguire validati e regolamentati con i dati provenienti da 21 aziende aggiuntive. Il BSIMM mette quindi insieme le esperienze di trenta imprese di sviluppo software - la maggior parte di essi si trovano negli Stati Uniti - che hanno implementato iniziative di sicurezza del software.

URL	http://bsimm.com/
Country of HQ location	US
Geographic Scope	International (mainly the US)

Type	Industry

BSIMM ha sviluppato il Security Framework Software (SSF). SSF fornisce un vocabolario comune per descrivere gli elementi più importanti di un quadro di sicurezza software all'interno di una società.

Sono stati identificati domini e pratiche comuni alla maggior parte delle esperienze. Il BSIMM descrive 109 attività che ogni organizzazione può mettere in pratica. Le attività sono descritte in termini di SSF, che identifica dodici pratiche raggruppati in 4 domini, 3 pratiche di dominio, come mostrato nella figura presa dal documento BSIMM2:

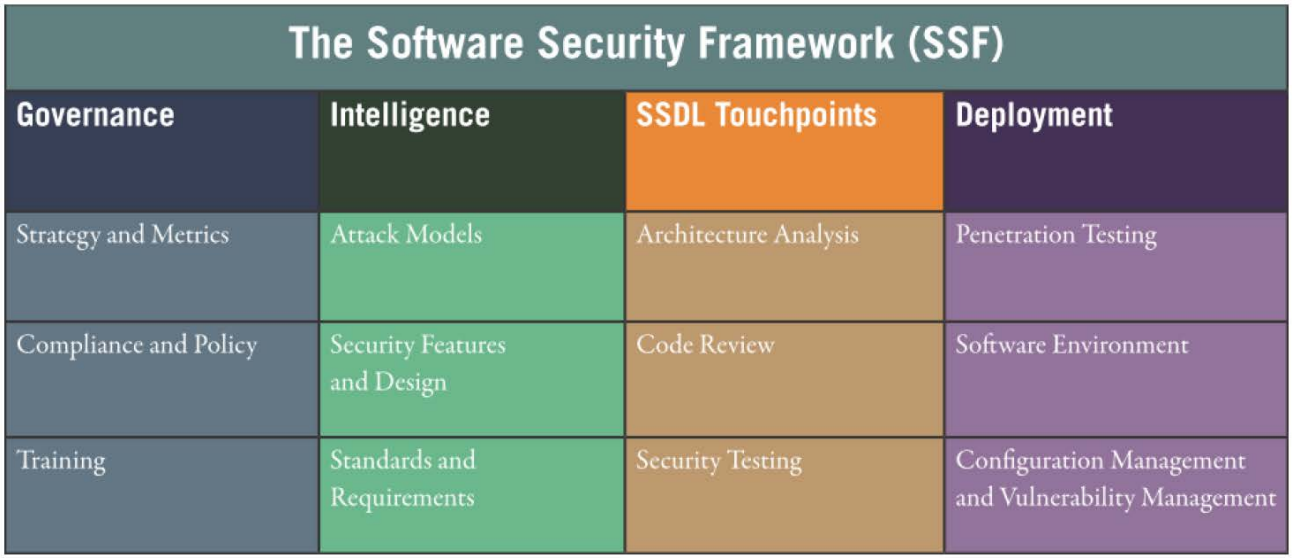


Figura 8: BSIMM SSF

Per ogni livello di pratica e di maturità vi è un'associazione "one activity - one objective". I domini sono:

1. Governance - Practices that help organise, manage, and measure a software security framework. Staff development is also a central governance practice.
2. Intelligence - Collections of corporate knowledge used in carrying out software security activities throughout an organisation. Collections include both proactive security guidance and organisational threat modelling.
3. SSDL Touchpoints - Practices associated with the analysis and assurance of particular software developments, artefacts and processes. All software security methodologies include these practices.
4. Deployment - Practices that interface with traditional network security and software maintenance. Software configuration, maintenance, and other environment issues have a direct impact on software security.

Il modello di maturità si presenta come una serie di attività connesse con le pratiche. Gli obiettivi per ogni livello di pratica sono identificati. Gli obiettivi possono essere ulteriormente suddivisi in obiettivi per la pratica/livello e sono associati alle attività. A titolo di esempio, la figura seguente, tratta dal documento BSIMM2, mostra il modello di maturità per la pratica di addestramento del dominio Governance.

GOVERNANCE: TRAINING		
Objective	**Activity**	**Level**
promote culture of security throughout the organization	provide awareness training	1
ensure new hires enhance culture	include security resources in onboarding	
act as informal resource to leverage teachable moments	establish SSG office hours	
create social network tied into dev	identify satellite during training	
build capabilities beyond awareness	offer role-specific advanced curriculum (tools, technology stacks, bug parade)	2
see yourself in the problem	create/use material specific to company history	
reduce impact on training targets and delivery staff	offer on-demand individual training	
educate/strengthen social network	hold satellite training/events	
align security culture with career path	reward progression through curriculum (certification or HR)	3
spread security culture to providers	provide training for vendors or outsource workers	
market security culture as differentiator	host external software security events	
keep staff up-to-date and address turnover	require annual refresher	

Figura 9: Training practice BSIMM

Risultati più rilevanti:

Maturity Model	BSIMM2 - https://www.bsimm.com/download/

8.2 Analisi dei Processi SSDLC

8.2.1 McGraw's Secure Software Development Life Cycle Process

McGraw[9] [1]si propone di accrescere il processo SDLC (cascata o iterativo) attraverso l'integrazione di alcune attività SSD. In sostanza, il processo di McGraw si focalizza su:

- incorporazione dei requisiti di sicurezza,
- esecuzione dell'analisi dei rischi durante le diverse fasi di sviluppo,
- applicazione di metodi di security assurance quali test di sicurezza risk-based,
- analisi statica e test di penetrazione.

Il processo suggerisce anche di utilizzare l'analisi dei rischi durante la fase di progettazione. Per la fase di security assurance, McGraw suggerisce di utilizzare gli abuse cases e i requisiti di sicurezza per guidare i test di penetrazione.

[9] G. McGraw, Software Security: Building Security In, Addison Wesley, 2006

8.2.2 Microsoft Software Development Life Cycle (MS SDL)

MS SDL è un modello a spirale potenziatocon diverse attività SSD. MS SDL pone molta attenzione alla fase di specifica dei requisiti durante la quale prevede di interagire con il cliente (end-user), al fine di identificare gli obiettivi di sicurezza e le caratteristiche di sicurezza richieste.

L'incorporazione di queste caratteristiche/funzionalità di sicurezza sono guidate da standard di settore e criteri di certificazione. Durante la fase di progettazioneMS SDL suggerisce di svolgere le seguenti attività: l'identificazione dei componenti critici per la sicurezza, l'identificazione di tecniche di progettazione e linee guida, l'identificazione dei punti di accesso degli attacchi, la modellazione delle minacce e analisi del rischio su base component-by-component, l'identificazione dei requisiti di sicurezza per mitigare le minacce, l'identificazione dei componenti che necessitano di particolare attenzione durante le fasi di test e review del codice, e i criteri di completamento per il software.

Figura 10: Microsoft SDL

MS SDL consiglia di seguire gli standard secure coding nella fase di implementazione. Il processo di MS SDL pone l'accento su:

- security assurance by recommending testing,
- analisi statica del codice utilizzando i tool SDL utili a tale scopo,
- review del codice nell'ultimo step della fase di implementazione. Terminata la fase di implementazione, il software completo viene nuovamente verificato attraverso un ulteriore test di sicurezza che si concentra principalmente sui componenti critici (ad esempio, punti di ingresso alle possibili aree di attacco).

URL	www.microsoft.com/security/sdl
Contact Method	support.microsoft.com/contactus/?ws=mscom#tab0 Email, chat, phone and address
Country of HQ location	US
Geographic Scope	International
Type	Industry (Microsoft)

Risultati più rilevanti:

Guidance	Microsoft SDL Process Guidance version 5.0 This guidance illustrates the way Microsoft applies the SDL to its products and technologies. It includes security and privacy requirements and recommendations for secure software development. It addresses SDL guidance for Waterfall and Spiral development, Agile development, web applications and Line of Business applications. IT policy makers and software development organisations can leverage this content to enhance and inform their own software security and privacy assurance programs.
	Microsoft SDL for Agile Development This documentation is not an exhaustive reference for the SDL process as practised at

	Microsoft, but is for illustrative purposes only.
	Microsoft SDL for Line-of-Business Applications This documentation is not an exhaustive reference on the SDL process as practised at Microsoft, but is for illustrative purposes only.
	The Security Development Lifecycle This is a book that provides guidance through each stage of the SDL, from education and design to testing and post-release. The authors are security experts from the Microsoft Security Engineering Team.
	Simplified Implementation of the Microsoft SDL This document illustrates the core concepts of the Microsoft SDL and discusses the individual security activities that need to be performed in order to claim compliance with the SDL process, including: roles and responsibilities, mandatory security activities, optional security activities and the application security verification process.
	SDL Quick Security Reference (QSR) With the SDL QSR, the SDL team introduces a series of basic guidance papers designed to address common vulnerabilities from the perspective of multiple business roles – business decision-maker, architect, developer and tester/QA.
	Securing Applications This documentation is aimed at developers of .NET Framework for writing security code. It includes: Key Security Concepts, Code Access Security, Role-Based Security, Cryptographic Services, Security Policy Management, Security Policy Best Practice, Secure Coding Guidelines and Security Tools.
Tools & Templates	Microsoft SDL Threat Modelling Tool Threat modelling allows software architects to identify and mitigate potential security issues early, when they are relatively easy and cost-effective to resolve. It is a free tool requiring Visio 7. The tool is focused on design analysis techniques.
	Microsoft SDL Process Template A downloadable template that automatically incorporates the policy, process and tools associated with the SDL into Visual Studio‟s software development environment.
	MSF-Agile+SDL Process Template A downloadable template that automatically incorporates the policy, process and tools associated with the SDL for Agile development guidance, into the Microsoft Solutions Framework for Agile software development (MSF-Agile) and the Visual Studio environment.
	The Microsoft SDL Tools A map of the available free tools and templates for each SDL stage.

8.2.3 Appropriate and Effective Guidance for Information Security (AEGIS)

AEGIS [10] [11] [2] [3]è un processo SSDLC basato sul modello a spirale e si concentra sulla specifica dei requisiti di sicurezza, identificando gli elementi principali ed eseguendo l'analisi dei rischi. Le fasi di analisi dei requisiti e di disegno sono strettamente collegati. Il modello propone quattro sessioni di progettazione tra gli sviluppatori e gli stakeholders del software. La prima e la seconda sessione modellano le caratteristiche principali del software e le lor relazioni identificando i requisiti di alto livello di riservatezza, integrità e disponibilità. Nella terza sessione vengono identificati rischi, vulnerabilità e minacce per il software. Nella quarta sessione, orientata alla progettazione, vengono identificati i requisiti di sicurezza per rimuovere le vulnerabilità identificate.

[10] I. Flechais, M.A. Sasse, and S.M.V. Hales, "Bringing Security Home: A Process for Developing Secure and Usable Systems," In Proc. of the New Security Paradigms Workshop (NSPW'07), Ascona, Switzerland, ACM Press, 2003, pp. 49-57.

[11] I. Flechais, C. Mascolo, and M.A. Sasse, "Integrating Security and Usability into the Requirements and Design Process," International Journal of Electronic Security and Digital Forensics, Inderscience Publishers, Geneva, Switzerland, 2007, vol. 1, no. 1, pp. 12-26.

AEGIS suggerisce anche una metodologia di analisi dei rischi da utilizzare durante le sessioni 3 e 4 finalizzate alla progettazione. Questo metodo di analisi dei rischi ha le seguenti fasi principali:

- Determinazione delle vulnerabilità;
- Determinazione del costo e della probabilità di un attacco in ambiente distribuito (inclusi i ruoli interpretati dalle persone che interagiscono con il software e i task che verranno eseguiti sul software).
- Selezione dei requisiti di sicurezza basate sulle indicazioni dell'esperto di sicurezza.
- Valutazione costi-benefici dei requisiti di sicurezza selezionati.
- Il confronto tra il costo e la probabilità di ogni attacco e il costo dei requisiti di sicurezza.
- Selezione dei requisiti di sicurezza sulla base dell'efficacia dei costi.

8.2.4 Secure Software Development Model (SSDM)

SSDM [12] è un processo che incorpora diverse attività di sicurezza in un modello SDLC a cascata. Secondo SSDM, la modellazione delle minacce dovrebbe essere eseguita in fase di specifica dei requisiti. Il risultato di questa modellazione dovrebbe essere una check-list contenente tutte le potenziali vulnerabilità e attacchi. Tali elenchi di fatto dovrebbero essere dati in input alla fase di sviluppo.

Dopo modellazione delle minacce, è necessario definire una politica di sicurezza che indichi chiaramente come saranno raggiunti gli obiettivi di sicurezza prefissati.

Tale politica, come sottolineato dal SSDM, è un insieme di decisioni di gestione di alto livello come ad esempio evitare errori in tutto il processo di sviluppo e la correzione degli errori non appena vengono rilevati. Per la fase di progettazione, SSDM consiglia di seguire la politica di sicurezza. I test di penetrazione rappresentano, nel modello SSDM, l'unica attività SSD per la fase security assurance.

8.2.5 Aprville and Pourzandi's Secure Software Development Life Cycle Process

[4]Aprville e Pourzandi[13] propongono un processo SSDLC sulla base della loro esperienza, maturata durante lo sviluppo di un software di instant messaging. Secondo il loro processo [5], il primo passo nella fase di specifica deu requisiti è quello di individuare gli obiettivi di alto livello di sicurezza (riservatezza, integrità e disponibilità) del software in fase di sviluppo, considerando il suo ambiente di distribuzione. Per gli obiettivi di sicurezza a basso livello, la modellazione delle minacce dovrebbe essere di supporto nella costruzione di un insieme di requisiti di sicurezza. Questi requisiti di sicurezza possono essere resi prioritari in base ai risultati dell'analisi del rischio. In fase di progettazione, si raccomanda l'uso di [6]UMLsec[14]. Per la fase di implementazione, si suggerisce di scegliere un linguaggio di programmazione che meglio soddisfa gli obiettivi di sicurezza. Inoltre, particolare attenzione deve essere posta su come evitare: (i) buffer overflow, (ii) format string vulnerabilities. Essi sottolineano di utilizzare per la crittografia algoritmi già verificati. Per la fase di security assurance: code reviews, static vulnerability code scanners, ad-hoc unit and system security testing, fuzz testing, testing tools.

[12] A.S. Sodiya, S.A. Onashoga, and O.B. Ajayi, "Towards Building Secure Software Systems," Issues in Informing Science and Information Technology, Informing Science Institute, California, USA, 2006, vol. 3, pp. 635-646.
[13] A. Apvrille and M. Pourzandi, "Secure Software Development by Example," IEEE Security and Privacy, IEEE CS Press, 2005, vol. 3, no. 4, pp. 10-17.
[14] J. Juerjens, Secure Systems Development with UML, Springer, 2005.

8.2.6 Secure Software Development Model (SecSDM)

SecSDM[15] utilizza l'analisi dei rischi nella fase di specifica dei requisiti al fine di dare priorità alla modellazione delle minacce software. Gli obiettivi di sicurezza di alto livello quali la riservatezza, l'integrità e la disponibilità sono poi identificati sulla base delle minacce identificate [7].

In fase di progettazione, vengono identificate e selezionate le funzionalità di sicurezza per mitigare le minacce e raggiungere gli obiettivi di sicurezza. SecSDM propone di seguire standard di secure coding durante la fase di implementazione.

