Hakin9

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

HACKING ORACLE

PROTECT YOURSELF FROM ARCHITECTURAL FLAWS

90+ PAGES

SECURITY IN AN ORACLE DATABASE

SECURE YOUR COMPANY'S NETWORK WITH THE JUNIPER NETSCREEN NS SERIES

IDENTITY THEFT

ORACLE'S ACHILLES' HEEL – ATTACK, DEFENSE AND FORENSIC RESPONSE

Vol.7 No.10
Issue 10/2012(58) ISSN: 1733-7186

PLUS

READING BETWEEN THE LINES – THE ARTICLE BY MALWAREBYTES' ADAM KUJAWA

Atola Insight

That's all you need for data recovery.

Atola Technology offers *Atola Insight* – the only data recovery device that covers the entire data recovery process: *in-depth* **HDD diagnostics**, **firmware recovery**, **HDD duplication**, and **file recovery**. It is like a whole data recovery Lab in one Tool.

This product is the best choice for seasoned professionals as well as start-up data recovery companies.

Emphasized features at a glance:

- Automatic in-depth diagnostic of all hard drive components
- Automatic firmware recovery and ATA password removal
- Very fast imaging of damaged drives
- Imaging by heads
- Case management
- Real time current monitor
- Firmware area backup system
- Serial port and power control
- Write protection switch

Atola TECHNOLOGY

Visit atola.com for details

HAKIN9 team

Editor in Chief: Ewa Dudzic
ewa.dudzic@hakin9.org

Managing Editor: Ewelina Sołtysiak
ewelina.soltysiak@hakin9.org

Editorial Advisory Board: *Rebecca Wynn, Matt Jonkman, Donald Iverson, Michael Munt, Gary S. Milefsky, Julian Evans, Aby Rao*

Proofreaders: Michael Munt, Rebecca Wynn, Elliott Bujan, Bob Folden, Steve Hodge, Jonathan Edwards, Steven Atcheson, Robert Wood, Ewelina Soltysiak

Top Betatesters: Hammad Arshed, Amit Chugh, Viswa Prakash, M.Younas Imran

Special Thanks to the Beta testers and Proofreaders who helped us with this issue. Without their assistance there would not be a Hakin9 magazine.

Senior Consultant/Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic@software.com.pl

Production Director: Andrzej Kuca
andrzej.kuca@hakin9.org

DTP: Ireneusz Pogroszewski
Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

Publisher: Software Press Sp. z o.o. SK
02-682 Warszawa, ul. Bokserska 1
Phone: 1 917 338 3631
www.hakin9.org/en

Dear Hakin9 followers,

This month's issue deals with oracle security and id theft. Julian Evans has prepared his column, ID Fraud Expert Says focusing on how to protect yourself from ID Fraud in the UK. We also have a special article written by Malwarebytes' Malware Researcher, Adam Kujawa, who will discuss the tips and tricks on how to get what you want from assembly code. His article, Reading Between the Lines will discuss:

- *How to make sure you are looking at real code and not garbage*
- *How to rename subroutines so they are easy to spot*
- *How to leave breadcrumbs in the code by making comments*
- *How to work backwards by 'finding the cheese first'*
- *How to make your map complete by forcing the code to work for you*

The article Security in an Oracle Database written by Andreas Chatziantoniou will discuss the various ways to secure an Oracle Database and prevent SQL injections.

Paul Wright will discuss Oracle's Achilles' Heel – in which he will outline the main weaknesses of the Oracle Database and will show how to secure against its most serious architectural flaw.

Douglas Berdeaux will discuss the link between identity theft and web applications, while Delyan Boychev and Ran Levi give you the details about ID theft – the risks and consequences and how to prevent this from happening. Massimiliano Sembiante from R.I.F.E.C will discuss side channel attacks with brain leading to data and ID Theft. We also have a feature on network security appliances by Chris Weber, who will discuss how to secure your company's network with the Juniper Netscreen NS Series.

I hope that you will enjoy reading this issue as much as the authors enjoyed writing their articles.

Go ahead and Get Hakin9!

Ewelina & the Hakin9 Team.

Oracle's Achilles' Heel

Attack, Defense and Forensic Response in A Distributed Database Estate

This article will highlight one of the main security weaknesses in Oracle Databases, it will then demonstrate a solution to this weakness and finally show how native auditing can be used to forensically identify the presence of this attack in a large distributed estate using a centralised syslog audit trail.

What you will learn…

- The reader will learn how to secure against the most serious architectural flaw in the Oracle RDBMS.

What you should know…

- Readers will have Oracle DBA/Dev, Unix and security knowledge.

If a remote user tries to guess the SYS password repeatedly using an automated tool then they are not slowed down, but for other accounts they are. This means that brute force protection is only in place for low privileged accounts not for the highest privilege account. This concept was published in an article by the Author back in 2007 (*http://www.rampant-books.com/art_wright_oracle_passwords_orabrute.htm*).

See basic PoC in Listing 1.

So Oracle DB protects the lower privileged accounts more than the highest privileged SYSDBA account. This is one of the greatest weaknesses in the Oracle DB. For SYS it is even more important to delay remote pw guessing, because it is immune to the security that profiles bring (e.g. password complexity verification function and lockout).

Listing 1. *Basic PoC*

```
[oracle@orlin dbs]$ while true;do sqlplus -S -L sys/wrongpw@orlin:1521/orcl_plug as
                sysdba;sleep 0;done;
ERROR:
ORA-01017: invalid username/password; logon denied
.... 8< .....snip
no failed logon delay for SYS account!

[oracle@orlin dbs]$ while true;do sqlplus -S -L system/wrongpw@orlin:1521/orcl_plug;sleep
                0;done;
ERROR:
ORA-01017: invalid username/password; logon denied
.... 8< ....snip
failed logon delay starts for non-SYS account
```

Listing 2. *Simplified PoC code*

```
Create user systhrottle identified by lowsec12;

Grant execute on dbms_lock to systhrottle;

create or replace trigger systhrottle.tra_servererror_ora1017

after servererror on database

declare

   l_db_usr varchar2 (32);

begin

   if (ora_is_servererror(1017)) then

      l_db_usr := upper (trim (sys_context ('userenv', 'authenticated_identity')));

      if l_db_usr ='SYS' then

            dbms_lock.sleep (1);

      else

          NULL;

        end if;

   end if;

end tra_servererror_ora1017;

/
```

Lising 3. *Oracle Syslog Audit Trail*

```
Sep 28 11:37:24 oracle Oracle Audit[23714]: SESSIONID: "24523"
ENTRYID: "57" STATEMENT: "8" USERID: "SCOTT" USERHOST: "ro-rac3"
TERMINAL: "pts/2" ACTION: "103" RETURNCODE: "0" OBJ$CREATOR: "SCOTT" OBJ$NAME:
"TEST" SES$ACTIONS: "---------S------"
SES$TID: "154816" OS$USERID: "oracle"
```

Defense – put a time delay on repeated SYSDBA attempts

One way to defend against this attack is to introduce a time delay to repeated guesses on the same account to slow the attacker's guesses down. Here is simplified PoC code that achieves this by adding a one second delay to every attempt. For full production code please contact the author on *paulmwright@oraclesecurity.com* (Listing 2).

Forensic incident response via centralised auditing

It is not well known is that Oracle is the only DB vendor that has the built-in ability to centralise it's audit trail *free of charge* by pushing syslog from all the Databases to a single collector. What this means is that compliance can be achieved for a large Oracle DB estate without having to spend money on a third party logging solution. This article will now show you how to do this based on the experiences of a large scale rollout. This is what the oracle syslog audit trail looks like: Listing 3.

The forensic signature for a remote brute force attack on SYS is as follows. The `1017 status code` is specific to the failed logon and there are multiple attempts at the same `time` for the `SYS` account, which shows someone is trying to brute force SYS access into the DB (Listing 4).

It may be the case that the syslog audit trail has been compressed into gunzip format on Unix as this results in great disk savings. The compressed audit trail records can then be searched using commands like the following.

```
for file in */*/*.gz; do gunzip -c "$file"; done |
          egrep -i '1017'
```

Centralised DB syslogging Implementation Overview

So Oracle has a great centralised syslog ability which can be used for distributed DB forensics from a single loghost... but is it easy to set up?

A. Run OS syslog commands and test.

As root on DB OS (breakglass):

```
#note the entry in syslog.conf should have a tab
in the middle not a space.
local4.info @dbsyslog01.svr.emea.mydomain.net
```

May need to replace spaces with tabs in `/etc/syslog.conf` on the address line (Listing 5).

Listing 4. *Attempt to brute force SYS access*

```
[root@localhost ~]# tail -f /var/log/boot.log
Mar  9 00:26:40 localhost Oracle Audit[15819]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15823]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15823]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15827]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15827]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15839]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15843]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
Mar  9 00:26:40 localhost Oracle Audit[15847]: LENGTH : '162' ACTION :[7] 'CONNECT'
                DATABASE USER:[3] 'sys' PRIVILEGE :[4] 'NONE' CLIENT USER:[6] 'oracle'
                CLIENT TERMINAL:[5] 'pts/1' STATUS:[4] '1017' DBID:[10] '1229390655'
```

Listing 5. *Running OS syslog commands*

```
#vi or vim this will show spaces as blank
:set list
#single line replacement of spaces by tabs.
s/ /<ctrl-TAB>/g
#and restart syslog after edit on Solaris
svcadm restart system-log
#or on linux
[root@lab2-5 etc]# service syslog restart
```

Listing 6. *Incident Responder*

```
--as SYS on DB
alter system set audit_syslog_level='local4.info' scope=spfile;
alter system set audit_sys_operations=true SCOPE=SPFILE;
alter system set audit_trail='DB' SCOPE=SPFILE;
shutdown immediate;
startup;
```

Then the logger command to test OS portion of the change works well before testing the DB syslog sending.

```
logger -t "Oracle Test" -p local4.info "test to
local4.info"
```

Configure Database syslog

That is all there is to it. You will note that only SYS actions are being sent to centralised syslog with the above settings. If `audit_trail` were set to OS, all audit trails could be sent to centralised syslog, but given that SYS can tamper local audit trail this is the main audit trail to centralise. For forensic purposes it is best to use an unprocessed audit trail that has not been changed, so the audit trail should be stored on a disk which only gives reading access to humans. Being able to demonstrate that the audit trail is read only will give it greater credibility. Once all the DBs have been configured to send DB audit trail via syslog to a centralised loghost then an effective incident response component is in place. Next is the business process to provide the Incident Responder (Listing 6).

Discussion

Native syslogging has a drawback in that DBA privilege can be used to turn it off silently using oradebug (*http://soonerorlater.hu/download/hacktivity_lt_2011_en.pdf*). An interesting point is the production databases produce audit trail on a steady basis – like a heartbeat – so gaps in the audit trail indicate that the audit trail has been turned off (or the DB is down).

Conclusion

The fact that Oracle has failed logon connection throttling for all accounts except the most privileged can be regarded as the Achilles' Heel of the database. To partly compensate for this, Oracle currently has the best audit trail of any RDBMS on the market. This can be used to forensically identify a brute force attack on the SYS account. The fact that this audit trail is easily centralisable means that Distributed Database Forensics can be carried out at a single loghost for an entire Oracle DB estate. This increases the ability to respond and also makes gaining compliance a lower cost.

PAUL M. WRIGHT

Paul M. Wright is an expert at securing 3-tier Oracle architectures, with over a decade of experience, including Pentest Ltd, NGSSoftware, Betfair, Markit Group and J.P. Morgan Investment Bank. This experience includes secure software development, deployment, configuration, monitoring, logging, forensic response, compliance audits, architecture and research leading to credit in 5 Oraclequarterly Security Alerts. He is Author of the first book on Database Forensics and has been published in IOUG's SELECT Journal, UKOUG's SCENE Journal and presented original research at ISACA, RSA, ISSD and SANS.Paul currently holds OCP for DBA and Development having already taught the first Oracle and Java Security courses for SANS in the UK. His current security work is for Oracle's 12.1 DB Beta and can be viewed through his blog at www.oraclesecurity.com.

Security in an Oracle Database

The most important asset of a company is their data. Losing customers addresses, orders, payroll information, research data, medical data or credit card numbers can easily become a major disaster for each organization. Naturally you want your data to be secure in various ways, be it, its acces, its integrity, possibilities to backup and restore it in secure ways, encrypt it and so on.

What you will learn…

- Various ways to secure an Oracle Database
- Secure data at rest
- Prevent SQL Injections

What you should know…

- No prerequisites required
- Examples are self-explanatory

This article shows how the various security features of the Oracle database work and how you should deal with your data in a secure way. On the other side this might also act as a (initial) checklist for the security conscious in order to see if the current security regime is up to its task.

To be honest, the Oracle database offers thousands of features of features, and most of them cover security in one way or another. Complete books have been written and they only give a glimpse, so the intention of this article is not to be complete but to give you a starting point into the journey around Oracle Database Security.

Basics

Oracle has a more than 30-year long history with its database, and with the CIA as one of its first customers it is not a surprise that security has always been a prime concern. Nowadays, the number of main security features covers a number of thick manuals, and tightly integrates with diverse topics like (Enterprise) Identity Management, Application Security and secure backups.

To come up with the biggest point up front, it is difficult to compromise a vanilla installation of the Oracle database, and next to impossible to penetrate a well configured database. Does that mean we can move on and look for other systems that might have more vulnerabilities? No – Oracle releases security patches frequently, although not all target the database.

Users

Two kind of users exist in a database: administrative users such as the *Database Administrator* (DBA) have accounts (SYS, SYSTEM) and normal users (called schemas). In a basic version a user is administered inside the database, created by the SYS users and granted certain rights.

```
sys@orcl> create user andreas indentified by
welcome1 default tablespace users;
sys@orcl> grant connect to andreas;
```

With these rights, the user "andreas" can get into the database and can pretty much do nothing. In order to access data, you need the right to select data from a table.

Now as data is stored in relational tables, the user cannot simply select or modify them. Before *that* can happen the SYS, user needs to grant certain rights on the table itself. But rights can also be revoked.

```
sys@orcl> grant select, insert on emp to andreas;
sys@orcl> revoke delete on departments from
                    andreas;
```

This system of grants is applied on all objects inside the database. The default is – obviously – that you have no rights. Of course a concept of groups exists as well. The above examples are still valid, but today it is typically only used for application specific users, as since the advent of three-tier-architectures no "real users" access the database directly.

Users can be administered externally – for example in an LDAP server. Unfortunately a direct integration with AD does not work, but with some features of the Oracle Identity Management Suite this is possible.

Administrator

So, as you cannot access the database without a normal user, you might want to use a different way to get access to the SYS account. As with all IT systems that have users with elevated rights, the SYS user is probably the most vulnerable part of the database. Evidently, the SYS user can do everything inside the database. What you want to accomplish is to lock out the SYS user from user data. Imagine a tablespace (where the data tables reside) contains confidential data that even the DBA is not allowed to see (e.g. credit card numbers). In this case you want to ensure that the DBA cannot access the data. Oracle offers an option that makes sure that the SYS user cannot access these data. This option is called "Database Vault". In essence the database becomes a no-go area for the DBA, except for the essential system data data that makes the database itself work.

With Database Vault you would secure a tablespace using a web-based application, based on realms on which the security can be defined. When the SYS user tried to access data of a table the following will happen:

```
SELECT FIRST_NAME, LAST_NAME, SALARY FROM
HR.EMPLOYEES WHERE SALARY >10000;

Error at line 1:

ORA-01031: insufficient privileges
```

The main concern with this security feature is pretty obvious: configuring the Database Vault is typically done by the DBA. So the DBA could go in there and temporarily disable the Database Vault, do his mischief and switch it back on.

Luckily, such an action will trigger an auditable event inside the database. Auditing in itself is a basic functionality of the database, however it is normally disabled, as a full blown database in action will create an enormous amount of auditing information, which will have a negative impact on performance.

So the DBA is locked out from production data. Are we safe now, when using the Database Vault? Nope, as the audit information remains inside the database, which the SYS user could get rid of as well. Are we in a situation that the SYS user is really locked out? No, but there is another database option that will catch the DBA red-handed. The Database Audit Vault comes to our rescue. Audit events – even switching the auditing on or off – will be shipped in a secure way to another database. If you make sure that the second database is not managed by the first DBA (but for example by the Internal Oversight Committee / Auditos/Internal Affairs) you are done. You cannot prevent the access completely, but at least you have forensic evidence that something has happened.

Storage

Good, so users and DBA are covered. So – a villain might just open the data file and try to read the data directly.

```
root> dd if=/u01/oradata/orcl/order.dbf \
      ibs=8192 skip=3449 count=1|strings

1+0 records in
3449+0 records out
Andreas Chatziantoniou 1234432112344321 435.32
                  1305.96 Gizmo 3
```

So apparently someone ordered three Gizmo's worth 1305.96 Euro using a credit card with the number `1234432112344321`.

How can we protect the data when it is at rest (on the disk)?

The database offers two possibilities for this: data encryption and Transparent Data Encryption (TDE).

Data encryption needs to be used in the application, for example as shown in the following PL/SQL block (very abbreviated): Listing 1.

With a key you can encrypt or decrypt data before you insert it into a table. Problem as usual is the key management. Such a key needs to be available in the program, so either it is hardcoded or stored in a wallet. The wallet technology resem-

Listing 1. *Example of DBMS_CRYPTO (abbreviated)*

```
G_CHAR_SET VARCHAR2(10) := 'AL32UTF8';
G_STRING VARCHAR2(32) := '12345678901234567890123456789012';
G_KEY RAW(250) := utl_i18n.string_to_raw
                      ( data => G_STRING,
                        dst_charset => G_CHAR_SET );

G_ENC_TYPE PLS_INTEGER := dbms_crypto.encrypt_aes256
                          + dbms_crypto.chain_cbc
                          + dbms_crypto.pad_pkcs5;

FUNCTION enc_creditcardnr( i_ccnr IN VARCHAR2 ) RETURN RAW
IS
    l_ccnr RAW(32) := UTL_I18N.STRING_TO_RAW( i_ccnr, G_CHAR_SET );
    l_enc RAW(32);
  BEGIN
    l_ccnr := utl_i18n.string_to_raw
              ( data => i_ccnr,
                dst_charset => G_CHAR_SET );

    l_enc := dbms_crypto.encrypt
                  ( src => l_ccnr,
                    typ => G_ENC_TYPE,
                    key => G_KEY );

...
```

bles the way in which Java or Apache store keys, however there is only a graphical interface for this.

A better alternative is to use the Transparent Data Encryption. With TDE the complete datafile is stored in an encrypted way. The following snippet shows how to create such a secured tablespace:

```
CREATE TABLESPACE my_enc_tbspc DATAFILE '/home
/user/oradata/my_enc_tbspc.dbf' SIZE 10M
ENCRYPTION USING 'AES192' DEFAULT STORAGE(ENCRYPT);
```

That means that the above example of the root user executing a "dd" command will deliver just garbage. Naturally, for the TDE a separate key is needed. Often companies combine the access mechanism to the key with a smart card. So only when the DBA is on site, the smart card is inserted into the reader, otherwise the card is put into a physical vault (banks do this as they have the infrastructure and certain applications that will operate only during certain hours). Therefore stealing the disks and trying to restart the database on a different computer will fail, as you miss the wallet with the key.

Now we know that the data is safe inside the database and even when it is on the disk (at rest). As a smart hacker you then want to try the back-up media (often enough these are still tapes, although they became less popular). Access to tapes, especially when they are stored off-site is dangerous. Now a similar technique can be (optionally) deployed with backups: secure backups – which mean that the backup data in itself is encrypted again. So only when you have access to the wallet you can restore the database. This poses an organizational issue, as you need to ensure that a copy of the wallet is stored in a very secure location onsite and off-site as well.

Secure access of the database

Typically, the database is accessed with either a client program called SQL*Plus or a JDBC connection. When you access the database with either one of those two possibilities you could see the data in plain text traveling over the cable. The following picture shows this: Figure 1.

Now – how can we secure this area? Again Oracle offers options that would encrypt the data when it travels over the line. Either the Advanced Security Option is used or a JDBC Connection is opened with SSL: Listing 2.

Your Java code will contain something like this: Listing 3.

After that the database traffic will be encrypted.

Lising 2. *Configuration of the Advanced Security Option*

```
(DESCRIPTION=
  (ADDRESS_LIST=
    (ADDRESS= (PROTOCOL = tcps) (HOST = my_orcl_server) (PORT = 1521)))
(CONNECT_DATA=
  (SERVICE_NAME= creditcarddata.securecompany.com))
(SECURITY=
  (SSL_SERVER_CERT_DN="cn=sec_guru,cn=OracleContext,c=nl,o=securecompany"))
```

Listing 3. *Usage of Oracle Wallet*

```
java.util.Properties japro = new java.util.Properties();
...
// Define key store, and password
japro.put ("javax.net.ssl.keyStore","/home/andreas/wallet.jks");
japro.put ("javax.net.ssl.keyStoreType","JKS");
japro.put ("javax.net.ssl.keyStorePassword","welcome1");

// Set the trust store,  and password
japro.put("javax.net.ssl.trustStore","/home/andreas/certs.jks");
japro.put("javax.net.ssl.trustStoreType","JKS");
japro.put("javax.net.ssl.trustStorePassword","welcome1");
```

Virtual Private Database/ Label Security

Often a database is offered in a multi-tenancy setup. Several customers might put their data into your database. This results in a problem, as a user who has the right to access a table, e.g. to select or update data, cannot be prevented from sticking to his own data organizations are stored in the same table.

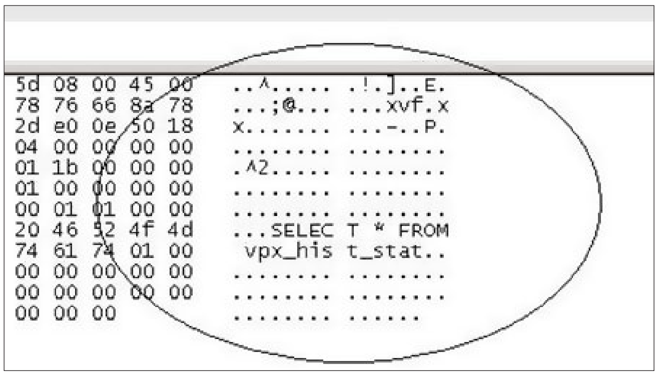
There are two solutions for this. The easy one would be to give every customer an independent tablespace. This results in a overhead as resources are not shared. A better way would be to use the Virtual Private Database. The VPD adds an "invisible" WHERE-clause to the table. This is called a policy. By doing this each SQL query towards that table will need to satisfy the additional WHERE-clause. Such a WHERE-clause will check for system-set data like a context, that gives information about the user.

An add-on (extra product) would be Label Security. Label Security is based on VPD but lets you define hierarchies. So you could implement a matrix with permissions like (PUBLIC, SECRET, EYES_ONLY) and match them with departments or organizations (EMP, HR, FINANCE), Then you define again policies and match them. E.g. EMP and HR might see PUBLIC information in the FINANCE schema, but cannot see higher protected data. If you move up the chain, then you get more rights.

SQL Injection

The final measure I'd like to introduce would be a prevention of the SQL Injection. SQL Injection is a technique in which a normal query is modified in a way that more records are returned. A normal SQL query would return the number of records it should retrieve (e.g. depending on a filter called a WHERE-clause). A normal query could look like this:

```
SELECT * FROM emp WHERE name = 'andreas';
```



```
5d 08 00 45 00    ..^.....  .!.]..E.
78 76 66 8a 78    ...;@...  ...xvf.x
2d e0 0e 50 18    x.......  ...-..P.
04 00 00 00 00    ........  ........
01 1b 00 00 00    .^2.....  ........
01 00 00 00 00    ........  ........
00 01 01 00 00    ........  ........
20 46 52 4f 4d    ...SELEC  T * FROM
74 61 74 01 00     vpx_his  t_stat..
00 00 00 00 00    ........  ........
00 00 00 00 00    ........  ........
00 00 00          ........  ......
```

Figure 1. *Plain Text DB Content*

A query with a SQL Injection looks like this:

```
SELECT * FROM emp WHERE name = 'andreas' OR
                '1'='1';
```

Quite simply the database is tricked to evaluate the statement in the WHERE-clause. Adding the part OR `'1'='1'` will result in a WHERE-clause that is always true, as 1 equals 1 and in combination with the logical OR we will always receive a true WHERE-clause.

This can easily happen with a Web application that sends queries to the database and does not sanitize the input.

The DBA might be pretty helpless as the application could come from an external party and could not be changed. The resolution to this issue would be the usage of a Database Firewall. This extra product checks (and learns) incoming queries and can either drop them or substitute them when it thinks it detects a SQL Injection.

Conclusion

As stated in the beginning, there is a large number of security possibilities inside that database. Not every company uses all of them, so you might come across a database which can easily have better protection. Securing a database is no rocket science, getting compromised data out of your system will typically require voodoo and good lawyers to fight off law suits. When you are responsible for the protection of the database use the security measures mentioned above, and add them to your database. When you are out "hacking" an Oracle database use the same list and see if your DBA was just lazy.

From there investigate how the database can be secured even tighter and add additional security measures.

ANDREAS CHATZIANTONIOU

Andreas has some 25 years of IT experience, of which the last 14 were exclusively spend with Oracle Technology (Database, Application Server, WebLogic Server, Fusion Middleware). Having worked for Oracle and Accenture and other companies, Andreas is now a freelance Oracle Consultant who – after all these years – thinks he partially understands the stuff he's working with. Contact data: Andreas Chatziantoniou, andreas@foxglove-it.nl.

idtheft protect

Be reactive...

- Your systems are being attacked 24 hours a day...
- You understand the threats and are protected against them...

Be proactive...

- My users' behaviour threatens our systems...
- I understand what motivates my users and what threats are coming my way...

ID Theft Protect provides information on threats from a user perspective.

Visit: **http://id-theftprotect.com**

Digital Security and Risk Analysis

Side channel attack with brain leading to data and ID Theft

Recent development of computer science integrated with neural engineering, allow detecting and decoding of brain activities via sophisticated interfaces devices. This may expose users to serious threats.

RIFE
Research Institute of Forensic and E-Crime

This article will provide a review of the latest research, will summarize the techniques used to interface brains with computers and will analyze potential risk exposures.

The beginning

Something that truly sounds like one of those futuristic, visionary Sci-Fi movies as Johnny Mnemonic is probably already a reality.

On the 8th of August 2012, a research team from some of the most renowned universities' such as: Oxford, Berkley UCLA and Geneva, announced a successful attempt to "hack" into human brains using a BCI (brain computer interface) device, with a technique similar to a Side-Channel attack.

Many people are not really aware of what this achievement may represent in terms of progress and future developments.

To have a clear understanding of what exactly is the risk we are exposed to and how the "Brain Hacking" is achieved, we need to have an overview of how the brain works and what technology has been used to attempt mind reading.

The Brain

Our Brain is an incredible machine composed by a number of features and functions. Almost everyone is aware that Neurons (Figure 1) are fundamental cells that through electro-chemical reactions allow exchanging information and performing major brain activity.

Our brain contains around 100 billion neurons. These cells are used to process electric signals.

Axons are extensions of neurons. The cell membrane of the axon contains voltage-gated ion channels that allow neuron to generate and propagate electrical signals and communicate with each other.

The ionic current flows within the neurons generate a fluctuation that can be measured and recorded using a device called *electroencephalograph* (EEG) that uses sensors applied over the scalp to detect the electric activities.

Generally the electric activity of the brain is represented by the summation of the synchronous activity of thousands or millions of neurons with similar orientation. If the cells do not have similar orientation, the ions do not line up and create waves that can be detected called Brain Waves.

Brainwaves are classified as the following:

- Beta – when the patient is alert or feels agitated or afraid.
- Alpha – during physical and mental relax.
- Theta – somnolence with reduced consciousness.
- Delta – when there is unconsciousness, sleep or even catalepsy

The BCI – Brain computer interface

Research on BCI device began in USA at UCLA Berkley around 1970. The project was commissioned by DARPA (Defense, Advanced Research Project Agency), the same agency that supported the ARPANET project, formally known as INTERNET. After years of experimentation on animal cavy, in the mid 1990 the project announced the first successful Neuroprosthetics device implanted on humans.

Neuroprosthetic devices are neural prosthesis capable of substiuting sensor and cognitive functions, generally implemented on patients affected by neural damages, with reduced or inexistent functionalities results of injuries or diseases (Figure 2).

Over the past decade, many researchers and laboratories started to explore the potentialities of BCI technology as it may be the source of a number of future applications.

Even if the BCI device seems quite complex and sophisticated, practically is based on the same concept of the *electroencephalograph* (EEG) recording brains activity from scalp electrodes, sampling signals (typically of 128 Hz – 512 Hz).

Each single variation of the electric waves is mapped and translated into performable tasks such as: controlling cursor movement, select letters or icons and so on.

The Mapping phase is important because the system requires to be previously tuned in order to interpret the signals. During the BEAM (*Brain Electrical Activity Mapping*) the BCI perform the following steps:

- Signal acquisition – user performs tasks to map EEG samples of brain activities.
- Samples Extraction – translate received signals to specific tasks

For instance, when we think or visualize an object, we produce electric activity over multiple areas, mainly because our brain is searching through our memories an association with the target object to identify it and schedule the next activity.

The process of sampling and training is obviously pertinent to the target use.

If the purpose is to provide help and support a disabled patient, the training will be specialized to select all those brain signals that are relevant to perform the assigned tasks.

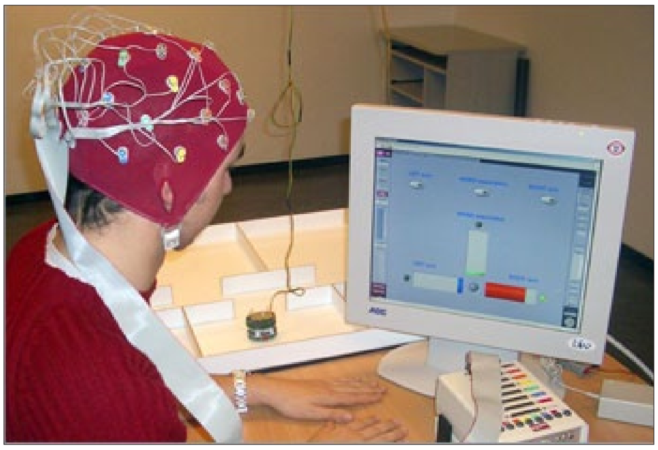


Figure 2. *Computer and brain activity*

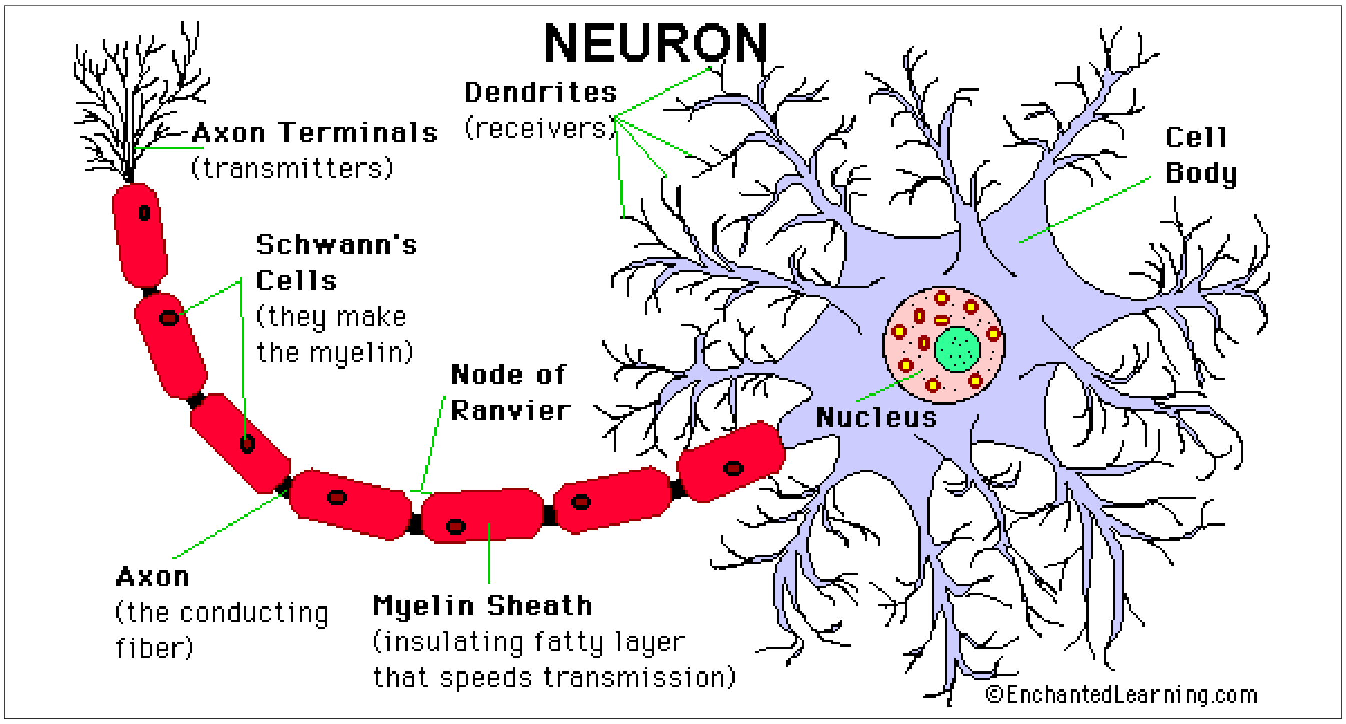

NEURON
Dendrites (receivers)
Axon Terminals (transmitters)
Cell Body
Schwann's Cells (they make the myelin)
Node of Ranvier
Nucleus
Axon (the conducting fiber)
Myelin Sheath (insulating fatty layer that speeds transmission)
©EnchantedLearning.com

Figure 1. *Neuron description*

For instance, if the patient needs to move an electronic arm, during the training he will visualize an image or will be asked to think to a specific action attempting to move the electronic arms as it would be his arm. The training can be performed over any type or muscular movement, emotions, thinking as well as any other operation that cause a variation in the activity.

