OsservatoriO
AttacchInformatici

L'errore nella programmazione genera vulnerabilità

Alla base dei più pericolosi attacchi informatici c'è spesso una svista nel codice software. Ecco, attacco per attacco, a cosa bisogna porre attenzione.

Marco Bozzetti

Sia nel Rapporto OAI 2009 sia negli articoli precedenti di questa rubrica si è evidenziato come uno dei maggiori pericoli per la sicurezza dei sistemi informativi sia il codice software con vulnerabilità intenzionali o non. Vulnerabilità che esistono nei programmi in vendita o in open source, e soprattutto in quelli sviluppati ad hoc, quando i programmatori non pongono l'adeguata attenzione su come programmare in maniera "sicura". CWE e SANS, due importanti istituzioni statunitensi per la sicurezza ICT, hanno stilato l'elenco dei principali errori nella programmazione che causano vulnerabilità, riportato nella seguente Tabella. Volutamente si è usato in molto casi il termine inglese, noto e usato anche in italiano, a fianco spiegandolo sinteticamente. Alcuni di questi errori, e cosa fare per porvi rimedio, verranno discussi nei prossimi articoli della Rubrica OAI (vedi tabella nella pagina seguente). Nella tabella sono elencate le prime dieci vulnerabilità per errori nella programmazione di codice secondo il CWE. Ma come far fronte a queste e a tutte le altre possibili vulnerabilità del software?
Due sono i livelli di intervento:
a) il controllo della sicurezza intrinseca del codice nel suo intero ciclo di vita da parte degli sviluppatori e dei loro supervisori; vari metodologie sono presenti, quali il CMMI, SSE-CMM, CLASP, Common Criteria, ecc., alcune delle quali integrati nei moderni ambienti di sviluppo;
b) il controllo della sicurezza intrinseca del codice da parte di chi lo mette in produzione nel proprio sistema informativo.

Le grandi case produttrici di software utilizzano tutte strumenti di cui al punto a), ma nonostante questo le vulnerabilità del codice continuano ad aumentare, e solo parzialmente sono sanate da opportune patch. Lato utente nella stragrande maggioranza dei casi non si fanno controlli sulla sicurezza del codice né per pacchetti acquisiti (soprattutto se di marche famose) né soprattutto per codici sviluppati ad hoc. In questo ultimo caso vengono fatti collaudi e test prevalentemente di tipo funzionale a livello interfaccia utente.
Per quanto riguarda gli sviluppatori, le prime generali regole da seguire per una programmazione sicura includono:
- seguire gli standard ed i framework per la sicurezza più consolidati, e non inventarsene din propri;
- introdurre logiche e strumenti di sicurezza nell'intero ciclo di vita del software;
- effettuare e mantenere il controllo di tutti i dati in input e in output;
- ritenere ogni componente ed ogni utente esterno inaffidabile;
- la sicurezza assoluta non esiste, e deve essere valutato, contesto per contesto, il rischio residuo che può/deve essere accettabile.

Per quanto riguarda gli acquirenti-clienti, il software acquisito o fatto sviluppare ad hoc dovrebbe essere provato in opportuni ambiti di staging-test anche per gli aspetti di sicurezza. La crescente disponibilità ed economicità di soluzioni as a service (SaaS, PaaS, IaaS, ecc.) può facilitare l'effettuazione di tali controlli anche per organizzazioni e aziende di medie piccole dimensioni.
Le conseguenze dell'ampia sottostima, sia a livello fornitori sia a livello acquirenti-utenti, dei problemi derivati dalla vulnerabilità dei programmi, in particolare di quelli applicativi, costituisce una delle principali cause della insicurezza globale dei sistemi informativi.

Tabella

GRADUATORIA	ERRORE	SPIEGAZIONE	PRINCIPALI IMPATTI
1	**Cross-site Scripting (XSS)**	Avviene lato client-browser con l'inserimento non controllato e neutralizzato di codice (es. script, ActiveX, HTML, Flash) nella generazione di una pagina web.	Viene normalmente inserito un XSS nella pagina di un forum o di un guestbook, e il codice maligno acquisito può apportare vari possibili danni al PC client: tra questi cattura di dati, manipolare cookie, manipolare il PC
2	**SQL Injection**	Sfrutta l'inefficienza dei controlli sui dati ricevuti in input ed inserisce codice maligno all'interno di un comando SQL	Tutti quelli possibili da un codice maligno; tipicamente tali attacchi sono usati per autenticare l'attaccante con ampi privilegi in aree protette del sito di visualizzare e/o alterare dati.
3	**Buffer Overflow**	Vengono trattati file buffer senza controllare le dimensioni dei dati in input	Tipico problema per linguaggi che non hanno controlli sulle dimensioni delle matrici, quali C e C++. Si deve conoscere bene sia il programma che l'elaboratore per provocare un buffer overflow che rappresenti un codice maligno, tipicamente per prendere il controllo del sistema
4	**Cross-Site Request Forgery (CSRF)**	Viene incluso un link o uno script in una pagina che accede un sito per il quale l'utente è ritenuto affidabile e autenticato, e coswì facendo si inserisce un codice maligno in un server web. Logicamente simile al XSS ma operante sul server, non sul client.	Tutti i possibili impatti dell'inserimento di codice maligno su un server. A rischio tutte le applicazioni web. L'attaccante si maschera da utente fidato e riconosciuto come tale. Può essere combinato con XSS ed avere effetti devastanti.
5	**Improprio controllo degli accessi e delle autorizzazioni**	Mancanza di o non corretta gestione degli strumenti di controllo degli accessi e di profilazione dei diritti degli utenti sui sistemi e sugli applicativi. Il problema è prevalentemente di tipo organizzativo, e dipende anche dai sistemi di identificazione, autenticazione e controllo accessi usati.	Un utente non autorizzato può accedere a risorse ICT; la criticità maggiore è l'accesso come amministratore: con tali diritti nella maggior parte dei casi l'attaccante può fare quello che vuole.
6	**Reliance on Untrusted Inputs in a Security Decision**	L'attacco deriva da una impropria fiducia su dati di input non sicuri-non affidabili, causata da logiche e/o meccanismi di controllo deboli e poco efficaci.	L'impatto è quello di un accesso non autorizzato.
7	**Path Traversal**	Impropria limitazione di un pathname in una directory, tipicamente quando si scambiano dati via file.	Nella costruzione di un file all'interno del proprio sistema e di una specifica directory, tramite un file acquisito dall'esterno, la risultante path può estendersi ad altri ambiti del sistema che l'attaccante può manipolare.
8	**Mancanza di restrizioni nel caricamento di file con tipi pericolosi**	Si ipotizza di caricare un file non rischioso, ad esempio un'immagine .gif, che invece maschera un codice maligno ad esempio in .php.	Acquisizione di codici e di contenuti maligni.
9	**OS Command Injection**	Non vengono ben controllati e neutralizzati elementi speciali nei comandi al sistema operativo, che possono essere inseriti da un attaccante in rete.	Il lancio di programmi nel proprio sistemi tramite comandi al sistema operativo può essere modificato da un attaccante in rete, con tutte le possibili gravi conseguenze.
10	**Mancanza di crittografia per informazioni critiche e/o riservate**	Informazioni confidenziali sia trasmesse in rete sia memorizzate in chiaro possono essere facilmente catturate e usate in maniera illegale e/o criminale.	La mancanza di confidenzialità e il furto di informazioni riservate può provocare seri danni anche a livello di compliance con le normative esistenti (es. dati sensibili per la privacy).