Sviluppo sicuro delle applicazioni: i requisiti

Date : 9 ottobre 2017

Questo articolo prosegue la serie dedicata a temi su cui ci sono i maggiori dubbi e perplessità sulle norme della serie ISO/IEC 27000.

Lo sviluppo sicuro è un argomento molto importante per la sicurezza.

Spesso, purtroppo, gli sviluppatori dimostrano di non avere chiaro l'oggetto della domanda.

Molte volte usano le raccomandazioni dell'OWASP Top ten vulnerabilities, anche se si tratta di un documento volutamente non esaustivo, altre volte richiedono un *vulnerability assessment* a società specializzate; ma questa attività si svolge alla fine del progetto e non all'inizio.

Si ricorda che un principio cardine della sicurezza delle informazioni prevede di considerarla sin dall'inizio di ogni progetto.

Queste pratiche, quindi, non rappresentano completamente lo sviluppo sicuro. Uno sviluppo sicuro include l'identificazione dei requisiti, la definizione e l'attuazione di processi, l'esecuzione di verifiche.

In questo articolo si tratterà dei requisiti. Essi si possono suddividere nei seguenti:

- requisiti di sicurezza funzionali;
- requisiti di sicurezza tecnici;
- requisiti di codifica.

Requisiti di sicurezza funzionali

Questi requisiti riguardano le funzioni solitamente visibili dagli utilizzatori di un programma software. Essi vanno documentati in documenti di *specifiche funzionali* o, quando si seguono metodi di tipo Agile, nelle *user story*. Non è necessario evidenziarli, anche se alcuni prevedono un capitolo dedicato alla sicurezza o impostano delle etichette (o *tag*) ai requisiti o alle storie.

Questi requisiti riguardano:

- le modalità di autenticazione dell'utente, che potrebbero basarsi su password, *token* o caratteristiche biometriche o loro combinazioni; i requisiti funzionali devono anche specificare come assicurare la sicurezza del meccanismo di autenticazione (caratteristiche delle password, mascheramento della password nella pagina di login, controllo delle credenziali solo sul server, cifratura della comunicazione); si può anche prevedere l'uso di un meccanismo di autenticazione esterno (per esempio Active Directory o un altro modulo dell'applicazione oggetto dell'analisi);
- profili, ruoli e autorizzazioni previsti dal software, incluse eventuali separazioni dei compiti;
- limitazione delle funzionalità e delle query agli utenti sulla base del loro profilo;
- utenze di amministrazione e loro autorizzazioni;
- modalità di interfacciamento con altri software; dovrebbe avvenire su canali cifrati impostati con lo scambio di certificati in modo da garantire l'identificazione delle diverse istanze;
- modalità di trasmissione su rete Internet, che dovrebbe sempre basarsi su canali cifrati;
- possibilità degli utenti ad accedere direttamente ai dati senza la mediazione dell'applicazione; questo dovrebbe essere impedito;
- scelte relative all'integrità referenziale nei db;
- scelte relative alle transazioni concorrenti nei db (per esempio, se una transazione è concorrente ad una già avviata, l'utente ne deve essere avvisato in modo che possa decidere se sottoporla nuovamente);
- modalità di controllo dell'integrità delle transazioni (per esempio attraverso caratteri di controllo, rilevazione duplicati non previsti, ACK all'utente);
- conferma delle azioni degli utenti dove opportuno (per esempio con messaggi di conferma quando si vogliono cancellare dei dati);
- *logging* delle attività, protezione delle registrazioni, tempi di conservazione;
- capacità previste (per esempio, numero massimo di utenti previsti) e possibilità di aumentarle (*scaling*);
- *misuse case*, ossia potenziali azioni degli utenti che potrebbero danneggiare l'applicazione o presentare problemi di sicurezza;
- la normativa vigente applicabile, altri regolamenti e obblighi contrattuali e loro requisiti applicabili (p.e. normativa sul trattamento dei dati personali, normativa bancaria, requisiti dei clienti);
- opzioni di configurazione dei meccanismi di sicurezza disponibili agli amministratori e agli utenti del sistema.

Requisiti di sicurezza tecnici

Questi requisiti sono divisi in due famiglie: quelli di raffinamento dei requisiti funzionali e quelli relativi all'architettura. Essi vanno documentati in documenti di *specifiche tecniche* o, quando si seguono metodi di tipo Agile, nelle *definition of done* o in *task* collegati alle *user story*.

Questi requisiti possono anche essere indicati come *Principi di ingegnerizzazione sicura*, anche se non tutti li intendono in questo modo.

Tra i requisiti tecnici di raffinamento di quelli funzionali, si possono citare:

- modalità di caching delle credenziali;
- controllo e sanificazione dei dati in input da utenti e da altri processi;
- controllo delle *query* degli utenti;
- scelta dei protocolli crittografici e delle loro impostazioni (per esempio per la lunghezza delle chiavi).

I requisiti architetturali sono quelli che ciascun sviluppatore si impegna a rispettare quando lavora nel progetto. La loro documentazione permette di assicurare la condivisione di tali principi, il loro aggiornamento quando necessario, il passaggio di consegne alle nuove persone che si inseriscono nel progetto e ad altri gruppi di sviluppo (per esempio quando il sistema è stato sviluppato e viene preso in carico da chi gestisce l'esercizio e la manutenzione).

Tra i requisiti architetturali vanno considerati i seguenti (alcuni possono essere approfonditi nelle pubblicazioni relative alle *OWASP Top ten vulnerabilities*):

- suddivisione in più livelli o *tier* (per esempio, in db, applicazione, presentazione);
- coesione dei moduli (ciascun modulo deve realizzare un'unica funzione);
- separazione del controllo accessi e delle autorizzazioni dagli altri moduli;
- separazione dell'interfaccia di amministrazione dagli altri moduli;
- disaccoppiamento dei moduli (*loose coupling*), che include il divieto di codificare dati, configurazioni e password negli oggetti software;
- verifica delle autorizzazioni per ogni operazione su un oggetto o sui dati (*mediazione completa*);
- utenze e relative autorizzazioni per l'interfacciamento con gli altri sistemi;
- gestione sicura delle sessioni e dei loro parametri;
- scelta di software e delle librerie di origine esterna (per cui deve essere assicurata l'affidabilità delle fonti e la disponibilità di manutenzione);
- gestione delle configurazioni in modo da permettere il *patching* e *fixing* tempestivo;
- gestione delle operazioni quando una dà errore; in particolare dovrebbero essere bloccate tutte le successive e l'utente dovrebbe essere avvertito (*fail safe*);
- intercettazione di tutti i possibili errori, in modo che siano gestiti.

Requisiti di codifica

Questi requisiti vanno documentati o in documenti specifici (anche in *wiki* ad uso interno o in *Regole di sviluppo sicuro*) o, quando si seguono metodi di tipo Agile, nelle *definition of done*.

Le regole di codifica da seguire dipendono dal linguaggio utilizzato, ma generalmente comprendono:

- uno standard per denominare classi, metodi, variabili e costanti in modo che ne sia comprensibile la finalità e la natura;
- gestione dei commenti (quantità, collocazione, attenzione affinché non riportino informazioni utili per comprendere l'architettura del sistema o altre informazioni critiche);
- gestione delle variabili per prevenire gli attacchi più diffusi;

- le funzioni da evitare (la più celebre è il GOTO, peraltro oggi spesso non più accettato).

A cura di: **Cesare Gallotti**