The image below (Figure 3) shows a voltage variation stimulated from a specific event. We can see the difference between the target stimulus and the non-target stimulus demonstrating how BCI distinguish correctly among signals.

Using a commercial BCI system produced by Emotiv or NeuroSky, we can have transfer rates around 5–25 b/min. Greater speed and accuracy may depend on translation algorithms, signal processing, and user training.

On the market more sophisticated devices are available, used in Neural Engineering for medical application, i.e.: the *Computed Tomography* (CT) or *Computed Axial Tomography* (CAT) or Positron emission tomography (PET).

Recently, researchers are developing and expanding a branch called "neuroimaging" that seriously dive into "Thought Identifications". Thus we are now able to reconstruct words, images, entire thought pattern or even to perform "Intentions Prediction" with an accuracy of 60%.

The experiment

What are potential risks in this scenario? How can this devices and software be abused by malicious?

Emotiv and NeuroSKY provide hardware and software together with a wide variety of applications, SDK kits and API for free testing and development of the products.

The API allows unrestricted access to EEG sampled signal, leaving the door wide open for malice to develop "spy" applications that attempt to capture and misuse user's brain signal, hence personal sensitive information.

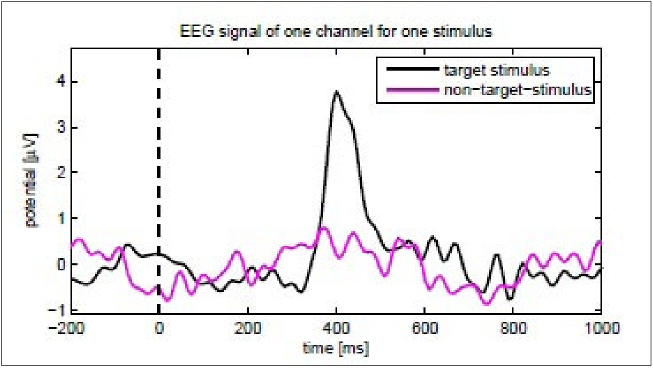


Figure 3. *EEG signal of one channel or one stimulus*

The experiment performed by the research team from Oxford, Berkeley UCLA and Geneva Universities has proven the feasibility of a side-channel attack using a commercial BCI device.

The test involved 30 "cavy" subjects and consisted of 3 main steps:

- The victim received a verbal explanation of the task by the operator
- Images and messages displayed on screen for 2 seconds to acquire the brain signal sample
- Images being flashed to victims in random order

No specific instructions were given to subjects. The participants had to watch the screen and visualize a sequence of images (Figure 4), such as:

- Pin Code
- Debit Card number
- Geographic Location
- Month of Birth
- Face Recognition – People
- ATM – Bank Information

Face recognition test

For this test a set of images were displayed for about 2 seconds and 1 of the images was the picture of President Obama. Later another image was displayed on the screen containing the following questions: "Do you recognize any of this people"? Then the images containing people were randomly flashed for the rest of the experiment detecting the brain responses.

The aim of the test was to verify whether would be possible to understand when the participant

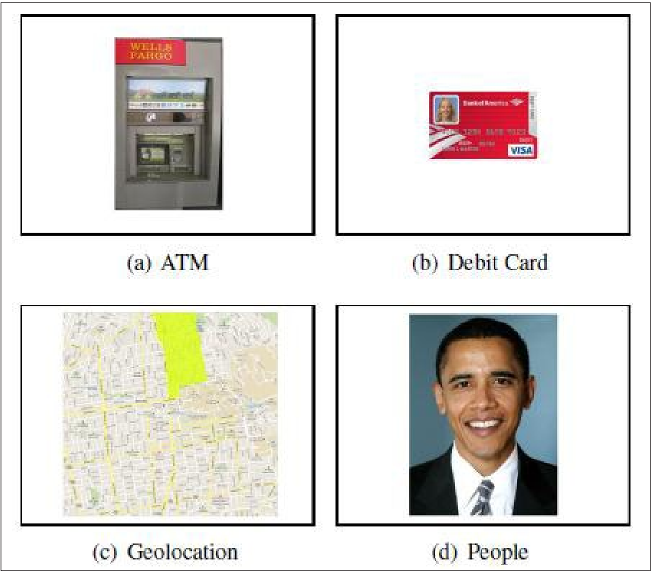


Figure 4. *Face Recognition Test*

recognizes the image among all the other displayed by reading their EEG responses.

The experiment had a similar procedure for all the other proposed tests. The results of the experiments are visible in the image below (Figure 5)

Obviously, the results of the test must be interpreted in terms of probabilities.

However, the experiment results show a 43% of probabilities of information leakage. This is a very high rate of success also because the researches declared that due to the simplicity of the test, it is possible to assume that a more sophisticated attack may result in a higher successful rate for the attacker.

The threat Scenario

The assumptions are that:

- BCI devices and EEG techniques will have a rapid progress and dissemination.
- More and more, programmers will develop applications forcing users to map and sample personal sensitive data.

It is not encouraging to admit that methods to steal sensitive data are sadly increasing. There are a number of reasons why personal data is continuously exposed to threats but mainly because it represents an amazing reward for the crime industry.

Identity Theft – Why IDs are stolen

Identity and personal information are valuable and strictly linked to identity.

Identity theft occurs when an unauthorized person obtains key pieces of personal information about you, such as your Social Security or driver's license number, and uses them to impersonate you. The information can be used to obtain credit, merchandise, and services in your name, or to provide false identification to police, creating a criminal record or leaving outstanding arrest warrants in your name.

We can categorize Identity theft in two ways:

- *Real ID* – i.e.: Thief uses stolen or cloned documents, such as passport or driving license to open new accounts, or use it to impersonate the victims during malicious activities.
- *Cyber ID* – i.e.: The imposter uses a cyber-ID to gain access to an existing account/service, or to create new accounts.

These types of malicious cyber activities may fall under crime such as:

- Unauthorized Use of Personal Identifying Information
- Unauthorized Access to Computer Material
- Unauthorized Access to Computer systems with intent to commit another offense
- Unauthorized Modification of Computer Material

In certain cases stealing a Real ID allows an attacker to gain access to cyber services or use it for online credibility. (id est: Passport ID used to buy flight tickets, ID cards to open online accounts or as evidence to provide when requested from online services)

What they Steal

Some of the "most wanted" personal information categories are:

- Financial
- Personal Identity
- Criminal
- Medical
- Education

How they Steal IDs

There are a number of techniques to steal personal data. Here is a list of the most common techniques to obtain personal information:

- Dumpster diving – practially the thief rummages in rubish and trash bins for identity
- Retrieving data from any kind of devices with storage/memory capabilities including mobile and portable devices.
- Pick-pocketing, housebreaking or mail theft to steal public records such a: bank or credit cards, identification cards, passports, authentication tokens
- Using generic questioning templates. (Real and Cyber information theft)
- Skimming bank cards and creating clone cards. Skimming techniques make use of devices capable of reading magnetic band and integrated chips.
- RFID copy and cloning using contactless devices to copy data.
- Shoulder Surfing or any the observation technique that allow to capture passwords or other type of sensitive information.
- Malicious code, Virus, malware, spyware etc. or any other type of malicious code and technique such as keystroke logging and hidden remote control.
- OS, Application and databases hacking (included hardware embedded firmware) to

Appendix 1

- *http://techcrunch.com/2012/08/27/brain-hacking-scientists-extract-personal-secrets-with-commercial-hardware/*
- *http://blogs.computerworld.com/cybercrime-and-hacking/20900/hacking-mind-3-new-brain-hacks-expose-new-realm-security-privacy-risks*
- *https://www.usenix.org/conference/usenixsecurity12/feasibility-side-channel-attacks-brain-computer-interfaces*
- *http://www.ocf.berkeley.edu/~anandk/neuro/bci-vertex-abstract.pdf*
- *http://sti.epfl.ch/page-1749-en.html*
- *http://en.wikipedia.org/wiki/Thought_identification*
- *http://en.wikipedia.org/wiki/Neuroimaging*

retrieve large quantity of personal data, as well as to grant access and abusing privileged accounts of users.

- Brute-force techniques to gain access break ciphers and encryption.
- Fake Advertisement offering bogus jobs or other type of services in order to accumulate resumes and applications typically disclosing applicants' names, home and email addresses, telephone numbers and sometimes their banking details.
- Phishing or scamming through Emails, SMS text messages, phone calls or other forms of communication in order to dupe victims into disclosing their personal information or login credentials, typically on a fake corporate website or data collection form. Spamming can also be vehicle of malware / spyware or so.
- Biometric falsification such as fingerprint identification/theft. Biometric identification methods are recently increasing. Facial reconditioning or retina scanning and other types of methods maybe subjected to complex theft mechanisms in the near future.
- Internet and Social network browsing and befriending, to obtain personal details published by users as well as for pictures used to create fake identification documents.
- Network attacks such as Man in the Middle or session hijacking to capture inbound and outbound traffic from a computer and obtain sensitive information. The list of the methods used from a malicious to steals personal ID and information has now a new entry.
- ID theft using brainwave samples used for BCI devices.

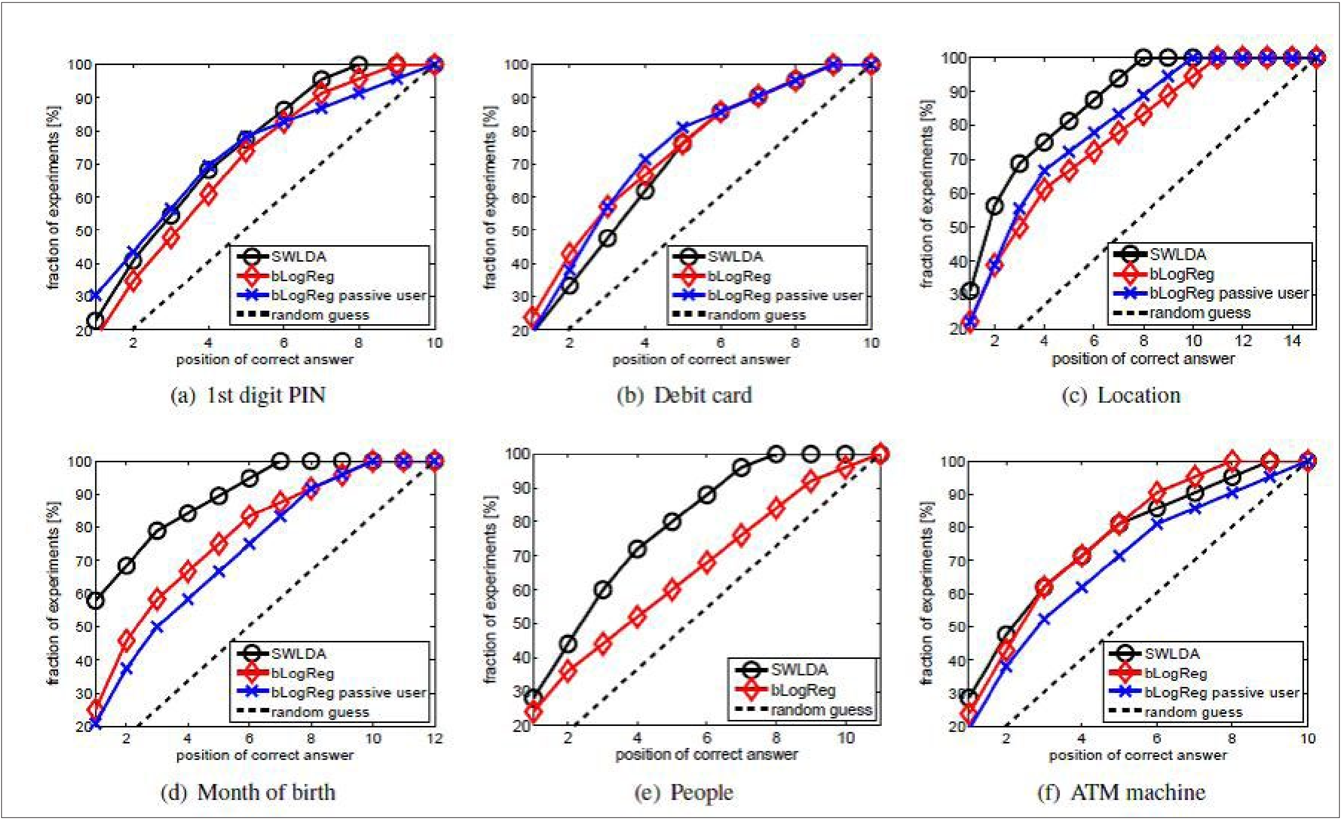


Figure 5. *Results of experiments*

Conclusions

As we have seen all these began around 1960, highly supported by US Military. Therefore, the question that spontaneously pops up is: “Why did the USA’s Department of Defense spend money on a project that was supposed to improve the living conditions of people affected by neural and physical definiciencies?” Think about it.

Progress can’t be stopped but at same time new research are sometimes opening to ethical paradox and moral dilemmas. For the first time, we are aware that we may be very close to achieve Mind Reading in a scientific way. We are likely to shed light on one of the darkest places in the universe and everything we know. Our secrets and past memories may be at risk.

Who will decide on the ethical implication of such powerful tools?

Who will protect us from inappropriate usage and limit potential misuses?

How can we establish standards to ensure human right are preserved, when the last bastion of privacy has fallen?

At the moment we are only able to detect and map brain activity by “reading” the electric movement, but in the near future with the help of technology progress, we may be able to interact with the brainwaves and presumably to “write” into our brain, mimicking sampled patterns and possibly modifying recorded information, hence modifying our personal information and memory.

This would result in more serious risks that may lead to tragic consequence, such as:

- Mind manipulation – by injecting manipulated patterns in our brain.
- Personality mutations – as repercussions on the above threat.
- Memory modification – by overwriting existing memorized information.

Is The Matrix few years away?

SEMBIANTE MASSIMILIANO
IT Sec&Risk Eng. at UBS Bank
M.Sc. Computer Security
Can be reached at: msembiante@rifec.com
RIFEC
Research Institute of Forensic and E-Crime

Identity

information theft and web applications

Identity theft is a process and, like any other process, needs a place to begin. For example, once the smallest amount of personal data is compromised, the next step an attacker can do is test for password or secret questions reuse. This could potentially open up more doors into email accounts, social network sites and much more.

What you will learn…

- The importance of securing web applications and identity information
- How the smallest vulnerability in a web application can lead to the largest identity information breach
- Security tips for database administration of CMS users
- Several web attack methods of hackers who target your data
- Things to never do with Identity information access

What you should know…

- General functionality of web applications
- Cross site scripting (XSS)
- Introductory database administration

A majority of people from studies, easily found using Google, reuse passwords. Some of these passwords are reused even from their bank accounts. Some people will use a password "scheme" in which they have an identifier intrinsic to the website in which they log into. Listing 1 is a simple example using Gmail and Facebook.

Can you tell which password is used for which site? g(mai)L – f(aceboo)K

Passwords and their corresponding usernames or email addresses are being dumped on a daily basis to popular pasting websites from hacker groups around the world. Most of which usually come from either SQL injection, or even spear phishing attacks on organizations. What's even scarier is that a lot of successful, large-scale attacks on companies that reveal all of their customer's identity information aren't made public by many hackers. This will leak the data and allow others to steal identities long before the victims, including the organization and their employees, know about it.

What is identity information? This information can include: *addresses*, *custom secret questions and answers*, *usernames*, *email addresses*, *phone numbers*, *employee information* and much, much more. For an organization to keep track of their customers or followers, it has become a very popular practice to employ some sort of web application for record keeping and reporting. A web application is any application in which we can manipulate data using only a web browser. Usually written in a web programming language such as PHP, ASP, Ruby, Coldfusion and supplemental Javascript code, a web application can connect to a databases, write to files, or even call other API's to access other applications or data outside of the organization.

Some organizations allow outside access to their web applications for their employees from anywhere in the world. *This should never be allowed*. Web applications which can read and write customer identity information to databases should be privately kept behind firewalls. If an organization must allow remote access to a web application to it's employees, a good practice would be to re-

Listing 1. *A simple password scheme*

```
mypass2012_gL
mypass2012_fK
```

quire that employee to connect to the domain via an encrypted VPN connection. The reason for this is that, no matter which programming language an organization uses, no matter how much they test the software, bugs can present that allow an attacker full access to all database information from anywhere in the world.

It is extremely easy to accidentally write semantic or logic errors into web code that could give an attacker complete control over all customer identity information.

Once the personal information is compromised, the attacker becomes one step closer to full identity theft of potentially thousands of victims.

let's imagine our customers information locked in a safe in which only one guard has a key. The bank is then locked up and an authenticator is standing at the front door of the bank which relays information from customers outside to the guard and then back to the customers. The authenticator has a specific set of rules and instructions in place by the programmer and the guard trusts them but doesn't know what they are specifically. In the case of SQL injection, the authenticator gets compromised and loses all control of these rules and since the guard on the inside of the bank in front of the safe already trusts the authenticator, he gladly gives the customer information to the authenticator who then passes it to the attacker outside.

Now, if we make our application only available to our internal network, we block off the front door of the bank to the world, but open another inside just for us. This helps us in our mission to keep the customers identity information safe but doesn't fully protect it alone. Most of the machines internally can have access to the internet, be compromised, and act as gateways for an attacker to the internal network.

Another thing to consider, is that an institution could use the same "*schema*" or authenticator for many web applications or web sites. *This should never be done*. If an SQL injection point is found

Listing 2. *MySQL PHP Function to connect to a database*

```
mysql_connect(where,username,password);
```

Listing 3. *Google dork for poor PHP syntax*

```
"Warning: mysql_query()" "invalid query"
                site:victimloser2012.com
```

on their web site, access can reveal not only the web content, but internal customer identity information as well. Some "*hosting*" websites, will allow one content management system (CMS) user to access all databases for all of the hosted web sites for all companies. This includes any web application software written by their customers! This is extremely devastating to happen.

Our next attack on the web application involves the spear phisher. If an attacker can find even the simplest cross-site scripting (XSS), vulnerability, he or she can then send a malicious link to the web application users. The URL could contain a link to evil javascript code, which rewrites the web page objects, changing the login forms and even sending cookies across the web.

Another way to gain access to databases is by stumbling upon the web application's source code on a compromised system. If our service software is out of date and a known remote exploit exists for one of our services, we could be leaving an even bigger door open to our customer's data. We could also gain access to these files via remote or local file inclusion attacks. The source code of interpreted web languages like PHP is in plain text. This means that our username and password of our web application is just sitting in a file on our server. Listing 2 shows how PHP can connect to a MySQL database.

Now if the attacker did a simple search with `grep` on UNIX/GNU, or `find` on Windows systems, for "*mysql_connect*", access could be just one more step away. Sometimes the password is stored in an include file and assigned to a variable. For most other languages, the attacker can just search for the case-insensitive string "password".

If a *local file inclusion* (LFI) attack exists, an attacker can take advantage of the fact that the web server UID can read and write to the server logs. After some digging, the attacker can find these logs, alter his browser's user agent to include interpreted code and then call the file with the browser to run the user agent code. Then all SQL commands can either be ran directly from the browser's user agent string, or from an uploaded shell-like application, giving him or her complete access to the customer information in our databases.

For an attacker to find these web applications and sometimes even their vulnerabilities, he or she has several options. Google as a search engine violently spiders the internet crawling through pages and links within. A special Google search string called a "*dork*" can be used to find usernames, email addresses, passwords, vulnerable network devices, and anything else from within a site including specific file types and any internet accessible web application. For example, Listing 3 will search for a MySQL vulnerability only within the site "*victimloser2012.com*"

This can reveal the information found in Listing 4.

which is very useful to the attacker. Another method is pure brute force. In this method, the attacker creates a simple script that tests an HTTP connection for each word in a list to check for 200 OK status codes. This can find tarball, zip files of backups, text, test, and developmental files that the web application programmer may create during development. A backup file can be downloaded without rendering and the source code can be extracted and read locally revealing passwords for our customer identity information. This method could also return directory traversal vulnerabilities. Sometimes a web server can be configured to allow the public to view the contents of a directory in which no index file resides. This can also reveal files that an attacker can download and view locally (Listing 5).

Listing 4 shows a simple Bash shell script that will test if the HTTP response from `curl` returns an "HTTP 200 OK" status in the headers. This is done by passing a filename to the application.

Listing 4. *MySQL Error from poor PHP syntax*

```
Warning: mysql_query(): supplied argument
              is not a valid MySQL-Link
              resource
in/directory/webapplications/accounting/
              login.php on line 188
Invalid query: Access denied for user:
              'adminuser2@10.0.13.37'
              (Using password: YES)
```

Listing 5. *A simple Bash script brute force method for finding hidden files on a web server*

```
#!/usr/local/bin/bash
function get {
       if [ "$(curl $URL -sIN | grep OK)"
              ]
              then echo "$URL is good."
       fi
}
function url {
       URL=http://weaknetlabs.com/$1
       get
}
url $1/
url $1.tar.gz
```

If not the headers from the HTTP response from weaknetlabs.com returns a 404, then the directory or gzipped tarball doesn't exist.

We can also pass a line by line dictionary file to the script that contains lines like "files" and "backup" using cat and xargs like so:

```
[trevelyn@shell ~]$ cat words.txt | xargs -I {}
                ./getscript.sh {}
http://weaknetlabs.com/main/ is good.
http://weaknetlabs.com/linux/ is good.
http://weaknetlabs.com/linux.tar.gz is good.
[trevelyn@shell ~]$
```

An astounding amount of identity information can be compromised from the result of one simple vulnerability found in an organization's web application.

Points to remember

- Although parallel computing has made it slightly easier for an attacker to decrypt passwords, we should *never* store all customer identity information in plain text. This includes answers to secret questions and passwords. They don't need to be in plain text in the database. We can hash their input and match that against what they have already stored with our data to verify their identity.
- We need to keep our software up to date for our servers and never stop testing our code for vulnerabilities. If we really care about the security of our customer's identity information then let our companies spend money on security. This includes trained professionals in debugging and penetration testing our softwares.
- Never allow direct internet access to our web applications or databases.
- We should never leave backup files anywhere accessible via the web.
- Never use the same schema for our databases for every user or client which plans on hosting customer information and remember to employ multi step validation processes when applicable. Sending a user a text message with a security code is a much better practice than simply asking what their favorite color is.

DOUGLAS BERDEAUX

Douglas Berdeaux is a certified wireless security professional and founder of WeakNet Laboratories. His coding includes WEAKERTH4N Linux, WiFiCake-ng, pWeb – Web Application Security Testing Suite, and several Android applications. He currently is a senior web programmer for Duquesne University in Pittsburgh, PA, USA.

HACKTIVITY

The IT Security Festival in Central and Eastern Europe

October 12-13, 2012. MOM Cultural Center, Budapest

THE LARGEST IT SECURITY FESTIVAL IN CENTRAL AND EASTERN EUROPE WILL BE HELD AGAIN! Real festival mood, peresentations, workshops, games, hardware hacking, lockpicking, big friday party and 1000+ hackers from all over the world!!!

Keynote Speaker:

Jeff Bardin, USA

Jeff is the Chief Intelligence Officer for Treadstone 71. In 2007, he was awarded the RSA Conference award for Excellence in the Field of Security Practices. He is the most respected expert in the field of cyber crime, cyber terrorism, cyber inteligence.
This talk covers the cyber intelligence lifecycle including examples of denial and deception. Open source intelligence (OSINT) is a critical aspect of asymmetric cyber warfare. It is part of the mosaic defense and one practiced as a method of unrestricted warfare. Methods of cyber espionage, sock puppet creation, infiltration, data collection and analysis are covered. Case studies on creating your own personas while using OSINT tools will be discussed.

...and who can you look forward to?

ZOLTÁN BALÁZS / HUNGARY --- Zombie browsers, spiced with rootkit extensions
ALEXANDER POLJAKOV / RUSSIA --- Top 10 SAP vulnerabilities and attacks
JOE MCCRAY / USA --- The Evolution of Pentesting High Security Environments
ANDRÁS KABAI / HUNGARY --- Hunting and exploiting bugs in kernel drivers
ALEXANDER KORNBRUST / GERMANY --- Self Defending Database
VIVEK RAMACHANDRAN / INDIA --- Malicious Wi-Fi Routers for Fun and Profit
MIROSLAV STAMPAR / CROATIA --- Spot the Web Vulnerability
BOLDIZSÁR BENCSÁTH / HUNGARY --- Duqu, Flame, Gauss malware analysis experiences
SHAY CHEN / ISRAEL --- Diviner the new OWASP ZAP extension

PAYPASS VULNERABILITIES · HSRP INSECURITIES · „CHIP-TWEET" · TRACING MOBILE PHONES · ALTERNATIVE USAGE OF PKI DEVICES · LOCKPICKING 2.0 · ALTERNATIVE INTERNET · USB = UNIVERSAL SECURITY BUG · iOS SECURITY · ANDROID SECURITY · NAT ATTACK · BROWSER BASED ATTACKS · DIGIPASS INSTRUMENTATION · SECURITY CODE REVIEW · GEEK GIRLS · ELITE SOCIAL NETWORKS CROOKS · AV INSECURITIES

AND WHAT ELSE?!

Hello Workshops. Jump from theory to practice: **Hello Injection** **Hello CA** **Hello Code Review**
Hardware hacking / Lockpicking (non-destructivelock-opening) workshop and Urban Warrior competition / **24 hours - Hacker road reloaded.** Get prepared. Never experienced any similar game. Form a team, with a good hacker, a good lockpicker, a good social engineer.

Tickets are available until 20th of September with 10% discount on www.hacktivity.com

Full price for adults: 68 EUR / for companies: 150 EUR / Cheap hotels offering also there!

Special packages:
2 days ticket & 2 nights in a hotel*** 199 EUR
2 days ticket & 2 nights in a hotel**** 299 EUR
packages.hacktivity.com

Sponsors:

Further information and registration: www.hacktivity.com

Deloitte. · rrc THINK VALUE. · biztributor · WEBSHARK · ARUBA networks · AdNovum · FORTINET · Hakin9 All About IT Security · asc

The Hidden Facts About Online ID Theft

Have you ever wondered what is ID, what is behind this word? Do you know what makes the difference between ID and Online ID?

What you will learn…

- what is ID and Online ID;
- the ways you could lose your Online ID;
- what are the risks and possible consequences of lost of your Online ID;
- how to protect yourselves.

What you should know…

- there are no specific technical terminology, but some basic knowledge about what is risk, encryption, would be nice;

What is ID? Well, this is something which ultimately identifies you. It doesn't matter how. The obvious answer could be a passport, ID card, *National Identification Number, Social Security Number* (SSN). But there are some not so obvious ways for identification. You should think about everything else which could be used like the combination of phone number and address, your birthplace, date of birth. A typical Online ID consists of Username and Password. But it could be also different types of tokens, SIM cards, PIN codes. Sometimes, even the public IP address of your computer or router could be used as your identification.

How it is different from the well known "regular" ID theft? Your proof of ID is something tangible, something you can touch and see. Your Online ID is something virtual. It doesn't exist in the real world. It is stored on some server, somewhere. You cannot touch it. In most of the cases, you don't even know in which country the server is located. When you register with some site, you typically, provide some personal information such as name, e-mail, phone number, address, job title. If the site is related to some purchase, you could also provide information about your bank account, credit or debit card information. In case the account is compromised all or part of the information could be lost as well.

Why your Online ID is interesting to someone? There are many reasons why your ID is interesting to the bad guys. Finances are the most obvious – they could transfer money from your bank account, they could buy something using your credit card information, they could even borrow money from a bank on your behalf. But there are some not obvious, which could have even greater negative impact to you like reputational, fraud, cheating on your behalf, even revenge. With the growing usage of Internet and online services, your data becomes more and more of a target to organized crime. Once stolen, it could be sold to a wide range of companies.

There are mainly two ways of compromising an identity. The first one is to steal the data from the operator or from the provider of the online services.

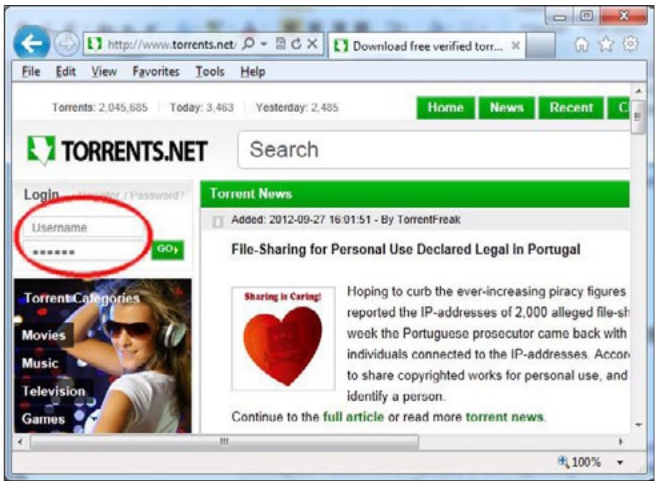
Typically the data, which contains information about your ID is stored in some kind of database. How well is this database protected? This is probably, the hardest way for someone to steal your ID. However it is the most efficient. With a single "strike" a hacker could steal thousands and thousands of Online IDs. While in the past such an action was just a question of pride, nowadays the data is sold to different companies. The target of such kind of attacks is the company, not a single person.

The second way is to steal the data directly from you. They would not normally steal your Online ID directly from you. Instead, your account would be compromised. Phishing (or its subtype vishing) is a typical example. Another way is using non-trusted sites – torrents, key-generators, illegal game sites. When you try to download illegal software or connect (Figure 1) to an illegal game site, you provide information about your ID to a company or person you don't know. There is a big chance the company will sell this information. Of course, there are some not so obvious ways to have your Online ID compromised like sending e-mail to a wrong recipient, easily guessed answers to Secret questions, using unprotected connections.

Another example would be the usage of wireless router (Figure 2) without or with minimal protection. In this case a bad guy could use your unprotected device in order to connect to Internet. Then he could do virtually everything like scanning a server, attack and steal government information, post information to a blog. And the source of all these actions would be your public IP address. Once the attack is discovered, can you guess what will be the first steps of the police – they will first knock on your door.

What are the risks associated with losing your Online IS? They are quite the same as the risks associated with your normal ID: for example someone could use your Online ID to borrow money from a bank, acquire credit card with your name on it or he could buy something expensive on your behalf. But your ID could be also used to write in a blog on your behalf, to make a bomb treat, to threaten the president. And with the growing number of Internet sites and your registrations, the probability of your Online ID being lost is becoming bigger and bigger.

Why it is harder to prevent? Unlike your ID card, the Online ID is something you cannot touch. If you lose your ID card, you will most probably this notice immediately. But if your Online ID is stolen, you may even not notice at all. People are still not used to protect their Online IDs in at least the same way as they are used to protect their ID. They don't



Added: 2012-09-27 16:01:51 - By TorrentFreak

File-Sharing for Personal Use Declared Legal in Portugal

Hoping to curb the ever-increasing piracy figures reported the IP-addresses of 2,000 alleged file-sh week the Portuguese prosecutor came back with individuals connected to the IP-addresses. Accor to share copyrighted works for personal use, and identify a person.

Continue to the full article or read more torrent news.

Figure 1. *Login to untrusted site*

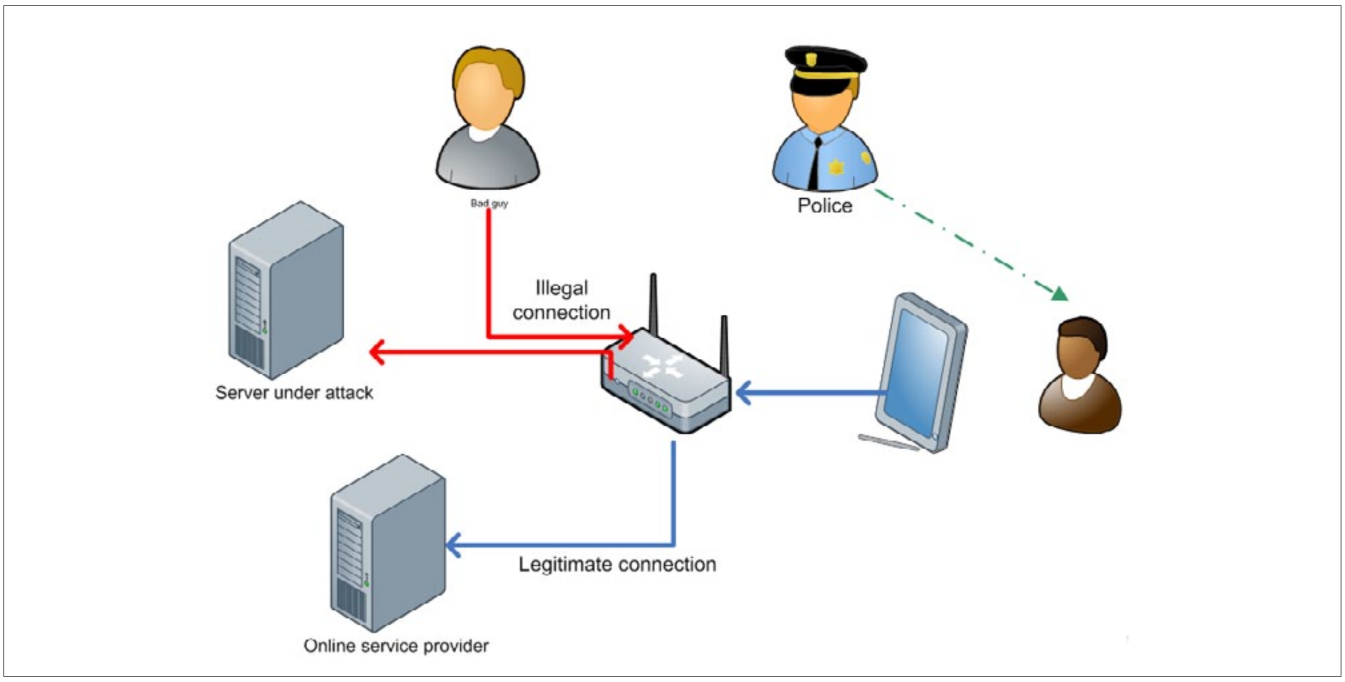


Figure 2. *Usage of unprotected wireless router*

realize the consequences of providing personal information to non-trusted parties. On top of everything, the protection of your Online ID doesn't depend entirely on yourselves. The online service providers are also heavily involved.