8.2.7 Software Security Assessment Instrument (SSAI)

SSAI[16][17] raggruppa un insieme di attività che utilizzano determinate risorse e strumenti per aiutare nello sviluppo di software sicuro. [8] [9] La prima risorsa che SSAI fornisce è un database delle vulnerabilità online [18] che contiene informazioni sulle varie vulnerabilità e le indicazioni per la loro mitigazione. [10] La seconda risorsa SSAI è una security checklist che può essere utilizzata come guida per lo sviluppo sicuro. Sono forniti i dettagli di come sviluppare una checklist e quali sono gli elementi potenziali che possono essere inclusi[19]. La terza risorsa è un elenco di tool per la scansione statica del codice accessibili pubblicamente. SSAI fornisce anche Flexible Modeling Framework (FMF), che è uno strumento di modellazione. Infine, SSAI fornisce un property-based testing tool (PBT), che utilizza le proprietà di sicurezza specificate nella security checklist o FMF come base di test per il software.

8.2.8 Hadawi's Set of Secure Development Activities

Hadawi [20] identifica 25 vulnerabilità (common vulnerabilities) da evitare durante lo sviluppo [11]. Egli propone anche una serie di requisiti di sicurezza per le fasi di progettazione e implementazione che, se incorporati, aiuterebbero ad evitare queste vulnerabilità [12].

Durante la fase di implementazione, l'unica attività SSD è la scelta di un appropriato linguaggio di programmazione (sicuro). Per la fase di security assurance, Hadawi consiglia di utilizzare: (i) security code reviews, (ii) static code analysis tools.

8.2.9 Comprehensive, Lightweight Application Security Process (CLASP)

Comprehensive, Lightweight Application Security Process (CLASP) [21] identifica un insieme di attività SSD che sono classificati in base ai ruoli svolti durante lo sviluppo. CLASP suggerisce l'impiego di un esperto di sicurezza fin dall'inizio dello sviluppo. Per la fase di specifica dei requisiti, CLASP sottolinea la necessità di

---

[15] L. Futcher and R.v. Solms, "SecSDM: A Model for Integrating Security into the Software Development Life Cycle," In IFIP International Federation for Information Processing, Volume 237, Proc. of the 5th World Conference on Information Security Education, Springer, 2007, pp. 41-48.

[16] D.P. Gilliam, T.L. Wolfe, J.S. Sherif, and M. Bishop, "Software Security Checklist for the Software Life Cycle," In Proc. of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03), Linz, Austria, IEEE CS Press, 2003, pp. 243-248.

[17] D. Gilliam, J. Powell, E. Haugh, and M. Bishop, "Addressing Software Security Risk and Mitigations in the Life Cycle," In Proc. of the 28th Annual NASA Goddard Software Engineering Workshop (SEW'03), Greenbelt, Maryland, USA, 2003, pp. 201-206.

[18] DOVES: Database of Vulnerabilities, Exploits, and Signatures, http://seclab.cs.ucdavis.edu/projects/DOVES/. Last Accessed March 2009.

[19] D.P. Gilliam, T.L. Wolfe, J.S. Sherif, and M. Bishop, "Software Security Checklist for the Software Life Cycle," In Proc. of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03), Linz, Austria, IEEE CS Press, 2003, pp. 243-248.

[20] M.A. Hadawi, "Vulnerability Prevention in Software Development Process," In Proc. of the 10th International Conference on Computer & Information Technology (ICCIT'07), Dhaka, Bangladesh, 2007,

[21] OWASP CLASP Project, http://www.owasp.org/index.php/Category:OWASP_CLASP_Project. Last Accessed March 2009

un'analisi dei rischi e la modellazione delle minacce. L'analisi dei rischi e la modellazione delle minacce devono essere eseguite anche nella fase di progettazione.

CLASP propone di annotare i diagrammi di classe con le informazioni di sicurezza. Nella fase di security assurance, CLASP consiglia: security code reviews, security code scanning, security testing.

CLASP fornisce anche un elenco di vulnerabilità (common vulnerabilities) con informazioni complete su come e quando possono essere introdotti durante lo sviluppo e come evitarli.

URL	www.owasp.org/index.php/Category:OWASP_CLASP_Project

Risultati più rilevanti:

Security Process	CLASP version 1.2

8.2.10 Secure Software Development Process Model (S2D-ProM)

S2D-PROM [22] specifica molteplici strategie possibili per avanzare da ogni fase di sviluppo all'altra [13]. L'idea principale alla base di questo processo è quello di fornire agli sviluppatori opzioni flessibili. Il processo si propone di condurre l'analisi dei rischi durante le fasi di specifica dei requisiti, progettazione, e implementazione. L'analisi del rischio, secondo S2D-PROM, può essere eseguita in modi diversi per ogni fase di sviluppo. I rischi identificati possono essere mitigati utilizzando varie strategie (ad esempio, definendo le norme di sicurezza o utilizzando meccanismi di sicurezza).

S2D-PROM non specifica se una sola strategia deve essere utilizzato mentre ci si sposta da una fase all'altra o più strategie possono essere utilizzate nello stesso tempo.

8.2.11 Team Software Process for Secure Software Development (TSP Secure)

[14]TSP-Secure[23] garantisce la sicurezza attraverso:

- la pianificazione per la sicurezza,
- la qualità e la gestione della sicurezza in tutto il ciclo di vita dello sviluppo,
- la formazione degli sviluppatori circa gli aspetti relativi alla sicurezza.

Durante la fase di progettazione il team identifica obiettivi di sicurezza e produce un piano dettagliato per guidare lo sviluppo. Le attività di sviluppo nel piano possono includere, ma non sono limitati, l'identificazione dei rischi per la sicurezza, l'identificazione dei requisiti di sicurezza, la progettazione sicura, le revisioni del codice, unit test, test fuzz, analisi statica del codice. Il team può scegliere qualsiasi attività SSD come ritenuto necessario.

Secondo TSP-Secure, un membro del team svolge il ruolo di responsabile della sicurezza che è responsabile di tutte le attività relative alla sicurezza in corso.

[22] M. Essafi, L. Labed, and H.B. Ghezala, "S2D-ProM: A Strategy Oriented Process Model for Secure Software Development," In Proc. of the 2nd International Conference on Software Engineering Advances (ICSEA'07), Cap Esterel, French Riviera, France, 2007, p. 24.
[23] N. Davis, "Secure Software Development Life Cycle Processes: A Technology Scouting Report", technical note CMU/SEI-2005-TN-024, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsalyania, USA, 2005.

9 LINEE GUIDA PER L'ADOZIONE DI UN CICLO DI SVILUPPO SOFTWARE SICURO

9.1 Definizione dei requisiti di sicurezza

I principali requisiti di sicurezza da definire sono:

- **Riservatezza e Integrità.** I due più importanti aspetti della sicurezza sono Riservatezza e Integrità. La Riservatezza significa che le risorse possono essere utilizzate solo dalla parte legittima. L'integrità dei dati significa che devono essere modificabili solo dalle persone autorizzate.
- **Autenticità.** Il terzo requisito di sicurezza principale è l'Autenticità: *Message authenticity* (o *data* origin authenticity), *entity authenticity* (chiamata anche *authentication*).
- **Non-ripudio.** Garantisce che qualsiasi azione sul sistema non possa essere successivamente rinnegata.
- **Flusso Informativo**. Il livello di sicurezza può avere regole diverse. Generalmente si considerano due livelli: alto (altamente sensibile o altamente attendibile) e basso (meno sensibile o meno attendibile). Laddove componenti di sistema considerati di alto livello interagiscono con parti meno attendibili, si deve garantire che non vi sia alcuno scambio di dati dall'alto verso il basso (vale invece il contrario ossia ci può essere lo scambio di dati dal basso verso l'alto *non up-flow*).
- **Controllo Accessi**. Uno dei requisiti di sicurezza principali è il controllo degli accessi, il che significa che solo un utente fidato può avere accesso ad un sistema sicuro. **Role-Based Access Control (RBAC)**: realizza un importante meccanismo di controllo degli accessi per tutelare i beni. I privilegi di accesso alle risorse dipendono dal ruolo che assumono nel tempo gli individui all'interno dell'Organizzazione. Ai ruoli sono associati profili che definiscono comandi, transazioni e accessi ai dati. L'assegnazione dei ruoli è centralizzata.

Le principali azioni di sicurezza da attuare sono:

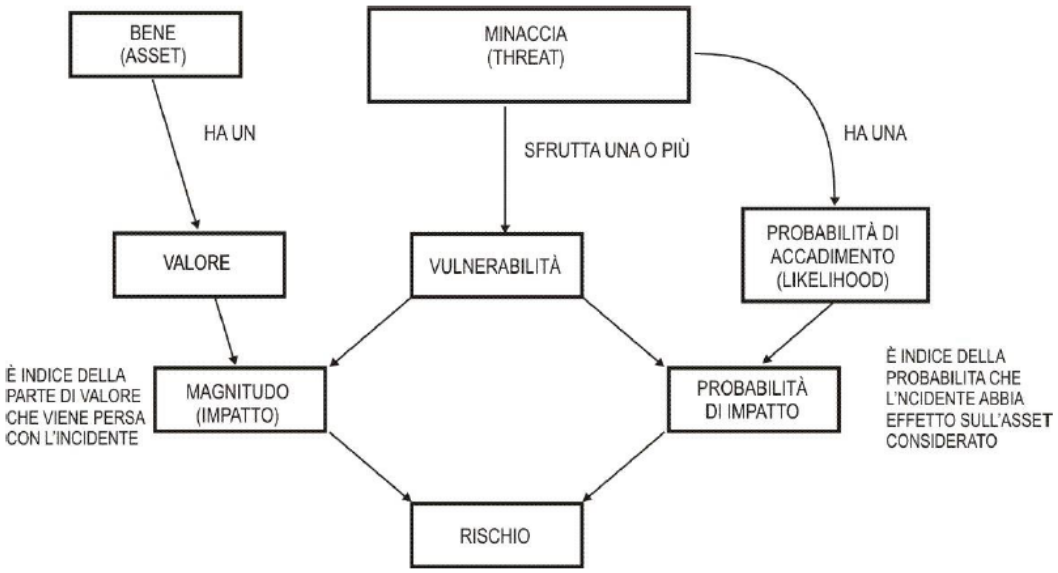
- **Definizione degli elementi di sicurezza applicativa**, finalizzata alla valutazione dei requisiti relativamente a**:**
  - Integrità,
  - Autenticità,
  - Riservatezza,
  - Disponibilità,
  - Non-ripudio,
  - Autorizzazione.
- **Definizione dei requisiti di privacy**, attraverso la raccolta strutturata delle seguenti categorie di informazioni:
  - Dati personali,
  - Servizi di terze parti,
  - Policy.
- **Risk assessment**, finalizzato alla valutazione del rischio (vedi Paragrafo 9.1.1).
  - **Consolidamento dei Requisiti**, che consiste nella review dei requisiti di sicurezza e privacy a seguito del Risk Assessment;
- A completamento di questa fase è necessario produrre la Reportistica/documentazione completa che sintetizza i risultati per ogni punto precedente.

Si evidenzia che in questa fase devono essere tenuti in considerazione anche gli aspetti di integrazione e di interfaccia con gli eventuali altri moduli dell'ecosistema software.

Inoltre vanno considerati i requisiti di sicurezza applicativa di carattere generale attraverso l'attuazione di best practices che individuano i requisiti di sicurezza che devono essere adottati nel processo di sviluppo di un'applicazione sicura: Performance, Password nel codice sorgente, Privilegi esecutivi minimi, Fattore di integrità, Input data validation, Gestione dell'output, etc. (per ulteriori dettagli si rinvia al deliverable ***D03.Fase1-SP2 - Linee Guida per lo sviluppo sicuro di codice***, paragrafo ***4.1 Progettazione e sviluppo dell'Applicazione: direttive standard***). Tali requisiti di sicurezza applicativa devono essere mutuati in questa fase sulla base dei requisiti, funzionali e non funzionali, individuati.

9.1.1 Risk Assessment

L'obiettivo dell'analisi del rischio è da una parte identificare, valutare e misurare la probabilità e la gravità dei rischi (ciò che viene generalmente indicato con il nome di *Risk Assessment*) e dall'altra decidere come comportarsi a fronte dei rischi identificati (ciò che viene generalmente indicato con il nome di *Risk Management*). Si riporta di seguito uno schema per il *Risk Assessment*:



Si sottolinea che la fase di Risk Assessment è quella che concorre a generare i requisiti di sicurezza che saranno gestiti, insieme con gli altri individuati non necessariamente durante la Risk Analysis, con l'ausilio dei "Software Requirements Tools".

La figura che segue sintetizza gli elementi in input e l'output prodotto dal processo di Risk Assessment:

Figura 11: Input ed Output della fase Risk Assessment

9.1.2 Identificazione degli strumenti a supporto

I tools riportati nel paragrafo 6.2.2 sono stati comparati[24] sulla base dei parametri riportati nella tabella che segue (in particolare vengono identificati 8 parametri per ogni sezione):

Software Functional Requirements **(Software Requirement Engineering)**	Requirement Elicitation • Interview • Joint Application Development (JAD) • Brainstorming • Questionnaire and Checklist • Case Modeling • Scalability Requirement Specification • Use Case Modeling • Tradition Requirement Specification • Templates • Glossary and Ontology • Prototype Requirement Validation • Audit • Walkthrough • Prototyping
Software Security Requirements **(Security Requirement Engineering)**	• Fair Exchange requirement • Non-repudiation security requirement • Role-based access control security requirement • Secrecy (Confidentiality) and Integrity • Authenticity • Freshness • Secure Information Flow • Guarded Access

Software Functional Requirements (Product)

Tools	Glossary & Ontology	Checklist	Templates	Use Case Modeling	Prototyping & Audit	TRS	Scalability	External Interface
RequisitePro	X	X	√	√	√	√	X	√
CaseComplete	√	X	√	√	√	√	X	√
Analyst Pro	X	X	X	√	X	√	√	√
Optimal Trace	X	X	√	√	√	√	X	√
DOORS	X	X	√	√	√	√	√	√
GMARC	X	X	√	X	√	√	X	√
Objectiver	√	√	√	X	√	√	X	√
RDT	√	X	√	X	√	√	√	√
RDD-100	√	X	√	X	X	√	X	√

[24] https://www.researchgate.net/publication/233952819

Tools	Glossary & Ontology	Checklist	Templates	Use Case Modeling	Prototyping & Audit	TRS	Scalability	External Interface
RTM	X	X	√	X	X	√	X	√
Reqtify	X	X	X	√	√	√	X	√
TcSE	X	X	X	√	X	√	X	√
Code Assure	X	X	X	X	X	√	X	√
IRqA	X	√	X	√	X	√	X	√

Software Security Requirements (Security)

Tools	Fair Exchange	Non-repudiation	Rbac	Secrecy & Integrity	Authenticity	Secure Informat. Flow	Guarded Access	Freshness
RequisitePro	√	X	X	X	X	X	X	√
CaseComplete	√	X	X	X	X	X	X	√
Analyst Pro	√	X	√	X	√	X	X	X
Optimal Trace	√	X	X	X	X	X	X	X
DOORS	√	X	√	X	√	X	√	√
GMARC	X	X	X	X	X	X	X	√
Objectiver	X	X	X	X	X	X	X	√
RDT	X	X	X	X	X	X	X	√
RDD-100	√	X	X	X	X	X	X	√
RTM	X	X	√	√	√	X	√	√
Reqtify	√	X	X	X	X	X	X	√
TcSE	X	√	√	√	√	√	√	√
Code Assure	X	√	√	√	√	√	√	√
IRqA	X	X	√	X	X	X	X	X

Ad ogni parametro viene assegnato un punteggio di 12,5 (100/8). Il tool che presenta un punteggio totale maggiore (il punteggio di 12,5 viene moltiplicato per il numero totale di parametri soddisfatti) è da considerarsi quindi il migliore.

