Programmazione Sicura

Indice

- L'importanza della protezione del codice
- Panoramica generale sui rischi
- Random Input is a Good Thing™
- Azioni di prevenzione
- Auditing del codice
- Conclusione
- Risorse

L'importante e' proteggersi

Cosa proteggere?

- Applicazioni per il trattamento di informazioni riservate
- Applicazioni amministrative (Utilizzate da root o Administrator)
- Servizi di rete (Daemons, servizi, non escludendo CGI)
- Applicazioni con elevazioni di privilegi (SUID)

Pitfall comuni

- Gratuita' nell'analisi del codice da parte dello sviluppatore stesso
- Mancanza di auditing del codice (dovuti a costi e tempi di sviluppo)
- Fattori esterni
  - Noncuranza da parte dell'amminstratore di sistema
  - Noncuranza da parte degli utenti

L'ambiente UNIX

- Presenta una forma di autenticazione basata su login e password.
- Utilizza una tipologia di controllo DAC, basata su permessi di lettura, scrittura esecuzione distinti tra utente, gruppo, altro.
- Ogni utente ha un userid ed un groupid numerico.
- Permette l'elevazione dei privilegi utilizzando lo suid bit sull'eseguibile.

L'ambiente NT

- Presenta una forma di autenticazione basata su login e password.
- Utilizza una tipologia di controllo MAC, basandosi su ACL; il sistema tratta le proprie risorse come oggetti aventi ognuno ACL per l'accesso.
- Un utente e' identificato da un UUID (Universal Unique IDentifier) e puo' appartenere a piu' di un gruppo, anch'essi identificati da un UUID.
- Concede l'elevazione dei privilegi mediante la negoziazione di token di autenticazione.

Rischi

- Penetrazione nel sistema
- Sottrazione di informazioni riservate
- Utilizzo illecito del sistema
- Distribuited Denial of Service

Elementi

- Validazione dell’input (boundary)
- Escaping
- Root directory (chrooted enviroment)
- Parametri
- Informazioni pregresse (enviroment)
- File temporanei (race condition,link)
- File descriptors

Insomma

Gli attacchi possono provenire da:

- Utente (input)
- Enviroment
- Altre applicazioni o librerie agganciate

Di chi mi fido?

- In teoria, nessuno
- Obbligatoriamente, il sistema sottostante
- Obbligatoriamente, delle restrizioni di accesso

- Mai, dell’utente
- Mai, di altri programmi agganciati
- Mai, del mio output

Auditing

- I vantaggi dell’Open Source
- Beta testing
- Utilizzo di random input generator per il test
- Tutto ciò che la vostra mente ha previsto non è nella mente dell’attaccante e viceversa

Sistemi di base

- Controllo dell’eseguibilità dello stack: StackGuard (gcc)
- Chrooting
- Sanity check di qualsiasi risorsa esterna
- Capabilities o drop dei privilegi
- Block by default

Variabili

- Boundaries
- Ordine degli argomenti sullo stack
- Tipo degli argomenti
- Parametri
- Variabili allocate dinamicamente
- Valori accettabili (sanity check)

OOP

- Non vi è particolare differenza
- Mantenere il **private** e il **public**
- Il garbage collector
- Gli oggetti importati sono sicuri?

Applets

- Apertura di altre finestre untrusted
- Operazioni sui file del client
- Aggancio di altre applet
- Utilizzo di risorse del sistemi

Risorse

- Sikurezza- http://sikurezza.org
- Mailing List- devel@sikurezza.org
- Security Focus- http://www.securityfocus.com
- Open Web Application Security Project- http://www.owasp.org
- Secure Programming for Linux and Unix HOWTO di David A. Wheeler

GRAZIE!!!

Vincenzo Calabrò