What measure could be taken in order to mitigate the risks? There are some simple measures, which will reduce considerably the probability of your Online ID being stolen:

- Provide your data only to trusted parties. Always try to do some basic investigation about the company you are registering with, especially if money transactions are required. Don't underestimate companies with simple registration process. Maybe it is just a free news site where you only read information. But in many cases these sites are not so well protected as the transactional sites. And in case we use the same password for both sites, losing the password becomes far bigger problem.
- Protect your online credentials, never disclose you passwords/PIN, be very careful about phishing sites. Not a single company will ask you for your password/PIN or other credential. They have the proper tools in place and they would reset your password instead of asking you. Only a bad guy would be interested in your credentials
- Close and if possible delete accounts which are not used anymore. Inactive accounts are normally not so well protected as the active ones.
- Check regularly your bank accounts and your statements for unusual transactions

On the Web

- *http://www.isaca.org/*
- *http://www.ftc.gov/bcp/edu/microsites/idtheft/consumers/defend.html*
- *http://www.idtheft.gov/*
- *http://www.idtheft.co.uk/*

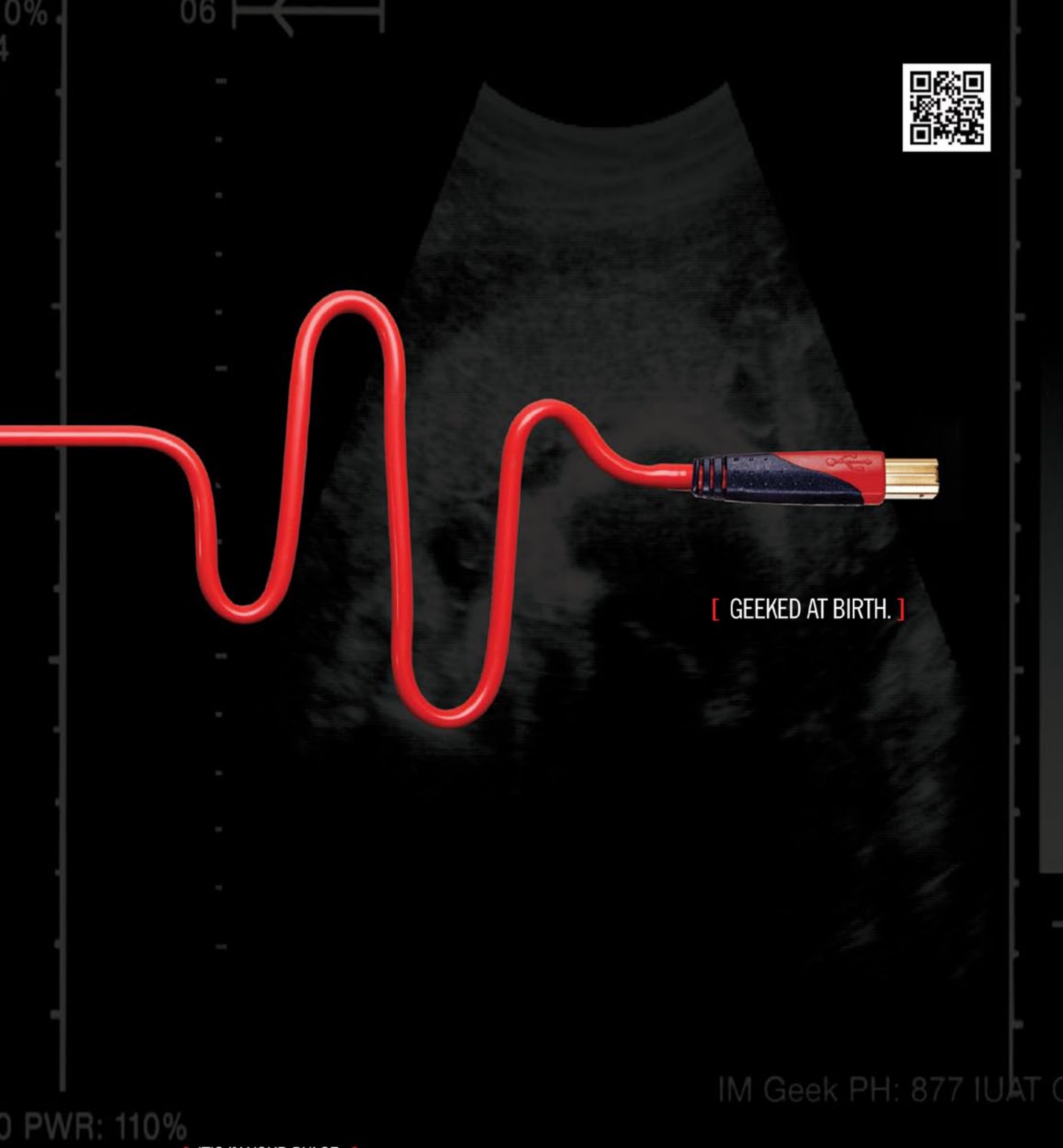
What to do in case you notice your ID has been used inappropriately? Besides all other actions, you should remember one more thing – the Online ID theft is a crime in the same way every other theft is. You should report it to the respective legal enforcement organizations.

Summary

Never underestimate the importance of your Online ID. As already mentioned, even the revealing your credentials for the most unimportant application could become a problem. Remember that the bigger is the number of the online services you use, the bigger is the risk of losing your online ID. On the other hand, the bigger is the number of the service providers, the bigger is the attacking vector for the bad guys.

DELYAN BOYCHEV

The author has been working as an Information security manager for a large financial institution since 2002. He was involved in the overall process of protecting the company's data as well as the customer's data, including personal identifiable information.



[IT'S IN YOUR PULSE.]

LEARN:
- Advancing Computer Science
- Artificial Life Programming
- Digital Media
- Digital Video
- Enterprise Software Development
- Game Art and Animation
- Game Design
- Game Programming
- Human-Computer Interaction
- Network Engineering
- Network Security
- Open Source Technologies
- Robotics and Embedded Systems
- Serious Game and Simulation
- Strategic Technology Development
- Technology Forensics
- Technology Product Design
- Technology Studies
- Virtual Modeling and Design
- Web and Social Media Technologies

University of Advancing Technology
UAT
Learn. Experience. Innovate.

You can talk the talk.
Can you walk the walk?

www.uat.edu > 877.UAT.GEEK

PLEASE SEE **WWW.UAT.EDU/FASTFACTS** FOR THE LATEST INFORMATION ABOUT DEGREE PROGRAM PERFORMANCE, PLACEMENT AND COSTS.

Identity Theft:

Stay Alert, Be Suspicious

Disclosure: This article is not intended to teach you what identity theft is or how much it is dangerous. That, you probably already know.

What you will learn…

- ID theft facts ands statistics
- ID threats and vulnerabilities
- The ways of coping and prevention.

What you should know…

- A comfortable feeling with this article does require light technical background.
- Information Security orientation can always be helpful.
- No pre-knowledge or working experience in cyber-crime is needed. Mostly important is the desire to be aware.

However, ask yourself if you take the proper precautions, especially when it limits your comfortability and your workflow. Yes, we are all smart people and some of us are professional IT personnel, but here is another question: How many people are you familiar with (could be family, friends or colleagues from work) who were victims of identity theft? Careful, this is a tricky question since the answer is: You cannot know.

What are we up against

Let's start at the end of a familiar story: An anonymous man contacts your bank representative, identifies himself with your name and ID number and after a brief small talk, asks to transfer a large amount of money from your investment portfolio, to a different account in another bank.

How could this happen? What went wrong here? How come the thief knew all these secret details and more disturbing is, what further personal information is in the thief's possession? This is a bad end to the story, but unfortunately its beginning was not so good either.

Identity theft also referred to as Identity Fraud or Impersonation, is a sophisticated crime carried out by the action of impersonating to someone else and making actions (mostly financial transactions) on his behalf. It is one of the most popular modern crimes in the 21st century, which victimize millions of people every year worldwide. Despite the intolerable ease with which one can search and find personal information about each and every one of us, taking simple precautions can significantly reduce our chances of being financially or personally damaged.

Back to our story: It is all about access. A successful ID Theft is based on accessibility to one's personal or classified information (ID, Credit Card & Social Security number or even details presented in your Children account on Social Networks). Once an intruder succeeds to find the desired security breach, it can be exploited to get valuable information as a part of the intruder's reconnaissance activity. Next, the intruder will use the victim's information to obtain permissions to databases and financial institutions (websites accounts, bank accounts, credit cards and more ...), withdraw funds from the victim's accounts and purchasing products / services, using the his name and his money.

Facts & Statistics

The following details have been carefully gathered at the past five years:

- on the *Federal Trade Commission's* (FTC) list of Top Consumer Complaints, identity theft has been the number one complaint for nine years running,

- in the USA, nearly one fifth of all complaints in 2010 were for identity theft,
- one out of four Americans was a victim of ID Theft is the past five years,
- although the number of reported cases of identity theft have slightly decreased in 2011 (relating to the year before) the cost of dealing with these cases has increased considerably, both in terms of time and financial resources,
- 80000 reports have been reported in the UK in year 2011, concerning identity theft,
- the average damage suffering victim of identity theft in the UK is approximately a thousand pounds.

According to the diagrams above (*source: Federal Trade Commission – Identity Theft Survey Report*), the vast majority of id theft reports, indicates the stolen information used for illegal activities on credit card accounts (it might suggest a way of prioritize the amount of resources IT personnel should invest for securing information assets). Note that in a significant percentage of cases, the crime is done while actually opening new accounts in the name of the victim.

Threats Types

There are three major types of identity theft known and common throughout the world. In this chapter I will describe a brief explanation for each type:

- financial Identity Theft,
- medical Identity Theft,
- criminal Identity Theft.

Financial Identity Theft

In this most common id threat, the intruder will use another's identity to obtain credit, goods and services.

Consequences from financial identity theft include:

- damaged credit,
- credit and debit card fraud,
- savings & Investment account fraud.

Medical Identity Theft

Medical identity theft occurs when someone uses another's personal information such as insurance information, to obtain medical services in order to make false claims for medical services or goods.

Consequences from medical identity theft include:

- false medical and pharmaceutical bills,
- false health insurance claims,
- denial different types of insurance claims or coverage.

Criminal Identity Theft

When a criminal fraudulently use the identity of the innocent victim during the commission of a crime, it is sometimes referred to as Criminal Identity Theft.

Consequences from criminal identity theft include:

- receiving a criminal record,
- obtaining an arrest warrant,
- imprisonment.

Weaknesses & Vulnerabilities Exploitation

Realization of the threats outlined in the previous chapter is possible by exploiting security breaches and weaknesses. These types of vulnerabilities do not belong only to the "electronic-world" of computing. In many cases, the realization of threats is possible by exploiting physical security weaknesses that exist in our daily habits & behavior.

In this chapter I will give an example of major security weaknesses (electronic & physical) that could lead to an identity theft.

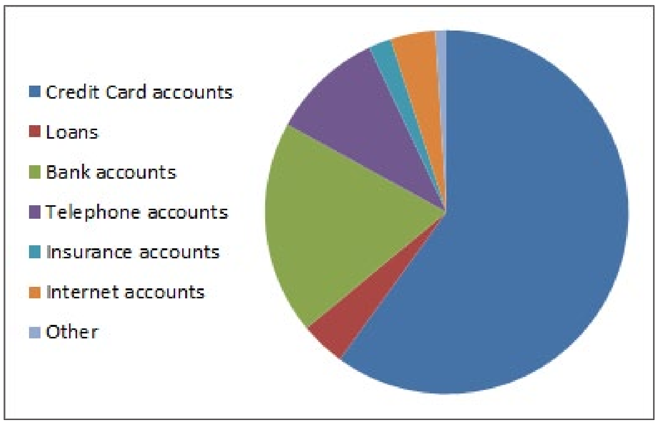


Figure 1. *How was stolen information used?*

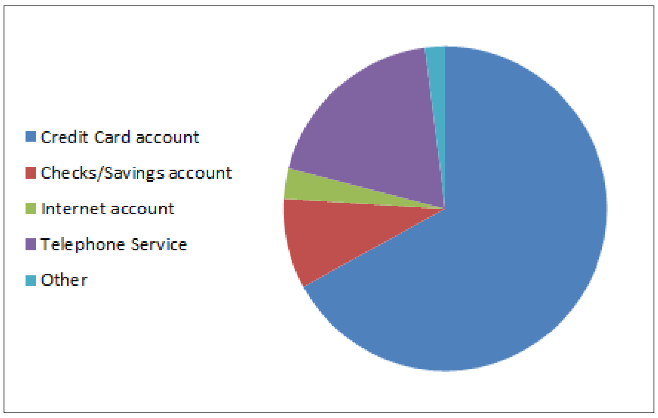


Figure 2. *Existing Accounts Misused*

Phishing
Phishing is a scam which links to fake site are being sent to the victims. These links are completely identical in content (coded via server-side languages & scripts such as HTML or JS) to the original site and is very similar to its URL address. The Hacker who created this forged website collects the user information of those who try to enter it.

Another method of phishing is a delivery of an innocent-looking email, bearing the details of the victim's bank. Sometimes, the message will report on security problems in the bank and the victim will be asked politely to go to a special site (which he sees as belonging to the bank itself) and enter his username, password & account information online. This method is similar to another theft method called "Social Engineering", which will be discussed later in this chapter (Figure 3).

In the picture above you can see that the URL does not address the original HSBC website (*www.expat.hsbc.com*). A user who types in his account details, actually sends his details to the hacker's fake website.

Pharming
Pharming is a hacker's attack intended to redirect a website's traffic to another bogus site. It can be conducted either by changing the hosts file on a victim's computer, or by exploitation of a vulnerability in DNS server software. Pharming requires unprotected access to target a computer, such as altering a customer's home computer, rather than a corporate business server.

Cookie &Session Theft
Session is being used to identify the user is connected. After signing-in the session-id is marked and accordingly we can identify the user each time the page is loading. If a Hacker finds the session ID he can easily visit the site by using it. A common way to get the session-id is steal the related cookie using Javascript code injection or XSS attacks.

The Figure 4 describes data from the cookie created following e-mail server connection. The session id is stored inside the field "roundcube_sessionid". This information could be used by hackers in order to get an access to the mail server, by identifying as the victim.

Malwares
Malicious software such as Viruses, Trojan horses and Spyware can be used in order to hack the victim's computer, sniff & scan for valuable personal data and then deliver it to the hackers remote machine.

Social Engineering
Social engineering is a term that describes a non-technical kind of intrusion that relies on human in-

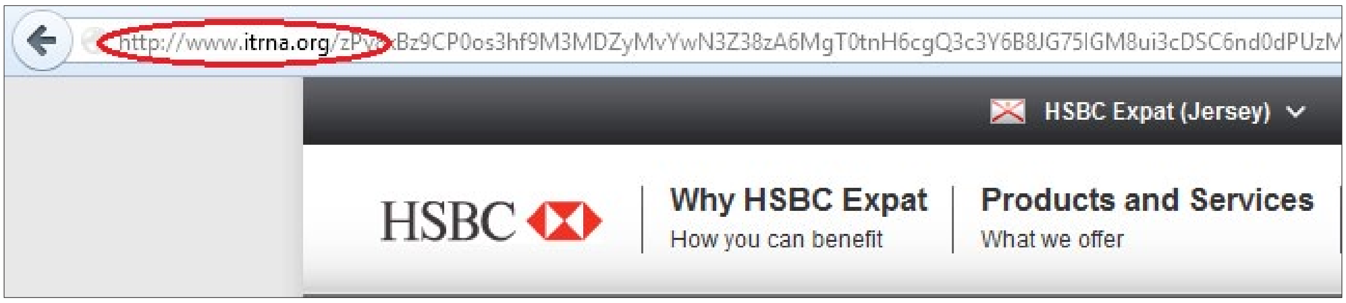
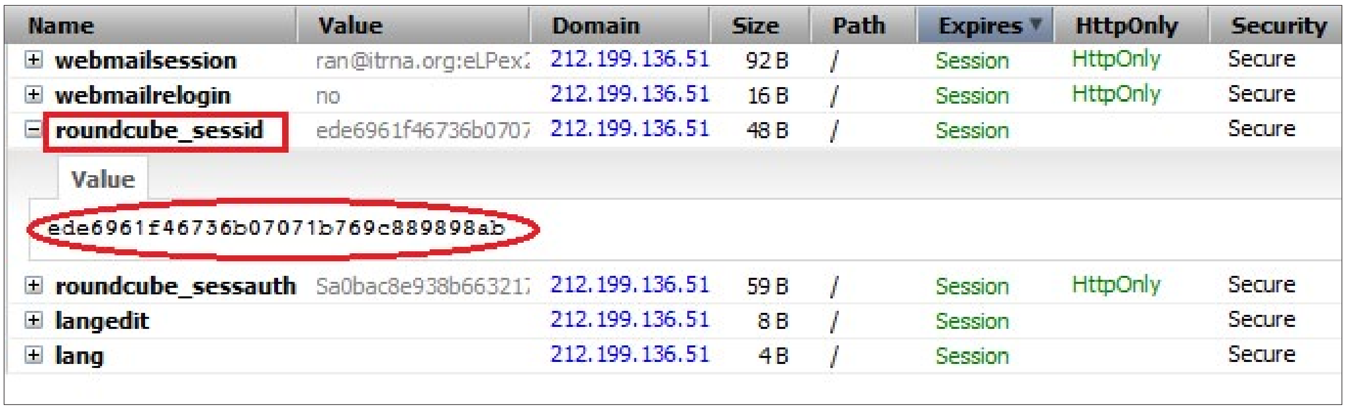

http://www.itrna.org/zPvxBz9CP0os3hf9M3MDZyMvYwN3Z38zA6MgT0tnH6cgQ3c3Y6B8JG75IGM8ui3cDSC6nd0dPUzM
HSBC Expat (Jersey)
HSBC
Why HSBC Expat
How you can benefit
Products and Services
What we offer

Figure 3. *An http URL, addressing for a fake HSBC website*



Name	Value	Domain	Size	Path	Expires	HttpOnly	Security
⊞ webmailsession	ran@itrna.org:eLPex	212.199.136.51	92 B	/	Session	HttpOnly	Secure
⊞ webmailrelogin	no	212.199.136.51	16 B	/	Session	HttpOnly	Secure
⊟ roundcube_sessid	ede6961f46736b0707	212.199.136.51	48 B	/	Session		Secure
Value							
ede6961f46736b07071b769c889898ab							
⊞ roundcube_sessauth	Sa0bac8e938b66321	212.199.136.51	59 B	/	Session	HttpOnly	Secure
⊞ langedit		212.199.136.51	8 B	/	Session		Secure
⊞ lang		212.199.136.51	4 B	/	Session		Secure

Figure 4. *32 digit session-id is stored in a cookie*

teraction and often involves tricking other people to break normal security procedures. This technique is used in many types of exploits.

Virus writers use social engineering tactics to persuade people to run malware email attachments, phishers use social engineering to convince people to enter sensitive information, and other scammers use social engineering into running software that is useless at best and dangerous at worst.

Social Networks

Let us use Facebook as an example: It begins with stealing virtual money on "Texas Hold'em" and can get to steal valuables assets in one way or another on "Mafia Wars". Facebook as a social network provides the ability for a programmer, to create applications in accordance with pre-determined rules. These programs can easily scan profiles and extract personal details such as: personal addresses, phone numbers, details about your family, your job, your plans for the future and of course, personal and intimate images.

Suspicious Phone Calls

Below are some "red flag" warnings that might suggest you have been a victim of identity theft:

- Calls from creditors or agencies demanding payment for items that you never bought or for accounts that you never opened,
- Calls from creditors about suspicious new accounts or a large volume of credit card activity,
- Calls from your Credit-Card company or your bank, asking for your password or user name.

Snooping around our House

Thief may watch you as you press some secret code or listen to you talking on the telephone providing some secret numbers. They may look through the garbage of our building searching for peaces of information such as forms, invoices, contracts and information that contains personal information such as phone number exact address registers and prescription drugs from pharmacies. Such information helps the thief to take over the accounts in and use your identity.

Preventive Actions & Coping Methods

Although all of the methods listed above are different and unique, the purpose of its realization is the same: Gathering information (reconnaissance) to carry out identity theft. Another similar thing is the way to deal with these weaknesses, meaning: The method of dealing with some weakness can certainly help dealing with a different one.

Following are some methods to deal with security breaches and minimize the chances of theft identifier.

Do not click on Hyperlinks within emails

Hyperlinks within emails could be cloaked, or hidden. The text you see as a hyperlink may not be where the hyperlink takes you. If you are unsure of the source of the email, you should not click on hyperlinks within emails that are apparently from a legitimate company. Instead, directly type in the URL in the Internet browser address bar, or call the company on a contact number previously verified or known to be genuine.

Use Anti-SPAM Filter Software

More than eighty percent of all emails sent is SPAM, with a majority fraudulent. Effective SPAM filters can reduce the number of fraudulent and malicious emails consumers are exposed to.

Use Different Passwords for Different Accounts

One successful password theft could gain the Hacker an access to many accounts you created before. Moreover, do not save passwords for websites and user accounts unsecured, use an encoding software which can be used to save the information secure or a portable hard drive that is not permanently attached to the computer.

Use Anti-Malware Programs

Programs such as Anti-Virus and Anti-Spywares are essential in order to search & destroy programs such as Viruses & Trojan horses that track and record your activity. Scan your computer regularly. Also, make sure the systems version and spyware signature files are current.

Update your Operating System & Browsers

Hackers are continually finding vulnerabilities in Operating Systems and Internet Browsers. Software vendors constantly publish Security-Patches, Software-Updates and fix vulnerabilities.

Harden the Social Networks Accounts

Here are some suggestions

- expose minimal details (remember the "Need to know basis"). Do not expose information that could be the answer to a secret question, or passwords,

Glossary

- cyber-crime (or computer-crime): The Department of Justice categorizes computer crime in three ways: 1) attacking computers of others (computer as a target) 2) using a computer to commit a crime (computer as a weapon) 3) using computers to store illegal or stolen information (computer as an accessory),
- dns servers: DNS servers are computers responsible for resolving Internet names into their real IP addresses,
- identity theft: Identity theft and identity fraud are terms used to refer to all types of crime in which someone wrongfully obtains and uses another person's confidential data in ways that involves fraud or deception typically for economic gain,
- reconnaissance: Any method of scanning and gathering information of an entity in order to profile the target organization or network or even his physical habitat for the efficient attack tactics,
- xss (or Cross-Site-Scripting): is a type of a security breach typically found in Web applications that enables attackers to inject client-side script into Web pages viewed by other users.

On the Web

- *http://www.identitytheft.info* – News & information of id theft trends & threats,
- *http://www.justice.gov/* – Great source for new Regulations & Clauses,
- *http://www.ic3.gov/default.aspx* – Getting updates & statistics of id thefts cases,
- *http://www.fraudwatchinternational.com/* – Technical info on phishing & fraud
- *http://www.fbi.gov/stats-services/publications/* – FBI's id threats publications.

- never enter your user name and password into third-party applications for any social network (such as Facebook). No Social Network will ask you for your username and password if you have already connected to its website.

Be Alert to your Surroundings

- when you need to transfer personal information over the telephone, do it while you are alone in a room or a place where you will not be able to be tapped,
- working with your Laptop in a coffee-shop? Take a look around, ensure no one is examine your work or any other data.

Be Aware of your Financial Status
Check your financial information regularly. Summary of your bank account and Recent Actions should be sent to you at least once a month, unless you specifically request otherwise.

Destroy sensitive documents before throwing them into the garbage
Before throwing it to trash, try to shred the documents into small pieces in order to prevent a reuse of sensitive information.

Be Alert & Aware of the Website you are visiting
Be careful when entering into a Fishy or an unfamiliar website (especially in areas such as porn or gambling). Before you provide confidential information, read a little about the website, try to find out some more details about the website owner and its members/visitors.

Keep your Wallet safe
Most of these cases lead to an identity theft. Criminals use the cards in your wallet/purse in order to steal your identity.

Summary

Identity theft is an annoying & dangerous phenomenon, especially the financial type. In my opinion, the issue of financial identity theft made headlines mainly because of the involvement of stakeholder and other interested parties: credit card companies, banks and insurance companies. In these cases it is not so hard to give a reasonable response and to impose security costs on the issuers of the payment instruments.

When I examine the issue of identity in a more abstract way, I cannot ignore the fact that identity theft is a part of a more comprehensive problem that could lead to Invasion of Privacy. As I mentioned earlier in the article, it is all about Access. Unauthorized access to personal information, whether done intentionally or accidentally, whether comprehensive or partial, violating the privacy of citizens and of course, equipped the Hacker with great tools for his crimes, using other identity.

RAN LEVI
The author has been working as an Information Security Consultant & Auditor in a variety of enterprises of all areas of industry, implementing security methods and techniques for the past eight years. The author is also involved in research & developing Bioinformatics algorithms and solutions for Computational Biology problems.

Learn ethical hacking › **Become a Pentester™**

- Get trained today through our exclusive 7-months hands-on course.
- Gain access to our complex LAB environment exploiting vulnerabilities across many platforms.
- Receive a trainer dedicated to you during the 7 months.
- 10 different hands-on engagements, 2 different certifications levels.

MONTH 1	› Vulnerability Assessment - level 1 › Vulnerability Assessment - level 2 › Vulnerability Assessment - level 3
MONTH 2	› Network Penetration Testing - level 1 › Network Penetration Testing - level 2
MONTH 3	› Network Penetration Testing - level 3
MONTH 4	› Web Application Penetration Testing - level 1 › Web Application Penetration Testing - level 2
MONTH 5	› Web Application Penetration Testing - level 3
MONTH 6	› Certification Exam 1 - Certified Cyber 51 Pentesting Professional - (CC51PP)
MONTH 7	› Certification Exam 2 - Certified Cyber 51 Pentesting Expert - (CC51PE)

~~Regular Price 1260 USD~~

Discounted Price **999 USD**

Sign Up Now

www.cyber51.com

Cyber 51

How to Secure your company's network with the Juniper Netscreen NS Series Security Appliance – Part 1

In last month's article we focused on the Cisco PIX Firewall, a cost effective yet still solid platform in the previously owned market. This month the focus is on a comparable unit from another top tier vendor that is also a great purchase in the enterprise resale market and still provides solid, fast efficient enterprise class stateful inspection at the perimeter with some advanced application layer features. The Juniper Netscreen.

What you will learn…

- How to cost effectively secure your company's network with enterprise class hardware
- How to operate with the Juniper Netscreen Security Appliance
- Differences and similarities between ScreenOS and PIX IOS

What you should know…

- Basic knowledge of network security
- General knowledge of network and network security equipment
- Understanding of IP subnetting and TCP and UDP services
- General understanding of NAT/PAT and Access Lists.

Comparable to the PIX 515 model and used in typical branch office, mid sized regional office deployments are the NS 25 NS 50 and NS 204 models which are the comparable model to the 515 series Cisco PIX and 5510 series of the ASA. These models are available via resellers often at incredible savings and still provide solid basic stateful inspection along with some advanced application layer capabilities. The capabilities between the two types of appliances are similar including stateful inspection, transparent layer 2 capability along with some application layer capabilities including acting as VPN tunnel endpoint or client endpoint. But there are some differences in the CLI between the two security operating systems and in how you approach the basic configuration steps.

This article will be presented in two parts so we can cover not only ScreenOS configuration and implementation of the Netscreen but how they relate to the Cisco PIX IOS and PIX/ASA platforms. In this first section we will take a look at the various features of the Netscreen, some basic configuration parameters and security parameters such as "policies" and how they relate to PIX IOS ACL's, Virtual Routers and their purpose as well as some of the various models limitations and features. In part two we'll take a more in-depth look at hardening the Netscreen particularly attack signatures and defensive measures (including a couple of really cool built in signatures not found on the PIX) and some standard branch and regional office deployment scenarios as well as take a closer look at some specific configuration variances and similarities to the PIX IOS.

Basic ScreenOS Differences from the PIX IOS

There are some differences between the ScreenOS and PIX IOS such as Policies as opposed to ACL's. Rather than using ACL's Screen OS uses policies to assign permissions to interfaces. Rather than NAT, ScreenOS uses MIPS\DIPS (*Mapped and Dynamic Mapped IP's*) but the end result is the same, a registered internet routable IANA address mapped to an internal RFC 1918 private address. We won't go through each and every syntactical difference between the PIX IOS and ScreenOS but we will review some of the more significant structural differences in the configurations between the two. We'll take a brief look at some of these differences in part 1 along with a general overview of the Netscreen and ScreenOS. In Part 2 we'll take a closer look at defense protections and capabilities, attack signatures and defenses, some basic deployment scenarios where these application layer defenses can be helpful and finally some basic steps to get your Netscreen up and running in a basic web application services deployment and how the configuration examples relate to similar configurations with the PIX IOS.

ScreenOS Zones

Perhaps the most obvious is the out of the box config. The PIX has gateway functionality out of the box. Interfaces are designated as internal and external (Inside, Outside) with security level assignments preassigned. Lower security level interfaces cannot pass traffic to higher level security interfaces without a specific rule (ACL) but a higher security interface can pass traffic to a lower security interface without a specific rule applied permitting that specific traffic. Rather than security levels the Netscreen uses Zones to accomplish security levels. Default zones are created and interfaces assigned out of the box. On an NS 50 for example, you'd have 3 zones that have interfaces assigned. Trust, Untrust and DMZ, the names being self explanatory.

Trust translates to the Inside interface on the PIX. Untrust to the Outside and of course DMZ to DMZ. However this does not mean traffic can by default go from one interface to another. Traffic between two interfaces assigned to different zones will not pass without a rule applied. Traffic between interfaces within the same zone however can pass without a rule, for any traffic to traverse any interface a specific rule must be applied.

VRs – Virtual Routers On the Netscreen

One of the somewhat unique features of the Netscreen in comparison to the PIX is the use of virtual routers which can be bound to zones and thus indirectly to interfaces. Virtual routers are referred to in the ScreenOS as "vr" and are bound to zones. Two vr's are created in the ScreenOS by default, one for Trust and one for Untrust, but they are not automatically assigned to the respective zones. You can create additional vr's as needed, such as for multiple routing protocols to multiple ISP's.

The primary purpose for the vr's are to conceal your internal route table from your external interface but they will conceal a routing table from any zone to any zone when those zones are placed in separate vr's. This can be helpful when utilizing routing protocols on the Netscreen. By placing the untrust zone in the untust-vr and the trust zone in the trust-vr you create two separate route tables separate from the other, concealing the internal routes to the untrust side of the appliance. This hides routing information from the internal network from the external interface which is particularly of help when routing protocols are implemented.

Naturally on some branch office implementations a single internal subnet might be deployed with the Netscreen acting as a gateway appliance to the ISP. In such implementations the advantages of separating the two are negligible since there's no actual routing going on the internal subnet. Just an interface to a private RFC 1918 subnet. So on most implementations of this type you can keep both default zones (trust – untrust) in the trust-vr for ease of use. On implementations where you decide to separate the zones via a virtual routing table you will need to remember to add routes between the zones otherwise you won't be able to traverse the zones.

To quickly view the configured virtual routers on your Netscreen use the following command;

```
netscreen-> get vr
```

The following is the output from an NS-50 running ScreenOS 5.4

```
* indicates default vrouter
A - AutoExport, R - RIP, O - OSPF, B - BGP, P - PIM

ID  Name         Vsys   Owner   Routes    MRoutes
                  Flags
  1 untrust-vr  Root   shared  0/max     0/max
* 2 trust-vr    Root   shared  9/max     0/max
```

The outputs fairly self explanatory however the one significant thing we learn from this output is this Netscreen is configured to only use the one virtual router (trust) for all interfaces. There are 9 routes in the route table comprised of routes from both the DMZ, untrust and trust zones. No other virtual routers have been implemented on this appliance. The other thing we see from this output is both the trust and untrust interfaces as well as any other interfaces configured on the appliance (in this example there is one additional interface acting as a DMZ) are all bound to the trust-vr. Thus if we examine the route tables we'll see that all routes appear in the trust-vr route table and no routes appear in the `untrust-vr` (Listing 1).
The output again is pretty straight forward (note the get route command. Same command on the PIX IOS except of course for the use of "show" commands as opposed to "get"). IP prefix (the part of the packet the route engine examines), the

Interface, Gateway, Metrics, etc. The Vs is the "virtual system", which is something we'll take a look at later on in part 2. The virtual system is however just what it sounds like. In this instance its "root". The interesting data here of course is again the lack of any routes in the untrust-vr, telling us that all active interfaces on this appliance are assigned to the trust-vr. We also see no OSPF, BGP, EBGP, (etc) routes in the table so it is not as important as if we were using these protocols. Another interesting thing we see here that we don't see in the PIX IOS, is the support for these routing protocols, another reason to love the Netscreen. The ability to do things like export BGP routes from the untrust-vr and propagate them into the trust-vr greatly expand the enterprise level functionality of the Netscreen.

Standard DMZ Architecture with the Netscreen & Hardware Options

One thing I like about the PIX 515 series is the ability to add up to 4 physical DMZ's via DMZ modules (either 1 or 4 port). The older smaller NS series Netscreen's like the NS 25, 50 and 204 come with fixed interfaces and no expansion slots. The larger models like the 5200 and 5400's are modular but you're spending in the thousands and tens of thousands depending on the unit.

Keeping with the "on a budget" approach for the mid sized branch or regional office looking for a solid stateful inspection perimeter appliance with application layer features the Netscreen NS series provides a fantastic bargain in the resale market that still provides solid stateful inspection including DOS, DDOS protection, VPN and Layer 2 capabilities and other advanced features. But it doesn't provide as many interfaces as the PIX 515 series with a 4 port DMZ expansion card. So if you have the money the SSG140 is a much better selection as it provides more interfaces and of course because it runs the much more desirable ScreenOS 6 and has many more application layer signatures, (200,000+) as well as other advantages, however the average price on these even in the resale market can run a few thousand dollars for a branch office size model (Figure 1).