Valutazione dei tool:

Tools	Product	Security
RequisitePro	62.5	25.0
Case Complete	**75.0**	25.0
Analyst Pro	50.0	37.5
Optimal Trace	62.5	12.5
DOORS	**75.0**	62.5
GMARC	50.0	12.5
Objectiver	62.5	12.5
RDT	62.5	12.5
RDD-100	50.0	25.0
RTM	37.5	50.0
Reqtify	50.0	25.0
TcSE	37.5	62.5
Code Assure	25.0	**87.5**
IRqA	50.0	12.5

Secondo la metodologia adottata, come si evince dalla tabelle di sintesi:

- **DOORS** risulta essere il tool migliore sotto entrambi i punti di vista;
- **CodeAssure** risulta essere il migliore dal punto di vista prettamente legato ad aspetti di sicurezza.

9.2 Progettazione di applicazioni sicure

Le azioni di sicurezza di questa fase possono essere così sintetizzate**:**

- **Analisi e modellazione delle minacce**, attraverso l'identificazione dei componenti applicativi coinvolti, delle interfacce e degli agenti che potrebbero minacciare il sistema;
- **Analisi della superficie d'attacco e della finestra di opportunità,** allo scopo di individuare le parti del sistema che possono essere esposte ad attacchi e pertanto lo rendono vulnerabile;
- **Piano di mitigation,** attraverso l'identificazione delle contromisure da adottare in questa fase al fine di mitigare le potenziali minacce individuate (utilizzando anche tool automatici e semiautomatici);
- **Secure Design Refactoring**, revisione progettuale che attua le contromisure individuate; produzione di un High Level Design conforme ai principi del Secure by Design;
- Questa fase produce come output finale la Reportistica/documentazione completa che sintetizza i risultati per ogni punto precedente (Specifiche Software comprensive delle contromisure).

Questa fase è inoltre responsabile della revisione dei requisiti di sicurezza individuate nella fase precedente di definizione dei requisiti di sicurezza (paragrafo 9.1).

La figura che segue sintetizza gli elementi in input e l'output prodotto dal processo di Progettazione di software sicuro:

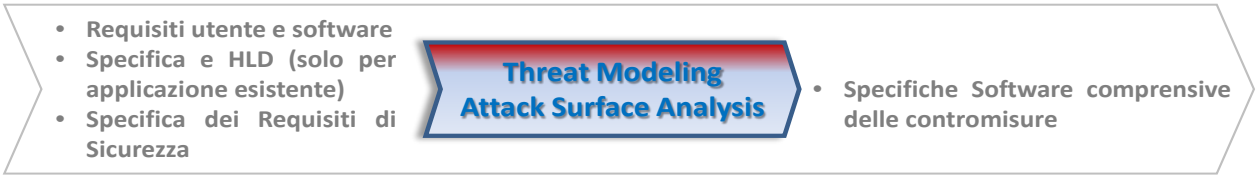


Figura 12: Input ed Output della fase Threat Modeling Attack Surface Analysis

Le linee guida di progettazione sicura sono oggetto del deliverable ***D04.Fase1-SP2 - Linee Guida per la modellazione delle minacce ed individuazione delle azioni di mitigazione conformi ai principi del Secure/Privacy by Design***. Si rinvia a quest'ultimo per ulteriori dettagli della metodologia da adottare.

9.2.1 Identificazione degli strumenti a supporto

Dopo aver identificato e documentato le esigenze di sicurezza, viene eseguita la modellazione delle minacce col fine di riconoscere e assegnare delle priorità a queste ed individuare le opportune contromisure per la loro mitigazione.

Nel paragrafo 6.3.2 sono stati presentati i tool a supporto di questa fase. Tra questi si evidenzia il **Microsoft Threat Modeling Tool** il quale consente di creare un modello di minacce tramite una rappresentazione del sistema o dei componenti del sistema, dei dati scambiati tra i componenti e i limiti di attendibilità nel sistema stesso.

A differenza delle tecniche di verifica, come ad esempio il penetration testing, il modello di minacce ottenuto attraverso la relativa modellazione, deve essere eseguito prima che un prodotto o un servizio venga implementato. Questo contribuisce a realizzare un prodotto finale più sicuro indirizzando problematiche di sicurezza ad un early-stage del ciclo di sviluppo. Il processo per la costruzione di un modello di minacce consiste dei seguenti step:

- Disegno dell'architettura del sistema;
- Individuazione dei confini di fiducia;
- Identificazione delle minacce;
- Individuazione delle contromisure da attuare per mitigare le minacce;
- Eventuale riprogettazione dei componenti per mitigare le minacce;
- Convalida del modello architetturale;
- Verifica dell'esistenza di una contromisura per ogni potenziale minaccia identificata.

9.3 Implementazione di applicazioni sicure

Le azioni di sicurezza che devono essere intraprese in questa sono così sintetizzate:

- **Data Validation:** verificare la presenza di vulnerabilità che possono riguardare eventuali dati corrotti in ingresso e che possono portare a un comportamento anomalo dell'applicazione;
- **Control Flow**: verificare i rischi collegati all'assenza di specifiche sequenze di operazioni che, se non eseguite nel corretto ordine, possono portare a violazioni sulla memoria o sull'uso scorretto di determinati componenti;
- **Analisi Semantica**: rilevare eventuali problematiche legate all'uso pericoloso di determinate funzioni o API (es. funzioni deprecate);

- **Controllo Configurazioni**: verificare i parametri intrinseci di configurazione dell'applicazione;
- **Buffer Validation**: verificare la presenza di buffer overflow exploitabile attraverso la scrittura o la lettura di un numero di dati superiore alla reale capacità del buffer stesso.

L'esame del codice sorgente applicativo deve portare alla produzione, mediante la Static Analysis, delle seguenti tipologie di documenti:

- **Report delle Vulnerabilità riscontrate**: report di dettaglio delle vulnerabilità riscontrate nella fase di analisi statica del codice tramite gli strumenti a supporto;
- **Remediation Plan**: report che analizza i falsi positivi ed indirizza la risoluzione delle problematiche riscontrate nell'analisi stessa.

La figura che segue sintetizza gli elementi in input e l'output prodotto dal processo SAST:

Figura 13: Input ed Output della fase Static Analysis

9.3.1 Identificazione degli strumenti a supporto

Nel paragrafo 6.4.16.3.2 sono stati presentati i tool a supporto di questa fase. Tra questi si raccomandano:

- closed source
  - IBM App Scan (versione SAST),
  - Checkmarx,
  - CodeDx,
  - HP fortify.
- open source
  - SonarQube,
  - FxCop (.NET),
  - BRAKEMAN (Ruby on Rails),
  - PMD (Java, XML e XSL),
  - PYLINT (Python),
  - CppCheck (C/C++),
  - FindBugs (Java),
  - JSHint (Javascript),
  - OWASP Dependency-Check (Java,.NET, Ruby, Node.js, Python, supporto limitato per C/C++).

L'utilizzo combinato dei tool sopra indicati consente una copertura ad ampio spettro eliminando quasi del tutto la revisione manuale che richiederebbe molto tempo.

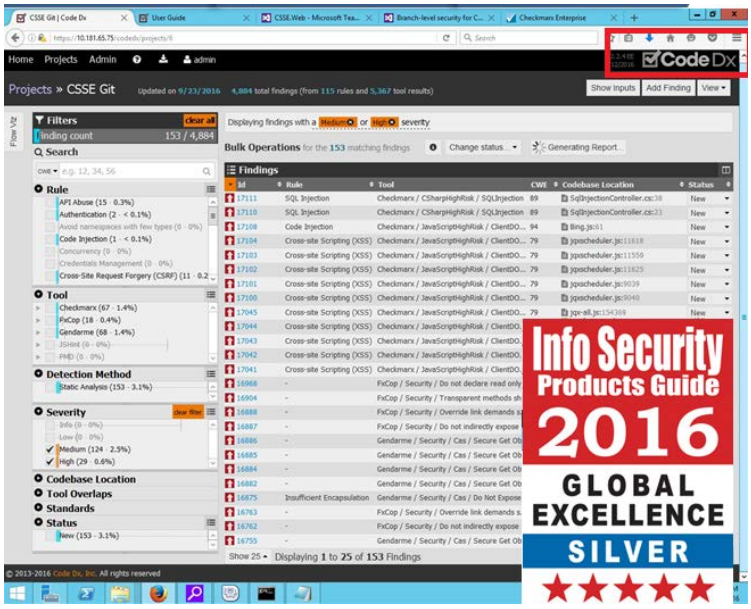
La tabella che segue mette a catalogo i risultati finali della valutazione di alcuni tool sopra indicati:

Tools	Categoria	Fase	Report
Checkmarx	SAST	Implementation / Verification	Vedi Appendice 2.a
CodeDx	SAST/DAST	Implementation/Verification	Vedi Appendice 2.b

Tools	Categoria	Fase	Report
SonarQube/SonarLint	SAST	Implementation	Vedi Appendice 2.c

1) **Checkmarx**, è un tool per l'analisi statica del codice, posizionato da Gartner nel quadrante Challengers nell'ambito dell'Application Security Testing (AST). Supporta numerosi linguaggi (vedi scheda nella tabella di cui sopra). Può essere integrato a vari livelli nell'ambito della fase di Implementation: IDE, build server, bug tracking tools.

2) **CodeDx** consente di effettuare la verifica di eventuali vulnerabilità di programmi e software presi in considerazione. CodeDx riunisce una serie di strumenti di analisi del codice (Checkmarx, **IBM App Scan**, **Veracode**), sia gratuiti che commerciali, che consentono a loro volta di individuare e correggere agevolmente eventuali bugs nel codice da analizzare. Uno screenshot dell'interfaccia CodeDx è riportata nella figura che segue:



3) **SonarQube**, consente di introdurre il controllo formale fin dall'inizio del ciclo di vita del software, attraverso l'introduzione di Quality Gate nelle fasi tipiche di passaggio tra lo sviluppo e la verifica e tra la verifica e la produzione.

9.4 Verifica della sicurezza delle applicazioni

Le azioni di sicurezza da intraprendere in questa fase sono così sintetizzate:

- **Analisi dinamica**: attraverso l'attuazione di test dinamici di sicurezza sull'applicazione in esecuzione in ambiente controllato;
- **Penetration Test**: attraverso l'esecuzione di scansioni ed analisi della superficie di attacco;
- **Test di autenticazione multilivello**: attraverso la verifica delle modalità di gestione dell'accesso degli utenti;
- **Business Logic test**: attraverso l'esecuzione di test manuali sulle applicazioni in fase di esecuzione;
- **Analisi dei risultati**: attraverso l'individuazione e la rimozione dei falsi positivi;

- **Remediation Plan**: attraverso la definizione del piano di rientro e la produzione di reportistica di sintesi e di dettaglio; Proof of Concept delle vulnerabilità riscontrate comprensiva di azioni per la riduzione della superfice d'attacco.

L'esame delle Applicazioni in esecuzione in ambiente di test, deve portare alla produzione, mediante la Dinamic Analisys delle seguenti tipologie di documenti:

- **Vulnerability Assessment**: report di dettaglio delle vulnerabilità riscontrate nella fase di analisi dinamica dell'applicazione tramite gli strumenti a supporto;
- **Remediation Plan:** report che analizza i falsi positivi ed indirizza la risoluzione delle problematiche riscontrate nell'analisi stessa.

La figura che segue sintetizza gli elementi in input e l'output prodotto dal processo DAST:

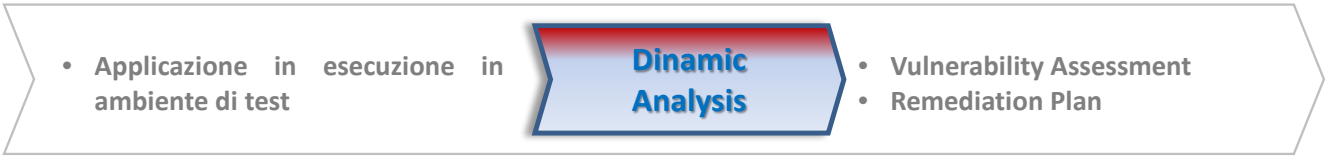


Figura 14: Input ed Output della fase Dynamic Analysis

9.4.1 Identificazione degli strumenti a supporto

Nel paragrafo 6.5.1 sono stati presentati i tool a supporto di questa fase. Tra questi si raccomandano:

- closed source
  - IBM App Scan (versione DAST),
  - Veracode,
  - CodeDx.
- open source
  - OWASP Zed Attack Proxy.

9.5 Supporto per la manutenzione di applicazioni sicure

L'obiettivo di questa fase è mantenere un prodotto sicuro, a partire dai nuovi trend sugli attacchi/minacce. Il team deve quindi analizzare le nuove minacce e individuare le contromisure necessarie rilasciando nuovi aggiornamenti/patch laddove necessario attraverso un processo di 'Continuous Security':

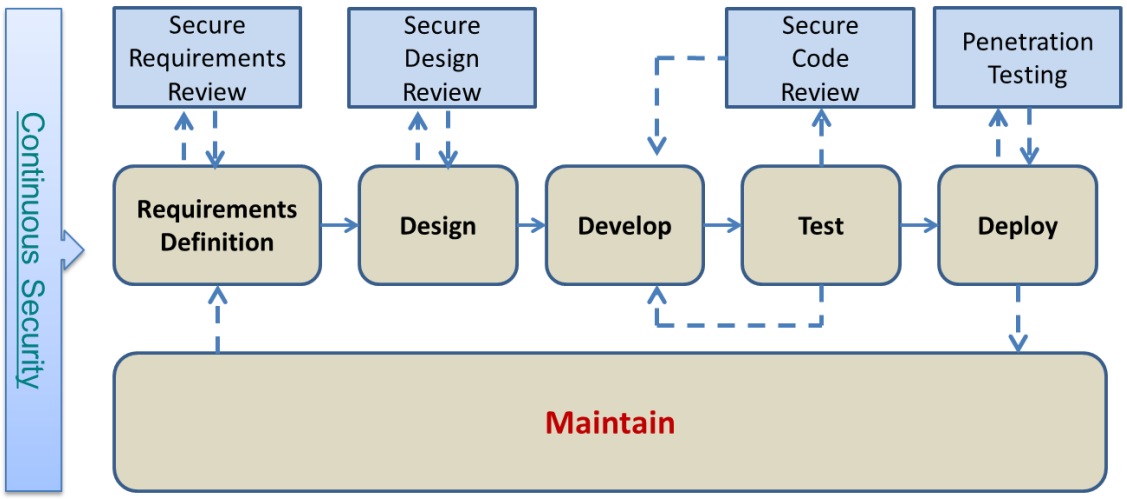


Figura 15: Continuous Security

9.5.1 Identificazione degli strumenti a supporto

In ottica di un processo di 'Continuous Security', in questa fase vengono riattuate le azioni afferenti alle diverse fasi di: Revisione dei requisiti di sicurezza, revisione dei risultati di progettazione, revisione degli aspetti di sicurezza del codice sorgente implementato, penetration test del codice rilasciato.

Gli strumenti per le fasi sopra menzionati sono stati già identificati e indicati nei precedenti paragrafi:

- Definizione dei requisiti di sicurezza [Par. 9.1.2];
- Progettazione di applicazioni sicure [Par. 9.2.1];
- Implementazione di applicazioni sicure [Par. 9.3.1];
- Verifica della sicurezza delle applicazioni [Par. 9.4.1].

10 LINEE GUIDA PER L'IMPLEMENTAZIONE DELLA PRIVACY BY DESIGN NEL SDLC

10.1 Introduzione e concetti base

10.1.1 Principi della Privacy

All'interno dalla ISO/IEC 29100:2011 (cfr. DR-2) sono descritti undici principi sulla privacy, che devono essere presi in esame per orientare la progettazione, lo sviluppo e l'implementazione dei requisiti di protezione per la privacy (10.1.4). Inoltre, devono essere ulteriormente utilizzati come riferimento per il monitoraggio e la misurazione delle prestazioni del software e come aspetti del controllo dei programmi di gestione della privacy in un'organizzazione.