A more cost effective option if you're on a strict budget is the End of Life Netscreen NS series units. Of course the NS series are end of life and will not provide the all the features of the newer

Listing 1. *Standard DMZ Architecture*

```
netscreen-> get route

IPv4 Dest-Routes for <untrust-vr> (0 entries)
--------------------------------------------------------------------------------
H: Host C: Connected S: Static A: Auto-Exported
I: Imported R: RIP P: Permanent D: Auto-Discovered
iB: IBGP eB: EBGP O: OSPF E1: OSPF external type 1
E2: OSPF external type 2


IPv4 Dest-Routes for <trust-vr> (9 entries)
--------------------------------------------------------------------------------
   ID          IP-Prefix      Interface         Gateway   P  Pref    Mtr     Vs
--------------------------------------------------------------------------------
*  10            0.0.0.0/0         eth3      10.200.1.1   S   20       1     Ro
    6         10.20.1.1/32         eth2         0.0.0.0   H    0       0     Ro
*   4       10.101.10.1/32         eth1         0.0.0.0   H    0       0     Ro
*   2        10.100.1.2/32         eth1         0.0.0.0   H    0       0     Ro
*   1        10.100.1.0/24         eth1         0.0.0.0   C    0       0     Ro
*   3       10.101.10.0/24         eth1         0.0.0.0   C    0       0     Ro
*   7        10.200.1.0/25         eth3         0.0.0.0   C    0       0     Ro
    5         10.20.1.0/24         eth2         0.0.0.0   C    0       0     Ro
*   8       10.200.1.11/32         eth3         0.0.0.0   H    0       0     Ro
```

SSG models but if you are trying to cut costs and just need an enterprise class firewall capable of protecting your gateway perimeter then the NS Series still offers basic protection with some cool extras. Online auctions usually let them go fairly cheap, between 50 and 100 bucks. But if you want one from a reseller then you'll stay pay a few hundred bucks normally, (or more depending on the reseller). The drawback of course (and why its so cheap) is they only run the older ScreenOS 5.x version whereas the SSG models run the newer more desirable (ScreenOS 6 provides a more feature rich environment and more functions and capabilities) version 6 with many more application layer features including many more attack signatures, AV signatures, etc. But the NS Series are still a solid perimeter device providing stateful inspection, DOS and DDOS and other attack protection as well as being cost effective enough for even the tightest of budgets.

The DMZ interface is usually going to be defined on Ethernet2, and functions similar to the PIX except again, no security levels. So instead of a default traffic pattern traversal being implemented you have to expressly define traffic permissions and direction to enable flow to and from the DMZ, usually to the external or untrust network in a typical branch office deployment. The DMZ will have a private RFC 1918 address assigned in most instances (unless you want to route to the DMZ subnet from the untrust zone) and a MIP assigned to it with a policy pointing to the MIP as the destination. When the DMZ host sends return egress traffic it will use the MIP address in the same way a host on the DMZ of the PIX uses a static NAT mapping. If you want hosts on the private trust subnet to access hosts on the DMZ subnet you merely need to establish the same type of policy you would for accessing the external untrust zone. We'll take a brief look at policies in the next few sections.

ScreenOS & Concurrent Sessions

Some networks of course can benefit from the SOHO models of the SSG (like the SSG 20 and Wireless models) but these are for small offices or home offices and again you'll be looking at the same issues if you have more than a few users that we saw on the smaller SOHO PIX models. With Cisco they're referred to as "concurrent connections" but on the Juniper appliance they're known as "sessions but the make up of them are similar. TCP or UDP connections equal a session, so one user with a few web pages, maybe an email client where the server's beyond the SSG, an FTP program or Database app, etc is going to use up a lot of sessions. I've seen 10 users use up as much as several thousand sessions.

The SSG 20 has a maximum session allocation of 8064 sessions which I personally wouldn't use in an office with any more than 5 or 6 users, although theoretically you could have around 20. But ideally you'll want to keep your load well below 85 percent on the firewall. In ScreenOS you can quickly

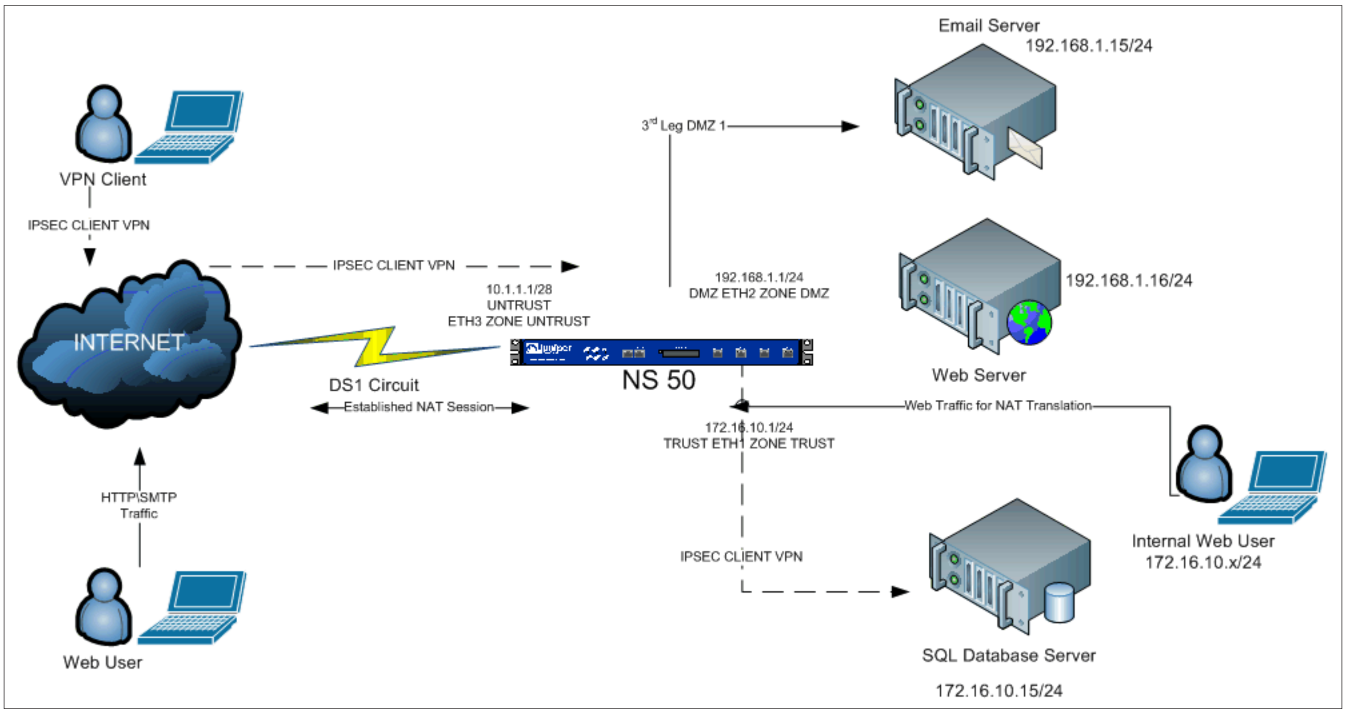


Figure 1. *Basic Inside, Outside Deployment with 1 DMZ*

determine the number of sessions available and if whether you're having dropped sessions using the following command.

```
netscreen-> get session info
```

The output seen below is the actual output from this command ran on an SSG 20 that was experiencing drops.

```
netscreen-> alloc 2588/max 8064, alloc failed 516,
                 mcast alloc 0, di alloc failed 0
total reserved 0, free sessions in shared pool
                 5476
```

We quickly see that while currently the allocated sessions are well below the 8064 total available, that the unit has dropped 516 sessions.

This means the appliance is not large enough to support the user base and its time to upgrade to a unit capable of more sessions.

You can also of course control the number of sessions available to users and services but on a smaller network you'll find its more trouble than its worth. Best just upgrade the device to the next size up. Once you take a quick look at all the services running on your network through a protocol analyzer you'll quickly see how just a handful of end users can quickly use up your available sessions (Figure 2).

An SSG 140 provides 48,000 sessions which is comparable to the NS 50 which has a max 64,000 sessions (Depending on the version of ScreenOS you are running. Older versions will have less sessions for the same model) which would be about the right size for 100 to 150 users comfortably. While you can of course have more users I've found that allocating 5000 sessions per 10 users is a good rule of thumb to follow, whether we're talking about Cisco or Juniper. You may find a different ratio works for you depending on your user base and their usage patterns but in general I think most will find that's a good ratio of users to session's if you like happy users and low maintenance. The important thing to remember here is just as it is with the PIX/ASA, users don't directly equate to sessions. Far from it. It is an indirect relationship and a dynamic one based on services and applications being offered and the usage patterns of your users.

ScreenOS Policies

Security Policies are the ScreenOS equivalent of PIX IOS ACL's (or conduits in older versions) that permit traffic to and from the various interfaces on the Netscreen appliance. Policies allow defined traffic to traverse interfaces in different zones. Policies are at the core very similar to Cisco ACL's.

A policy will define 4 actions;

- Permit
- Deny
- Reject (sends TCP reset)
- Tunnel (encrypts for VPN, either LT2P or IPSEC)

A policy is generally defined by 4 basic parameters.

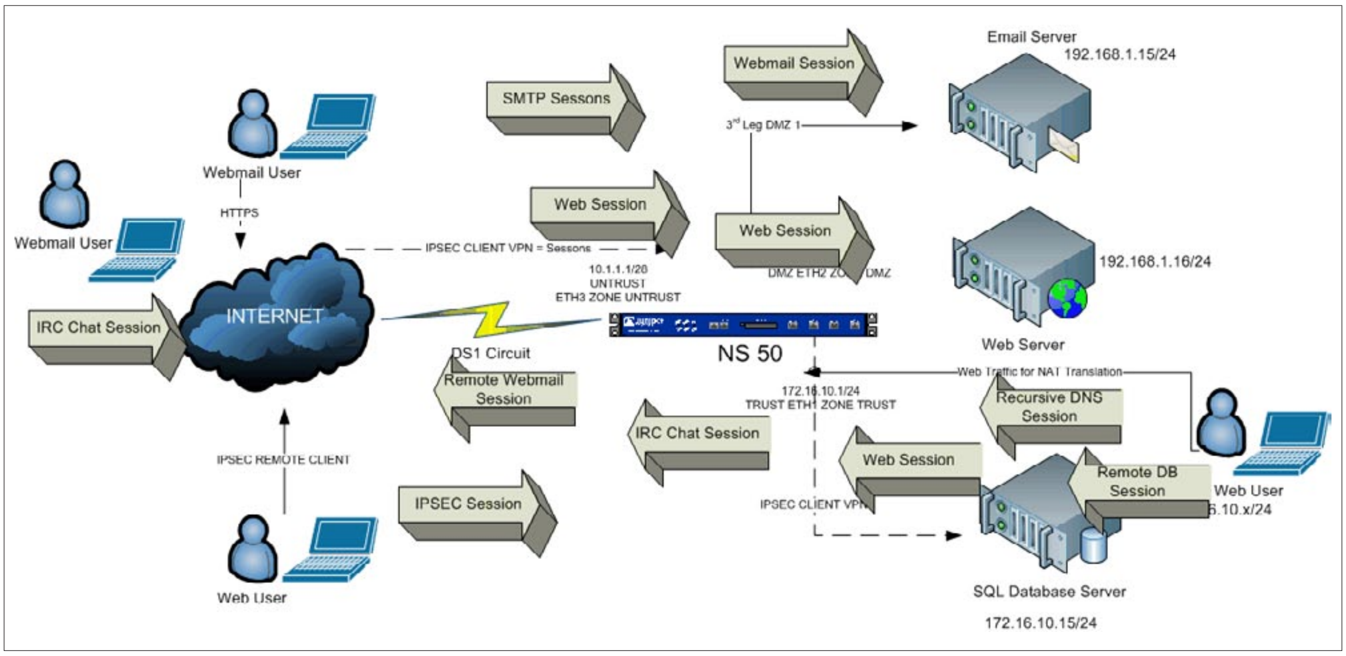


Figure 2. *1 Power user = Multiple Sessions*

- Source
- Destination
- Service
- Action

The source will be a network (wire) or host address. Same for the destination (Combined the source and destination define the direction). The Service will be the protocol. Together they make up the socket for the connection. The source and destination of course provide direction and the action defines whether to permit or deny (Or “encrypt” for VPN tunneling). These are the same general types of variables that define a standard Cisco ACL.

The difference between them lies more in their implementation order and syntax, a prime example being the default traffic traversal behavior between interfaces. In the PIX IOS trust traffic is automatically permitted to the untrusted interface once the interfaces are brought up and basic NAT or PAT rules applied. This is due to varying security levels being assigned to the interfaces. The higher the security levels have access to the lower security levels. Lower security levels need ACL’s to access the higher security levels. But in ScreenOS interfaces are assigned to security zones but these zones do not have security levels assigned to them by default. Instead total lockout is in effect on any interfaces in different security zones until a policy is assigned. Thus really there is no “outside, inside, DMZ”, etc. The default names are out of convenience and not functional. Without an accompanying security policy defined, no traffic from the “Trust” network can traverse to the “Untrust”. In other words unlike the PIX, the outside isn’t automatically outside nor is the inside automatically inside. You have to make them that way. You can’t just bring up the interfaces and apply PAT and expect to be surfing the web with the Netscreen. You will need policies defined first.

An example of an outbound policy permitting all traffic from the trust to the dmz zone would look like this.

```
Netscreen->set policy id 1 from “Trust” to
“Untrust” “Any” “Any” “ANY” permit log
Netscreen->set policy id 1
```

The same sort of policy is used to permit traffic to the DMZ zone or for any zone. A common issue with first time Netscreen users who are used to the PIX or other gateway type appliances is to forget to permit traffic in both directions that they want it to travel. Of course you can be much more specific and define multiple services and parameters.

The Syntactical Structure of Security Policies compared to PIX/ASA IOS ACLs

ScreenOS security polices are at the end of the day quite similar to PIX IOS Access Lists, and like ACL’s are tied to an interface and define a state, either permit or deny. However unlike Cisco ACL’s they also define an additional state “encrypt” which is used for VPN encryption. It differs from the application of the Cisco ACL used for VPN encryption by defining encryption in the state whereas PIX IOS simply defines the ACL. In PIX IOS the ACL is not applied to the interface but instead simply defined, then referenced in the Crypto map. But the differences don’t stop there at least syntactically. Policies “look” a lot different on the surface from ACL’s but they accomplish pretty much the same things when dealing with basic in and out firewall functions.

Let’s take a quick look at a PIX IOS ACL compared to a ScreenOS policy. We’ll define a simple basic inbound ACL to provide SMTP traffic to a standard mail server and compare it to a ScreenOS policy that defines the same access. For the purposes of this article we’ll pretend that 192.168.1.1 (In actuality 10.1.1.5 is an RFC 1918 address however for the purposes of the article we will pretend it’s a registered address) is the outside (registered IANA) address and that the internal (RFC 1918) addresses sit on the 10.1.1.0/24 network.

PIX IOS

```
PIX# access-list 101 extended permit tcp any host
                 10.1.1.5 eq smtp
access-group 101 in interface outside
```

ScreenOS

```
netscreen->set policy id 3 from “Untrust” to
“Trust” “Any” “MIP(10.1.1.5)” “SMTP” permit log
netscreen->set policy id 3 application “SMTP”
netscreen->set policy id 3
```

The ACL is pretty straight forward as we can see. The ACL is defined first, then the state (permit in this instance), then the protocol (TCP), the source (any) and the destination (10.1.1.5) and finally the actual protocol (service as Juniper refers to it) is defined, in this case SMTP (SMTP defines the standard TCP port of 25. If another port is in use by your mail server then the actual port number would be stated in place of the protocol). The Screen OS Policy looks a bit different but accomplishes the same thing. First we see it defines the policy and sets an "ID" to reference it, (3 in this example, similar to the ACL id of 101). Then it defines the source of the traffic (Untrust zone), the destination (Trust zone), the source address (any) and the actual destination address (MIP(MIP defines the mapped IP or NAT'd address in use, in most instances on the untrust zone mapped to a private RFC 1918 address in the trust or dmz zones) (10.1.1.5). We also see that we must specify the service and bind the service application (SMTP) to the policy;

```
"netscreen->set policy id 3 application "SMTP"
```

ScreenOS comes with many services predefined (port numbers already mapped). To see the predefined services already installed on your unit use the following command.

```
netscreen-> get service pre-defined
```

The following output is from a live Netscreen NS50 running ScreenOS 5.4 (Listing 2).

The output is truncated for brevity but we see there are a total of 136 predefined services on this unit. And we must define the policy ID. These are

Listing 2. *Live Netscreen NS50 running ScreenOS 5.4*

```
netscreen-> get service pre-defined

Name                 Proto           Port Group       Timeout(min)     Flag
ANY                            0     0/65535 other         30          Pre-defined
AOL                            6     5190/5194 remote      30          Pre-defined
BGP                            6     179 other             30          Pre-defined
CHARGEN                       17     19 other               1          Pre-defined
DHCP-Relay                    17     67 info seeking        1          Pre-defined
DISCARD                       17      9 other               1          Pre-defined
DNS                           17     53 info seeking        1          Pre-defined
ECHO                          17      7 other               1          Pre-defined
FINGER                         6     79 info seeking       30          Pre-defined
FTP                            6     21 remote             30          Pre-defined
FTP-Get                        6     21 remote             30          Pre-defined
FTP-Put                        6     21 remote             30          Pre-defined
GNUTELLA                      17     6346/6347 remote       1          Pre-defined
GOPHER                         6     70 info seeking       30          Pre-defined
GTP                           17     2123 remote            3          Pre-defined
H.323                          6     1720 remote           30          Pre-defined
HTTP                           6     80 info seeking        5          Pre-defined
HTTP-EXT                       6     7001 info seeking      5          Pre-defined
HTTPS                          6     443 security          30          Pre-defined
ICMP Address Mask                    0/65535 other         30          Pre-defined
ICMP Dest Unreachable          1     0/65535 other         30          Pre-defined

ICMP Address Mask              1     0/65535 other         30          Pre-defined
ICMP Dest Unreachable          1     0/65535 other         30          Pre-defined
ICMP Fragment Needed           1     0/65535 other         30          Pre-defined
ICMP Fragment Reassembly       1     0/65535 other         30          Pre-defined
ICMP Host Unreachable          1     0/65535 other         30          Pre-defined
ICMP Parameter Problem         1     0/65535 other         30          Pre-defined
ICMP Port Unreachable          1     0/65535 other         30          Pre-defined
ICMP Protocol Unreach          1     0/65535 other         30          Pre-defined
ICMP Redirect                  1     0/65535 other         30          Pre-defined
Total number of pre-defined services shown: 136
```

done as separate commands but are part of the defined policy.

To set a simple rule permitting all internal traffic from the inside to the outside on the Netscreen we would set a policy statement resembling this one (although the ID number can vary)

```
Netscreen->set policy id 1 from "Trust" to
"Untrust" "Any" "Any" "ANY" permit log
Netscreen->set policy id 1
```

Static NAT & MIPS

NAT and PAT functions including one to one static NAT translations, Dynamic NAT Pools and Port Address Translation aren't much different than they are in the PIX IOS except for the syntax. They look a bit different but accomplish the same thing so we wont' spend a lot of time on them in this section. Basic NAT is basic NAT. Later we will look at other features of NAT but in this introduction we'll just look at a typical one to one translation, in this instance permitting traffic from the external interface to a server on the inside. In PIX IOS we defines static one to one NAT translations via the "static inside,outside" command. For ScreenOS we'll use the "MIP" (mapped IP) command.

PIX IOS

```
PIXFIREWALL(config) # static (inside,outside)
10.1.1.5 192.168.1.15 netmask 255.255.255.255
```

This maps a static IP to the internal IP of the mail server sitting on the DMZ, note the 32 bit mask identifying the host, and not the actual subnet mask of the wire.

ScreenOS

```
NETSCREEN-> set interface "ethernet3" mip
10.1.1.5 host 192.168.1.15 netmask
                255.255.255.255 vr "trust-vr"
```

Like the PIX IOS the outside IP is mapped first to the inbound IP, but in this case rather than binding the internal IP to the interface on which it sits it binds the entire process on the outside (ethernet3) interface to the mapping. But it does map it to the virtual router, in this case the trust-vr (remember vr's are used to isolate route tables). Similar to the PIX it uses the 32 bit network mask to define the host.

Syslogging

One keyword shared with the PIX/ASA ACL's is the "log" command at the end, which instructs the appliance to log the when the action defined is met (usually a deny). Identifying the Syslog server is pretty straight forward as well. Like the PIX IOS you define the server IP, port, protocol (TCP or UDP, default is the standard UDP 514), here we set it to TCP for a better guarantee of delivery (Listing 3).

Vulnerabilities in ScreenOS 5.x

Like the PIX there are some documented vulnerabilities with ScreenOS 5 (and before) however again like the PIX these are able of being mitigated by either upgrading the OS or via work-arounds or configuration options (and omissions).

Only a handful of vulnerabilities are identified by CVE and all of those are not pertinent to versions of ScreenOS above 5.0, therefore we won't spend a lot of time on them other than to look at a few and how they can be easily mitigated or in most cases negated.

Cross Site Scripting (XSS) Vulnerability (CVE-2008-6096)

Prior to ScreenOS 5.4 (r10) ScreenOS is vulnerable to a cross site scripting attack which allows attackers to inject arbitrary script or HTML into the Username field in both the telnet and web interface login.

Listing 3. *Netscreen*

```
Netscreen->set syslog config 192168.1.10 port 1000
Netscreen->set syslog config 192.168.1.10 log all
Netscreen->set syslog config 192.168.1.10 facilities local0 local0
Netscreen->set syslog config 192.168.1.1.10 transport tcp
Netscreen->set syslog src-interface ethernet1/1 - Sets the interface used to reach the
                syslog server
Netscreen->set syslog enable
```

FIX

The solution to this vulnerability is either upgrade to release 10, or simply don't open HTTP/HTTPS access or Telnet access to the Untrusted side of your Netscreen, which by the way is a no brainer anyway. Opening telnet or http management to your Netscreen from the outside is just a bad way to fly. It is always advised to use a VPN or RAS connection to enter the network, and then access your firewall from either your client session or a jump host. Of course you'll also want to restrict access on the internal trusted network to your systems using a trust side policy permitting access to the appliance only from trusted hosts. This negates the vulnerability.

Behavioral discrepancy information leak

The Behavioral Discrepancy Information Leak documented on CVE impacts the appliance when using IKE with pre-shared key authentication. It allows an attacker to enumerate valid usernames using an IKE Aggressive Mode packet. This generates a response if the username is valid but does not respond when the username is invalid helping an attacker identify valid user names to attempt to exploit.

FIX

The solution is simple. The vulnerability only impacts version 5.2 and below so again getting a unit with a valid build negates the issue. Of course this doesn't give an attacker access to the device or the VPN, it merely gives them valid user names to attempt to exploit but to fully mitigate the issue simply get a model running 5.4 or upgrade yours.

Unknown Vulnerability in ScreenOS (CVE-2004-1446)

This is the one that many reference with the older Netscreens as it is in the dreaded "unknown" category. It is a DoS attack in that it allows remote attackers to cause a reboot or hang using a crafted SSH v1 packet.

FIX

Usually you probably don't want to permit SSH direct to your Netscreen from the untrust zone. Even though it's encrypted it is preferable to have a different method of accessing the internal network then use either a jump host or a VPN NAT'd address to ssh into the appliance. Of course this is not always possible and it can often be necessary to SSH into the appliance and that's fine as again the issue is limited to version 5.0 and below (*http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2004-1446*) so quick fix is to upgrade to ScreenOS 5.4.

There are other documented vulnerabilities on ScreenOS 5 but most are either at 5.0 or below and therefore are not an issue unless you don't have access to an upgrade. Of course the easy fix is to pick up a unit with 5.4 installed. Preferably you'll want 5.4r10, however any 5.4 version is fine since the few documented vulnerabilities are easily mitigated or negated.

This concludes part 1 of our brief look at the Netscreen and ScreenOS and "some" of its features (there are many) and how it compares to the Cisco PIX and PIX IOS. In part two we'll take a look at hardening the Netscreen, attack signatures (including a couple of really cool built in signatures) and defense protections and capabilities, some basic deployment scenarios where these application layer protections can be useful and finally some basic steps to get your Netscreen up and running in a basic web application services deployment and of course how the configuration examples compare with similar configurations both in implementation and syntax, with the PIX IOS.

CHRIS WEBER

Chris Weber is a freelance Network Security Consultant with more than 15 years in the field of network security, analysis and design and has consulted on network security issues and incident response for multiple commercial organizations including fortune 500 and fortune 100 firms as well as US Federal Government organizations. Mr. Weber has held multiple industry certifications throughout his career including Cisco, Microsoft and others, and is currently an authorized Juniper Consulting Partner. Currently Mr. Weber consults via his own freelance consulting firm "Layer 9" located in Northern Virginia and can be reached at cw@layer9security.com.

Lint Center
for National Security Studies, Inc.™
EMPOWER, ENHANCE, ENABLE

Need a scholarship?

White hats, Ninjas, Grinders, and Engineers – listen up!

The Lint Center for National Security Studies awards merit-based scholarships semi-annually in both July and January. A streamlined, web-based application form is available on our main portal. Undergraduate and post-graduate students pursuing technical degrees in computer security, computer science, diplomacy, and linguistics are encouraged.

LintCenter.org

About the Lint Center: The Lint Center for National Security Studies in the United States is a Veteran and Minority directed, all-volunteer 501(c)(3) non-profit organization, dedicated to fostering the educational development of the next generation of the National Security and Intelligence communities by providing passionate individuals with scholarship opportunities and mentorship from experienced National Security personnel.

About the Lint Center's Mentoring Program:
In addition to the scholarship award, winners will acquire an experienced security practitioner-mentor. With over 150 mentors, the Lint Center is well positioned to match emerging leaders with practitioners to streamline the learning curve.

Check out our blog: LintCenter.info
Follow us on Twitter: @LintCenter
Become a fan: facebook.com/LintCenter

EMPOWER, ENHANCE, ENABLE…
(*Script Kiddies need not apply*)

Reading Between the Lines

How to quickly obtain what you are looking for when reverse engineering assembly code

Much like pondering time travel and staring into the Matrix, reverse engineering assembly code can be an overwhelming task. Many novice reversers get lost in the ocean of instructions and can take weeks to identify a simple print statement. However, there are shortcuts one could take to make the process less daunting and more efficient. By employing these methods, even the freshest RE stands a chance of defeating the monsters within the code.

What you will learn…

- How to make sure you are looking at real code and not garbage
- How to view the code in a way that is pleasing to the brain
- How to rename subroutines so they are easy to spot
- How to leave breadcrumbs in the code by making comments
- How to work backwards by 'finding the cheese first'
- How to make your map complete by forcing the code to work for you

What you should know…

- Have a firm understanding of operating system fundamentals
- Have a working knowledge of high level programming (C/C++)
- Be familiar with x86 Assembly
- Be familiar with the concepts, methods and tools of reverse engineering

An aspiring reverse engineer goes through an immense learning regiment, which includes learning about the inner workings of an operating system, mastering an arsenal of new and complicated tools and learning an array of different programming languages, the most important of which being assembly code.

Assembly is inherently easy to learn, there are only a few instructions that make up the dialed down version of a higher level programming language. However, even after arming themselves with all the tools required to reverse engineer a program, they are bombarded with their first look at disassembled code, a waterfall of PUSH's and POP's, MOV's and RETN's, the garbled, uncommented and (most of the time) not-completely intact nightmare that is RE.

The solution

If you work in a setting where you are allowed to take 3 months to tear apart a program, good for you, enjoy dreaming about the code! For everyone else who usually works on a deadline or does not have the time to examine every line of code, there is a better way. Just like when trying to write a report on a 500-page book in an hour, an RE needs to use a few parsing tricks in order to understand the flow and function of the program. These tricks seem simple enough in their execution but it is only with practice that they will become second nature and therefore most effective. The goal is to take shortcuts to find out parts of what you want to know, to understand the flow of the code and predict what the program might be doing next, which will allow you to skip the boring redundant instructions put in by the compiler and get straight to the good stuff.

Tools

In this article, I will be talking about two tools in particular, they are my favorite and I always use them whenever I need to do some RE:

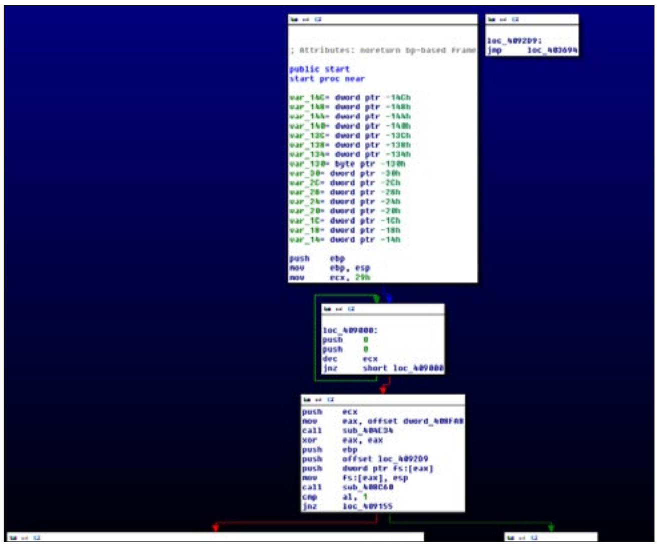
- OllyDBG – My Favorite Debugger
- IDA Pro – My Favorite Disassembler

In addition, here are some tools that I will be mentioning in this Article:

- PEID – Awesome binary packer identifier
- QUnpack – Awesome unpacking application
- GUNPacker – Another awesome unpacking application
- MSDN Library – The Microsoft Developers Network Library, you can install a local version that comes with Visual Studio or you can just search on the Web and find the web version, either way it is incredibly useful and you should definitely have access to it.

Note

This article will also only focus on reverse engineering Intel x86 Assembly, it is the most common type of assembly you are likely to run into when reversing anything written in C/C++.



```
; Attributes: noreturn bp-based frame

public start
start proc near

var_14C= dword ptr -14Ch
var_148= dword ptr -148h
var_144= dword ptr -144h
var_140= dword ptr -140h
var_13C= dword ptr -13Ch
var_138= dword ptr -138h
var_134= dword ptr -134h
var_130= byte ptr -130h
var_30= dword ptr -30h
var_2C= dword ptr -2Ch
var_28= dword ptr -28h
var_24= dword ptr -24h
var_20= dword ptr -20h
var_1C= dword ptr -1Ch
var_18= dword ptr -18h
var_14= dword ptr -14h

push    ebp
mov     ebp, esp
mov     ecx, 29h
```

```
loc_4092D9:
jmp     loc_403694
```

```
loc_409000:
push    0
push    0
dec     ecx
jnz     short loc_409000
```

```
push    ecx
mov     eax, offset dword_408FA8
call    sub_404E34
xor     eax, eax
push    ebp
push    offset loc_4092D9
push    dword ptr fs:[eax]
mov     fs:[eax], esp
call    sub_408C60
cmp     al, 1
jnz     loc_409155
```

Figure 1. *IDA Pro Function Graph View*

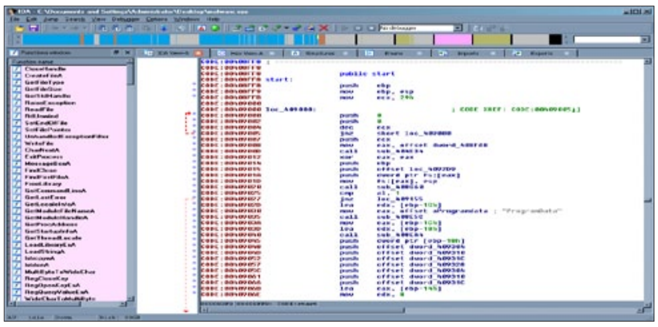
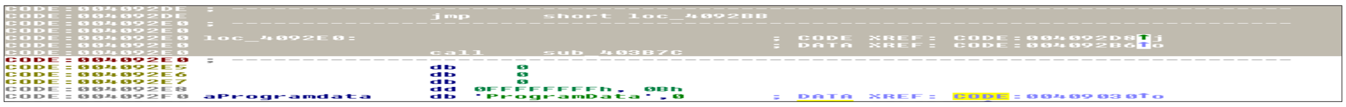


Figure 2. *An example of IDA Pro not recognizing a function*



```
CODE:004092DE ; ---------------------------------------------------------------------------
CODE:004092DE                 jmp     short loc_4092BB
CODE:004092E0 ; ---------------------------------------------------------------------------
CODE:004092E0
CODE:004092E0 loc_4092E0:                             ; CODE XREF: CODE:004092D8↑j
CODE:004092E0                                         ; DATA XREF: CODE:004092B6↑o
CODE:004092E0                 call    sub_403B7C
CODE:004092E0 ; ---------------------------------------------------------------------------
CODE:004092E5                 db    0
CODE:004092E6                 db    0
CODE:004092E7                 db    0
CODE:004092E8                 dd 0FFFFFFFFh, 0Bh
CODE:004092F0 aProgramdata    db 'ProgramData',0      ; DATA XREF: CODE:00409030↑o
```

Figure 3. *The end of a function without a RETN*

Pre-Game Stretches

Before you attempt doing any kind of reversing, you need to make sure you are going to be looking at real code, not garbage. To do this, I usually do a few things:

- Throw the binary into PEID, it will tell you if the binary is packed.
  - If the binary IS packed, unpack it when one of the unpack applications listed in the Tools section.
- If PEID says everything is okay, try opening the binary in OllyDbg, you need to make sure that the file is valid.
  - If OllyDbg cannot open the file, it might be corrupt or just not executable.
- Finally, open the file in IDA Pro and check to see if everything is loaded properly, you should see at least one export function, maybe some imports and sections of real code.
  - If everything in IDA Pro is garbled or all you have is a tiny function and no imports, you might have a custom packed binary.
  - If you have a custom-packed binary, there is one thing to remember: *every packed binary must be unpacked before it can run*. Therefore, a good way to unpack something unique would be to let it unpack itself.

To sum up, the only time you should be attempting to reverse engineer a packed binary in IDA Pro is if you are trying to figure out how to unpack it; NEVER try to parse through packed code, it will get you nowhere and you will waste a lot of time.

Creating order from chaos

One of the greatest things about IDA Pro is its ability to automatically create a flow-chart type graph to view the assembly code (Figure 1). It takes a function and puts it into a box with all the assembly inside of it, and then when it encounters a conditional jump, IDA will create new boxes for the results of that jump, one for true and one for false. The purpose of this functionality is so that humans can easily determine the flow of a function instead of having to manually do it ourselves and waste brain power by keeping a mental note of whether the "Jump if Equal" statement is true and do we need to jump to the code at `0x00401123`.

While that is just one of the many amazing things IDA Pro can do, it does not make IDA as smart as a human. You have to keep in mind that it does not always recognize a function when it sees one. You might load up a binary and find the 'Start' function is in line-by-line assembly mode rather than the graph. Figure 2 shows you exactly what this looks like. You can hit the space bar a hundred times and IDA will refuse to make it look nice for you. In this case, you need to know how to make IDA realize that a function is there.

IDA has a command called "Create Function" which it allows you to execute if you select a portion of code from the start of it to the end. All you need to do is highlight the code using your cursor. After that, scroll down with your trackball (if you have one) to where you find either no-code or another function starting, we hope that there is a `RETN` instruction to neatly signal the end of the function but sometimes there isn't and you just have to deal with that. Figure 3 shows an example of that disappointing situation.

When you find the end of the function, right click on the selected area and select "Create Function." The resulting graph will represent the function you just selected. This allows you to worry less about the flow of the application and more about the purpose of the function (Figure 4).

Keeping Track

If we are going to be hopping around the code constantly looking for new and interesting functionality, we need to be able to keep track of where we are for our own notes as well as being able to navigate the same straits later on when using OllyDbg. Unfortunately, when we switch to graph mode, we lose the ability to see the address of the code we are looking at. Beyond hitting the space bar every time we need to check our location, we can just modify some of IDA Pro's general options. In order to do this, make sure you are in graph mode and click on the "Options" menu then click "General" (Figure 5).

This opens the IDA options interface where you can select the "Line Prefix" option (Figure 6); this will show us the location of each line of code on the graph display. Another nifty option you could select is "Auto Comments" which gives very general and

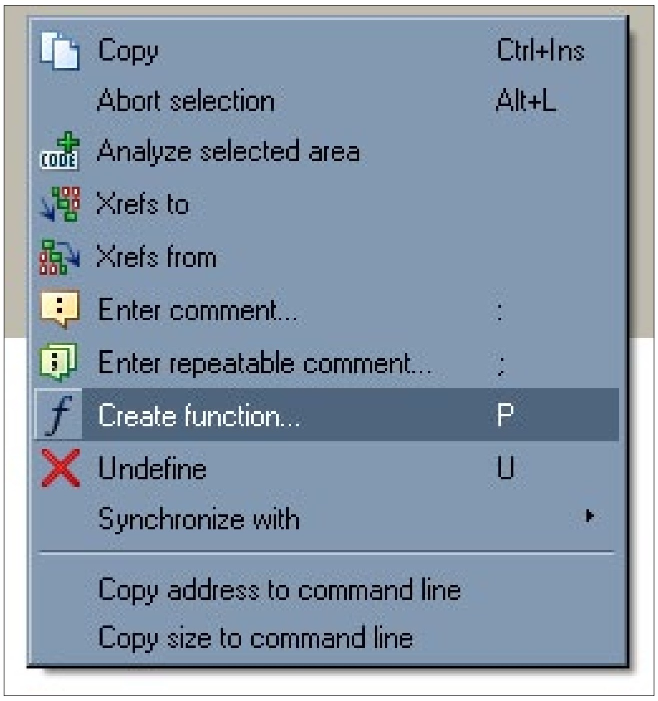
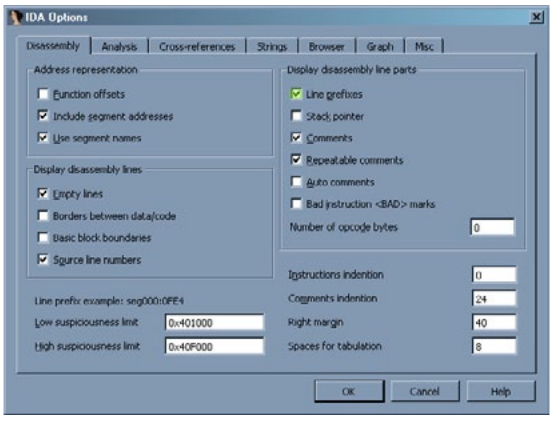


Figure 4. *'Create Function' Option*


IDA Options

Disassembly | Analysis | Cross-references | Strings | Browser | Graph | Misc

Address representation
- [ ] Function offsets
- [✓] Include segment addresses
- [✓] Use segment names

Display disassembly lines
- [✓] Empty lines
- [ ] Borders between data/code
- [ ] Basic block boundaries
- [✓] Source line numbers

Display disassembly line parts
- [✓] Line prefixes
- [ ] Stack pointer
- [✓] Comments
- [✓] Repeatable comments
- [ ] Auto comments
- [ ] Bad instruction <BAD> marks

Number of opcode bytes: 0

Line prefix example: seg000:0FE4
Low suspiciousness limit: 0x401000
High suspiciousness limit: 0x40F000

Instructions indention: 0
Comments indention: 24
Right margin: 40
Spaces for tabulation: 8

OK | Cancel | Help

Figure 6. *IDA Options interface*

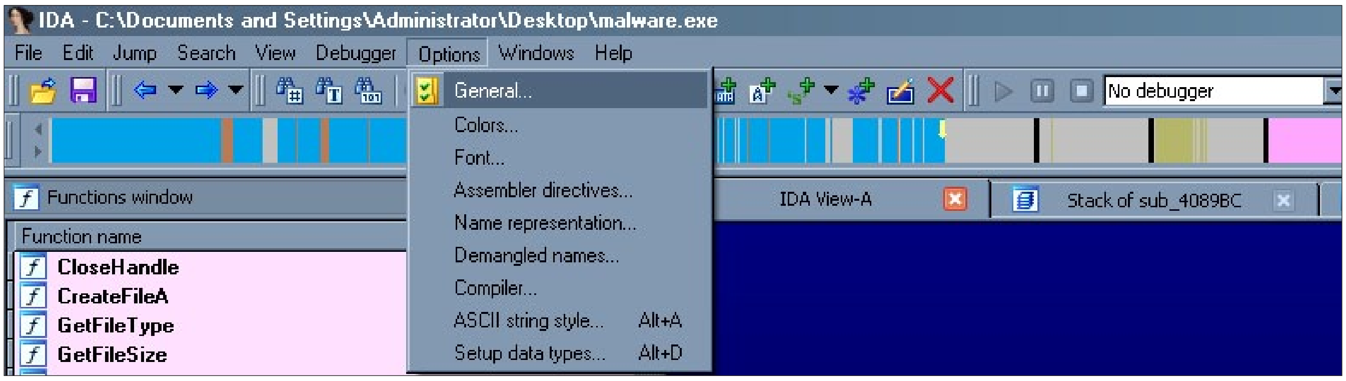

IDA - C:\Documents and Settings\Administrator\Desktop\malware.exe

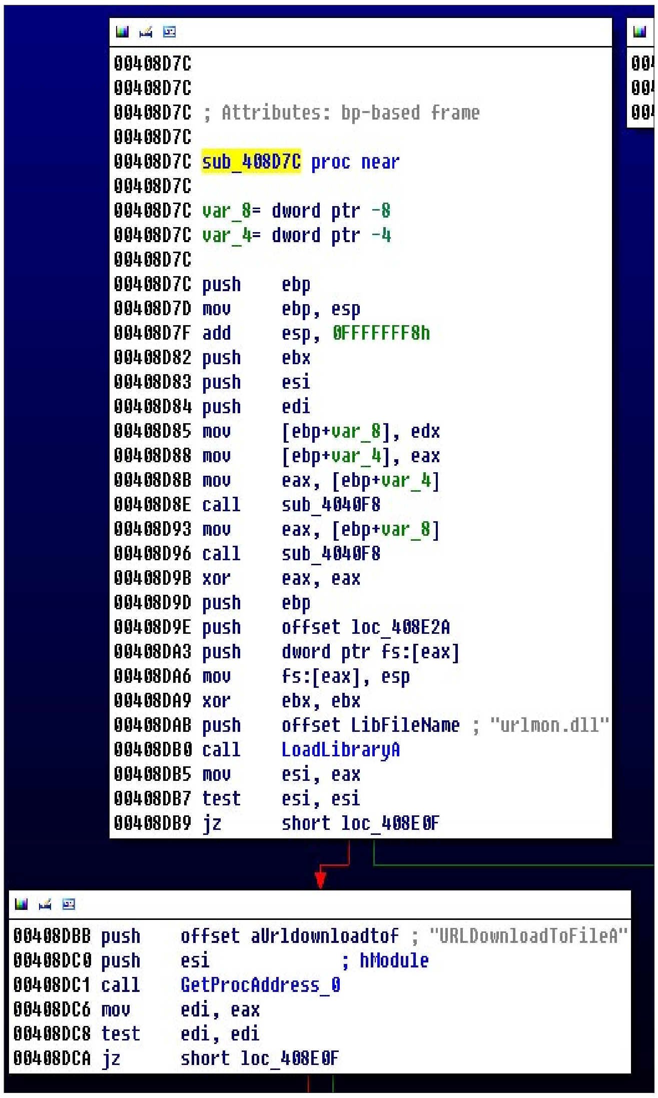
Figure 5. *The Options->General Menu Item*

technical comments on certain lines of code; this would be useful to someone who is not as familiar with assembly.

Keeping track of the location of the code is incredibly useful when trying to get a deeper look into the code by using a separate application such as a debugger or just keeping track of interesting functions that you might want to examine further later on.

Give me the Gist

Imagine walking through a small city you have never been to before and all of the buildings have no descriptions other than "Building `<number>` E. Street", how would you know what function they served and how could you tell if they were important or not? To make it even more frustrating, imagine that some of the buildings are duplicated at different intervals throughout the city so you would not know if you had even been to that building before or not.



```
00408D7C
00408D7C
00408D7C ; Attributes: bp-based frame
00408D7C
00408D7C sub_408D7C proc near
00408D7C
00408D7C var_8= dword ptr -8
00408D7C var_4= dword ptr -4
00408D7C
00408D7C push    ebp
00408D7D mov     ebp, esp
00408D7F add     esp, 0FFFFFFF8h
00408D82 push    ebx
00408D83 push    esi
00408D84 push    edi
00408D85 mov     [ebp+var_8], edx
00408D88 mov     [ebp+var_4], eax
00408D8B mov     eax, [ebp+var_4]
00408D8E call    sub_4040F8
00408D93 mov     eax, [ebp+var_8]
00408D96 call    sub_4040F8
00408D9B xor     eax, eax
00408D9D push    ebp
00408D9E push    offset loc_408E2A
00408DA3 push    dword ptr fs:[eax]
00408DA6 mov     fs:[eax], esp
00408DA9 xor     ebx, ebx
00408DAB push    offset LibFileName ; "urlmon.dll"
00408DB0 call    LoadLibraryA
00408DB5 mov     esi, eax
00408DB7 test    esi, esi
00408DB9 jz      short loc_408E0F
```

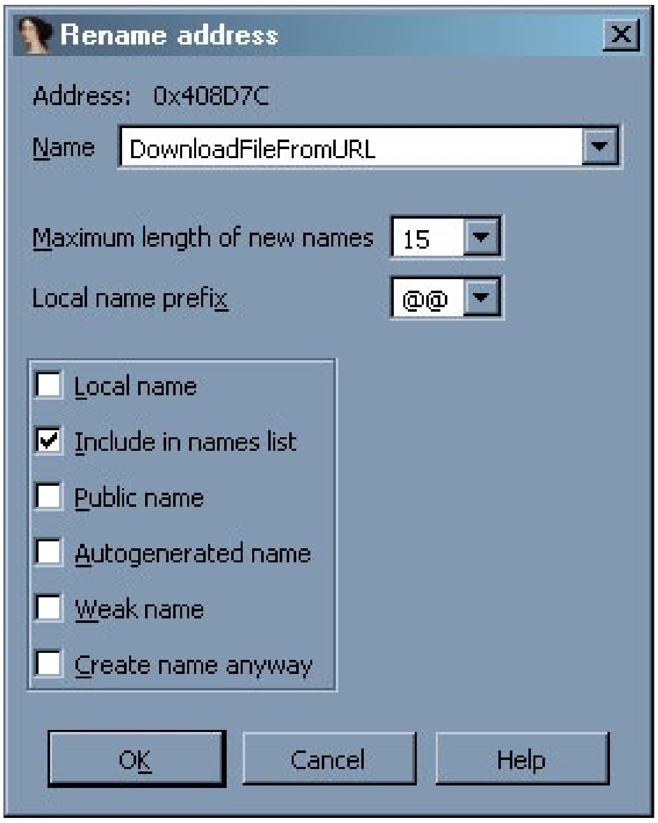
```
00408DBB push    offset aUrldownloadtof ; "URLDownloadToFileA"
00408DC0 push    esi             ; hModule
00408DC1 call    GetProcAddress_0
00408DC6 mov     edi, eax
00408DC8 test    edi, edi
00408DCA jz      short loc_408E0F
```

Figure 7. *A function with a default name*

Well this is not that different from reversing a new binary. Each function that is part of the original programming and not a Windows API usually starts with the prefix "Sub" (for subroutine) and then the address where the function is located in memory.

The best way to avoid overlooking functions with non-descriptive labels is to rename them something that means something to you. In Figure 7 you can see an example of what IDA Pro will automatically name a function, if the function is repeated numerous times you may begin to remember the address but what if it's a unique function, only used once or twice in the program? Well we can rename the function by simply right clicking the function name and selecting the "Rename" option or pressing the "N" shortcut key (Figure 8).

Now that you can rename the function, what should you rename it to? I generally try to use the purpose of the function, as you would if you were programming a new function. If you notice on Figure 7, one of the functions of this routine is to resolve the API `URLDownloadToFile` and later on it actually makes that API call. Based on that information, I called this function `DownloadFileFromURL` because that is what it does. The name is up to you, I enjoy using this method but in case of running into a function that I cannot identify, I will often use the name `IDontKnow`.



Rename address

Address: 0x408D7C

Name: DownloadFileFromURL

Maximum length of new names: 15

Local name prefix: @@

- [ ] Local name
- [✓] Include in names list
- [ ] Public name
- [ ] Autogenerated name
- [ ] Weak name
- [ ] Create name anyway

Figure 8. *The rename address interface*

Another benefit of renaming the functions is being able to identify them in other functions, which can help you rename those to something more important and help you understand the flow of the program even better (Figure 9).

Commentary

Commenting is a programmer's best friend or worst enemy. Some programmers hate to do it and others do it a little too much, either way it is an important part to understanding the code and how it all works together. When it comes to reverse engineering, the same principals apply; commenting will always benefit you. You can comment anything, be it what values are being passed, the purpose of a certain group of API calls or arithmetic operations or even just your own theories on the where the code is leading. Personal notes are useful in case you lose your train of thought or if you step away from the code for a while. If you really enjoy commenting, you could even write comments insulting the code for being so weak against your Jedi reversing skills!

IDA Pro makes comments easy, it does not matter if you are using the graph interface or going through the code line-by-line.

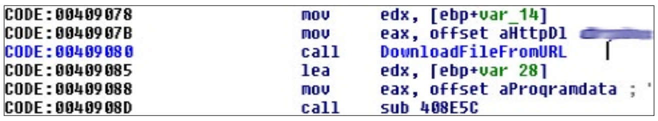
In Figure 10, a function that I named "CheckOSVersion" is checking for a specific value after calling the API `GetVersionExA`. I found out what the compared value (0x06) referred to by looking up the API call in the MSDN library. I discovered that the value refers to any Windows operating system released *after* Windows XP (Vista/Win 7/etc). If I were to stop reversing at this function and go on a vacation or even just look away momentarily, I might forget what that value means. If so, it is important that I make a note to myself in the comments that the code is checking for a certain type of operating system.

To make a comment, right-click the line of code you want to make a note on (to the right of it in the whitespace) and select the "Enter Comment" option. When you do that, a text box appears where you can write in your comment. Figure 11 shows what I wrote and Figure 12 is the result. After adding the comment, I even felt inclined to change the name of the function to reflect its purpose.

Commenting is not just something that school kids and people with bad memories should use. It is very easy to get lost in the code and if you have not already left yourself a note of why a particular function is important or what it could do, you might end up looking up the same values repeatedly. Do your best to be efficient and not duplicate your work if you do not have to.

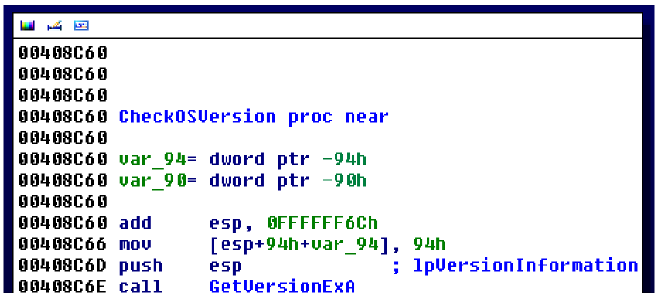
Finding the Cheese First

Okay, after renaming your functions, commenting the code heavily and making sure that every piece of code is in an easy to look at graph format, you still have no idea what is going on with this program. This happens a lot and even with the previously mentioned techniques, you could spend hours wandering aimlessly in the code. In these situations, it is best to think about your exact title, Reverse Engineer, literally meaning to engineer something in reverse. So if you are sup-



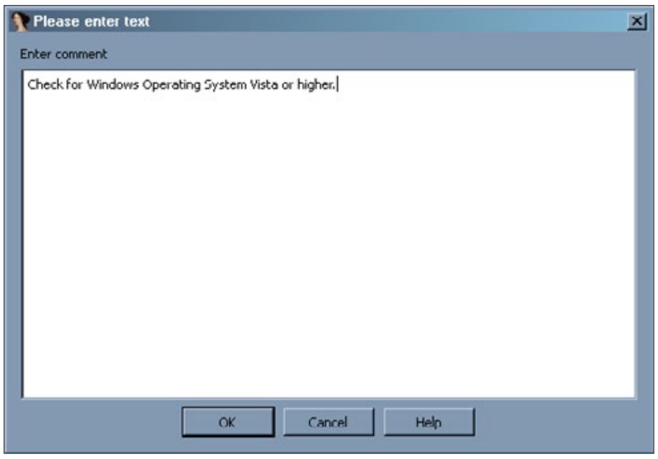
```
CODE:00409078                 mov     edx, [ebp+var_14]
CODE:0040907B                 mov     eax, offset aHttpDl
CODE:00409080                 call    DownloadFileFromURL
CODE:00409085                 lea     edx, [ebp+var_28]
CODE:00409088                 mov     eax, offset aProgramdata ;
CODE:0040908D                 call    sub_408E5C
```

Figure 9. *The renamed function shown inside of another function*



```
00408C60
00408C60
00408C60
00408C60 CheckOSVersion proc near
00408C60
00408C60 var_94= dword ptr -94h
00408C60 var_90= dword ptr -90h
00408C60
00408C60 add     esp, 0FFFFFF6Ch
00408C66 mov     [esp+94h+var_94], 94h
00408C6D push    esp             ; lpVersionInformation
00408C6E call    GetVersionExA
```

Figure 10. *CheckOSVersion calls an API to determine the OS Version*

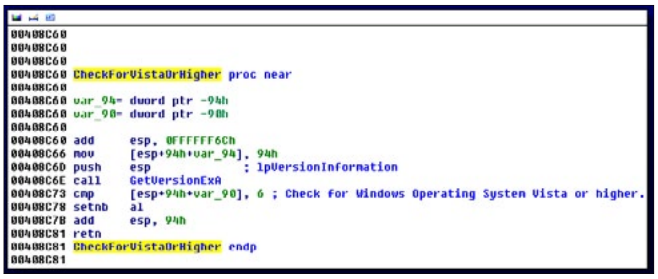


Please enter text

Enter comment

Check for Windows Operating System Vista or higher.

Figure 11. *Comment entry interface*



```
00408C60
00408C60
00408C60
00408C60 CheckForVistaOrHigher proc near
00408C60
00408C60 var_94= dword ptr -94h
00408C60 var_90= dword ptr -90h
00408C60
00408C60 add     esp, 0FFFFFF6Ch
00408C66 mov     [esp+94h+var_94], 94h
00408C6D push    esp             ; lpVersionInformation
00408C6E call    GetVersionExA
00408C73 cmp     [esp+94h+var_90], 6 ; Check for Windows Operating System Vista or higher.
00408C78 setnb   al
00408C7B add     esp, 94h
00408C81 retn
00408C81 CheckForVistaOrHigher endp
00408C81
```

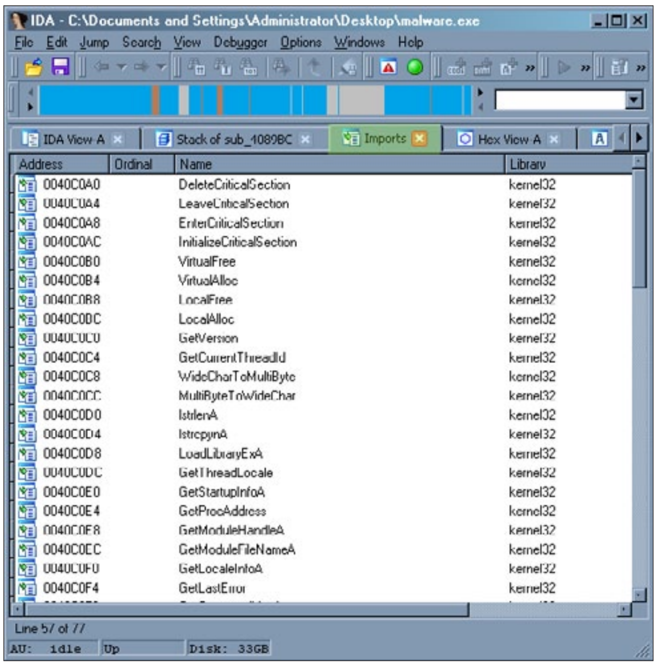
Figure 12. *The result of my commenting*

posed to be doing this all backwards, why are you moving through the code in one direction? This is where the trick of finding your end goal and working backwards comes into play. In my opinion, this trick is the best way to figure out exactly what is going on with a binary in the shortest amount of time.

Imagine that you were a mouse trying to find your way through a maze to find a piece of cheese, the cheese in this metaphor of course being the execution a particular function. You make your way somewhat into the maze but find that you do not know where you are going, so instead you start at the other end, where the cheese is, and work your way backwards to a familiar part of the maze. On the way, you leave pieces of the cheese as markers to illuminate the correct path. At this point, you would know how to navigate through the maze successfully and could easily do it again if you needed to.

The same method in which the mouse found the cheese and worked backwards can be employed in reversing. In order to accomplish this, use the "Imports" tab in IDA Pro. The imports are all of the API functions that the binary needs to use in order to do complete its operations; Figure 13 shows you what this tab looks like.

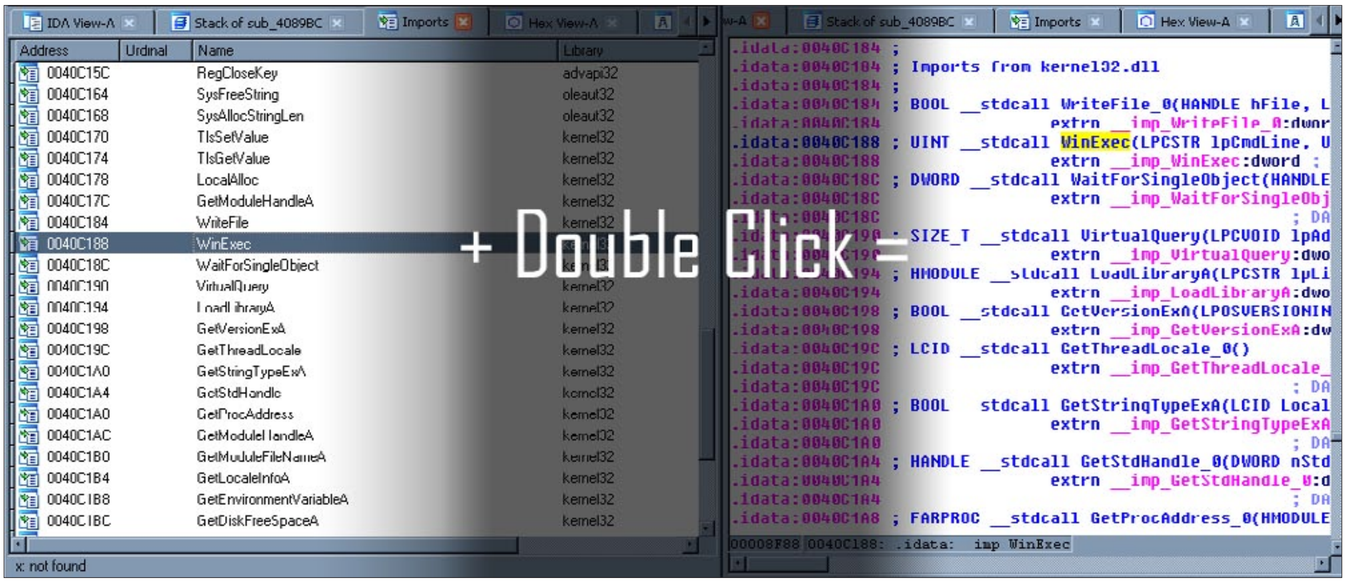
IDA Pro not only knows about these functions but also knows where they are being called in the code; you can use this to your advantage. By double clicking a function related to what you want to


IDA - C:\Documents and Settings\Administrator\Desktop\malware.exe
File Edit Jump Search View Debugger Options Windows Help
IDA View-A
Stack of sub_4089BC
Imports
Hex View-A

Address	Ordinal	Name	Library
0040C0A0		DeleteCriticalSection	kernel32
0040C0A4		LeaveCriticalSection	kernel32
0040C0A8		EnterCriticalSection	kernel32
0040C0AC		InitializeCriticalSection	kernel32
0040C0B0		VirtualFree	kernel32
0040C0B4		VirtualAlloc	kernel32
0040C0B8		LocalFree	kernel32
0040C0BC		LocalAlloc	kernel32
0040C0C0		GetVersion	kernel32
0040C0C4		GetCurrentThreadId	kernel32
0040C0C8		WideCharToMultiByte	kernel32
0040C0CC		MultiByteToWideChar	kernel32
0040C0D0		lstrlenA	kernel32
0040C0D4		lstrcpynA	kernel32
0040C0D8		LoadLibraryExA	kernel32
0040C0DC		GetThreadLocale	kernel32
0040C0E0		GetStartupInfoA	kernel32
0040C0E4		GetProcAddress	kernel32
0040C0E8		GetModuleHandleA	kernel32
0040C0EC		GetModuleFileNameA	kernel32
0040C0F0		GetLocaleInfoA	kernel32
0040C0F4		GetLastError	kernel32

```
Line 57 of 77
AU:  idle   Up      Disk: 33GB
```

Figure 13. *The IDA Pro Imports tab*

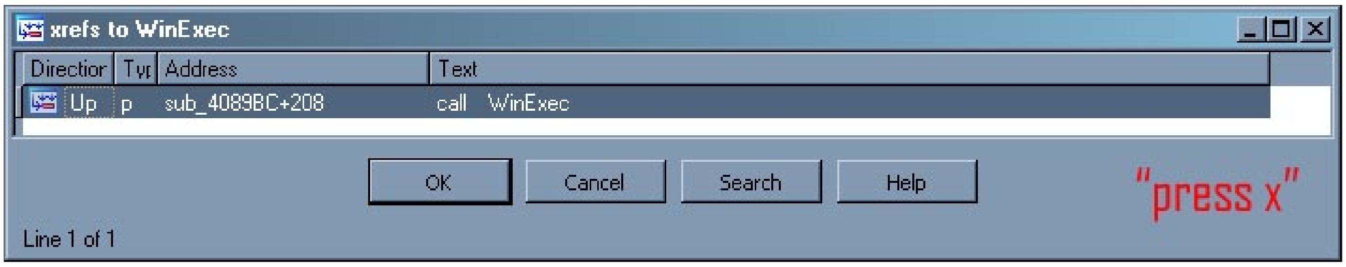


Address	Ordinal	Name	Library
0040C15C		RegCloseKey	advapi32
0040C164		SysFreeString	oleaut32
0040C168		SysAllocStringLen	oleaut32
0040C170		TlsSetValue	kernel32
0040C174		TlsGetValue	kernel32
0040C178		LocalAlloc	kernel32
0040C17C		GetModuleHandleA	kernel32
0040C184		WriteFile	kernel32
0040C188		WinExec	kernel32
0040C18C		WaitForSingleObject	kernel32
0040C190		VirtualQuery	kernel32
0040C194		LoadLibraryA	kernel32
0040C198		GetVersionExA	kernel32
0040C19C		GetThreadLocale	kernel32
0040C1A0		GetStringTypeExA	kernel32
0040C1A4		GetStdHandle	kernel32
0040C1A8		GetProcAddress	kernel32
0040C1AC		GetModuleHandleA	kernel32
0040C1B0		GetModuleFileNameA	kernel32
0040C1B4		GetLocaleInfoA	kernel32
0040C1B8		GetEnvironmentVariableA	kernel32
0040C1BC		GetDiskFreeSpaceA	kernel32

```
x: not found
```

```
.idata:0040C184 ;
.idata:0040C184 ; Imports from kernel32.dll
.idata:0040C184 ;
.idata:0040C184 ; BOOL __stdcall WriteFile_0(HANDLE hFile, L
.idata:0040C184                 extrn __imp_WriteFile_0:dwor
.idata:0040C188 ; UINT __stdcall WinExec(LPCSTR lpCmdLine, U
.idata:0040C188                 extrn __imp_WinExec:dword ;
.idata:0040C18C ; DWORD __stdcall WaitForSingleObject(HANDLE
.idata:0040C18C                 extrn __imp_WaitForSingleObj
.idata:0040C18C                                          ; DA
.idata:0040C190 ; SIZE_T __stdcall VirtualQuery(LPCVOID lpAd
.idata:0040C190                 extrn __imp_VirtualQuery:dwo
.idata:0040C194 ; HMODULE __stdcall LoadLibraryA(LPCSTR lpLi
.idata:0040C194                 extrn __imp_LoadLibraryA:dwo
.idata:0040C198 ; BOOL __stdcall GetVersionExA(LPOSVERSIONIN
.idata:0040C198                 extrn __imp_GetVersionExA:dw
.idata:0040C19C ; LCID __stdcall GetThreadLocale_0()
.idata:0040C19C                 extrn __imp_GetThreadLocale_
.idata:0040C19C                                          ; DA
.idata:0040C1A0 ; BOOL   stdcall GetStringTypeExA(LCID Local
.idata:0040C1A0                 extrn __imp_GetStringTypeExA
.idata:0040C1A0                                          ; DA
.idata:0040C1A4 ; HANDLE __stdcall GetStdHandle_0(DWORD nStd
.idata:0040C1A4                 extrn __imp_GetStdHandle_0:d
.idata:0040C1A4                                          ; DA
.idata:0040C1A8 ; FARPROC __stdcall GetProcAddress_0(HMODULE
00008F88 0040C188: .idata:  imp WinExec
```

Figure 14. *How to find the API function declarations*



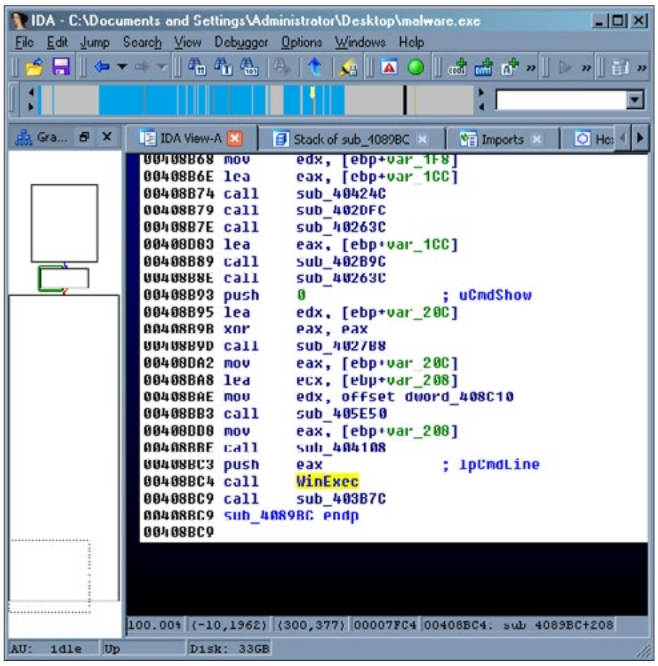
Direction	Typ	Address	Text
Up	p	sub_4089BC+208	call WinExec

```
Line 1 of 1
```

Figure 15. *The IDA Cross-Reference results for WinExec*

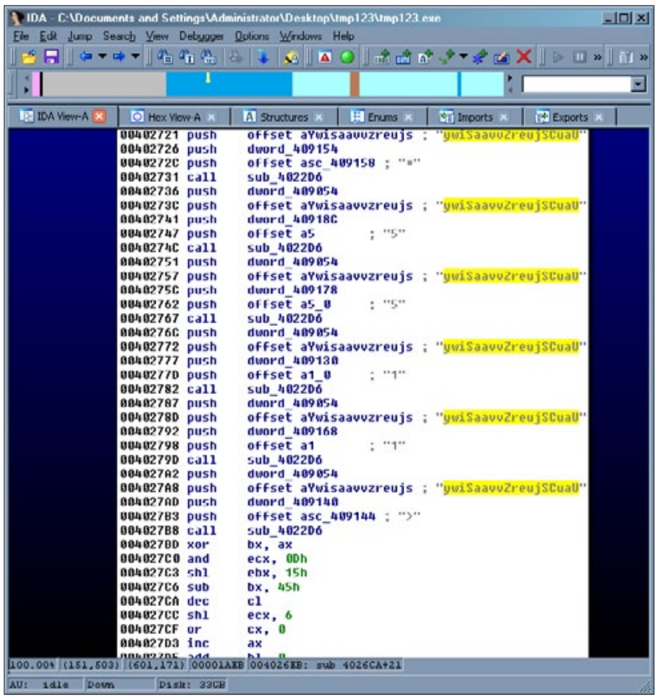
find out about, it will take you to a very pink group of API functions declarations. From there you can use IDA to cross-reference the API functions with the calling code.