Principi	Descrizione
1. Consenso e scelta	Il principio del consenso prevede di presentare all'interessato la scelta se acconsentire o meno al trattamento dei propri dati personali (Consenso Informato). Aderire al principio della scelta significa fornire all'interessato - in maniera chiara, facilmente comprensibile, accessibile e conveniente - i meccanismi per esercitare la scelta e di fornire il consenso in relazione al trattamento dei suoi dati personali al momento di raccolta, al primo utilizzo o non appena possibile.
2. Scopo legittimo e specifico	Il principio di legittimità e specificità dello scopo assicura che quest'ultimo sia conforme alla legge applicabile e si basi su una base giuridica ammissibile.
3. Limitazione della raccolta	Il principio della limitazione della raccolta prevede la limitazione della raccolta dei dati personali a ciò che è strettamente necessario per gli scopi specificati.
4. Minimizzazione dei dati	Il principio della minimizzazione dei dati prevede la progettazione e l'implementazione e l'elaborazione dei dati, attraverso procedure o sistemi ICT, in modo da ridurre al minimo i dati personali che vengono elaborati e il numero di parti interessate della privacy.
5. Limitazione dell'utilizzo, conservazione e divulgazione	Tale principio prevede la limitazione dell'utilizzo, della conservazione e della divulgazione (incluso il trasferimento) dei dati personali a ciò che è necessario per soddisfare gli scopi specifici, espliciti e legittimi del trattamento.
6. Precisione e qualità	Il principio di accuratezza e qualità assicura che i dati personali elaborati siano accurati, completi, aggiornati (a meno che non vi sia una base legittima per mantenere dati obsoleti), e adeguati e pertinenti ai fini del trattamento.
7. Apertura, trasparenza e preavviso	Tale principio prevede di fornite informazioni chiare e facilmente accessibili sulle politiche stabilite dal titolare del trattamento e delle procedure e delle pratiche relative al trattamento dei dati personali.
8. Partecipazione individuale e accesso	Il principio della partecipazione e dell'accesso individuale prevede di fornire agli interessati la possibilità di accedere e di rivedere i proprio dati personali, a condizione che la loro identità autenticata con un livello adeguato di garanzia e tale accesso non sia vietato dalla legge applicabile.
9. Responsabilizzazione	Il principio della responsabilità prevede di documentare e comunicare in modo appropriato tutte le politiche, le procedure e le pratiche relative alla privacy. Prevede altresì l'assegnazione ad un individuo specifico all'interno dell'organizzazione del compito di attuare le politiche, le procedure e le best practice relative alla privacy.
10. Sicurezza delle informazioni	Il principio di sicurezza delle informazioni prevede di proteggere i dati personali sotto la propria autorità con dei controlli appropriati a livello operativo,

	funzionale e strategico. Al fine di garantire l'integrità, la riservatezza e la disponibilità dei dati personali e proteggerli dai rischi, quali l'accesso non autorizzato, la distruzione, l'utilizzo non consentito, la modifica, la divulgazione o la perdita in tutto il ciclo di vita dell'informazione.
11. Conformità alla privacy	Il principio della conformità alla privacy prevede di verificare e dimostrare che il trattamento rispetti la protezione dei dati e la tutela della privacy, attraverso requisiti specifici e mediante verifiche periodiche – anche attraverso il ricorso a revisori interni o esterni.

Tabella 5 - Principi generali della privacy

10.1.2 Obiettivi di protezione

Gli obiettivi di protezione mirano a fornire delle proprietà astratte, ossia indipendenti dal contesto per i sistemi IT. Nella sicurezza ICT la triade della riservatezza, dell'integrità e della disponibilità è stata ampiamente accettata. Sebbene siano state proposte diverse estensioni e perfezionamenti, questi obiettivi di protezione *core* sono rimasti stabili per decenni e sono serviti da base per molte metodologie di sicurezza ICT. (cfr. DR-4). A completamento di questi obiettivi di protezione della sicurezza, sono stati proposti tre obiettivi di protezione specifici per la privacy: l'incollegabilità, la trasparenza e l'intervenibilità.

Obiettivo	**Descrizione**
Incollegabilità	L'incollegabilità garantisce che i dati rilevanti per la privacy non possano essere collegati tra domini costituiti da uno scopo e contesto comuni. Ciò significa che i processi devono essere gestiti in modo tale che i dati rilevanti per la privacy non siano collegabili a qualsiasi altro insieme di dati rilevanti sulla privacy al di fuori del dominio.
La trasparenza	La trasparenza garantisce che tutte le elaborazioni dei dati rilevanti per la privacy, comprese le impostazioni legali, tecniche e organizzative, possano essere comprese e ricostruite in qualsiasi momento. Le informazioni devono essere disponibili prima, durante e dopo l'elaborazione. Pertanto, la trasparenza deve riguardare non solo l'elaborazione effettiva, ma anche l'elaborazione pianificata (trasparenza ex ante) e il tempo trascorso dall'elaborazione per sapere cosa è successo esattamente (trasparenza ex post)
L'intervenibilità	L'intervenibilità garantisce l'intervento in relazione a tutti i trattamenti di dati relativi alla privacy in corso o pianificati, in particolare da parte di coloro i cui dati vengono elaborati. L'obiettivo dell'intervenibilità è l'applicazione di misure correttive e controbilanci ove necessario. L' intervenibilità è legata ai principi relativi ai diritti degli individui, ad es. i diritti di rettifica e cancellazione dei dati, il diritto di revocare il consenso o il diritto di presentare un reclamo o di sollevare una controversia per ottenere il rimedio.

10.1.3 Privacy by design

10.1.3.1 Definizione della Privacy by design

La Privacy by Design (PbD) è definita come "un approccio olistico concettuale che può essere applicato - end-to-end - all'interno di un'organizzazione, includendo le sue tecnologie informatiche, le sue pratiche commerciali, i suoi processi, la progettazione finisca e le infrastrutture di rete" (cfr. DR-11). Secondo questa impostazione, l'utente dovrebbe essere considerato il centro di un sistema di protezione dei dati personali (per definizione, quindi il sistmea è "user centric"). Qualsiasi progetto - sia strutturale, sia concettuale - andrebbe realizzato considerando, sin dalla fase di progettazione, la riservatezza e la protezione dei dati personali. La PbD comprende una trilogia di applicazioni:

- i sistemi IT;
- Le pratiche di business;
- La progettazione delle reti (cfr. DR-5).

Ed è in questo contesto che si inserisce la necessità di prevedere l'ingegnerizzazione della privacy by design in ogni fase del ciclo di vita del software.

10.1.3.2 I sette principi della privacy by design

Principio	**Descrizione**
Proattivo non reattivo; Preventivo non correttivo	L'approccio di *Privacy by Design* (PbD) è caratterizzato da misure proattive piuttosto che reattive. Essa è diretta ad anticipare e previene gli eventi invasivi della privacy prima che accadano. PbD non attende che i rischi per la privacy si materializzino, né offre rimedi per la risoluzione delle infrazioni della privacy una volta che si sono verificati, in quanto è diretta ad impedire che si verifichino.
Privacy come impostazione predefinita	La *Privacy by Design* è diretta a garantire il massimo grado di privacy prevedendo che i dati personali siano automaticamente protetti in qualsiasi sistema IT o di business. Nessuna azione è richiesta da parte dei singoli per proteggere la loro privacy, in quanto è integrata nei sistemi per impostazione predefinita.
Privacy incorporata nel design	La *Privacy by Design* è incorporato nel design e nell'architettura dei sistemi IT e di business. Non è attuata successivamente ad un evento. Il risultato è che la privacy diventa una componente essenziale delle funzionalità principali. La privacy è parte integrante del sistema, senza diminuirne la funzionalità.
Funzionalità completa; somma positiva, non somma zero	La *Privacy by Design* cerca di tutelare tutti i legittimi interessi e gli obiettivi in un'ottica *win-win*, senza prevedere delle soluzioni a somma zero che includano degli inutili trade-off. *Privacy by Design* evita la pretesa di false dicotomie, come la sicurezza a discapito della privacy, in quanto dimostra che è possibile averle entrambe.
Sicurezza end-to-end - Protezione completa del ciclo di vita	La *Privacy by Design* che è stata incorporata in un sistema sin dal primo momento, si estende in modo sicuro durante l'intero ciclo di vita dei dati coinvolti: prevedendo robuste misure di sicurezza - essenziali per la privacy - dall'inizio alla fine di un ciclo di vita. Ciò garantisce che tutti i dati vengano conservati e distrutti – in modo sicuro e tempestivamente - alla fine del processo. Pertanto, la *Privacy by Design* garantisce una gestione delle informazioni sicura end-to-end.
Visibilità e trasparenza - Keep it Open	La *Privacy by Design* cerca di assicurare a tutti gli stakeholder che qualunque sia la pratica aziendale o la tecnologia coinvolta, essa opererà secondo le promesse e gli obiettivi dichiarati, anche assoggettandosi a verifiche indipendenti. Le sue componenti e le sue operazioni rimangono visibili e trasparenti, sia per gli utenti che per i fornitori.
Rispetto per la privacy degli utenti - Mantenerlo incentrato sull'utente	La *Privacy by Design* richiede ai progettisti e agli operatori di garantire gli interessi dei singoli, offrendo robuste misure di privacy per impostazione predefinita. Prevedendo degli avvisi appropriati e potenziando le opzioni user-friendly, pertanto garantendo l'impostazione user-centric.

Tabella 6 - I sette principi della Privacy by Design

10.1.4 Data protection Impact Assessment

La progettazione di qualsiasi software che coinvolga il trattamento dei dati personali deve essere preceduta da un'identificazione dei requisiti di protezione per la privacy, in quanto dal trattamento o dall'elaborazione dei dati personali potrebbero derivare dei rischi. I rischi per la privacy negli applicativi software che comportano il trattamento dei dati personali, dovrebbero essere trattati prima della loro implementazione, ossia sin dalla fase di progettazione (*Engineering Privacy by Design*). Dovranno quindi essere analizzati i rischi collegati alle applicazioni software.

In linea con i requisiti di attuazione previsti dal Regolamento (UE) 679 del 2016 (cfr. DR-1), di seguito indicato come **GDPR**, qualora un trattamento dei dati personali possa presentare un rischio elevato per i diritti e le libertà delle persone fisiche, i titolari di quest'ultimo dovranno effettuare una valutazione dell'impatto del trattamento sulla protezione dei dati personali o *Data Protection Impact Assessment*, di seguito indicata come "DPIA" (cfr. Art. 35 DR-1), quest'obbligo è applicabile anche al ciclo di vita del software.

Sulla base di quanto stabilito dal WP Art. 29 (cfr. DR-10), sarà necessario effettuare una valutazione della necessità di svolgere una DPIA, basandosi sulla mappa concettuale definita nella Figura 16**.**

In particolare, al fine di valutare se il trattamento - posto in essere all'interno di un'applicazione software - possa comportare un rischio elevato per i diritti e le libertà delle persone fisiche (Cfr. ART. 35 DR-1) sarà necessario determinare se rientra tra quelli indicati nella Tabella 8- in cui sono descritte alcune tipologie di trattamento che obbligano il titolare a svolgere una Data Protection Impact Assessment DPIA.

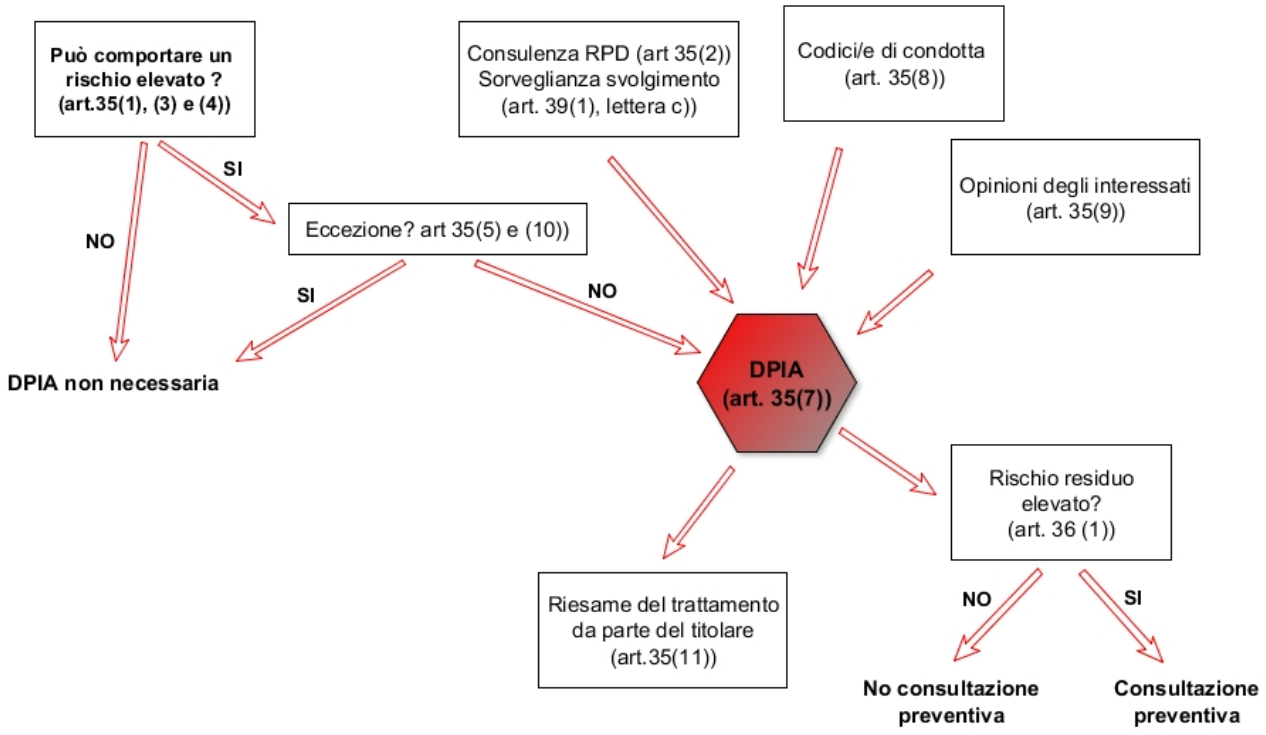


Figura 16 – Flusso di valutazione necessità DPIA

Pertanto, se il trattamento, le sue modalità di attuazione o i dati trattati rientrano in quelli descritti nella Tabella 8, e non si configurano eccezioni – individuate all'interno di elenchi che dovranno essere redatti dagli Stati Membri (ad oggi non risultano essere stati ancora individuati) - sarà necessario svolgere una DPIA.

Tipologia di trattamento	Descrizione
1 - Valutazione di profilazione o scoring	Tutti quei trattamenti che analizzano i dati presenti all'interno dei propri archivi allo scopo di trarne informazioni riguardo il rendimento professionale, la situazione economica, la salute, le preferenze o gli interessi personali, l'affidabilità o il comportamento, l'ubicazione o gli spostamenti dell'interessato
2 - Decisioni automatizzate	Tutti quei trattamenti che producano effetti giuridici sulla persona fisica ovvero che incidono in modo analogo significativamente su dette persone fisiche
3 - Monitoraggio sistematico	Tutti quei trattamenti che sono utilizzati per osservare, monitorare o controllare gli interessati, compresa la raccolta di dati attraverso reti o la sorveglianza di un'area accessibile al pubblico
4 - Dati sensibili o estremamente personali	Tutti quei trattamenti che si riferiscono a particolari categorie di dati sensibili o estremamente personali
5 - Dati trattati su larga scala	Tutti i trattamenti che gestiscono dati personali su larga scala, in relazione al numero di soggetti interessati, al volume dei dati, alla durata o all'ambito geografico
6 - Combinazioni o raffronto di insieme di dati	Tutti quei trattamenti nei quali è prevista una presenza congiunta di due o più titolari distinti, secondo modalità che esulano dalle ragionevoli aspettative dell'interessato
7 - Dati relativi a interessati vulnerabili	Tutti quei trattamenti in cui la tipologia delle informazioni trattate determina uno squilibrio fra interessato e titolare, nel senso della mancanza del potere, in capo al primo, di acconsentire o di opporsi al trattamento. Si inseriscono in questa categoria i dati dei minori, dei dipendenti o delle persone richiedenti specifiche tutele
8 - Utilizzi innovativi	Tutti quei trattamenti che utilizzano tecnologie o tecniche innovative per la raccolta o l'utilizzo dei dati personali, dato che il livello di conoscenza tecnologica, in un dato momento storico, non è in grado valutare il livello di rischio connesso all'innovazione
9 - Trattamenti che impediscono di esercitare un diritto o avvalersi di un servizio o contratto	Tutti quei trattamenti che impediscono agli interessati di esercitare un diritto di avvalersi di un servizio o di un contratto, ossia tutti i trattamenti dai quali l'interessato non può esimersi qualora volesse accedere a detto servizio o concludere detto contratto

Tabella 7 - Tipologie di trattamento che rappresentano un rischio elevato

Nel caso in cui la DPIA sia stata valutata come necessaria (cfr. DR-10), si potrà procedere con l'analisi degli impatti potenziali sui diritti e le libertà dell'interessato (persone fisiche), a fronte del trattamento dei relativi dati personali, allo scopo di porre in essere le opportune attività di trattamento dei rischi per la protezione dei dati personali.