In Figure 14, I want to know where the API function `WinExec` is being called. I double clicked the function name and was taken to where the function was being declared. In Figure 15, I have pressed the "x" key and IDA told me all the places where `WinExec` is being referenced. In this case, there is only one function using it.



```
00408B68 mov     edx, [ebp+var_1F8]
00408B6E lea     eax, [ebp+var_1CC]
00408B74 call    sub_40424C
00408B79 call    sub_402DFC
00408B7E call    sub_40263C
00408B83 lea     eax, [ebp+var_1CC]
00408B89 call    sub_402B9C
00408B8E call    sub_40263C
00408B93 push    0                ; uCmdShow
00408B95 lea     edx, [ebp+var_20C]
00408B9B xor     eax, eax
00408B9D call    sub_4027B8
00408BA2 mov     eax, [ebp+var_20C]
00408BA8 lea     ecx, [ebp+var_208]
00408BAE mov     edx, offset dword_408C10
00408BB3 call    sub_405E50
00408BB8 mov     eax, [ebp+var_208]
00408BBE call    sub_404108
00408BC3 push    eax              ; lpCmdLine
00408BC4 call    WinExec
00408BC9 call    sub_403B7C
00408BC9 sub_4089BC endp
00408BC9
```

Figure 16. *The portion of code that calls WinExec*



```
00402721 push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
00402726 push    dword_409154
0040272C push    offset asc_409158 ; "*"
00402731 call    sub_4022D6
00402736 push    dword_409054
0040273C push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
00402741 push    dword_40918C
00402747 push    offset a5        ; "5"
0040274C call    sub_4022D6
00402751 push    dword_409054
00402757 push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
0040275C push    dword_409178
00402762 push    offset a5_0      ; "5"
00402767 call    sub_4022D6
0040276C push    dword_409054
00402772 push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
00402777 push    dword_409130
0040277D push    offset a1_0      ; "1"
00402782 call    sub_4022D6
00402787 push    dword_409054
0040278D push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
00402792 push    dword_409168
00402798 push    offset a1        ; "1"
0040279D call    sub_4022D6
004027A2 push    dword_409054
004027A8 push    offset aYwisaavvzreujs ; "ywiSaavvZreujSCuaU"
004027AD push    dword_409140
004027B3 push    offset asc_409144 ; ">"
004027B8 call    sub_4022D6
004027BD xor     bx, ax
004027C0 and     ecx, 0Dh
004027C3 shl     ebx, 15h
004027C6 sub     bx, 45h
004027CA dec     cl
004027CC shl     ecx, 6
004027CF or      cx, 0
004027D3 inc     ax
```

Figure 17. *The function that calls a decryption routine to resolve dynamic API function names*

Next, I double clicked that result and was taken to the actual function, where I saw exactly how `WinExec` was being used and what other types of functions were being used in conjunction. My next step was to comment and rename the function, then use the "x" key again to find any cross-references in other functions, renaming and commenting until I work back far enough to somewhere I have been before or can easily get to. This method does not just work with API calls either, if you look in the "Strings" tab, you will find a list of different strings that IDA has extracted from the binary and it will tell you where those are being referenced as well (Figure 16).

Filling in the Blanks

There are certain types of programs, namely malware, which will do everything in its power to make it more difficult for you to "find the cheese." There are numerous reasons for this, maybe the author does not want someone else stealing their code or maybe they do not want someone to find out the true capability of their program, either way it creates a big problem for reverse engineers. These anti-reversing techniques usually take the form of dynamically created addresses, function calls and even mathematical operations. If you encounter something like this, you might not be able to continue using only IDA Pro. In these cases, I recommend using a debugger like OllyDbg. In Figure 17, the program uses a decryption routine using a hard-coded key value and a single character value to represent an API call.

At this point, I would open the binary in OllyDbg or another debugger and using what I already know about the flow of the program, navigate to that portion of code and let the binary decrypt the values for me. Figure 18 reveals what the API function names are and from there I can comment the

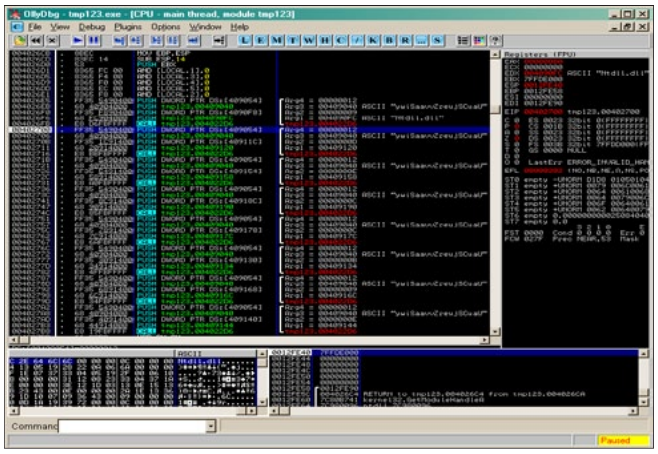


Figure 18. *OllyDbg executing the code and resolving the dynamic values for me*

code in IDA or rename the variables where the results are stored so I know which one is being referenced, no matter where I am in the code.

Keep in mind that IDA Pro also has a very powerful debugger and many Reverse Engineers swear by it, however I recommend keeping your debugger and disassembler separate only because when dealing with certain types of programs (malware) you might damage or corrupt your IDA file and lose all your work. I keep them in two separate virtual machines, frequently save, and backup the saved IDA file to a remote drive.

Conclusion

So there you have it, an array of different tips and tricks that you can use to help you get what you want out of assembly code. As far as what you should do next, that is up to you. The method in which an RE completely tears apart a binary is always their own. Each person is different and the way we perceive and come to conclusions are different. Being an RE is like being a detective, putting the pieces of a puzzle together in order to see the bigger picture. I know many Reverse Engineers who prefer to take a completely static approach and only use IDA Pro. I also know plenty who prefer to use IDA Pro as a map, making as much sense of it as possible beforehand and then using it to guide them through the code using a debugger, editing and adding to the map as they go. In the end, it is whatever works best for you; remember that RE is an art form as much as it is a technical skill and it takes practice, passion and continuous education to become good at it.

ADAM KUJAWA

Adam Kujawa is a computer scientist with over eight years' experience in reverse engineering and malware analysis. He has worked at a number of United States federal and defense agencies, helping these organizations reverse engineer malware and develop defense and mitigation techniques. Adam has also previously taught malware analysis and reverse engineering to personnel in both the government and private sectors. He is currently the Malware Intelligence Lead for the Malwarebytes Corporation. akujawa@malwarebytes.org.

Malwarebytes

ISAF

The **Most Comprehensive** Exhibition
of the Fastest Growing Sectors of recent years
in the **Center of Eurasia**

ISAF IT SECURITY
INFORMATION, DATA AND NETWORK SECURITY EXHIBITION

ISAF SAFETY&HEALTH
OCCUPATIONAL SAFETY AND HEALTH EXHIBITION

ISAF SMART HOUSES
SMART HOUSES AND BUILDING AUTOMATION EXHIBITION

ISAF SECURITY
16th INTERNATIONAL SECURITY AND RFID EXHIBITION

ISAF FIRE
16th INTERNATIONAL FIRE, EMERGENCY RESCUE EXHIBITION

www.isaffuari.com

SEPTEMBER 20th - 23rd, 2012
IFM ISTANBUL EXPO CENTER (IDTM)

KOSGEB

MARMARA TANITIM FUARCILIK
T. +90 212 503 32 32 | marmara@marmarafuar.com.tr
www.marmarafuar.com.tr

THIS EXHIBITION IS ORGANIZED WITH THE PERMISSIONS OF T.O.B.B. IN ACCORDANCE WITH THE LAW NUMBER 5174.

How to Protect Your Identity in the UK from Fraud

Information is being collected about us every second of every day without us ever realizing what happens to it. Most of us don't really care what happens to our personal data as long as it isn't misused. So let's go up close and personal by taking a brief glance at how you can protect your personal data if you are a UK citizen.

Worth remembering, your data held in the UK is also shared with other countries, mainly the English speaking world i.e. Canada, New Zealand, USA, South Africa and Australia to name a few. The credit reporting agencies share this data with these countries and in particular when people migrate to these countries. Every country has its own data protection laws but for the benefit of this article we will concentrate on the UK.

UK data regulation

Regulating our personal data is more important than ever these days, especially given the sensitive nature of the data that is collected. The first attempt at a data protection law was with the *Data Protection Act* (DPA) 1984 which started by authorising organisations to take accountability for your personal data privacy. Check any UK registered website and they should highlight the DPA 1984 and 1998 (amendment). The 1998 amendment tightened the DPA which now allows everyone to see the data that is stored about them on either hardcopy (paper) or a computer.

The personal data held by third parties is used in many instances to make key life changing decisions without you ever realizing it – i.e. credit referencing agencies, people tracking websites, banks, mortgage lenders, employers etc. I will discuss this in more detail later. The DPA provides a safeguard for people so people can ask for the data held about them and dispute any inaccuracies. The way the data is collected and used is also covered under the DPA 1984/1998 Acts. As is the case with most laws, it's there as a protection but that doesn't stop data breaches or inaccurate data being held about people.

Keep in mind

You can use the DPA to request information from a financial provider if you suspect for example that the data about you is inaccurate. It doesn't have to be your data that stops you from being accepted for a new loan or credit card. It can also be where you live and who you live with. More often than not people fail to tick or un-tick the 'do not receive any marketing communication from a company or its third parties' box. You should always remember to 'opt-out' if you value your privacy.

The Electoral Register

There are many instances of people applying for credit cards and loans being refused simply because they are not recorded on the electoral roll. The electoral register should highlight your current address, so it's important you make sure it's up to date if you have recently moved. The names and addresses of all UK citizens over the age of 18 registered to vote are kept on the electoral register *http://bit.ly/qcw51*. For the past few years organisations and individuals could obtain this information and use it for any legal purpose, but privacy concerns have meant that regulation was introduced in 2002.

The regulation introduced two electoral registers. The full register lists everyone who is entitled to vote. Only certain people and organisations (i.e. UK Direct Marketing Association *http://bit.ly/BM-RXz*) can have copies of the full register, and they can only use it for specified purposes. These include electoral purposes, the prevention and detection of crime and checking your identity when you have applied for credit. The edited register leaves out the names and addresses of people who have asked for them to be excluded from that version of the register. The edited register can be bought by anyone who asks for a copy and they may use it for any purpose.

Everyone on the full register goes on the edited version by default, but you can 'opt out' this when you return the 'Annual Voter' registration form. This means commercial organisations will not be able to have access to your name and address and on that year's register. Remember, that you will not be removed from the previous year's registers. Organisations may still have your personal details as well as the people you live with. If you want to stop cold calling, direct marketing mail and tempting credit card offers, this is a first positive step to protecting your personal data.

Preference Services

Register with the free MPS (Mailing Preference Service) *http://bit.ly/3xksTZ* if you want to manage and control what marketing mail marketing telephone calls (includes silent calling) you receive. This list of people who don't want their publicly available details to be used for direct marketing purposes is administrated by the *UK Advertising Standards Authority* (ASA). There is though one small issue with this and that is organisations are not legally obliged to use it.

UK organisations can buy in the lists but should check the data against the Mailing Preference Service opt-out list. The problem is a number of organisations don't actually do this. That said if the organisation is a DMA member (and you can check to see if an organisation is a member of the DMA *http://bit.ly/7tzoa0*) they are bound by the code of practice, so must screen the data against the MPS database.

The Royal Mail also has an 'opt-out' door to door service *http://bit.ly/UU3g6* which will stop all those unaddressed mail being posted through your mailbox. This service doesn't stop the mail addressed 'the occupier' though. If an organisation continues to send you unsolicited marketing mail after you asked them to stop, that organisation will be in contravention of the Data Protection Act and ASA regulations, which means that the ICO and ASA can be asked to intervene.

Another really useful mailing preference service is 'The Bereavement Register' *http://bit.ly/hmlbcW* which can help reduce the amount of direct mail sent to your address, stopping painful daily reminders. Unless companies are informed of a death, they will continue to send promotional mailings about their products and services. By registering with this free service the names and addresses of the deceased are removed from mailing lists, stopping most direct mail within as little as six weeks.

Telephone marketing, silent calls and filling in forms

Telephone marketing calls are something we all have experienced. Sometimes having an ex-directory number can help as can signing up for the *Telephone Preference Service* (TPS) *http://bit.ly/qOsft* Organisations are not obliged to use the TPS list but it does help reduce the marketing calls from personal experience. If you are repeatedly hassled by these marketing and silent calls then complain to the ICO.

Cold calling that originate overseas can also be stopped, but only if those companies calling are UK registered/owned and are using foreign call centres to make these calls – so these companies will still be bound by the DPA code of practice. If you are still receiving cold calls then there is an EU Data Protection Directive *http://bit.ly/QqxGe* which the ICO routinely liaises with.

Silent calls are made by automated dialling systems that fail to connect the call when answered, however it might a good idea to register with a service called SilentCall-Gard: *http://bit.ly/eH5aG* – it's totally 100% free. In the UK, new legislation introduced by the regulatory body Ofcom (Independent regulator and competition authority for the UK communications industries) has just revamped the automated dialling systems. After February 2011, all automated calls must be connected within two seconds of the recipient speaking or there should be a recorded message that states the organisation's name and how to opt out of future calls.

Form filling is something we are all fond of – well not really. This is where we get caught out as so far as allowing others access to our personal data by forgetting to tick or un-tick a simple box. Be sure to tick the appropriate boxes when filling out any forms for goods and services. Look for opt-out statements which use euphemisms to confuse you. Read the opt-out a couple of times so you fully understand what you are opting out of and that you are actually not opting in. It's very important you have the opportunity to prevent your details being passed to third parties – but this is in your control. Consider this; magazine subscriber lists are routinely sold / rented as are our high level data from our credit files to marketing and other agencies (including people tracking websites). Form filling will never go away, whether it's online or a paper copy, so stay completely vigilant.

HINT
Worth noting and not everyone knows this – third-party organisations are not legally allowed to sell on the details of consumers on these sold-on lists, unless that is they convert these consumers into consumers of their own.

Checking your credit report

Under the DPA you can request information from a financial provider i.e. bank, credit card provider if you suspect the information about you is incorrect or has caused you problems when applying for a loan, credit card or opening a bank account for example. The Data Protection Act allows for you to obtain a 'statutory credit report' from all the credit reporting agencies – Experian: *http://bit.ly/hNL28g*, Equifax: *http://bit.ly/fY7yZq* and CallCredit: *http://bit.ly/gRIv2c*. The information on credit reports is the most important information about you. Credit information is used to decide whether you are financially viable. In other words these agencies decide whether you can have a loan, mortgage, credit card and so on. Without a credit history (which takes time to build – and the only way to build this is to have credit in your name) you are unlikely to be able to borrow money.

Credit card companies are not that interested in people who don't rack up debt – so long as they can make money on your interest payments they are more than happy to give you credit to spend. To access your statutory report will cost £2.00 (correct as of September 2012) but this will not give you your credit score – which determines how risky you are to loan money too. All three credit reporting agencies will have some different information about you, so it's important to obtain the reports and credit score from all three.

If you spot an error, you should send the credit reporting agency a 'correction' letter or if you notice that you have some late payments showing or you have an unpaid debt that was a result of an uncorrected billing error. You can also apply to the agencies for a 'notice of correction' to your credit report which will clarify the in correction to future lenders.

If you have recently divorced or left your partner you should also 'financially disassociate'. Once a disassociation has been created, lenders requesting your report no longer see details of the disassociated family member or members. You will need to notify each agency about the disassociation.

If you don't do this then your ex-partner may obtain a loan or credit card in your good name. It has happened and continues too, even when people are legally divorced. Here are the links for financial disassociation: Experian – *http://bit.ly/dIakAB* Equifax: *http://bit.ly/eoU9Ds* CallCredit – *http://bit.ly/hDm03Z*.

Protecting your email address

Unsolicited direct marketing mail is not only sent to a letter box. As we all know well it is also sent to our mail inbox on our PCs. This unsolicited email is called spam. Since 2003 sending spam is a criminal offence, but beware it all depends on whether you remembered to tick or un-tick that box on the web form that asks you for permission to use your personal details for marketing purposes.

Savvy surfers use two email addresses – one for email communication and the other with everything else. Disposable email addresses are a must have if you value your email addresses. There are many but the general idea is that you open a web page and click a get link for a randomly generated email address that exists for a specified time period. Here are three popular websites: Guerrillamail *http://bit.ly/2LVUNc* Spamgourmet: *http://bit.ly/PcE9K* and Mailinator: *http://bit.ly/1WHWBe*.

Some of these sites only allow you to send and receive email using their webmail system – but some not all allow you to manage the spam and forward any relevant emails to your actual email address.

Facebook and Google data privacy

Facebook privacy has been the subject of much discussion in recent months. It isn't the only social website that is facing criticism. Google, the world leader in Web search, has been in trouble recently for collecting information from unsecured wireless networks all over the world. This was done as specially equipped vehicles took pictures for the Google mapping feature called Street View. Google said it never meant to collect people's private information, like e-mails and passwords.

Some of the main problems have been linked to the default privacy settings in Facebook. Facebook now opt out users in to allowing third party sites like Yelp to 'personalise' a user's experience, and there are questions about how much information is being given away. One suggestion here is to make instant personalization which exports users content to third-party Web sites, opt-in by default. Another data issue circulating is the one concerning third-party applications Facebook currently stores the data for no more than 30 days and does not use it for advertising or selling to third-parties. One suggestion here is for Facebook not to keep data about user visits to third-party sites that use social plug-ins, such as the "Like" button.

Facebook data privacy could also be enhanced if it was allowed to degrade or fade in time. The idea of "degrading" data about visitors isn't a new concept. A database could be developed that would gradually swap user details for more general information and help guard against accidental disclosure. See my blog entry regarding Facebook scraping *http://bit.ly/gWhq7W* for further information on this threat.

Facebook has recently addressed a major security issue – surrounding HTTPS. I wrote about this in *Managing your Facebook Privacy* back in June 2010 feature See my blog entry regarding how you can setup a HTTPS connection: *http://bit.ly/hVkkCB* – if we all value our data then we should all be using HTTPS.

Protecting your identity and your personal data from identity theft

So – how do you go about protecting your good name, both in the cyber world and the offline world? I'm going to highlight the UK service options and then you can decide which service is best for you.

UK Identity Theft Protection Service Options

Here is what you should look for if you are living in the UK:

- Credit reporting / scores i.e. providing single report or triple reports analysis*
- Computer protection i.e. anti-malware/firewall/anti-virus/password protection
- 24/7 access to trained ID Theft Resolution Specialists – includes identity recovery
- Identity theft Insurance (up to £50,000)
- Lost wallet/cards protection – will cancel and replace your cards/passport etc
- CIFAS Protective Registration – places a warning flag against your credit file(s)** Also available for directors whose company is at risk of corporate identity fraud and against the name of any deceased party (by a relative or executor) who may be at risk of impersonation attempts.

*If an application for credit is made in your good name you also have the option of receiving an EMAIL or SMS. The three leading credit reference agencies in the UK are: Experian, Equifax and CallCredit. **CIFAS Protective Registration can also be purchased separately for £20 for one year. Please check the CIFAS *http://bit.ly/hHORFO* website for further information. (September, 2012).

The average cost of UK identity theft protection services varies from £8-10 per month (this mainly applies to credit monitoring only). In the UK there is only one company that offers an identity protection service, similar to what is on offer in the US – called Garlik. Garlik charge £4 per month for one year for individuals to DataPatrol. Garlik DataPatrol *DOES NOT* offers an online credit monitoring service. (September 2012).

Worth remembering

If you do decide to purchase just a credit monitoring service you will have to pay extra for your credit score.

Worth remembering

Section 75 of the Consumer Credit Act 1974 protects consumers on any credit card purchases (this includes loss or theft) which cost over £100 and under £30,000. Note: This also applies when someone else fraudulently uses your credit card i.e. Chip & Pin fraud, Card Not Present (CNP) fraud etc.

If you are a UK citizen and value your personal data, I'm sure what I have written here will be of considerable interest. I hope to cover this feature for US citizens very soon...

JULIAN EVANS
ID Theft Protect

NoSQL

Database Security with Cassandra

This article will discuss what you should know about Cassandra and other NoSQL databases. Since the var-ious applications which can use Cassandra are out-side the scope of this article, we are only going to look at the exploitation of the database.

What you will learn…

- Difference between SQL and NoSQL databases
- What security issues exist within NoSQL databases
- How to setup Cassandra and use CQL
- How to scan for and remotely access Cassandra

What you should know…

- Basic Ubutnu linux administration and installation of software.
- Usage of Nmap, and basic understanding of TCP/IP.
- Familiarity with SQL databases such as MySQL or Postgres.

Recently, a new type of database, that differs from a traditional relational database in many ways has appeared. In a relational database such as MySQL or Postgres, *Standard Query Language* (SQL) is used to perform operations on the data, which is stored as rows, and organized by columns and tables. This new type of database is referred to as a "NoSQL" database. This is because SQL is not used to perform operations on the data, such as to add, remove, update, or delete a record. The datastore is also not organized into rows with various columns, but rather as a multi-dimensional hash instead. The reason for this is to optimize performance, and achieve very high scalability with a large datasets. Also, developers can make changes to the application, without having to change of the structure of the data within the DB. You can find these types of database being used in applications such as social networking, online games, and applications which require storing large amounts of data very quickly.

Some Background

The term NoSQL was first used in 1998 by Carlo Strozzi to refer to his *Relational Database Management System* (RDBMS) of the same name (NOSQL). His database was relational, but did not use SQL as it's language for operations. It was not until 2009 when Eric Evans (of Rackspace) used the term NoSQL to describe various distributed datastore projects, that the term gained acceptance to refer to those database management systems which were distributed, not relational and did not use SQL [1]. Currently, there are several projects which can be considered NoSQL databases. Some of these are MongoDB, Neo4j, CouchDB, riak, Facebook's Cassandra, Amazon's SimpleDB and Google's BigTable. In this article, the issues which affect the type of database in general will be discussed, and Cassandra will be looked at in detail, with specific examples.

NoSQL and Security

One issue that affects all of these NoSQL databases, is that they were not designed with security in mind. This can impact the Confidentiality and Integrity of the data, which relies on a DBMS ability to perform *Authentication*, *Authorization*, and *Accounting* (AAA) functions. In a relational database, a high level of control can be wrapped around tables, and specific access can be given to certain users. For example, to achieve confidentiality of data at rest within Postgres, you can create a user and grant them access to a certain database, and determine if they should be able to read/write or just read the data. You can also specify which tables are able to perform operations against. Additionally, you can tell who has access to specific data, based on their username and a timestamp. This level of control does not exist within a NoSQL database, because the data is only organized as hashes, and is not normalized into tables. Also, many of them are still developing the user security model, and some do not use authentication at all. Native encryption support does not yet exist, so data must be encrypted before storage into a NoSQL database, and in some cases can even break the data searching model completely. Transport Layer Security (SSL) Encryption of the data in motion between nodes is a new feature to some, and non-existent in others. Since there is no SQL or tables, controls such as transactions, referential integrity, and foreign key constraints do not exist either, and cannot enforce data integrity. However, keep in mind that these are all trade-offs for the performance and scalability achieved.

NoSQL on the Network

On the network side, care must be taken to avoid access to the NoSQL database server directly. Since there is no user authentication in most cases, anyone can connect to the open management ports, and perform any operation they would like. Additionally, new attacks such as "Node in the Middle" have began to appear. This occurs when a rogue database node announces that it is part of the distributed NoSQL database. Even though it might only contain part of the total dataset, it has access to perform operations across the entire cluster. All this is due to the distributed nature of NoSQL databases, since they are designed to be highly scalable, and allow other nodes to quickly join the cluster.

Securing NoSQL

As you can see from the concerns outlined above, serious issues can occur if these are not addressed. The advice from each of the NoSQL database vendors so far has been that all of the same precautions should be taken as when developing any application. This includes following practices such as sanitizing input, proper error handling, securing communications when possible, and validating business logic. In addition, the database server(s) themselves should be kept within a "walled garden", allowing communication between nodes to take place only behind a firewall, or through a vpn. This approach, although somewhat effective, works by essentially relying on perimeter security to keep unauthorized users out. This can be defeated by probing for clusters which are not protected, misconfigurations, or in some cases pivoting through a *Cross Site Request Forgery* (CSRF) or another application level attack. All that is needed is to locate a trusted system behind the firewall, which can reflect the necessary query from behind the secure perimeter. For example, if a system administrator falls victim to clicking a link which sends a malicious request directly to the NoSQL server. Although the desktop might be fully patched and not vulnerable to client side attacks, JSON requests can be sent on their behalf to the restful APIs of certain NoSQL systems [2].These same type of NoSQL injection attack can be made through the improperly sanitized input of a web form, possibly affecting other systems behind the firewall.

Example: Cassandra

A closer look will now be taken at a specific NoSQL database, Cassandra. Cassandra itself was first created by Facebook for use in their social networking application, and was open sourced as an Apache project in 2008. Although it is no longer used to power the main application (since they are using a fork of the open sourced version), it is still used within inbox search. As you can imagine from the sheer size of data that must be processed, this NoSQL database has the ability to scale to very large datasets, and to also be run across distributed systems. Since this software is under very active development, it is possible that the developers may choose to resolve some of the issues being described. The Apache project page can be found here at: *https://cassandra.apache.org/*.

The best way to learn more about this NoSQL implementation is to examine it hands on through some examples. Specifically we are going to use version 1.1.5., as provided by a company called Datastax (3). This version of the software is a free community version of their Enterprise Cassandra Server. You can install a single node version of Cassandra, using the Datastax distribution, which allows you to get up and running quickly. In this way, you can learn more about the defense of it, as well as possible new attack vectors. Since it is an open source, the code can be examined first hand as well, for possible direct application vulnerabilities. Since it is run across these multiple systems on a network, it requires certain ports to communicate across nodes. The Datastax distribution also includes an application called ops center which can be used to manager a Cassandra cluster, and can be downloaded here: *http://www.datastax.com/download/community*.

Options are available here to download install Cassandra on CentOS, Debian, OS X, and even Windows 7. For this example, Ubuntu 12.04 LTS will be used, running on a Rackspace Cloud Server, along with the guide to installing from Debian packages.

If you would like a dedicated or virtual machine of your choice can be used, though a clean installation is recommended. 2GB of RAM is also necessary, preferably with two processor cores.

Installing Cassandra

Once you have access to a fresh install of Ubuntu, the first step is to become root and check the version of java on the system:

```
# sudo -s
# java -version
```

which should return the following:

```
java version "1.6.0_24"
```

Listing 1. *Keyspaces*

```
[default@unknown] create keyspace test;
dd7862b9-1aa2-30eb-b34a-9bcac6c0c8c8
Waiting for schema agreement...
... schemas agree across the cluster

[default@unknown] use test;
Authenticated to keyspace: test

[default@test] create column family testfamily;

926b2a9f-1197-316b-b3c1-615497e166af
Waiting for schema agreement...
... schemas agree across the cluster

[default@test] describe test;
Keyspace: test:
  Replication Strategy: org.apache.cassandra.locator.NetworkTopologyStrategy
  Durable Writes: true
    Options: [datacenter1:1]
  Column Families:
    ColumnFamily: testfamily
      Key Validation Class: org.apache.cassandra.db.marshal.BytesType
      Default column value validator: org.apache.cassandra.db.marshal.BytesType
      Columns sorted by: org.apache.cassandra.db.marshal.BytesType
      GC grace seconds: 864000
      Compaction min/max thresholds: 4/32
      Read repair chance: 0.1
      DC Local Read repair chance: 0.0
      Replicate on write: true
      Caching: KEYS_ONLY
      Bloom Filter FP chance: default
      Built indexes: []
      Compaction Strategy: org.apache.cassandra.db.compaction.
                SizeTieredCompactionStrategy
      Compression Options:
        sstable_compression: org.apache.cassandra.io.compress.SnappyCompressor
```

If not, it is necessary to install java:

```
# apt-get install openjdk-6-jre-headless
```

Once Java is installed, the following must be added to the `/etc/apt/sources.list`:

```
deb http://debian.datastax.com/
                  community stable main
```

and the repository key must be added also:

```
# curl -L http://debian.datastax.com/debian/
                  repo_key | sudo apt-key add -
```

Next, apt must be updated, and the packages can then be installed:

```
# apt-get update
# apt-get install libssl0.9.8 python-cql dsc1.1
                  opscenter-free
```

Once these packages are installed, Cassandra will be automatically started, and the cassandra command line interface can be used to test connecting to a server. To start a connection to the locally running instance on TCP port 9160, use the following command:

```
Using CQL on the CLI

# cassandra-cli -h localhost -p 9160
Connected to: "Test Cluster" on localhost/9160
Welcome to Cassandra CLI version 1.1.5
```

Listing 2. *NoSQL System*

```
[default@test] set testfamily[ascii('creditcard')]
[ascii('number')] = ascii('123456789000000');
Value inserted.
Elapsed time: 81 msec(s).
[default@test] set testfamily[ascii('creditcard')][ascii('expiration')] = ascii('01/13');
Value inserted.
Elapsed time: 41 msec(s).

[default@test] get testfamily[ascii('creditcard')];
=> (column=657870697261746966e, value=01/13, timestamp=1348213446666000)
=> (column=6e756d626572, value=123456789000000, timestamp=1348213374055000)
Returned 2 results.
Elapsed time: 502 msec(s).
```

Listing 3. *Nodetool*

```
# nodetool -h 127.0.0.1 info
Token            : 133960192202570926894249588632529377034
Gossip active    : true
Thrift active    : true
Load             : 30.45 KB
Generation No    : 1348196041
Uptime (seconds) : 18393
Heap Memory (MB) : 82.57 / 486.00
Data Center      : datacenter1
Rack             : rack1
Exceptions       : 0
Key Cache        : size 96 (bytes), capacity 25165824 (bytes), 3 hits, 5 requests, 0.600
                 recent hit rate, 14400 save period in seconds
Row Cache        : size 0 (bytes), capacity 0 (bytes), 0 hits, 0 requests, NaN recent hit
                 rate, 0 save period in seconds
```

```
Type 'help;' or '?' for help.
Type 'quit;' or 'exit;' to quit.

[default@unknown]
```

The prompt should show that a connection is now established to the local cluster "Test Cluster" which is setup by default. Next, these commands can show you which environment you are connected into:

```
[default@unknown] show cluster name;
Test Cluster
[default@unknown] show keyspaces;
Keyspace: system:
  Replication Strategy: org.apache.cassandra.
                locator.LocalStrategy
  Durable Writes: true
    Options: []
  Column Families:
```

Each of the column families listed are what is used by Cassandra to store it's data, and they are similar to a table within a SQL database. These column families live within a Keyspace, which are either "system" or user defined Keyspaces. The default is system, but another keyspace and column family can be created and viewed by using these commands: Listing 1.

Using Keyspaces

Those commands create a Keyspace called test, and a column family called testfamily, that define the basic schema. Authorization within Cassandra is done the family level, and currently two permissions exist, READ and WRITE. Plans exist to add support for stronger authentication schemes, but are not fully implemented yet. Now, in order to actually test writing and reading some data within the NoSQL system you would issue the following commands: Listing 2.

As you can see, these commands and their results look very strange if you are used to seeing SQL queries and their output. These commands work through the cli running on the same host as the database, or through connecting to another host on port 9160. The language used for these commands is called *Cassandra Query Language* (CQL), which includes support for server side functions. Additionally, if it is possible to escape any input sanitizing, these CQL commands can be injected through a web application directly. Each programming language has different ways in which you can connect to Cassandra, but the standard inferface is known as "Thrift", the lowest level official API for accessing the NoSQL datastore. Example of using thrift within PHP, Python, Java, Perl, C#, and C++ can be found here: *https://wiki.apache.org/cassandra/ThriftExamples.*

Probing the Network

When probing for versions of Cassandra running on a network, certain open ports will help to determine a vulnerable system. These are the ports can typically be found open on a linux node running the Datastax Community version of Cassandra:

Port	Description
22	SSH
8888	Ops Center
1024+	JMX Loopback
7000	Cassandra Intra-Node (Gossip)
7199	JMX Monitoring
9160	Cassandra Client (Thrift)
61620	Ops Center Monitoring
61621	Ops Center Agent

If you are not sure if Cassandra is running on a system, NMAP can be used to scan for these open ports. If it is know that Cassandra is running on a remote system, you can use the bundled program, nodetool, which gives detailed information as follows: Listing 3.

Nmap and Cassandra

The latest version of NMAP also has some new features which can assist in both fin-gerprinting, and performing a brute force attack against the Cassandra authentication (which is open by default). You can download the latest version of NMAP by using the svn client, and installing a few prerequisite packages:

```
# apt-get instal subversion g++ gcc make
                libssl0.9.8 libssl-dev lua5.1
# svn co https://svn.nmap.org/nmap
# cd nmap
# ./configure && make && make install
```

Once Nmap is installed and working properly, the plugin for Cassandra can be used to attempt a brute force attack:

```
# /usr/local/bin/nmap -p 9160 127.0.0.1
                --script=cassandra-info
Starting Nmap 6.02 ( http://nmap.org ) at 2012-09-
                21 08:07 UTC
Nmap scan report for localhost (127.0.0.1)
```

```
Host is up (0.00091s latency).
PORT     STATE SERVICE
9160/tcp open  cassandra
| cassandra-info:
|   Cluster name: Test Cluster
|_  Version: 19.32.0
Nmap done: 1 IP address (1 host up) scanned in
                 0.36 seconds
```

The above command and output shows that information can be determined using the cassandra-info script.

```
# /usr/local/bin/nmap -p 9160 127.0.0.1
                 --script=cassandra-brute

Starting Nmap 6.02 ( http://nmap.org ) at 2012-09-
                 21 08:09 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00022s latency).
PORT     STATE SERVICE
9160/tcp open  apani1
|_cassandra-brute: Any username and password would
                 do, 'default' was used to test.

Nmap done: 1 IP address (1 host up) scanned in
                 0.21 seconds
```

As you can see from the above output, nmap was run against the localhost, and did not find it necessary to run a full brute force since it could be determined any username and password would work. If password authentication was setup, output would be similar to the following:

```
# nmap -p 9160 <remoteip> --script=cassandra-brute

PORT     STATE SERVICE VERSION
 9160/tcp open cassandra
 cassandra-brute:
 |   Accounts
 |     admin:lover - Valid credentials
 |   Statistics
 |_    Performed 4581 guesses in 1 seconds,
                 average tps: 4581
```

Summary of Issues

Overall, it can be seen that many issues exist not only with NoSQL databases and their lack of authentication options, but also within the default configurations. Since these type of database are meant to be used when rapidly developing a big data application, developers might tend to overlook such things as configuration of security mechanisms. To address these issues, many developers have opted to store sensitive data, such as AAA functions of their application, into a traditional RDBMS while using a NoSQL datastore for "Big Data" only. Others might opt to layer their security mechanisms throughout the application, or by using middleware frameworks to perform security functions. Finally, although it is a weak option, securing the perimeter is definitely necessary until better authentication is standardized for NoSQL applications. In the meantime, penetration testers and black hats alike can use this situation to their advantage.

References

- *https://en.wikipedia.org/wiki/NoSQL*
- *https://blogs.adobe.com/asset/2011/04/nosql-but-even-less-security.html*
- *http://www.datastax.com/download/community*

DAVID LISTER

David Lister is an Application Security Architect for Rackspace, an Open Cloud hosting company headquartered in San Antonio, Texas. In previous roles he was a software developer on various projects written in a mix of Php, Python, Perl, Java, Ruby, C#, and Asp.net. He has also been active in many areas throughout the past nine years at Rackspace in various roles, involving systems administration, network security, incident response, penetration testing, and application security. David holds a Master's degree from the University of Texas San Antonio in Infrastucture Assurance, and is a member of the ISSA, Austin Hackers Anonymous, OWASP and the IEEE Computer Society.

Code & Platform Level SQL Injection Defenses

How to prevent from SQL Injection using Code-Level and Platform Level Defenses

SQL Injection is neither a secret technique nor a recent one. For years, thousand of sites have been compromised because of this attack. SQL Injection attack has successfully breached many big organizations website such as Sony Pictures, PBS, Microsoft, Yahoo, LinkedIn, NASA and a lot more.

What you will learn…
- Techniques of SQL injection prevention.
- Hardening your platform methods.

What you should know…
- Basic understanding of SQL implementation.
- SQL injection attacks methodology.

SQL injection attacks is a serious threat to any SQL based website as these attacks are easy to learn. However, the damage caused, varies from a simple login bypass to the whole system compromise. Nevertheless, it can be prevented through a right programming technique and a secure database platform.

What is SQL Injection

According to Wikipedia, SQL Injection is a technique used to attack databases through website by exploiting the code vulnerability of website application. This can be done by including portions of SQL statements in a web form entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database. The objective is to fool the database system into running malicious code that will reveal sensitive information or otherwise compromise the server.

Usually, there are four main categories of SQL Injection attacks against databases:

- SQL Manipulation: manipulation is a process of modifying the SQL statements by using various operations such as UNION. Another way for implementing SQL Injection using SQL Manipulation method is by changing the where clause of the SQL statement to get different results.
- Code Injection: Code injection is a process of inserting new SQL statements or database commands into the vulnerable SQL statement. One of the code injection attacks is to append a SQL Server EXECUTE command to the vulnerable SQL statement. This type of attack is only possible when multiple SQL statements per database request are supported.
- Function Call Injection: Function call injection is a process of inserting various database function calls into a vulnerable SQL statement. These function calls could make operating system calls or manipulate data in the database.

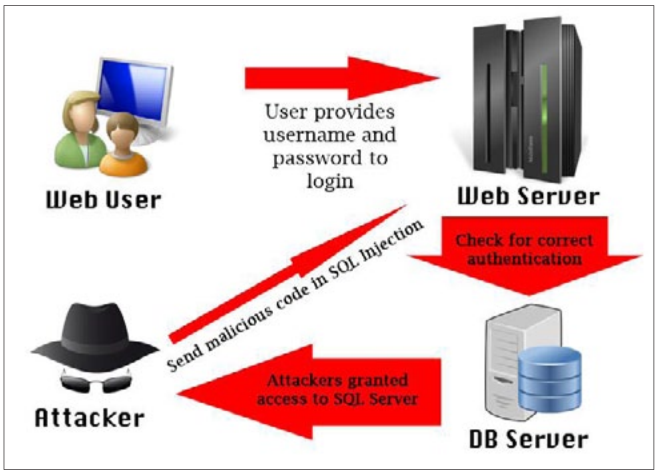

Web User
User provides username and password to login
Web Server
Check for correct authentication
Send malicious code in SQL Injection
Attacker
Attackers granted access to SQL Server
DB Server

Figure 1. *Illustration of how the attacker attacks using SQL Injection*

- Buffer Overflows: The usage of the call injection function causes buffer overflow. For most of the commercial and open source databases, patches are available. This type of attack is possible when the server is un-patched (Figure 1).

How it happens

Sometimes a programmer will make a mistake by making a web application, which is vulnerable to SQL Injection such as web form, cookie, parameters and etc. However, it does not validate values from application before passing them to be executed to SQL queries on a database server. As a result, the attacker will be able to manipulate the values by making the data visible as a code instead of data (Figure 2).

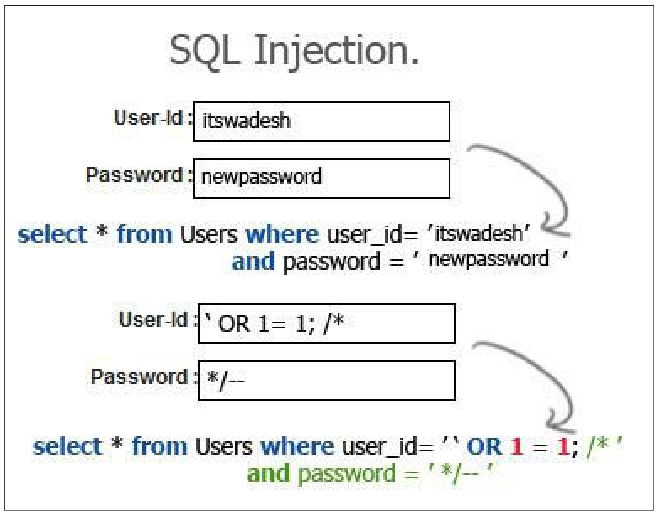
Meanwhile, the attacker can bypass authorization and authentication of the system and gain control of the database. He can destroy, change or steal all sensitive data for personal use or to cause harm to the victim. In the example below, simple SQL Injection attacks show how an attacker can manipulate SQL query to perform attacks.

Let's look at usual PHP query for login:

```
$sql="SELECT * FROM user_table WHERE username=
'".$_POST['username']."' AND password= '".$_POST
['password']."'"; $result=mysql_query($sql);
```

Usually people might think that only authorized user can log in into the system, but that's not true. Attackers might be able to log in into the system using a simple trick.

Let's say an attacker exploits the query by injected *A' OR 'A'='A* in the username field and *A' OR 'A'='A* in the password field. Then the final query will be like this:



```
SQL Injection.

User-Id : itswadesh
Password : newpassword

select * from Users where user_id= 'itswadesh'
        and password = ' newpassword '

User-Id : ' OR 1= 1; /*
Password : */--

select * from Users where user_id= '' OR 1 = 1; /* '
        and password = ' */-- '
```

Figure 2. *The explanation how attackers can manipulate SQL Query*

```
SELECT * FROM user_table WHERE username='A' OR
'A'='A' AND password='A' OR 'A'='A';
```

The above query is always true and returns the row from the database and the attacker can log in into the system.

In the worst scenario, attackers could make damage to the database by using *DROP TABLE* command if the database user has drop privilege into that database.

Lets suppose a query in an order page:

```
$sql="SELECT * FROM order WHERE order_id= '".$_
         GET['order_id']."'";
```

The attacker will inject SQL command in the URL of the page, the code might be like this `12'; DROP TABLE order; #` and the URL looks like this:

```
http://victim.com/order.php?id=12'; DROP TABLE
         order; #
```

Now the final query will become like this:

```
SELECT * FROM order WHERE order_id='12'; DROP
         TABLE order; #';
```

The hash symbol `#` tells the MYSQL server to ignore the rest of the query. In this query, it simply ignores the last single quote (') of the query.

How to prevent

Despite thousands of attacks of SQL Injection on Internet websites, there are also numerous methods of its prevention. The most popular techniques used are code level and platform level defenses.

Code Level Defenses

Secure coding behavior is very important in order to prevent SQL injection attacks. According to The *Open Web Application Security Projects* (OWASP), in order to avoid SQL injection flaws, developers need to either stop writing dynamic queries or prevent user-supplied input, which contains malicious SQL from affecting the logic of the executed query.

Therefore, in order to reduce or eliminate the threat of SQL injection, you can do some code-level prevention along with secure coding behavior.

Parameterize Statements

One of root cause of SQL injection is the vulnerable design of SQL queries that are sent to the database for execution, which is also known as a dynamic string building or dynamic SQL.

Frequently known as parameterize statements, a more secure alternative to dynamic string building by providing parameters to an SQL query through the use of placeholders, or bind variables, instead of working directly with the user input. These techniques are the most secure alternative that can help to avoid or prevent common SQL injection problems within an application to replace an existing dynamic query. They are also very well organized as it optimized the query based on the supplied prepared statement and increasing the performance of subsequent queries.

Parameterize queries require the developer to make a prior definition of all the SQL code, and then pass in each parameter to the query. Using this method of programming, it allows the database to differentiate between code and data; regardless of what user input is given.

They are also easiest to adopt, and work in fairly similar ways among most web technologies in use today, including:

- Java
- .NET
- PHP

However, not all dynamic statements can be parameterized. You cannot parameterize SQL identifiers or keywords besides data values. The following examples show statements that can't have parameterized statements:

```
SELECT * FROM ? WHERE username = 'zaidi'
SELECT ? FROM users WHERE username = 'zaidi'
SELECT * FROM users WHERE username LIKE 'z%'
ORDER BY ?
```

Safe Java Parameterized Statements

Java provides Java Database Connectivity (JDBC) framework that has the ability to use parameterized statements through *PreparedStatement* class.

A *PreparedStatement* is a precompiled SQL statement that can be implemented multiple times without having to recompile for every execution.

Vulnerable Java Statement examples

The example code below is vulnerable to SQL Injection because it uses dynamic queries to concatenate malicious data to the query itself. Notice that it uses the Statement class instead of the *PreparedStatement* class.

```
String query = "SELECT * FROM users_list WHERE
username ='"+ username + "'" + " AND password='" +
                password + "'";
Statement stmt = connection.createStatement();
ResultSet rs = stmt.executeQuery(query);
```

The below code is also vulnerable to SQL Injection. Although it uses the *PreparedStatement* class it is still query dynamic SQL via string concatenation.

```
String query = "SELECT * FROM users_list WHERE
username ='"+ username + "'" + " AND password='" +
                password + "'";
PreparedStatement stmt = connection.
                prepareStatement(query);
ResultSet rs = stmt.executeQuery();
```

Secure Java Statement Example

The example below is one of secure code to SQL Injection because it properly uses parameterized queries. By applying Java's *PreparedStatement* class, bind variables (using the question marks) and the corresponding *setString* methods, we can defense from SQL Injection attacks.

```
PreparedStatement stmt =
connection.prepareStatement("SELECT * FROM
users_list WHERE username=? AND password=?");
stmt.setString(1, username);
stmt.setString(2, password);
ResultSet rs = stmt.executeQuery();
```

Safe .NET (C#) Parameterized Statements

Microsoft .NET have various methods to parameterize statements by using ADO.NET framework that allow us to check the parameters passing in or supplied. There are four data providers of ADO.NET such as *System.Data.SqlClient* for Microsoft

Table 1. *Syntax of Parameterized Statement*

Database Type	Data Provider	Parameter Syntax
Microsoft SQL	System.Data.SqlClient	@parameter
Oracle	System.Data.OracleClient	:parameter (only applies in parameterized SQL command text)
OLE	System.Data.OleDb	Positional parameters with a question mark placeholder (?)
ODBC	System.Data.Odbc	Positional parameters with a question mark placeholder (?)

SQL Server, *System.Data.OracleClient* for Oracle databases, *System.Data.OleDb* for OLE DB and *System.Data.Odbc* for ODBC data sources.

However, each provider has different syntax for parameterized statement implementation as shown in Table 1.

Vulnerable .NET Statement Examples

The below instance is vulnerable to SQL Injection if the data input is not properly validated and sanitized. If an attacker could inject a single quote (') character, it would be possible to modify the SQL statement (Listing 1).

The below example is also similar and vulnerable as the previous example, though the stored procedure is not vulnerable to SQL Injection. The attacker could inject a single quote (') character that would be possible to append on to the EXEC statement and execute extra commands (Listing 2).

Listing 1. *Modify SQL Statement*

```
using (SqlConnection connnection = new
                SqlConnection(ConnString))
{
string sql = "SELECT UserId FROM User WHERE
                " +"UserName = '" + UserName
                + "' AND " + "Password = '"
                + Password + "'";
using (SqlCommand cmd = new SqlCommand(sql))
{
cmd.Connection = connnection;
try
{
cmd.Connection.Open();
var userId = cmd.ExecuteScalar();
}
catch (SqlException sx)
{

}
}
}
```

Listing 2. *EXEC Statement*

```
string sql = "EXEC usp_login '" + UserName
                + "','" + Password + "'";
using (SqlCommand cmd = new
                SqlCommand(sql))
{
cmd.Connection = connnection;
try
{
cmd.Connection.Open();
var userId = cmd.ExecuteScalar();
}
catch (SqlException sx)
{

}
}
```

Listing 3. *SQL Injection*

```
// Build the query statement using
                parameterized query.

string sql = "SELECT UserId FROM User WHERE
                " + "UserName = @UserName
                AND Password = @Password";

using (SqlCommand cmd = new
                SqlCommand(sql))

{
// Create the parameter objects as specific
                as possible.
cmd.Parameters.Add("@UserName", System.
                Data.SqlDbType.NVarChar,
                50);
cmd.Parameters.Add("@Password", System.
                Data.SqlDbType.NVarChar,
                25);

// Add the parameter values.  Validation
                should have already
                happened.

cmd.Parameters["@Password"].Value =
                Password;
cmd.Connection = connnection;
try
{
cmd.Connection.Open();
var userId = cmd.ExecuteScalar();
}
catch (SqlException sx)
{

}
}
```

Secure .NET Parameterized Statement Examples (Using SqlClient provider)
The below code is an example of SqlClient provider parameterized statement which is safe to SQL Injection because it properly uses parameterized queries (Listing 3).

Secure .NET Parameterized Statement Examples (Using OracleClient provider)
The below code is an example of OracleClient provider parameterized statement which is safe to SQL Injection because it properly uses parameterized queries (Listing 4).

Secure .NET Parameterized Statement Examples (Using OleDbClient or Odbc provider)
The below code is an example of OleDbClient or Odbc provider parameterized statement which is safe to SQL Injection because it properly uses parameterized queries (Listing 5).

Safe PHP Parameterized Statements
PHP has various frameworks that we can use to access the database. The most common frameworks that facilitates parameterized statements are mysqli package for MySQL databases, `PEAR::MDB2` package and *PHP Data Objects* (PDO) framework.

Listing 4. *SQL Injection II*

```
// Build the query statement using
               parameterized query.
string sql = "SELECT UserId FROM User WHERE
               " + "UserName = :UserName
               AND Password = :Password";

using (OracleCommand cmd = new
               OracleCommand(sql))

{

// Create the parameter objects as specific
               as possible.
cmd.Parameters.Add("UserName", System.Data.
               OracleType.NVarChar, 50);
cmd.Parameters.Add("Password", System.Data.
               OracleType.NVarChar, 25);

// Add the parameter values.  Validation
               should have already
               happened.
cmd.Parameters["UserName"].Value =
               UserName;
cmd.Parameters["Password"].Value =
               Password;
cmd.Connection = connection;

try
{
cmd.Connection.Open();
var userId = cmd.ExecuteScalar();
}
catch (SqlException sx)
{

}
}
```

Listing 5. *SQL Injection III*

```
// Build the query statement using
               parameterized query.
string sql = "SELECT UserId FROM User WHERE
               UserName =? AND Password
               =?";

using (OleDbCommand cmd = new
               OleDbCommand(sql))

{

cmd.Parameters.Add("@UserName", System.
               Data.OleDbType.NVarChar,
               50);
cmd.Parameters.Add("@Password", System.
               Data.OleDbType.NVarChar,
               25);

// Add the parameter values.  Validation
               should have already
               happened.
cmd.Parameters["@UserName"].Value =
               UserName;
cmd.Parameters["@Password"].Value =
               Password;
cmd.Connection = connection;

try
{
cmd.Connection.Open();
var userId = cmd.ExecuteScalar();
}
catch (SqlException sx)
{

}
}
```

Vulnerable PHP Statements Example

The below code is using an old mysql library, which does not support prepared statements vulnerable to SQL injection. However, by either properly escaping or validating the user input, we could still avoid the vulnerability.

```
$input=$_GET["input"];
$con = mysql_connect('localhost', 'zaidi', 'pass123');
mysql_select_db("zaidi_demo", $con);
$sql="SELECT * FROM user WHERE id = '".$input."'";
$result = mysql_query($sql);
```

Secure PHP Statement Example (Using MySQLi Package)

One of the most commonly used database interfaces that support parameterized statements through placeholder question marks is mysqli package. The following example shows a parameterized statement using the mysqli package: Listing 6.

Secure PHP Statement Example (Using PEAR::MDB2 Package)

The `PEAR::MDB2` package is widely used and it supports named parameters using the colon character and using placeholder question marks. The example below shows the use of *MDB2* with placeholder question marks to build a parameterized statement (Listing 7).

Secure PHP Statement Example (Using PDO Package)

The PDO package is an object-oriented vendor-independent data layer. It supports named parameters using colon character and placeholder question marks. The following example demonstrates the usage of PDO with named parameters: Listing 8.

Stored Procedures

Stored procedures are programs stored within the database. You need to define the SQL code first, and pass in the parameters then. Stored procedure is different than prepared statements as the SQL code for a stored procedure is defined and stored in the database itself, and then called from the application.

Stored procedures can be very effective for mitigating the attack of SQL injection, as it is possible to configure access controls at the database level on most databases. As the result, the attackers are unable to access sensitive information if the permission is properly configured within the database.

Whitelist Input Validation

Input validation is one of prevention measure from SQL injection attacks. It can be used to detect unauthorized input before it is processed by the

Listing 6. *MySQL Package*

```
$con = new mysqli ("localhost","zaidi","pas
            s123","db");
$sql = "SELECT * FROM user WHERE username=?
            AND password=?;
$cmd= $con->prepare($sql);

//Add parameters to SQL query
$cmd->bind_param("ss", $username,
            $password); //bind parameter
            as strings
$cmd->execute ();
```

Listing 7. *MDB2 Usage*

```
$mdb2= & MDB2::factory ($dsn);
$sql ="SELECT * FROM user WHERE username=?
            AND password=?";
$types=array ('text','text'); //set data
            types
$cmd=$mdb2->prepare($sql,$types,MDB2_
            PREPARE_MANIP);
$data=array ($username, $password); //
            parameters to be passed
$result=$cmd->execute ($data);
```

Listing 8. *PDO Usage*

```
$sql="SELECT * FROM user WHERE
            username=:username AND"
            +"password=:password";
$stmt= $dbh->prepare ($sql);

//bind values and data types
$stmt->bindParam(':username',$username,
            PDO::PARAM_STR, 12);
$stmt->bindParam(':password',$password,
            PDO::PARAM_STR,12);

$stmt->execute();
```

application. In other words, it is a process of testing input received by comparing a standard compliance defined within the application. Whitelist input validation is a method that accepts only the input that is known to be good.

While constructing whitelist validation compliance, below specification should be considered:

- Data expected type
- Data length or size
- Data numeric range
- Data content

Building regular expression for whitelisting is quite complicated. Luckily, there are several website on the Internet providing tutorials on how to develop regular expression such as OWASP Validation Regex Repository (*https://www.owasp.org/index.php/OWASP_Validation_Regex_Repository*) and Regular Expression Info (*http://www.regular-expressions.info/*).

Example of PHP Input validation

In PHP, there are several of functions that can be used as input validation such as:

- `Preg _ match` (regex, matchstring) – regular expression match with *matchstring* using regex expression
- `Is _ <type>(input)` – Check whether the input is `<type>` i.e: `is _ numeric()`
- `Strlen(input)` – Check the length of the input

Example of `preg _ match` expression to validate form parameters in PHP:

```
$username = $_POST ['username']
if (!preg_match("/^[a-zA-Z]{8,12}$/D", $username)
{
//handle failed validation
}
```

Escaping User Input

User input can become additional threat to our system for SQL injections attacks. While you can't always avoid user input completely, the next best thing is to escape it. Escaping user input is not as excellent as limiting dynamic queries but still it can stop many SQL injection attacks.

For instance, in the MySQL, the extension for PHP provides the function `mysql_real_escape_string()` to escape input characters that are special to MySQL.

Example of MySQL Escape Input

```
if (get_magic_quotes_gpc())
{
```

Listing 9. *Example of Java Stored Procedures*

```
try {
CallableStatement cs = connection.prepareCall("{call sp_getAccountBalance(?)}");
cs.setString(1, custname);
ResultSet results = cs.executeQuery();
}
catch (SQLException se) {

}
```

Listing 10. *Example of .NET Stored Procedures*

```
Try
Dim command As SqlCommand = new SqlCommand("sp_getAccountBalance", connection)
command.CommandType = CommandType.StoredProcedure
command.Parameters.Add(new SqlParameter("@CustomerName", CustomerName.Text))
Dim reader As SqlDataReader = command.ExecuteReader()
 ' …
 Catch se As SqlException
 ' error handling
End Try
```

```
$name = stripslashes($name);
}
$name = mysql_real_escape_string($name);
mysql_query("SELECT * FROM users WHERE
                  name='{$name}'");
```

Turn Magic Quotes Off

Turning the `magic_quotes_gpc` variable off can also stop some SQL injection attacks. Unfortunately, this isn't always a reliable measure because sometimes the magic quotes might be off and you are unaware of this. Still, it is better than nothing.

In any case, you need to have code to substitute quotes with slashes. Here is the simplest way to do it:

```
if (!get_magic_quotes_gpc()) {
$username = mysql_real_escape_string ($username);
$password = mysql_real_escape_string s($password);
}
```

Handling Sensitive Data

Other method can be given greater emphasized is the storage and access of sensitive information within the database. Examples of types of information that might interest the attackers may include username and passwords, personal information or financial information such as credit card details. Therefore, additional controls over sensitive information should be considered. The following examples show controls or design decision to be considered:

- *Passwords*: If possible, you need to store all users' password outside the database or use another alternative – to store a salted one-way hash encryption (using a secure salted hash algorithm such as *SHA2*) for each user's password instead of the password itself. Salt, is an additional small piece of random data and it should be stored separately from the password hash.
- *Credit card or other financial information*: Store details such as credit card need to follow requirement of the *Payment Card Industry Data Security Standards* (PCI-DSS) in order to enhance data security. Encryption algorithm such as FIPS-certified approved should be implemented to credit cards information. Furthermore, other financial data such as bank account details need to be encrypted too.
- *Archiving*: Archiving or purging history of not needed sensitive information after some period of time is highly recommended. Where the application does not require this information after initial processing, it needs to be removed immediately.

Avoiding Obvious Object Names

The choice of names for critical objects such as encryption functions, password columns, and credit cards columns should be considered wisely. The attackers will try to guest the common tables and columns name in order to search for sensitive data such as *passwords*, *pass* or *passw*. Therefore, to make the attack more difficult, it would be a good idea to use uncommon table names for saving password information. Although this approach don't stop the attacks, it will help to make the attacker unable to identify this information immediately.

Platform Level Defenses

Deploying a secure platform to host the website is a very important component in order to prevent from SQL injections or other attacks. Some developers are not very much concerned about choosing or implementing a secure platform and just focusing on employing prevention at the code level only. As the result, the attacker may be able to compromise the system and cause harm to all information within application database. Hence, the below strategies and implementation techniques of securing the platform level from SQL injection are necessary.

Web Application Firewall

The most well-known runtime solution in web application security is the use of Web Application Firewall or known as WAF. A *web application firewall* (WAF) is a network appliance, server plugin, software-based solution or filter that applies a set of rules to an HTTP conversation and adds security features to a web application. By customizing the rules to your application, many attacks can be identified and blocked.

Nowadays, there are thousands of WAF either free open source WAF or commercial products on the market. In this article, we will cover some type of WAF for securing our application such as ModSecurity, WebKnight and some cloud-based WAF.

ModSecurity

ModSecurity (*http://www.modsecurity.org*) is one of most popular open source WAF that supports both Unix and Windows platform. It is implemented as an Apache module; however, it can protect virtually any web applications. ModSecurity can be used for attack prevention, monitoring, intrusion detection, and general application hardening. The strength of ModSecurity is its rule language, which is a combination of configuration directives and a simple programming language applied to HTTP request and response. The ModSecurity Core Rule Set includes blacklist rules for SQL injection and blind SQL injection, which depends on the application. However, it could generate false positives as other WAFs if they are not properly tuned (Figure 3).

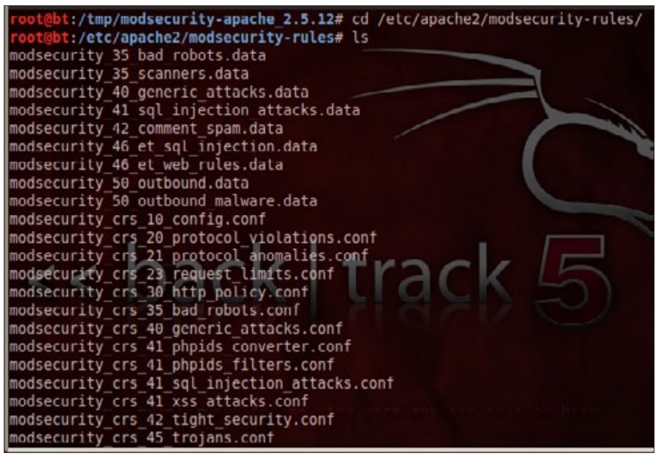
WebKnight

WebKnight (*http://www.aqtronix.com*) is an IIS ISAPI filter that secures your web server by blocking certain malicious requests. It can check POST data for malicious input, highly configurable and comes with GUI. This is done through the process of scanning all requests and processing them based on filter rules, set by the administrator. These rules are not based on a database of attack signatures that require regular updates. Unfortunately, WebKnight does not support regular expressions and it is limited to blacklist keyword validation.

Cloud-based WAF

Cloud-based Web Application Firewall is the newest web application security technologies and part of the *web application firewall* (WAF). This technology is unique due to the fact that it is platform elasticity and does not require any hardware or software changes on the host. Most Cloud-based WAF requires a DNS change, where all web traffic is routed through the WAF then it is inspected and threats are thwarted.

Cloud-based WAFs are typically centrally orchestrated, which means that threat detection information is shared among all the users of the service and this collaboration results in improved detection rates and lower false positives.



```
root@bt:/tmp/modsecurity-apache_2.5.12# cd /etc/apache2/modsecurity-rules/
root@bt:/etc/apache2/modsecurity-rules# ls
modsecurity_35_bad_robots.data
modsecurity_35_scanners.data
modsecurity_40_generic_attacks.data
modsecurity_41_sql_injection_attacks.data
modsecurity_42_comment_spam.data
modsecurity_46_et_sql_injection.data
modsecurity_46_et_web_rules.data
modsecurity_50_outbound.data
modsecurity_50_outbound_malware.data
modsecurity_crs_10_config.conf
modsecurity_crs_20_protocol_violations.conf
modsecurity_crs_21_protocol_anomalies.conf
modsecurity_crs_23_request_limits.conf
modsecurity_crs_30_http_policy.conf
modsecurity_crs_35_bad_robots.conf
modsecurity_crs_40_generic_attacks.conf
modsecurity_crs_41_phpids_converter.conf
modsecurity_crs_41_phpids_filters.conf
modsecurity_crs_41_sql_injection_attacks.conf
modsecurity_crs_41_xss_attacks.conf
modsecurity_crs_42_tight_security.conf
modsecurity_crs_45_trojans.conf
```

Figure 3. *ModSecurity list of rules*

Cloud-based WAF is and ideal for cloud-based web applications and website owner that need web application security with low budget and did not want to or was not able to make software or hardware changes to their systems. Some of the most popular Cloud-based WAF are Cloudflare, WAPPLES V-Series, XyberShield and Amazon EC2 Cloud-based Web Application Firewall (Figure 4).

Application Intrusion Detection and Prevention Systems

Traditionally network-based IDSs can be used to detect SQL injection attacks; however, they are often not optimal as they are not inside the webserver and the application. Yet, you could still use them for initial line of defense (Figure 5).

A WAF can serve as good IDS because it runs on the application layer and can be tuned for the specific application being protected. Good IDS/IPS capable of detecting attacks, blocks them and alerts administrators who can then make decision whether to be done about the vulnerability.

PHPIDS

PHPIDS (*https://phpids.org/*) is one of the good options of IDS as it does not filter or sanitize input, but rather detect attacks and takes action based on its configuration. Meanwhile, PHPIDS can be

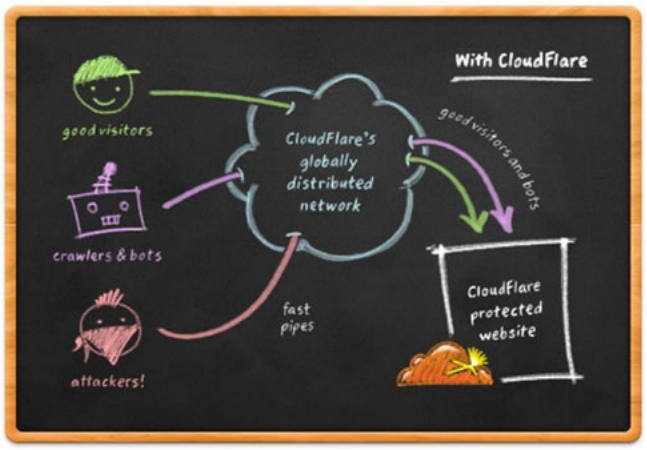


Figure 4. *CloudFlare is one of Cloud-Based WAF*

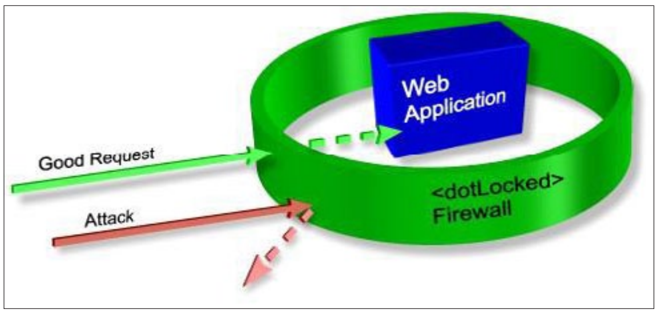


Figure 5. *How WAF functions to defense from attacks*

integrated with ModSecurity in order to perform excellent prevention. PHPIDS functions range from simple loggings to sending out an emergency email to the development team, displaying a warning message for the attackers or even sending the user's session.

MyPHPIPS

MyPHPIPS (*MyPHP Intrusion Prevention System*) is an open source PHP Web Application Intrusion Prevention System. It was based on PHPIDS (*http://phpids.org*) and distributed under the LGPL License. This work is supported by CyberSecurity Malaysia.

MyPHPIPS intends to assist the web developer/maintainers to secure their PHP CMS/application deployments without having with minimal resources (i.e time and money). It is a portable and less-hassle framework that serves as an extra security layer to defend against invalid/malicious requests to the web application or content management systems. It uses PHPIDS to calculate an impact value that can come from malicious requests and reacts to them. The attack recognition is based on a set of approved and heavily tested filter rules where any attack is given a numerical impact rating which makes it easy to decide what kind of action should follow the hacking attempt.

MyPHPIPS is available for download at the Google Code repository *http://code.google.com/p/myphpips/downloads/list*.

Disable Shell

Using somewhat advanced SQL injection; the attacker could inject a new PHP file into the web root of the PHP server using SQL injection vulnerability in web application. The injection is a command shell written in PHP that gives root access to the operating system. The example of PHP shell code is as below:

```
<?php echo shell_exec('cat '.$_GET['command']); ?>
```

Luckily, disabling all shell functions can prevent it. Get rid of any database functionality that you don't need to prevent a hacker taking advantage of it. For example, the *xp_cmdshell* extended stored procedure in MS SQL spawns a Windows command shell and passes in a string for execution, which could be very useful for a hacker. The Windows process spawned by *xp_cmdshell* has the same security privileges as the SQL Server service account.

Limit Discovery

Search engine such Google, Yahoo or Bing is very useful in our daily life especially to search for some information. However, the attackers could use this tool to find SQL injection vulnerabilities in our website. Most of the major search engines provide steps and online tools for removing your website contents from their indexes and caches.