In linea con quanto previsto da regolamenti e standard applicabili in materia (cfr. DR-6), tale attività costituisce un processo composto da un insieme di attività ben definite, da compiersi in sequenza ordinata, nell'ambito delle seguenti fasi:

1) **Definizione del contesto**, tramite la comprensione dell'organizzazione, dell'architettura tecnologica e dei fattori che potrebbero influenzare la gestione del rischio privacy;

2) **Privacy risk assessment**, attraverso cui si identificano, si analizzano e si valutano i rischi per gli interessati;
3) **Privacy risk treatment**, in cui si identificano le strategie e le modalità operative per l'implementazione delle misure di sicurezza adeguate alla copertura dei rischi rilevati in sede di risk assessment. (I requisiti di protezione per la privacy, da implementare all'interno del piano di trattamento dei rischi individuati per il software possono essere ricavati dai controlli descritti nella ISO/IEC 29151 (cfr. DR-7)

10.1.4.1 Riconoscere le informazioni personali identificabili (PII)

Per poter definire adeguatamente il trattamento del rischio privacy per i software, sarà necessario individuare le tipologie di informazioni personali identificabili (PII), ossia quelle da cui possono essere ricavati dei dati personali, che potrebbero essere trattate da un applicativo software. Per determinare se una persona fisica debba o meno essere considerata identificabile, sarà necessario prendere in considerazione diversi fattori. In particolare, si dovrebbe tenere conto dei mezzi che possono ragionevolmente essere utilizzati dai software per il trattamento dei dati personali. I software dovrebbero supportare meccanismi adeguati ad informare l'interessato, raccogliere il consenso e progettare i suoi dati personali. Le seguenti specificazioni forniscono degli ulteriori chiarimenti su come determinare se una informazione possa essere considerata PII.

Identificativi

In alcuni casi, l'identificabilità dell'interessato potrebbe essere molto semplice (e.g. quando l'informazione contiene o è associata ad un identificatore che è usato per riferirsi o per comunicare con l'interessato). Le informazioni possono essere considerate PII almeno nei seguenti casi:

- se contiene o è associato a un identificatore che fa riferimento a una persona fisica (ad esempio, il codice fiscale);
- se contiene o è associato a un identificatore che può essere correlato a una persona fisica (ad esempio, numero del passaporto, numero di conto);
- se contiene o è associato a un identificatore che può essere utilizzato per stabilire una comunicazione con
- una persona fisica identificata (ad esempio, una posizione geografica precisa, un numero di telefono);
- se contiene un riferimento che collega i dati a uno degli identificatori di cui sopra.

Altre caratteristiche identificative

Le informazioni non devono necessariamente essere associate a un identificatore per poter essere considerate PII. Le informazioni saranno considerate PII anche se contengono o sono associate a una caratteristica che distingue una persona fisica da altre persone fisiche, ad esempio i dati biometrici. Qualsiasi attributo che assume un valore che identifica univocamente un l'interessato deve essere considerato come una caratteristica identificativa. Si noti che indipendentemente dal fatto che una determinata caratteristica distingue una persona fisica da altre potrebbe cambiare a seconda del contesto di utilizzo. Ad esempio, mentre il cognome di una persona fisica potrebbe essere insufficiente per identificare quella persona naturale su una globale scala, sarà spesso sufficiente distinguere una persona fisica su una scala aziendale. Inoltre, ci possono anche essere situazioni in cui una persona fisica è identificabile anche se non c'è singolo attributo che lo identifica in modo univoco. Questo è il caso in cui una combinazione di diversi attributi presi insieme distingue questa persona da altre, ad esempio la combinazione degli attributi "femmina", "45" e "avvocato" può essere sufficiente per identificare una persona fisica all'interno di una determinata organizzazione, ma spesso sarà insufficiente per identificare quella persona fisica al di fuori di tale contesto.

La Tabella 9 fornisce alcuni esempi di attributi che potrebbero essere PII, a seconda del dominio.

Età o bisogni speciali delle persone fisiche vulnerabili	Posizione derivata dai sistemi di telecomunicazione
Accuse di condotta criminale	Storia medica
Qualsiasi informazione raccolta durante i servizi sanitari	Nome
Conto bancario o numero di carta di credito	Identificativi nazionali (ad es. Numero di passaporto)
Identificatore biometrico	Indirizzo e-mail personale
Estratto conto della carta di credito	Numeri di identificazione personale (PIN) o password
Condanne penali o reati commessi	Interessi personali derivati dall'utilizzo di tracciamento di siti Web
Rapporti di indagini penali	Profilo personale o comportamentale
Numero cliente	Numero di telefono personale
Data di nascita	Fotografia o video identificabili con una persona fisica
Informazioni sanitarie diagnostiche	Preferenze di prodotto e servizio
disabilità	Origine razziale o etnica
Fatture del medico	Credenze religiose o filosofiche
Stipendi dei dipendenti e file di risorse umane	Orientamento sessuale
Profilo finanziario	Appartenenza sindacale
Genere	Bollette
Posizione GPS	
Traiettorie GPS	
Indirizzo di casa	
indirizzo IP	

Tabella 8 - Esempi di Attributi per indentificare una persona

Dati pseudonimizzati

Al fine di limitare la capacità del titolare o del responsabile di identificare l'interessato, la sua identità e le sue informazioni possono essere sostituite da degli pseudonimi. Questa sostituzione viene solitamente eseguita da un soggetto terzo prima di trasmettere le PII a un destinatario. Questo è spesso il caso in cui i dati sensibili devono essere elaborati titolari che non li hanno raccolti. La sostituzione è considerata pseudonimizzazione quando:

(a) gli attributi collegati allo pseudonimo non sono sufficienti per identificare l'interessato;

(b) l'assegnazione degli pseudonimi è tale da non poter essere invertita da parte delle persone che l'hanno eseguita.

La pseudonimizzazione evita il collegamento. Ma esseno diversi i dati collegabili allo stesso pseudonimo, ci sarà il rischio che la pseudonimizzazione sia violata, in quanto più grande è il set di dati associato a un dato pseudonimo, maggiore è il rischio che la proprietà (a) sia violata. Inoltre, più piccolo è il gruppo di persone fisiche a cui un insieme di dati pseudonimi si riferisce, maggiore sarà la probabilità che un interessato sia identificabile. Gli attributi contenuti direttamente nelle informazioni in questione e gli attributi che possono essere facilmente collegati a queste informazioni (ad es. utilizzando un motore di ricerca o dei riferimenti incrociati con altri database) devono essere presi in considerazione nel determinare se l'informazione si riferisce a un elemento identificabile dell'interessato.

La pseudonimizzazione è in contrasto con l'anonimizzazione. I processi di anonimizzazione soddisfano anche le proprietà (a) e (b) sopra, ma eliminano il collegamento. Durante l'anonimizzazione, le informazioni sull'identità vengono cancellate o sostituito da pseudonimi per i quali la funzione di assegnazione viene distrutta. Quindi, i dati resi anonimi non sono più PII.

Metadati

Le PII possono essere memorizzate in un sistema ICT in modo tale da non essere facilmente visibili all'utente del sistema. Ad esempio, la memorizzazione del nome dell'interessato come metadato nelle proprietà di un documento, nei commenti o nelle modifiche. L'interessato deve essere a conoscenza

dell'esistenza delle PII sotto forma di metadati o del trattamento delle PII per tale scopo, in quanto potrebbe preferire che le PII non vengano elaborate in questo modo o condivise pubblicamente.

Dati non richiesti

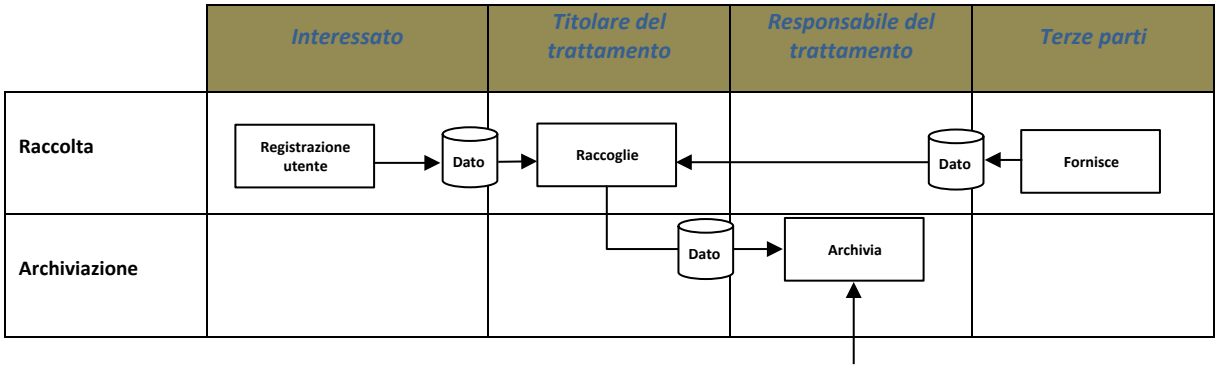
Anche le PII non richieste da un titolare, cioè non intenzionalmente ottenute, potrebbero essere memorizzate da un software. Ad esempio, l'interessato potrebbe fornire delle PII anche quando non è stato richiesto dal trattamento (ad es. ulteriori informazioni personali fornite nel contesto di un modulo di feedback anonimo su un sito Web). Il rischio di raccogliere informazioni personali indesiderate può essere ridotto considerando le misure di tutela della privacy al momento della progettazione del software.

I dati personali stabiliti dal GDPR (cfr. DR-1) sono suddivisi nelle seguenti categorie di dati personali:

Categorie di dati personali	Descrizione
Dati identificativi	I dati identificativi rappresentano tutti quei dati che possono identificare, direttamente o indirettamente una persona, con particolare riferimento a un identificativo come il nome, un numero di identificazione, dati relativi all'ubicazione, un identificativo online.
Dati Sensibili	I dati sensibili sono tutti quei dati personali che rivelino l'origine razziale o etnica, le opinioni politiche, le convinzioni religiose o filosofiche, o l'appartenenza sindacale, nonché dati genetici dati biometrici intesi a identificare in modo univoco una persona fisica, dati relativi alla salute o alla vita sessuale o all'orientamento sessuale della persona.
Dati giudiziari	I dati giudiziari rappresentano tutti quei dati relativi alle condanne penali e ai reati o a connesse misure di sicurezza.

10.1.5 Flusso informativo del trattamento

Per definire l'architettura e il design di un software i progettisti dovranno prendere in considerazione la struttura del flusso informativo, descrivendo le interazioni tra interessato, titolare, responsabile e terze parti all'interno dell'applicativo software. Gli attori identificati possono interagire tra loro in vari modi, secondo i seguenti scenari, maturati dalla ISO/IEC 29134:2017 (cfr. DR-6):



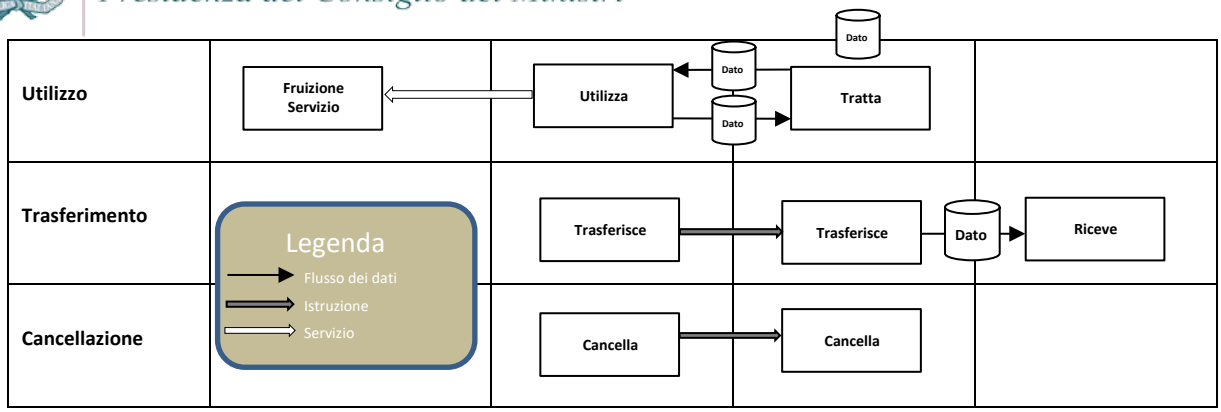


Figura 17 - Esempio di flusso informativo del trattamento

10.1.6 Privacy Implementation Strategy

La Privacy Implementation Strategy prevede che i progettisti del software definiscano e selezionino un modello di ciclo di vita adeguato all'ambiente di produzione e di sviluppo, all'ambito, all'ampiezza e alla complessità del progetto, parametrato sulle necessità emerse dai risultati della data protection impact assessment per la privacy (10.1.4).

Dovranno essere documentate:

- I principi generali della privacy applicabili alla progettazione del software (10.1.1)
- Gli obiettivi di protezione che il software dovrebbe garantire (10.1.2)
- I principi della privacy by design applicabili alla progettazione del software (10.1.3.2)
- I risultati della data protection impact assessment per il software e l'individuazione dei requisiti di protezione per la privacy (10.1.4)
- Le tipologie di Informazioni Personali Identificabili (PII) trattate nell'ambiente software (10.1.4.1)
- La descrizione del flusso informativo derivante dal trattamento all'interno del software (10.1.5)

10.2 Implementazione della strategia nelle fasi di sviluppo del software

10.2.1 Scopo

Gli elementi definiti all'interno della Privacy Implementation Strategy (10.1.6), i requisiti di protezione della privacy e le strategie di design per la privacy (ricavabili sulla base di quelli individuati da ENISA in DR-4), dovranno essere inquadrati all'interno di ciascuna fase della Engineering privacy by design (10.2.3) e rimappati per ciascuna fase del ciclo di vita dei software (10.2.2), così come definiti nelle fasi *Software life Cycle Processes* (cfr. DR-3).

10.2.2 Le fasi di implementazione della Engineering Privacy by Design

La seguente impostazione è stata maturata dal *Privacy Engineering Framework* del MITRE (cfr. DR-9), prevedendo le seguenti attività:

Attività	Descrizione
Definizione dei requisiti privacy:	Definizione delle proprietà della privacy di un software in modo che possa essere integrato con il design e lo sviluppo
Design e sviluppo privacy:	Definizione del design e sviluppo dei requisiti previsti
Verifica e validazione privacy:	Riscontro della conferma che i requisiti di privacy sono stati correttamente implementati e validati attraverso delle verifiche

Tabella 9 - Fasi dell'Engineering Privacy by Design

10.2.2.1 Definizione dei requisiti privacy

Input: Requisiti di privacy di base e test; Normative, best practice e procedure applicabili sulla privacy; requisiti funzionali; Profili di rischio per la privacy.

Attività: Svolgere una Data Protection Impact Assessment parametrata sugli obiettivi di protezione individuati; Selezionare e perfezionare i requisiti di protezione per la privacy di base e effetturare dei test; Sviluppare dei requisiti di protezione della privacy personalizzati e testarli sulla base dei risultati della DPIA.

Output: Requisiti di protezione per la privacy specifici per il software.

10.2.2.2 Design e sviluppo privacy

Input: Requisiti Architetturali e funzionali specifici per la privacy

Attività: Identificare delle strategie e dei modelli di design della privacy; Identificare dei controlli di privacy, dei criteri tecnici e delle policy; Sviluppare dei dati e dei modelli di processo che riflettano i controlli di privacy identificati; Allineare, integrare e implementare i controlli di privacy con gli elementi funzionali; Analizzare il rischio del design di privacy complessivo.

Output: Componenti del software implementati; Mitigazione dei rischi accettabili per la privacy residua

10.2.2.3 Verifica e validazione privacy

Input: Componenti del software implementati; Requisiti di privacy specifici del sistema e test]; Politiche e procedure di privacy applicabili.

Attività: Sviluppare / perfezionare dei test sulla privacy; Eseguire delle verifiche sulla privacy; Verificare il comportamento operativo rispetto alle politiche e alle procedure sulla privacy applicabili.

Output: Risultati dei test di privacy; Documentazione delle Incoerenze sulla privacy documentate; Descrizione del piano di trattamento della privacy.

10.3 Integrazione della Engineering Privacy by Design nel Software Life Cycle Process

Il diagramma illustrato nella Figura 18, definisce la mappatura delle fasi della Engineering Privacy by Design sulle fasi del Software Life Cycle Process:

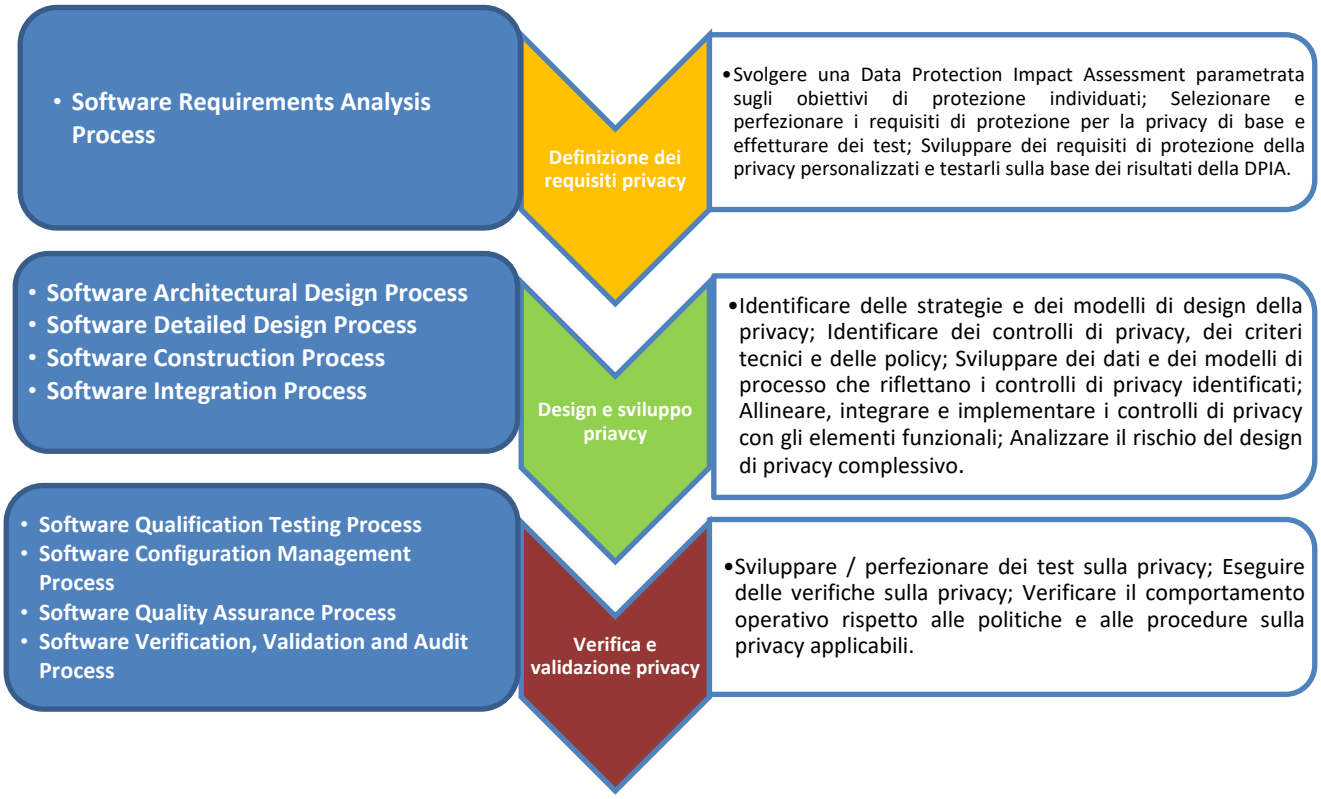


Figura 18 - Integrazione della Engineering privacy by design nel Software Life Cycle Process

La seguente impostazione è stata maturata dal *Privacy Engineering Framework* del MITRE (cfr. DR-9), prevedendo le seguenti attività:

Attività	Descrizione
Definizione dei requisiti privacy:	Definizione delle proprietà della privacy di un software in modo che possa essere integrato con il design e lo sviluppo
Design e sviluppo privacy:	Definizione del design e sviluppo dei requisiti previsti
Verifica e validazione privacy:	Riscontro della conferma che i requisiti di privacy sono stati correttamente implementati e validati attraverso delle verifiche

Tabella 10 - Fasi dell'Engineering Privacy by Design

10.4 Definizione dei requisiti privacy

Input: Requisiti di privacy di base e test; Normative, best practice e procedure applicabili sulla privacy; requisiti funzionali; Profili di rischio per la privacy.

Attività: Svolgere una Data Protection Impact Assessment parametrata sugli obiettivi di protezione individuati; Selezionare e perfezionare i requisiti di protezione per la privacy di base e effetturare dei test; Sviluppare dei requisiti di protezione della privacy personalizzati e testarli sulla base dei risultati della DPIA.

Output: Requisiti di protezione per la privacy specifici per il software.

10.5 Design e sviluppo privacy

Input: Requisiti Architetturali e funzionali specifici per la privacy

Attività: Identificare delle strategie e dei modelli di design della privacy; Identificare dei controlli di privacy, dei criteri tecnici e delle policy; Sviluppare dei dati e dei modelli di processo che riflettano i controlli di privacy identificati; Allineare, integrare e implementare i controlli di privacy con gli elementi funzionali; Analizzare il rischio del design di privacy complessivo.

Output: Componenti del software implementati; Mitigazione dei rischi accettabili per la privacy residua

10.6 Verifica e validazione privacy

Input: Componenti del software implementati; Requisiti di privacy specifici del sistema e test]; Politiche e procedure di privacy applicabili.

Attività: Sviluppare / perfezionare dei test sulla privacy; Eseguire delle verifiche sulla privacy; Verificare il comportamento operativo rispetto alle politiche e alle procedure sulla privacy applicabili.

Output: Risultati dei test di privacy; Documentazione delle Incoerenze sulla privacy documentate; Descrizione del piano di trattamento della privacy.

APPENDICE 1. CATALOGO SECURITY TOOLS

Prodotto	Categoria	Fase SSE	Tipo Licenza	Sito Web
Acunetix Web Vulnerability Scanner	DAST, IAST	Verification	14 Day Free Trial	www.acunetix.com
Adallom	Cloud Access Security Broker	Verification	Available by Request	adallom.com
Airlock Suite by Ergon Informatik	WAF, Authentication, Identity Management	Response	Available by Request	https://www.airlock.com
Akamai	CDN, DDoS Protection, WAF	Response	N/A	www.akamai.com
Alert Logic Security-as-a-Service	Intrusion Prevention System, Cloud Access Security Broker, WAF	Response	Available by Request	www.alertlogic.com
Amazon WAF	WAF	Response	N/A	https://www.aws.amazon.com
Analyst Pro	Requirements management	Requirements		http://www.analysttool.com
aNimble	Requirements management	Requirements	Free	http://sourceforge.net/projects/nimble/
AppMobi Security Kit	Apache Cordova App Encryption and Authentication	Response	Available by Request	www.appmobi.com
AppSpider Pro by Rapid7	DAST	Verification	Available by Request	https://www.rapid7.com
Appthority	Mobile AST	Verification	Available by Request	https://www.appthority.com

AppWall by Radware	WAF, DDoS Protection	Response	Available by Request	www.radware.com
Arbor Networks APS	DDoS Protection	Response	N/A	https://www.arbornetworks.com
Armor Complete	Cloud Security Platform	Release	Available by Request	https://www.armor.com
Arxan Application Protection	Anti-Tamper Software	Response	Available by Request	www.arxan.com
AuditMyApps by Pradeo	Mobile AST	Verification	Available by Request	https://auditmyapps.com
Backtrack-linux	Penetration Testing	Verification	Open Source	www.backtrack-linux.org
Barracuda Firewal	WAF	Response	N/A	barracuda.com
BeEF	Penetration Testing	Verification	Open Source	beefproject.com
Bit9 + Carbon Black	Endpoint Security	Verification / Response	Available by Request	www.bit9.com
Black Duck Hub	Open Source Scanning	Verification	Available by Request	https://www.blackducksoftware.com
Blue Coat Cloud	Cloud Access Security Broker, WAF	Response	Available by Request	https://www.bluecoat.com
Bluebox	Mobile Access Security Broker	Response	Available by Request	https://www.bluebox.com
BRAKEMAN	SAST	Implementation	Open Source	https:// brakemanscanner.org
BrightCloud Threat Intelligence by Webroot	DAST	Verification	N/A	https://www.brightcloud.com
Burp Suite by PortSwigger	SAST, DAST, Penetration Testing	Implementation / Verification	Free Tier	https://portswigger.net
CaseComplete	Requirements management	Requirements		http.//casecomplete.com
CppCheck	SAST	Implementation	Open Source	cppcheck.sourceforge.net
CD Protection by CD Networks	CDN, WAF, DDoS Protection	Response	N/A	https://www.cdnetworks.com

Checkmarx CxSAST	SAST, DAST, RASP	Implementation / Verification	Available by Request	checkmarx.com
Cigital	SAST, DAST	Implementation / Verification	N/A	https://www.cigital.com
CipherCloud	Cloud Access Security Broker	Response	Available by Request	https://www.ciphercloud.com
Cisco ACE WAF	WAF	Response	N/A	www.cisco.com
CloudFlare	CDN, DDoS Protection, WAF	Response	N/A	www.cloudflare.com
CloudFront by Amazon	CDN, DDoS Protection	Response	N/A	https://www. aws.amazon.com
CloudLock Security Fabric	Cloud Access Security Broker	Response	Available by Request	https://www.cloudlock.com
CloudPassage Halo	Cloud Access Security Broker	Response	Available by Request	https://www. cloudpassage.com
CloudSOC by Elastica	Cloud Security Testing/Scanning	Verification	Free Risk Assessment	https://www.elastica.net
Code Assure Solo	Requirements management	Requirements		
CodeDx	SAST, DAST	Implementation / Verification	Available by Request	https://codedx.com
CodeProfiler by Virtual Forge	SAST	Implementation	Available by Request	https://www.virtualforge.com
ContextIntelligence by Yottaa	CDN, DDoS Protection, WAF	Verification	N/A	www.yottaa.com
Contrast Enterprise	IAST, RASP	Implementation / Verification	Available by Request	https://www.contrastsecurity.com
Coras	Threat Modeling tool/practies	Design	Open Source	coras.sourceforge.net
DDoS Strike by Security Compass	DDoS Protection	Response	Available by Request	https://www.securitycompass.com
Defendpoint by Avecto	Endpoint Security	Verification /	Available by Request	https://www.avecto.com

		Response		
DenyAll WAF	WAF	Response	N/A	www.denyall.com
Dependency Checker	Library Inspection	Implementation	Open Source	https://www.owasp.org
F5 Big-IP ADC platform	WAF, DDoS Protection	Response	N/A	https://f5.com
Falcon Host by CrowdStrike	Endpoint Security	Verification / Response	Available by Request	https://www.crowdstrike.com
FindBugs	SAST	Implementation	Open Source	findbugs.sourceforge.net
FireEye NX	Web Server Scanner, WAF	Response	N/A	https://www. fireeye.com
Fortigate Firewall Platform by Fortinet	WAF	Response	Available by Request	https://www.fortinet.com
FortiWeb by Fortinet	WAF	Response	Available by Request	https://www.fortinet.com
Gendarme	SAST	Implementation	Open Source	www.mono-project.com/Gendarme
GMARC	Requirements management	Requirements		
HP Fortify Static Code Analyzer	SAST, DAST, IAST, RASP	Implementation / Verification	Available by Request	www.hp.com
IBM Security AppScan	SAST, DAST, IAST	Implementation / Verification	Available by Request	https://www.ibm.com
IBM DOORS Next Generation	Requirements management	Requirements		https:/.ibm.com
IBM Rational RequisitePro solution	Requirements management	Requirements		https:/ibm.com
Imperva Incapsula	WAF, DDoS Protection	Response	N/A	www.imperva.com
InfoBlox DNS Firewall	WAF	Response	60 Day Free Trial	www.infoblox.com

Intelligent Next-Gen T-Series Firewall by Hillstone Networks	WAF	Response	N/A	https://www.hillstonenet.com
		Verification	Available by Request	https://www.hillstonenet.com
IrqA	Requirements management	Requirements		jshint.com
JSHint	SAST	Implementation	Open Source	
Kali Linux	Penetration Testing	Verification	Open Source	kali.org
Klocwork by Rogue Wave Software	Code Quality Scanning	Response	Available by Request	https://www.klocwork.com
Kona Site Defender by Akamai	WAF, DDoS Protection	Response	N/A	www.akamai.com
Level 3 Content Delivery Network	CDN, DDoS Protection	Response	N/A	www.level3.com
LogRhythm Security Intelligence Platform	Predictive Security Analytics	Verification / Response	Available by Request	www.logrhythm.com
Malwarebytes Endpoint Security	Endpoint Security	Verification	N/A	www.malwarebytes.org
MetaFlows	Cloud Security Scanning	Implementation	14 Day Free Trial	www.metaflows.com
Metascan by OPSWAT	SAST	Implementation	Available by Request	https://www.opswat.com
Metasploit by Rapid7	Penetration Testing	Verification	Open Source	www.metasploit.com
Microsoft Application Verifier	DAST	Verification	Free	www.microsoft.com
Microsoft Attack Surface Analyzer	Intrusion Prevention	Verification	Free	www.microsoft.com
Microsoft BinScope	SAST	Implementation	Free	www.microsoft.com

Microsoft Code Analysis Tool	SAST	Implementation	Free	www.microsoft.com
Microsoft FxCop	Library Inspection	Implementation	Free	www.microsoft.com
Microsoft SDL Regex Fuzzer	SAST	Implementation	Free	www.microsoft.com
Microsoft SDL MiniFuzz File Fuzzer	SAST	Implementation	Free	www.microsoft.com
Microsoft Security Assessment Tool (MSAT)	Risk Management	Risk Assessment	Free	https://technet.microsoft.com/it-it/security/cc185712.aspx
Microsoft Threat Modeling Tool	Threat Modeling tool	Design	Free	https://www.microsoft.com
ModSecurity	WAF	Implementation / Verification	Open Source	modsecurity.org
MyAppSecurity ThreatModeler	Threat Modeling tool	Design	Available by Request	myappsecurity.com
N-Stalker Cloud Web Scan	SAST, DAST	Implementation / Verification	Free Tier Available	https://www.nstalker.com
NetScaler AppFirewall by Citrix	WAF	Verification	N/A	citrix.com
Netsparker Web Application Security Scanner	DAST	Response	Available by Request	www.netsparker.com
Neustar	DDoS Protection	Response	N/A	www.neustar.biz
Nevis Security and Compliance Suite by AdNovum	WAF, Authentication, Identity mngt	Verification	Available by Request	www.adnovum.ch

Nikto2	Web Server Scanner	Verification	Open Source	cirt.net
Nmap	Penetration Testing and Network Mapping	Verification / Response	Open Source	www.nmap.org
NSFOCUS Web Application Firewall	DAST, WAF	Verification	N/A	www.nsfocus.com
Objectives	Requirements management	Requirements	Available by Request	http://www.objectiver.com
OWASP Dependency Check	SAST	Implementation	Open Source	www.owasp.org
OWASP Zed Attack Proxy (ZAP)	Penetration Testing	Verification / Response	Open Source	www.owasp.org
Open Source Requirements Management Tool (OSRMT)	Requirements management	Requirements	Open Source	http://sourceforge.net/projects/osrmt/
Optimaltrace	Requirements management	Requirements		www.compuware.com/products/optimaltrace
PA-7000 Series Firewall by Palo Alto Networks	WAF	Verification Verification	N/A	https://www.paloaltonetworks.com
Palo Alto Enterprise Security Platform	RASP WAF	Response	Available by Request	https://www.paloaltonetworks.com
Peach Fuzzer	Penetration Testing	Verification / Response	Available by Request	www.peachfuzzer.com
PYLINT	SAST	Implementation	Open Source	https://www www.pylint.org
PMD	SAST	Implementation	Open Source	https://pmd.github.io

Polarion[1]	Application Lifecycle Management (ALM)	Requirements		http://www.emerasoft.com/agile-application-lifecycle-management/polarion-alm/
Prevoty	RASP	Verification / Response	Available by Request	www.prevoty.com
ProAccel by Bricata	Intrusion Prevention System	Response	Available by Request	www.bricata.com
ProtectWise Cloud Network DVR	CDN, App Security Scanning	Verification	Available by Request	http://www.protectwise.com
Qualys Security & Compliance Suite	DAST, WAF	Verification / Response	Available by Request	https://www.qualys.com
Reqtify	Requirements management	Requirements		http://users.reqtify.tni-software.com/?p=home
Risk Fabric by Bay Dynamics	Predictive Security Analytics	Implementation / Verification / Response	Available by Request	https://baydynamics.com
rmtoo	Requirements management	requirements	Free	http://sourceforge.net/projects/rmtoo/
RSA ECAT by EMC	DAST	Implementation / Verification	Available by Request	https://www.emc.com
RTD	Requirements management	Requirements		http://www.igatech.com/rdt
RTM	Requirements management	Requirements		http://www.serena.com/Products/rtm/home.asp
Samurai Web Testing Framework	DAST, Penetration testing	Verification	Open Source	https://www.samurai-wtf.org

SeaMonster	Requirements management	Requirements		https://sourceforge.net/projects/seamonster/
Security AppScan by IBM	SAST, DAST, IAST	Implementation / Verification	Available by Request	https://www.ibm.com
SiteLock TrueCode SAST	SAST, DAST	Implementation / Verification	Available by Request	https://www. sitelock.com
SonarLint	SAST	Implementation	Open Source	https://www.sonarlint.org
SonarQube	SAST	Implementation	Open Source	https://www.sonarqube.org
Sophos Next-Gen Firewall	WAF	Response	30 Day Free Trial	www.sophos.com
SRX Series Firewall by Juniper Networks	WAF	Verification	N/A	www.juniper.net
Sucuri	WAF	Verification	N/A	www.sucuri.net
Sucuri Website Firewall	WAF, DDoS Protection, App Security Scanning	Response	Available by Request	www.sucuri.net
Symantec Advanced Threat Protection	IAST, RASP	Implementation / Verification	60 Day Free Trial	https://www.symantec.com
Tanium Endpoint Platform	Endpoint Security, App Security Scanning	Implementation / Verification	Available by Request	https://www.tanium.com
TcSE (Teamcenter Systems Engineering)	Requirements management	Requirements		
Telelogic DOORS	Requirements Management	Requirements	Free	http://telelogic-doors.software.informer.com/

Thunder TPS by A10 Networks	DDoS Protection	Verification / Response	N/A	https://www.at10networks.com
Trend Micro Deep Security Platform	SAST, DAST	Implementation / Verification	N/A	https://www.trendmicro.com
TRIKE	Threat Modeling tool/practies	Design	Open Source	http://octotrike.org/tools.shtml
Tripwire Enterprise	IAST, RASP	Implementation / Verification	Available by Request	https://www.tripwire.com
Trustwave Secure Web Gateway	CDN, DAST	Verification	N/A	www.trustwave.com
Trustwave Web Application Firewall	WAF, Penetration Testing	Verification	N/A	www.trustwave.com
Veracode Cloud Platform	SAST, DAST, Mobile AST, Penetration Testing	Implementation / Verification	Available by Request	www.veracode.com
vSentry by Bromium	Endpoint Security	Verification / Response	Available by Request	www.bromium.com
vThreat Platform	Penetration Testing, App Security Scanning	Verification	Available by Request	www.vthreat.com
WhiteHat Sentinel	SAST, DAST	Implementation / Verification	30 Day Free Trial	whitehatsec.com
Wireshark	Penetration Testing and Packet-level Monitoring	Verification	Open Source	www.wireshark.org
Ziften	Endpoint Security	Response	30 Day Free Trial	www.ziften.com

[1] Già adottato in azienda;

APPENDICE 2. VALUTAZIONE STRUMENTI

a. CHECKMARX

PRODOTTO	CATEGORIA	FASE SSE	SITO WEB	
CHECKMARX	**SAST**	Implementation	https://www.checkmarx.com/	
DESCRIZIONE				
È un tool a pagamento, per l'analisi statica del codice, posizionato da Gartner nel quadrante Challengers nell'ambito dell'Application Security Testing (AST). Supporta numerosi linguaggi (vedi oltre). Può essere integrato a vari livelli nell'ambito della fase di Implementation: IDE, build server, bug tracking tools. Orientato alla facilità d'uso da parte del team di sviluppo.inserire una descrizione del prodotto				
Tainted analysis, Pattern matching , "scan rules" (customizable)inserire il meccanismo d'azione				
ANALISI DEL VALUTATORE				**SCORE**
Livello di integrazione con i seguenti prodotti				
a. IDEs	Esistono plugin per i seguenti IDE: Eclipse, Visual Studio e IntelliJ. I plugin consentono la scansione del codice, l'analisi e la navigazione dei risultati in modo integrato con l'IDE.			7
b. source repository,	TFS, SVN, GIT, Perforce.			7
c. build server,	Jenkins, Bamboo, TeamCity, TFS, Anthill Pro, Maven.			7
d. bug tracking tools	Jira.			4
I linguaggi di programmazione supportati	Java C# JavaScript and commonly used frameworks Node.JS and commonly used frameworks VB.NET ASP.NET VB6 PHP			7

	C/C++ Apex and VisualForce Ruby VBScript Perl HTML5 Python Groovy Scala	
I framework applicativi supportati (es. Spring, Hibernate, …)	**[*] Requires minor adjustments** **Platform/Enviroment: Java** Struts, Spring MVC, iBatis*, GWT, Hibernate, OWASP ESAPI, JSTL FMT Taglib, ATG DSP Taglib, Java Server Faces (JSF), JavaScript **Platform/Enviroment: .NET** Enterprise Libraries, Telerik, ComponentArt, Infragistics, FarPoint, iBatis*, Hibernate.Net [*], Entity framework up to 4.3.1 **Platform/Enviroment: PHP** Zend, Kohana, CakePHP, Symfony, Smarty, OWASP ESAPI **Platform/Enviroment: C/C++** MISRA **Platform/Enviroment: Ruby** Ruby on Rails **Platform/Enviroment: JavaScript** JQuery, Node.js, Ajax, Knockout, AngularJS, ExpressJS, Jade, Backbone, Handlebars, Hapi.JS **Platform/Enviroment: iOS** iOS mobile applications **Platform/Enviroment: Python** Django **Platform/Enviroment: Groovy** Grails	7
Le tipologie di applicazione supportate (Web, Mobile, Client-Server…)	Web application, Mobile (Android, iOS, Windows mobile), Client-Server	7

Le vulnerabilità riconosciute (Sql injection, Cross-site scripting, Code injection…)	SQL Injection, Cross-site scripting, Code injection, Buffer Overflow, Parameter tampering, Cross-site request forgery, HTTP splitting, Log forgery, DoS, Session Fixation, Session poisoning, Unhandled exceptions, Unreleased resources, unvalidated input, URL redirection attack, Dangerous Files Upload, Hardcoded password	7
Gli Standard supportati (OWASP Top 10, SANS 25, …)	OSWAP Top 10 2013, OSWAP Mobile Top 10, SANS 25, HIPAA, FISMA, BSIMM, PCI DSS, Mitre CWE	7
L'integrazione di "Custom rules"	E' possibile definire Custom Rules (per esempio per dichiarare che una funzione esegue sanitizzazione)	4
L'incidenza dei "Falsi positivi"	In primo luogo, è possibile "spegnere" falsi positivi estendendo la lista dei "sanitizer" fornita out of the box da checkmarx (con pochi colpi di click). In secondo luogo, è possibile "spegnere" falsi positivi dichiarandoli come "Not Exploitable" . In terzo luogo, è stato possibile apprezzare un approccio messo in atto da Checkmarx atto a limitare il numero di segnalazioni. La prova eseguita ha evidenziato che: in presenza di codice evidentemente prono a una SQL INJECTION, ma in assenza di un vettore di attacco, la segnalazione della vulnerabilità viene soppressa. Viceversa la segnalazione viene prodotta se viene individuato anche un vettore di attacco. Il side effect è che in una scansione parziale che considera il codice vulnerabile ma esclude in tutto o in parte il vettore d'attacco, non vengono prodotte segnalazioni.	4
La capacità di analisi "raw source code" vs "need to compile"	Lo strumento è in grado di funzionare in modalità "raw source code". E' quindi possibile sottoporre anche porzioni di codice "out-of-context". Tuttavia, in questo caso potrebbero non essere segnalate certe vulnerablità che invece si manifestano in una scansione "in-context". E' una scelta by design per limitare falsi positivi.	Raw Source
La capacità di analizzare le dipendenze da librerie esterne al fine di controllare se sono presenti vulnerabilità note	Esiste un add-on di CheckMarx (acquistabile a parte) che analizza le dipendenze da librerie esterne al fine di controllare se sono presenti vulnerabilità note, interrogando una base dati esterna.	1
La capacità di correlare lo scan statico con l'esito di uno scan dinamico (correlazione White Box con Black Box)	NO	1
LE PERFORMANCE		
a. Full scan vs Incremental scan	Sono supportati sia Full sia Incremental scanning	7

b. Client-side scan vs Server-side scan	Server-side scanning: i sorgenti in tutti le configurazioni (anche in quella di plug-in integrato con l'ambiente di sviluppo) vengono compressi e inviati al server dove avviene effettivamente lo scan. Ne consegue il beneficio dell'allegerimento dell'occupazione della potenza di calcolo dei Client.	7
Eventuali funzionalità di prioritizzazione delle remediation	Le vulnerabilità individuate vengono ordinate secondo 4 livelli: High, Medium, Low, Information che indirizzano la priorità della remediation.	7
La facilità d'uso	Lo strumento è fortemente orientato alla facilità d'uso da parte del team di sviluppo (partendo dalla cosapevolezza -preso dalla homepage- che "Getting developers to use application security testing is one of the biggest challenges facing security professionals today"). Alla prova dei fatti, lo strumento è davvero molto fruibie	7
I costi di licenza	Esistono varie forme di licenza: per numero progetti e per numero di sviluppatori. Ad esempio, la licenza a 6 mesi, per 5 sviluppatori e senza limiti sul numero di progetti (ma con un numero di linee di codice (LOC) fino a 1.000.0000) costa per 11.062,50 €	ALTO
Il supporto alla reportistica	E' supportata una reportistica di tipo custom (non sono espressamente disponibili report pre-definiti, ad esempio specificamente orientati a CWE SANS Top 25, OWASP Top 10, PCI Data Security Standard, ecc). I formati supportati sono: PDF, CSV, RTF, XML.	4
La classificazione degli errori riportati	Sono riferiti agli standard supportati (es. "PCI DSS (3.1) - 6.5.1 - Injection flaws - particularly SQL injection", OWASP Top 10 2013 - A1-Injection)	7
CONSIDERAZIONI FINALI DEL VALUTATORE		
CONSIDERAZIONI GENERALI		

Considerazioni generali:
1. Installazione agevole
2. Fruizione da Browser agevole (apprezzabile il riconoscimento automatico del linguaggio: è sufficiente eseguire lo zip dei sorgenti)
3. Fruizione da plug-in integrato con IDE agevole e intuitiva (tasto destro su un punto del progetto e si può eseguire lo scan)
4. Supporto alla remediation in tutti gli ambienti: CxAudit, plug-in, browser
5. Inserimento di regole custom agevole (esaminato il caso "sanitizer")
6. Reporting custom
7. Sinottico minimale
8. Scan full e incrementale
9. No need to compile (ma anche nessun check sulle librerie linkate, a meno di integrare un componente licenziato a parte)
10. Integrazione con Jenkins, come step aggiuntivo della fase di build (Continuous Integration), agevole attraverso plug-in

Punti di forza:
1. Vettore di attacco
2. Funzionlità "Full Graph" che raccorda più vettori di attacco mostrando eventuali punti di intersezione (candidati ideali per il fix)

APPROCCIO PER LA VALUTAZIONE

L'approccio seguito è stato quello di costruire un programma di benchmark che presentasse XSS e SQL Injection: XSS e SQL Injection sono le vulnerabilità rispettivamente al primo e al secondo posto nella OWASP TOP Ten 2010. Condizione necessaria per candidare un tool alla Leonardo Suite è la capacità del tool di identificare (anche solo parzialmente) le vulnerabilità inserite. In caso contrario l'analisi del tool si conclude con esito negativo.
La scelta di costruire un programma di benchmark (in vece di utilizzare benchmark preconfezionati, come ad esempio https://github.com/OWASP/Benchmark) nasce dalla volontà di evitare overfitting dei tool sottoposti ad analisi.
L'utilizzo dello stesso programma di benchmark consente di avere risultati comparabili tra i vari tool sottoposti ad analisi.
Nel caso di Checkmarx, le SQL Injection sono state individuate ad eccezione di una (1 falso negativo). Segnalato il problema all'assistenza, si è stabilito che la mancata individuazione della SQL Injection era dovuta al fatto che essa sfruttava una feature introdotta in Java 7 ("With Java 7, you can create one or more "resources" in the try statement. A "resources" is something that implements the java.lang.AutoCloseable interface. This resource would be automatically closed and the end of the try block." Vedi https://dzone.com/articles/java-7-new-feature-%E2%80%93), non ancora integrata nel Virtual Compiler di Checkmarx. Putroppo nella successiva versione di Checkmarx (8.1.0) il problema non risulta ancora risolto.
Nel caso di Checkmarx, l'XSS (di tipo reflected) è stato individuato insieme a 2 problemi di "Sensitive Cookie in HTTPS Session Without Secure Attribute". Entrambi i problemi, tuttavia, vengono classificati come "Low" (benché l'XSS sia sfruttabile).
Estremamente interessante è l'esito della scansione con Checkmarx a valle della risoluzione dei problemi XSS e Cookie attraverso l'impiego delle ESAPI (OWASP): le segnalazioni correttamente scompaiono, segno che Checkmarx riconosce nativamente la sanitizzazione del codice attraverso l'adozione del framework ESAPI.

INTERPRETAZIONE DEI RISULTATI

Valutazione molto positiva, eccetto il falso negativo, al cui riguardo si svolgono ancora queste considerazioni (avallate da ulteriori prove). Se si elimina l'uso del nuovo costrutto sintattico introdotta in Java 7, Checkmarx individua la SQL INJECTION. Quindi sembra verosimile che la mancata interpretazione del nuovo costrutto sintattico, impedisca in sostanza a Checkmarx di individuare l'attack vector, senza il quale –by design- un vulnerabilità non viene segnalata.	
TEAM DI VALUTAZIONE	Leonardo Software Security team

b. CodeDX

PRODOTTO	CATEGORIA	FASE SSE	SITO WEB
CodeDx	**SAST/DAST**	Implementation/Verification	https://codedx.com/
DESCRIZIONE			
CodeDx e' un Tool commerciale che serve ad effettuare la verifica di eventuali vulnerabilità di programmi e software presi in considerazione. CodeDx riunisce una serie di strumenti di analisi del codice (sia gratuiti che commerciali) che consentono a loro volta di individuare e correggere agevolmente eventuali bugs nel codice da analizzare.inserire una descrizione del prodotto			
Source analysis, Pattern matching , "scan rules" (customizable)inserire il meccanismo d'azione			
ANALISI DEL VALUTATORE			**SCORE**
Livello di integrazione con i seguenti prodotti			
a. IDEs	CodeDx si integra con i seguenti ide: Eclipse, Visual Studio		8
b. source repository,	CodeDx si integra i seguenti repository: Git (direttamente); Subversion, Mercurial, o Team Foundation Version Control (TFVC) (tramite zip del "source outside" di CodeDx e successivo upload verso CodeDx)		8
c. build server,	CodeDx si integra con i seguenti build server: Jenkins, Maven		7
d. bug tracking tools	CodeDx supporta il tool Bug Issue Tracker JIRA		
Le tipologie di applicazione supportate (Web, Mobile, Client-Server…)	Client Server, Web, Mobile (Android Studio)		7
I linguaggi di programmazione supportati	C/C++, Java, Javascript, JSP, .NET(C#, Visual Basic), Python, Ruby		8
I framework applicativi supportati (es. Spring,	Il tool supporta i piu' popolari frameworks tra i quali Spring-MVC, JQuery e molti altri.		7

Hibernate, …)		
Gli Standard supportati (OWASP Top 10, SANS 25, …)	CodeDx supporta sia lo standard CWE che altri standard come OWASP Top 10, SANS Top 25 e PCI-DSS.	8
Le vulnerabilità riconosciute (Sql injection, Cross-site scripting, Code injection…)	Tutte le vulnerabilità descritte negli standard di cui al punto 5	7
L'integrazione di "Custom rules"	E' possibile all'interno di CodeDx creare delle regole personalizzate	7
Possibilita' di inibire la segnalazione di particolari vulnerabilita'	E' possibile all'interno del Tool gestire la segnalazione di una particolare vulnerabilità	7
L'incidenza dei "Falsi positivi"	Dai riscontri, l'incidenza di falsi positivi è accettabile	8
La capacità di analisi "raw source code" vs "need to compile"	CodeDx (a seconda dei tool embedded che vengono invocati) permette di analizzare il codice in entrambe le modalita' (sia source-code che raw-code).	entrambe
La capacità di analizzare le dipendenze da librerie esterne al fine di controllare se sono presenti vulnerabilità note	CodeDx permette (tramite l'utilizzo di tool embedded come Dependency Check) di analizzare le dipendenze da librerie esterne al fine di controllare se sono presenti vulnerabilità note	8
La capacità di correlare lo scan statico con l'esito di uno scan dinamico (correlazione White Box con Black Box)	Il prodotto è in grado di effettuare correlazioni tra entrambe le tipologie di scan del codice	7
Le performance		
a. Full scan vs Incremental scan	Il prodotto è in grado di effettuare entrambe le tipologie di scan del codice	8
b. Client-side scan vs Server-side scan	Il prodotto consente di effettuare scan sia lato server che client	7
Supporto alla Remediation	Il tool guida nella localizzazione del problema ed offre supporto informativo utile per sanarlo.	6
Funzionalità di prioritizzazione delle Remediation	Il tool permete di evidenziare i bugs in base a delle priorita' di intervento	7

La facilità d'uso	Il prodotto è piuttosto facile da installare e assolutamente intuitivo da utilizzare	8
I costi di licenza	CodeDx e' un prodotto commerciale a pagamento dai costi non eccessivi rispetto a strumenti similari commerciali. L'argomento andrebbe comunque analizzato in una logica commerciale complessiva aziendale.	MEDIO
Il supporto alla reportistica	Il tool consente di produrre un'ottima reportistica in vari tipi di formato (Pdf, xml, Excel)	8
La classificazione degli errori riportati	Il Tool CodeDx permette di classificare gli errori secondo quattro tipologie di gravita': High, Medium, Low e Info	7
CONSIDERAZIONI FINALI DEL VALUTATORE		
Dopo aver preso in considerazione tutti i vari punti descritti nella scheda si ritiene che il Tool CodeDx sia un ottimo tool di facile uso e integrabile con molti altri tool sia gratuiti che a pagamento. Il tool permette agli sviluppatori di software, tester e analisti della sicurezza diindividuare e gestire con modalita' abbastanza semplici le vulnerabilità nel software. Il tool permette di integrare una quantita' molto ampia di plugin e altri tool che danno una copertura quasi completa di tutti i linguaggi e gli ide presenti sul mercato. La Reportistica e' molto dettagliata e disponibile in vari formati. Dalle evidenze riscontrate, è emerso che il prodotto sia adeguatamente affidabile. Si ritiene pertanto che CodeDx sia utilizzabile proficuamente per gli scopi aziendali.		
TEAM DI VALUTAZIONE	Leonardo Software Security team	

c. SAST

PRODOTTO	CATEGORIA	FASE SSE	SITO WEB
SonarQube	**SAST**	Implementation	http://www.sonarqube.org
DESCRIZIONE			
SonarQube è un prodotto avanzato per l'analisi statica del codice sorgente, finalizzato alla ricerca di errori di programmazione e di costrutti che costituiscono delle bad practise. I Bug rilevati sono tracciati ed evidenziati in un'interfaccia web intuitiva, in modo da poter seguire e gestire il processo di remediation. Dato che si tratta di un prodotto open source, il miglioramento dei pattern per il riconoscimento dei problemi è demandato all'ampia community in rete.inserire una descrizione del prodotto			
SonarQube effettua le sue analisi attraverso appositi plugin che applicano al codice sorgente dei pattern match pre-definiti.inserire il meccanismo d'azione			
ANALISI DEL VALUTATORE			**SCORE**
Livello di integrazione con i seguenti prodotti			

a. IDEs	Si integra tramite il plugin SonarLint con Eclipse, Visual Studio, IntelliJ. SonarLint è uno strumento che analizza il codice dal punto di vista della qualità, ma è possile utilizzarlo in collegamento con SonarQube, per sfruttare le regole di sicurezza di quest'ultimo.	8
b. source repository,	Si integra, tramite plugin, a Git, Svn, CVS, TFVC, Jazz RTC, ClearCase	8
c. build server,		
d. bug tracking tools	SonarQube comprende la gestione completa dei bug riscontrati (tracciamento incluso)	8
Le tipologie di applicazione supportate (Web, Mobile, Client-Server…)	Web, Mobile Android	8
I linguaggi di programmazione supportati	ABAP, C/C++, C#, COBOL, Flex, Groovy, HTML, Java, JavaScript, JSP, JSF, Objective-C, PHP, PL/I, PL/SQL, Python, RPG, Swift, VB.NET, Visual Basic 6, XML	10
I framework applicativi supportati (es. Spring, Hibernate, …)		
Gli Standard supportati (OWASP Top 10, SANS 25, …)	SonarQube comprende fra le sue rules CWE, SANS TOP 25 e OWASP TOP 10	10
Le vulnerabilità riconosciute (Sql injection, Cross-site scripting, Code injection…)	Le SQL Injection non sono state individuate (2 falsi negativi). L'XSS (di tipo reflected) non è stato individuato (1 falso negativo) così come non sono stati individuati i 2 problemi relativi a "Cookie without the secure flag" (2 falsi negativi).	10
L'integrazione di "Custom rules"	SonarQube offre la possibilità di creare delle regole personalizzate, attraverso dei custom templates	10
Possibilita' di inibire la segnalazione di particolari vulnerabilita'	Il tool consente di "sopprimere" la segnalazione di una particolare vulnerabilità in maniera agevole	9
L'incidenza dei "Falsi positivi"	Coloro che scoprono un falso positivo possono segnalarlo alla Community. Per questo motivo l'incidenza dei falsi positivi è tenuta bassa.	7
La capacità di analisi "raw source code" vs "need to compile"	SonarQube fa le sue valutazioni su bytecode, per cui presuppone un rebuild del codice modificato	Need to Compile
La capacità di analizzare le dipendenze da librerie esterne al fine di controllare se sono presenti vulnerabilità note	Non dispone di questa funzionalità	1

La capacità di correlare lo scan statico con l'esito di uno scan dinamico (correlazione White Box con Black Box)		
LE PERFORMANCE		
a. Full scan vs Incremental scan	Il prodotto è in grado di effettuare entrambe le tipologie di scan del codice	8
b. Client-side scan vs Server-side scan	Il prodotto consente di effettuare scan sia lato server, sia lato client	8
Supporto alla Remediation	SonarQube offre la possibilità di organizzare e seguire la fase di correzione dei bugs	9
Funzionalità di prioritizzazione delle Remediation	SonarQube classifica i bugs in base all'urgenza con la quale debbono essere corretti	8
La facilità d'uso	Il prodotto è piuttosto facile da installare e assolutamente intuitivo da utilizzare	7
I costi di licenza	SonarQube è Open Source, con licenza GNU Lesser GPL License, Version 3, quindi non comporta alcun costo di licenza	free
Il supporto alla reportistica	Si realizza tramite plugin open source o commerciali. La dashboard e l'interfaccia web costituiscono, di per sé, una valida reportistica	7
CONSIDERAZIONI FINALI DEL VALUTATORE		
Riguardo a XSS, SonarQube non implementa nessuna regola per l'individuazione di XSS, quindi il falso negativo è atteso a priori. Riguardo a SQL Injection: SonarQube implementa la regola "S2077 SQL binding mechanisms should be used" per l'individuazione di SQL Injection. Eseguendo ulteriori prove, atte ad introdurre nel codice altri pattern di SQL Injection, si è osservato che la regola funziona segnalando le ulteriori vulnerabilità. Riguardo alla problematica "Cookie without the secure flag" SonarQube implementa la regola "S2092 Cookies should be secure" per l'individuazione di tali cookie. Tale regola tuttavia non ha individuato le 2 vulnerabilità presenti nel benchmark. Nonostante queste attuali carenze, SonarQube rimane una scelta da considerare, nell'analisi statica, sia pure affiancandolo ad altri strumenti che colmano le lacune sopra citate. È infatti uno strumento completo, dall'interfaccia moderna e user-friendly, gratuito e aperto alla collaborazione di una grande e attiva community. SonarQube, tramite plugin, si integra con i più importanti ambienti di sviluppo per consentire quello che viene definito "continuous inspection" del codice. Un'altra caratteristica che rende SonarQube molto interessante è la sua capacità, attraverso altri plugin, di poter analizzare il codice scritto in un'ampia gamma di linguaggi, compresi quelli di Microsoft. Altro punto a favore di SonarQube è la gestione dei bug rinvenuti, classificati per priorità e tracciabili; è possibile infatti pianificare e seguire la fase di remediation, con una valutazione del tempo richiesto e l'assegnazione dei task ai vari sviluppatori. user-friendly e soprattutto gratuito e aperto alla collaborazione di una grande e attiva community. Il tool, tramite plugin, si integra con i più importanti ambienti di sviluppo per consentire quello che viene definito un "continuous inspection" del codice. Un'altra caratteristica he rende SonarQube molto interessante è la sua capacità, attraverso, appositi		

plugin, di poter analizzare il codice scritto in un'ampia gamma di linguaggi, compresi quelli di Microsoft. Altro punto a favore di SonarQube è la gestione dei bug rinvenuti, classificati per priorità; è possibile infatti pianificare e seguire la fase di remediation, con una valutazione del tempo richiesto e l'assegnazione dei task ai vari sviluppatori. Per contro, il tool non fa l'analisi delle vulnerabilità delle librerie che vengono utilizzate nel codice. Per questo scopo si possono utilizzare tool appositi, quali l'open source.friendly. Il punto di forza del tool è rappresenta	
TEAM DI VALUTAZIONE	Leonardo Software Security team

11 BIBLIOGRAFIA

[1] G. McGraw, «Software Security: Building Security In, Addison Wesley,» 2006.

[2] S. H. Flechais, « Bringing Security Home: A Process for Developing Secure and Usable Systems," In Proc. of the New Security Paradigms Workshop (NSPW'07),» Switzerland, 2003, pp. 49-57.

[3] C. M. a. M. S. I. Flechais, in *"Integrating Security and Usability into the Requirements and Design Process," International Journal of Electronic Security and Digital Forensics, Inderscience Publishers, vol. 1, no. 1,* , Geneva, Switzerland, 2007, pp. 12-26.

[4] A. A. a. M. Pourzandi, in *"Secure Software Development by Example," IEEE Security and Privacy vol. 3, no. 4*, IEEE CS Press, 2005, pp. 10-17.

[5] S. O. a. O. A. A.S. Sodiya, in *"Towards Building Secure Software Systems," Issues in Informing Science and Information Technology vol. 3.*, California, USA, Informing Science Institute, 2006, pp. 635-646.

[6] J. Juerjens, « Secure Systems Development with UML, Springer,,» 2005.

[7] L. F. a. R. Solms, in *"SecSDM: A Model for Integrating Security into the Software Development Life Cycle," In IFIP International Federation for Information Processing, Volume 237, Proc. of the 5th World Conference on Information Security Education,* .

[8] T. W. J. S. a. M. B. D.P. Gilliam, « "Software Security Checklist for the Software Life Cycle," In Proc. of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03), Linz, Au,» Linz, Austria, 2003, pp. 243-248.

[9] J. P. E. H. a. M. B. D. Gilliam, « "Addressing Software Security Risk and Mitigations in the Life Cycle," In Proc. of the 28th Annual NASA Goddard Software Engineering Workshop (SEW'03), Greenbelt,» Maryland, USA, 2003, pp. 2001-206.

[10] «Database of Vulnerabilities, Exploits, and Signatures, http://seclab.cs.ucdavis.edu/projects/DOVES/,» 2009.

[11] T. W. J. S. a. M. B. D.P. Gilliam, in *"Software Security Checklist for the Software Life Cycle," In Proc. of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03),*, Linz, Austria.

[12] M. Hadawi, in *"Vulnerability Prevention in Software Development Process," In Proc. of the 10th International Conference on Computer & Information Technology (ICCIT'07)*, Dhaka, Banglades, 2007.

[13] L. L. a. H. G. M. Essafi, in *"S2D-ProM: A Strategy Oriented Process Model for Secure Software Development," In Proc. of the 2nd International Conference on Software Engineering Advances (ICSEA'07), Cap Esterel, French Riviera*, France, 2007, p. 24.

[14] N. Davis, in *"Secure Software Development Life Cycle Processes: A Technology Scouting Report", technical note CMU/SEI-2005-TN-024, Software Engineering Institute, Carnegie Mellon University, Pittsburgh*, Pennsalyania, USA, 2005.

[15] T. W. J. S. a. M. B. D.P. Gilliam, « "Software Security Checklist for the Software Life Cycle," In Proc. of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03), Linz, Au,» Austria, 2003, pp. 243-248.