One of the methods which is common across all major search engines is the use of *robots.txt* file in the root directory of your website, which can be used to prevent crawlers from indexing the site. Other method is to put *noindex* in Meta tag of the web applications. The examples of each method are shown below:

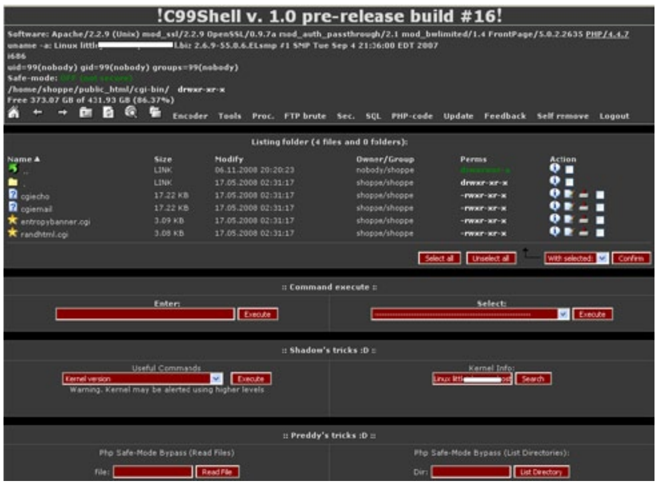
Example of robots.txt

```
User-agent: *
Disallow: /
```

```
Example of Meta Tag
<meta name="robots" content="noindex">
```

Least Privileges

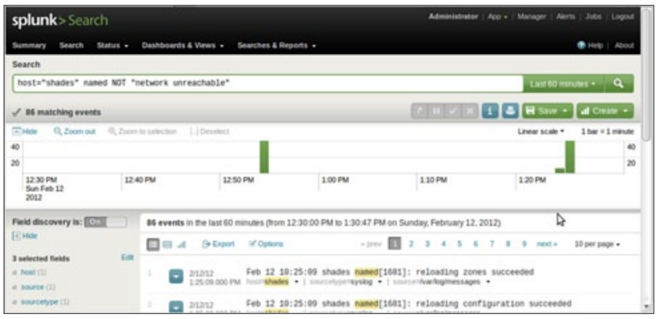
To prevent the potential damage of a successful SQL injection attack, you should limit the privileges assigned to every database account in your system. Do not assign DBA or admin type access



```
!C99Shell v. 1.0 pre-release build #16!
Software: Apache/2.2.9 (Unix) mod_ssl/2.2.9 OpenSSL/0.9.7a mod_auth_passthrough/2.1 mod_bwlimited/1.4 FrontPage/5.0.2.2635 PHP/4.4.7
uname -a: Linux littl[redacted].biz 2.6.9-55.0.6.ELsmp #1 SMP Tue Sep 4 21:36:00 EDT 2007
i686
uid=99(nobody) gid=99(nobody) groups=99(nobody)
Safe-mode: OFF (not secure)
/home/shoppe/public_html/cgi-bin/   drwxr-xr-x
Free 373.07 GB of 431.93 GB (86.37%)

Listing folder (4 files and 0 folders):
Name               Size      Modify               Owner/Group      Perms
..                 LINK      06.11.2008 20:20:23  nobody/shoppe    drwxr-xr-x
.                  LINK      17.05.2008 02:31:17  shoppe/shoppe    drwxr-xr-x
cgiecho            17.22 KB  17.05.2008 02:31:17  shoppe/shoppe    -rwxr-xr-x
cgiemail           17.22 KB  17.05.2008 02:31:17  shoppe/shoppe    -rwxr-xr-x
entropybanner.cgi  3.09 KB   17.05.2008 02:31:17  shoppe/shoppe    -rwxr-xr-x
randhtml.cgi       3.08 KB   17.05.2008 02:31:17  shoppe/shoppe    -rwxr-xr-x
```

Figure 6. *PHPShell have shell command functions*



```
host="shades" named NOT "network unreachable"
86 matching events
86 events in the last 60 minutes (from 12:30:00 PM to 1:30:47 PM on Sunday, February 12, 2012)
1  2/12/12 1:25:09.000 PM  Feb 12 10:25:09 shades named[1681]: reloading zones succeeded
   host=shades | sourcetype=syslog | source=/var/log/messages
2  2/12/12                 Feb 12 10:25:09 shades named[1681]: reloading configuration succeeded
```

Figure 7. *Splunk can monitor web logs realtime*

References

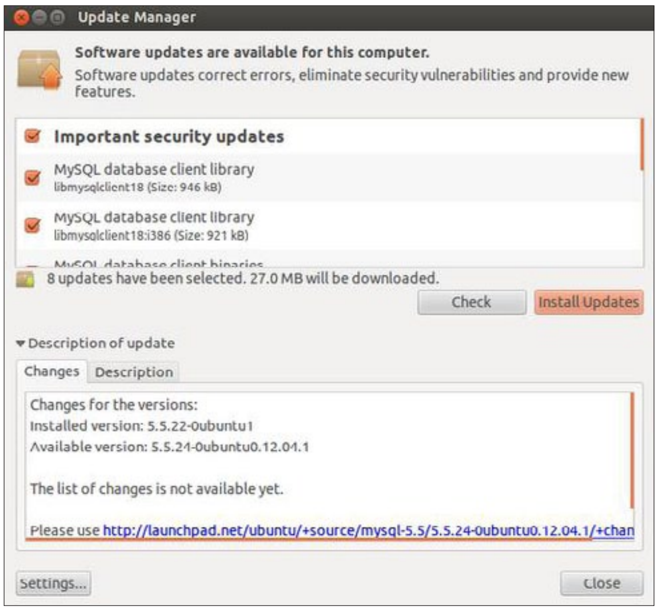
SQL Injection Attacks and Defense, 2009, Justin Clarke, Syngress Publishing Inc,

- *http://roshanbh.com.np/2007/12/sql-injection-attack-examples-and-preventions-in-php.html*
- *https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet*
- *https://www.owasp.org/index.php/Preventing_SQL_Injection_in_Java*
- *https://www.owasp.org/index.php/Guide_to_SQL_Injection*
- *http://software-security.sans.org/developer-how-to/fix-sql-injection-in-java-using-prepared-callable-statement*
- *http://software-security.sans.org/developer-how-to/fix-sql-injection-microsoft-.net-with-parameterized-queries*

rights to your application accounts. It is very dangerous if you assign DBA or admin type access rights to your application accounts. To determine the permission assigned to a database login, find its role membership and remove any unnecessary or privileged roles such as the public or database administrator role. Make sure that accounts that only need read access are only granted read access to the tables they need access to. If an account only needs access to portions of a table, consider creating a view that limits access to that portion of the data and assigning the account access to the view instead, rather than the underlying table.

Web Server Logs

Web server log files can provide us with some insight into potential SQL injection attacks. Apache and IIS log the vulnerability in a URL parameters by default. You might consider to log the Referrer and Cookie headers but this will increase the size of the log files, but it is provide potential security benefits which are another potential location for SQL injections vulnerabilities to materialize.



Software updates are available for this computer.
Software updates correct errors, eliminate security vulnerabilities and provide new features.

Important security updates
MySQL database client library
libmysqlclient18 (Size: 946 kB)
MySQL database client library
libmysqlclient18:i386 (Size: 921 kB)
MySQL database client binaries

8 updates have been selected. 27.0 MB will be downloaded.

Changes for the versions:
Installed version: 5.5.22-0ubuntu1
Available version: 5.5.24-0ubuntu0.12.04.1

The list of changes is not available yet.

Please use http://launchpad.net/ubuntu/+source/mysql-5.5/5.5.24-0ubuntu0.12.04.1/+chan

Figure 8. *Frequent systems update can harden your platform*

You can use some automated logging facilities too. The most popular automated web server logging facility is Apache Scalp (*http://code.google.com/p/apache-scalp/*) and Splunk (*http://www.splunk.com/*) (Figure 7).

Keep latest updates and patches

It is really necessary to keep updates and having the latest patches. The platform such as Linux or Windows will have frequent updates and we need to download these and install them into our systems. These updates will patch any bugs and security vulnerability in our platform (Figure 8).

Summary

Overall security measurement should be given a serious thought in order to prevent from SQL injection attacks whether by implementing Code-Level Defenses or Platform-Level defenses. SQL injection is one of most favorite attacks methods used by the attackers and thousand prevention methods have been invented in order to avoid these attacks. You can look at all of the solutions presented and determine the best and integrate it into your applications. All these methods need to be combined in order to keep you secure from SQL injection attacks.

AHMAD ZAIDI SAID

Ahmad Zaidi Said held position as an Intrusion Analyst at Malaysian Computer Emergency Response Team (MyCERT), CyberSecurity Malaysia. His education background comprises Degree in Engineering, majoring in Manufacturing from International Islamic University Malaysia. He has much experience in the areas of network security, penetration testing, web security, incident/exploit analysis, code obfuscation, honeypot technology, system infrastructure development & deployment and cloud computing security. He has obtained an Ec-Council Network Security Administrator (ENSA) certificate in 2008.

He can be contacted at LinkedIn http://my.linkedin.com/in/ahmadzaidi or email to zaidi@cybersecurity.my.

Secure Thy Databases

In today's world, databases are the backbone of IT enabled enterprises. Due to the importance of data stored, it has been recognized as a business need to protect these databases from adversaries. However, there still appears to be lack of information on what the various risks to databases are and how these risks can be mitigated effectively.

What you will learn…

- Major threats to databases and respective countermeasures
- Designing and developing un-injectable database interfaces
- Things to remember while storing PII in the databases
- Secure coding techniques for database interaction
- Securing database to database communication

What you should know…

- The importance of securing databases
- Web application programming

We have come a long way since a bunch of non-interactive pages written in plain HTML were considered to be a website and it was more than enough for an organization to show its web presence.

Today's websites are highly interactive, developed using sophisticated server side scripting code and with a relational database at the back storing and serving the data needed for a web application to work.

More often than not, this data is of sensitive nature and usually contains:

- Financial data of the organization
- Customer records including PII
- Organization's intellectual property related to their existing or secret project

It is not hard to figure out that a leak in such databases can have a huge impact on the organization and hence it needs to be protected. Fortunately, the majority of organizations understands this as a business requirement and is putting efforts as well as money on trying to be secure.

But the question is – Are these efforts enough?

'Heartland Payment Systems' and 'Sony PlayStation' hacking incidents are prime examples of how

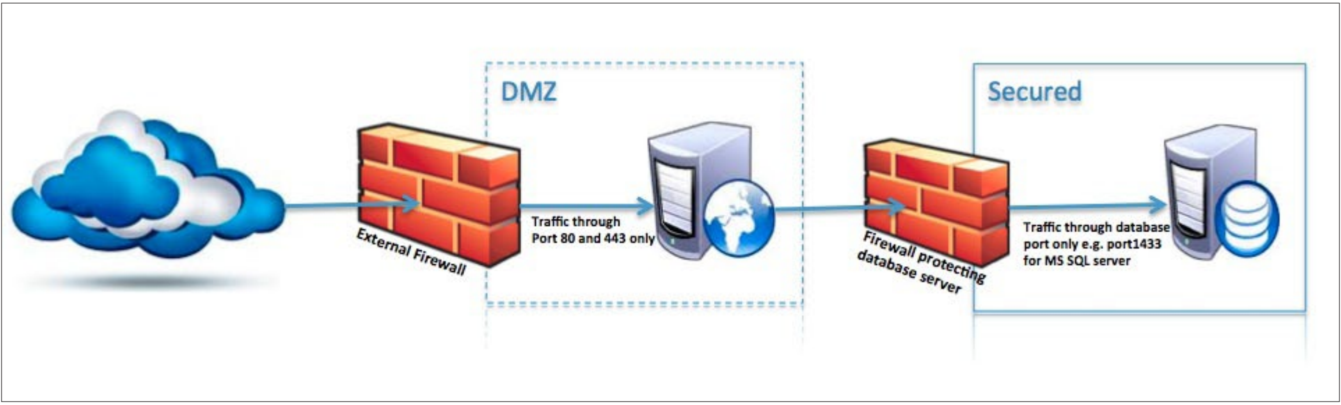

DMZ
Secured
External Firewall
Traffic through Port 80 and 443 only
Firewall protecting database server
Traffic through database port only e.g. port1433 for MS SQL server

Figure 1. *Common web application deployment scenario*

lack of appropriate security measures has lead to big data breaches'

To answer this question, let's look at the following figure depicting a common web application deployment scenario: Figure 1.

The Figure 1 demonstrates a web application residing on the web server located in DMZ. This is firmly protected using a firewall that only allows data through port 80 and 443.

The application interacts with a database server located behind another firewall that only allows communication through database communication ports such as 1433 for MS SQL Server.

Even though the database is behind two firewalls configured with appropriate rules, *it still does not make the database secure* and the reason is:

"Databases tend to trust the applications and users interacting with it"

This means the real threats to databases are:

- Insecure applications,
- Other 3rd party databases linked for various business purposes
- Authorized (Internal) users with direct access to databases

Considering databases trust and respond to all the requests coming from authorized users, securing them appears to be a non-trivial task – but it's not.

In the following sections, we will discuss various techniques we can implement to protect our databases from these threats.

Avoiding Application Insecurities

There are various ways adversaries may take advantage of insecurities in the applications to gain access to underlying databases such as:

- SQL Injection attacks
- SQL Truncation attacks
- Exposed database connection strings

The key to protect from such attacks is to carefully design and develop a defensive application's database interface with a proactive security approach.

(To stay within the scope of the article we will only be covering the techniques to secure the databases and not the details on the attack techniques.)

Designing Secure Database Access Layer

If you are an application architect and you are responsible to design the database access layer, it is highly recommended that you apply the following principles in your next application:

- All the database access calls must be handled at one place only and the tier calling the database should be a different than the web tier e.g. *Database Access Objects* (DAO) which decouples the web tier from database.
- All the table attributes must be strongly typed i.e. defined with appropriate data-type and size (minimum length, maximum length, Null). Data types such as *varchar* or *nvarchar* should be used sparingly and must not be defined with unnecessary long sizes.
- Input validation becomes utmost necessity while trying to develop un-injectable interfaces. All the input must be filtered before sending to database as per the following considerations:
  - Input data MUST be filtered against a white list (using *regex* or *other string functions*). All the special characters such as ', # and – must be stopped unless escaped using proper methods or functions provided by the vendor (Be careful while using self coded functions to escape the malicious characters).
  - Length of input data MUST be validated against the attribute-size before passing it on to database. The value being passed MUST be less than or equal to attribute size specified in the table/database.
  - Special care must be taken while validating the size of the input value if special characters are being escaped at database or application level. The size of resultant value (after adding escape characters) must not exceed the size of attribute.
    - *This is because escaping (manual or using system function such as QUOTENAME) is usually done via inserting extra characters in front of the special (malicious) characters which increases the size of input value which increases the possibility of SQL Truncation attacks. (See OWASP Testing Guide and ADSR for more details on this type of attack.)*
  - Client side validation is not enough and should not be trusted.
- Data flowing across a trust boundary (e.g. *from the web tier to business logic tier* or *from business logic tier to database tier*) must be thoroughly validated. Even the data coming from other existing databases (e.g. Legacy applications) must be thoroughly verified and validated.

- *Often, database is considered as a part of the application and therefore the database accounts used by web applications often are given privileges more than required which disallow database server to defend itself against access to or modification of unauthorized schema or information and because of this an application level breach may result in a full database level security incident.*
  - Database administrator level accounts such as *sa* or *db_owner* have privileges more than those required by the application. Such accounts can be misused to access system objects and/or modify database schema. Therefore, web applications should use lesser-privileged accounts which are not allowed to make schema changes, call system objects (e.g. *xp_cmdshell* stores procedure) or request data not related to application.
- All of the database control should be on database layer itself and not the application layer.
- Another important feature web applications must provide is *Transaction Traceability*. *Transaction Traceability* means that all the operations on the database could be traced back to the user who requested the operation. This can easily be achieved by passing the user's identity information such as *User name* with the SQL Statement which then be stored in the database in appropriate table.
- Before performing any operation on the database, application MUST verify if the user is authorized to perform the operation. Before allowing any sensitive transaction, User should be asked to provide the *password* once again even if the authentication has been performed earlier.
- Web applications must be *fail safe* and should not leave the application in an unknown state if database throws an error. Application must have adequate error handling routines and central error logging mechanism which must cater for database generated errors as well such as query failures, connection problems, low memory etc. Error handling routine must:
  - Handle all the database generated errors and prevent showing any sensitive or crucial database information.
  - Must not display database generated errors to the user and should display a generic and friendly *error message* instead.
  - The error must be logged to a central system with all the related information e.g.
    - Error No and Error message,
    - Module name where error occurred.
    - Username (No passwords should be logged)

Securing PII (Personally Identifiable Information)

Encrypting PII (collected through the application) in transition and at rest strengthens the security of the application and mitigates the risk of details being stolen. The application architect must identify the personal data being collected and determine if it really needs to be collected and if so, this data must be categorized as *highly-sensitive*, *sensitive* and *non-sensitive* data.

All the information classified as 'sensitive' must be in encrypted form while in transit i.e. the data should be encrypted at one endpoint and decrypted at the another end point. This data is stored in the database unencrypted. Examples of such type of data include User names, Details collected via registration form (Name, Address, Date of Birth and Phone Number).

Whereas, the *highly-sensitive* information must not transit through layers in *clear text* and should also be stored in the database in encrypted form only. Examples of such data are Passwords, SSN and Credit card information.

The encryption of data in transit can be achieved via tunneling protocols such as SSL which will prevent it from being sniffed and MITM attack.

To protect the *highly-sensitive* information it must be encrypted (or decrypted) on the application layer only using strong industry standard 'encryption' (e.g. *AES*) algorithms.

For passwords, Hash values must be calculated using one-way hashing algorithms (e.g. *MD6*, *SHA256*) and this hash value should be stored in the database.

The protection of the PII through encryption cannot be considered complete unless there is a good *encryption key management* process in place. For this purpose, an enterprise-wide key management solution must be employed.

Secure Database Connection Strings

The most common method for an application to connect to the database is by using connection strings. These connection strings contains vital information about the database and type of connection established between web application and the database. This data includes *Database Name*, *Database location i.e. Hostname or IP Address*, *Type of technology used to communicate with database e.g. ODBC, OLEDB etc*, *Username* and *Password*. And as these are rarely encrypted, they allow a remote attacker who has shell access to perform direct operations against the database or back end systems, thus providing a leap point for total compromise.

Although there are several places where developers like to store the connection strings e.g.

- Within the source code
- Windows registry
- include (Plain text) files
- web.config and global.asa

But none of these methods ensure the 100% safety of connection string. We suggest that these connection strings should be stored outside the code and MUST not be stored in plain text. If the application is being developed using JAVA, *OWASP ESAPI's* encryption classes are highly recommended for this purpose.

For Microsoft.Net, the developers may also use *DPAPIProtectedConfigurationProvider* (DPAPI) or *RSAProtectedConfigurationProvider* to encrypt the connection strings stored in web.config file.

Avoiding SQL Injection Attacks

SQL Injection is not a new vulnerability and has been around for more than 10 years. But it is still one of the most widely exploited vulnerability (as per OWASP Top 10). The primary cause for SQL Injection is use of Dynamic SQL Statements. A very simple example of dynamic query is as below:

```
$sql = "select * from USER where username='"+
          username + "'";
```

Such dynamic SQL statements are nearly impossible to make secure and hence must be avoided. The recommended way of interacting with databases is using strongly typed parameterized queries.

Let's go through some of the techniques to code secure database interfaces:

Writing safe parameterized queries

When using parameterized queries, application passes the parameter values separately along with the query to the database. This allows the database to distinguish between executable SQL statements and untrusted data at the execution time. Therefore, the actual query is not modified i.e. any input passed as parameter will remain a parameter during the execution rather than becoming a part of the SQL Statement thus suppressing many varieties of SQL Injection attacks.

Validating the data against a white list as well as for size before assigning to the STRONGLY

Listing 1. *Writing Parameterized Queries in Java*

```
String selectStatement = "SELECT * FROM User WHERE useriD=?";
PreparedStatement prepStmt=con.prepareStatement(selectStatement);
prepStmt.setString(1,userId);
ResultSet rs = prepStmt.executeQuery();
```

Listing 2. *Writing Parameterized Queries in .Net*

```
myCommand=new SqlCommand("select * from User where userID=@userId");
myCommand.Connection=con;
myCommand.Parameters.Add(new SqlParameter("@UserId",System.Data.SqlDbType.VarChar,
                12,"UserId"));
myCommand.Parameters["@UserId"].Value=txtUserId.Text;
reader= myCommand.ExecuteReader();
```

Listing 3. *Writing Parameterized Queries in PHP (Using PDO)*

```
$stmt = $con->prepare ("INSERT INTO user (userId, userName, userEmail)
                         VALUES (:userid, :username, :useremail)");
$stmt->bindValue (":userid", $id);
$stmt->bindValue (":username", $name);
$stmt->bindValue (":useremail", "$email");
$stmt->execute ();
```

TYPED parameters further reduces the chances of SQL injection (Listing 1-3).

Writing safe stored procedures

Developers very often use stored procedures rather than embedding SQL Statements in the source code due to variety of reasons that include encapsulation, modularity and execution speed. Other than these reasons, there is a belief in developer community that stored procedures are injection-safe. This may not be true unless the stored procedures are created with due care and diligence (Listing 4).

The stored procedure given above is vulnerable to SQL Injection as the query is formed by concatenating the parameter (user's input) with the SQL Statement and passed directly to the '*EXEC*' function which takes a 'SQL statement' as a parameter.

The stored procedure given above can be made secure by using *sp_executesql* instead of *EXEC*. The following snippet demonstrates the secure way of writing the above example stored procedure: Listing 5.

This is a safer way to write a SQL Statement within stored procedures. Unlike '*EXEC*', '*sp_executesql*' is a system stored procedure which takes 'first parameter' as 'parameterized SQL statement' and second parameter as a list of parameters for that query (passed as first parameter). Therefore any input passed as parameter will remain a parameter during the execution rather than becoming a part of the query.

Similarly, For Oracle and MySQL bind variables should be used.

Using ORM Safely

It is commonly thought that ORM layers, like Hibernate are immune to SQL injection. This is not the case as Hibernate includes a subset of SQL called HQL, and allows "native" SQL queries. Often the ORM layer only minimally manipulates the inbound query before handing it off to the database for processing.

If using Hibernate, do not use the depreciated `session.find()` method without using one of the query binding overloads. Using session.`find()` with direct user input allows the user input to be passed directly to the underlying SQL engine and will result in SQL injections on all supported RDBMS.

```
Payment payment = (Payment) session.find("from com.
                    example.Payment as
payment where payment.id = " + paymentIds.get(i));
```

The above Hibernate HQL will allow SQL injection from paymentIds, which are obtained from the user. A safer way to express this is:

```
int pId = paymentIds.get(i);
TsPayment payment = (TsPayment)
session.find("from com.example.Payment
as payment where payment.id = ?", pId, StringType);
```

It is vital that input is properly escaped before use on a SQL database. Luckily, the current Oracle JDBC driver escapes input for prepared statements and parameterized stored procedures. However, if the driver changes, any code that assumes that input is safe will be at risk.

Listing 4. *Example of an Injectable Stored Procedure*

```
CREATE PRCEDURE SP_SearchUser @username varchar(55) = NULL AS
DECLARE @sql nvarchar(255)
SELECT @sql='SELECT UserId, UserEmail, UserAddress FROM User Where'
If @username is NOT NULL
        SELECT @sql=@sql + 'userName Like `'' + @username +''''
EXEC (@sql);
```

Listing 5. *Example of a safe Stored Procedure*

```
CREATE PRCEDURE SP_SearchUser @username varchar(55)=NULL AS
DECLARE @sql nvarchar(255)
SELECT @sql='SELECT UserId, UserEmail, UserAddress FROM User Where'
If @username is NOT NULL
        SELECT @sql=@sql + 'UserName Like @username'
EXEC sp_executesql @sql, N'@username varchar(55),@username
```

The application should:

- Ensure that all native queries are properly escaped or do not contain user input
- Ensure that all ORM calls which translate into dynamic queries are re-written to be bound parameters
- escape data as per the persistence layer's requirements to avoid SQL injections
- Have at least one automated test which should try to perform a SQL injection. This will ensure that the code has an extra layer of defense against SQL injections, and ensure that if this control fails, that the likelihood of the injection working is known.

Securing Database to Database Communication Links

Database links are used to expose objects of one database to another database. It is a widely adopted practice to create 'database links' to transfer data between two or more databases.

This is mostly seen in situations in which databases are located in different security zones e.g. the database located in DMZ and serving the company's e-commerce website is linked to company's ERP system's database (sometimes known as core database) located in highly protected internal network. If not created properly, such links can prove to be very dangerous as you might be opening access to core database for an unauthorized user as shown in Figure 2. How secure your ERP system's database may be, if the external database is compromised, the credentials to access the link and ultimately the core database could be compromised. Other possible risks posed by database links are:

- If there is SQL Injection vulnerability in the web application, the attacker may leverage that vulnerability to execute queries on the core database with privileges of DBLink account.
- Most of such database to database communication is performed over an unencrypted channel. Therefore, an attacker may intercept the traffic to perform MITM attack enabling him to sniff using sniff the data using tools such as Ethereal.

To minimise the risks from database links, ensure that:

- Private database links are used instead of public database links which will restrict all the users to access the database link except the owner of the link.
- Only *current user* DB Links are created and *fixed user* DB Links are avoided.
- The DBLink connection is encrypted e.g. via SSL.

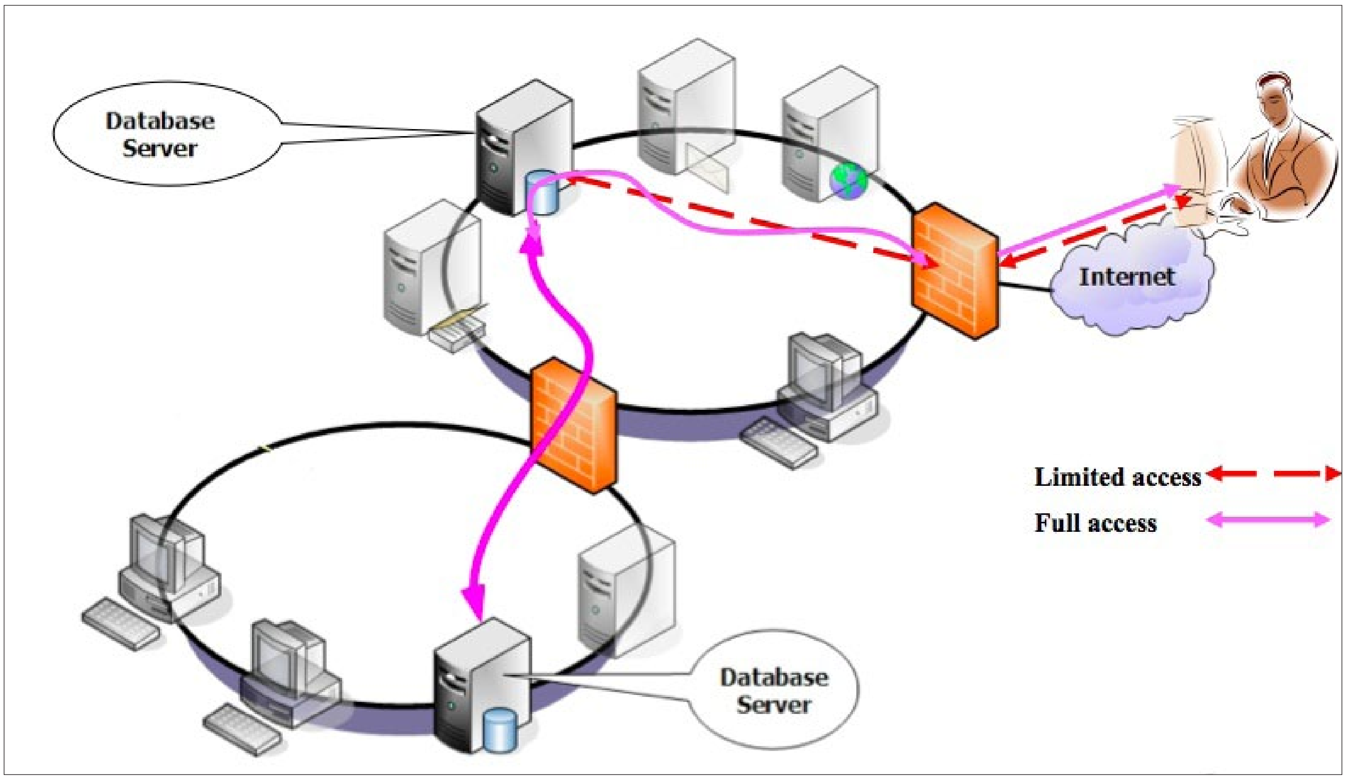


Figure 2. *Database link being abused to gain unauthorized access to database in high security zone*

- DBLinks are separated according to the operations performed. E.g. If *read operation* is to be performed on some tables and *write/update operation* on some other tables, two DBLinks should be created. One with *read access* to tables and other with *write access* to the tables.
- All the databases involved are properly hardened against all the known threats and patched on time.

Defending against malicious users with direct access

Defending against authorised users is most tricky as it is impossible for the database to identify if the user performing the operation has malicious intentions. This can only be achieved through adequate database activity monitoring controls.

Database Activity Monitoring is referred to reviewing the audit trails of database events and ensuring only legitimate operations have been performed on the database. There are several examples where proactive database activity monitoring has helped in identifying illicit activities from insiders and prevented full breach from taking place by catching the culprit in early stages of the attack.

Some of the events you must consider to include in audit trails:

- DBA Actions (DDL Statement execution, administration commands)
- Access to sensitive data (direct access to tables containing customer details)
- After hours access
- Remote administration of database

Other Best Practices

In addition to all the controls mentioned in this article so far, following are the best practices for keeping the databases secure. We highly recommend that these practices should be adopted by every organization:

- Database management systems are software products and therefore come with their own share of vulnerabilities, such as buffer overflows and remote command execution. Hence, it is important to ensure that latest patches are applied as soon as possible. If a patch is not available for identified vulnerability, try to reduce the risk by limiting the access to the vulnerable component.
- Similar to operating systems, rootkits have been created for databases. These rootkits allow adversaries to create database backdoors, hide illegally logged in users, replace system objects with malicious ones. To minimize the risk from rootkits:
  - Baseline the database on regular basis and compare with previous baselines. Inspect if suspicious change is identified.
  - Always try to use absolute execution paths for critical objects e.g. SYS.dbms_crypto
- Provide an additional layer of security to database by encrypting the file system on which data is stored.
- Vendors are trying to add various bells and whistles to database management system these days, such as inbuilt web servers, stored procedure to run system commands, HTTP end points and more. More often than not these features open up unnecessary attack vectors than being helpful. It is highly recommended that such features should be disabled in DBMS if there is not required by the business.

Conclusion

In this article we looked at different threats to databases and discussed various security controls that can be implemented on different layers to mitigate the risk from these threats. Although all the security controls and techniques suggested in the article are really important in ensuring the safety of the data within the database, we suggest that the businesses should take a risk-based approach in selecting the controls to be implemented.

SANDEEP NAIN

Sandeep Nain is a known software security professional based in Melbourne, Australia. He works alongside high-profile national and international enterprises enabling them to produce secure software. Academically, he holds a Master of Technology degree in Information Technology and several industry certifications including CISSP, CSSLP and CEH.

NETCLARITY
PREEMPTIVE, PROACTIVE PROTECTION

NG
INTRUSION DEFENSE
NextGen
Network Access Control

Harden your Network from the Inside Out

Network Access Control

Asset Vulnerability Management

Compliance Auditing and Reporting

Info Security Products Guide 2011 GLOBAL PRODUCT EXCELLENCE AWARDS CUSTOMER TRUST ★★★★★

[illegible] CRN

SC MAGAZINE BEST BUY ★★★★★

CVE COMPATIBLE cve.mitre.org

RSA TOP THREE INNOVATORS [illegible] RSA CONFERENCE 2007

TOLLY Up to Spec CERTIFIED

www.netclarity.net

Available through Partners Worldwide

Web Server Security Essentials

Over a period of time, Internet has become an important part of our daily lives. Security threats are always bright for web server roles, due to their usual way of providing service access to their customers. We all have been hearing the news about web sites being hacked. These sites usually belong to small and medium size organizations, and multi-national companies. These hacks sometimes result in changing the legitimate content of the website and putting a message from the hacking organization for their cause.

What you will learn…

- understanding the problems around securing web server from internal and external threats,
- understanding the different types of web server security threats and vulnerabilities,
- going through a statistical data about the top web application attacks, top web application weaknesses, and top impacts and outcomes of web security attacks,
- understanding the methodologies and the tools needed to identify the vulnerabilities in the web application and web servers,
- what it requires to remove and secure the web server from web server related threats and vulnerabilities,
- and last but not least, how to keep checking your web server for its most up-to-date security solutions for securing it from the threats.

What you should know…

- basic understanding of operating systems, web servers, web sites, and web applications,
- this article assumes that you should possess the knowledge about the installation, configuration and management tasks related to the web servers and web applications.

In this article, we will see a quick summary about types of web security threats and the recent web and data security breach incident statistics. Once we will cover this section, we will move ahead with discussing the best practices to secure the web servers environment from these types of security threats.

Anxiety Around of Web Services Security

Sometimes easy things become difficult to manage properly. This is what we see usually with web servers, as service or application web servers are quite straightforward to configure and bring them online on the network. But making sure that our web server is fully secure before it faces the public users is not as easy, and often overlooked. In this section of our article, we will be taking a look at the problems around web server security from the following perspectives, one is the types of threats to web server security, and the second is Year 2012 data and web security breach incident statistics.

Types of Threats to Web Server Security

Below are few of the common web server security threats, which are often get carried out from the dark side of the world (hackers) for military and non-military organizations, these victims of these types of cyber-attacks sometimes also incurred heavy fines from the governing bodies for having weak security of their systems. Let's take a look at these threats, which are:

- vandalism
- monetary Fraud
- gaining Access to Corporate Data
- tampering Online Transaction
- theft of One's Intellectual Property
- carrying out Denial-of-Service (DoS) Attack

An overview of past Web Security Breaches and Incident Statistics

Now it's time to go through some statistics for the past web security breaches and incidents. The statistical data we will see in this article for the web server security breaches are taken from the WHID (*Web Application Security Consortium*).WHID is an online web portal and service, which allows the companies and individual users to update the statistics about the web server security breaches and hacking.

Below we will see few of the statistics related to the Top Attack Methods, Top Application Weaknesses, and, last but not least, Top Impacts and Outcomes of these past top web server security breaches. Okay, so let's start now.

Top Attack Methods

Top attack methods for breaching the web servers' security are not new, they belong to the traditional web server security hacking categories, but the thing changed here is the technique and to some-how the attack is engineered. These top attacks were carried out in two ways. In the first choice, the attacker penetrates the security of the client, where a infected cookie or infection of the web browser can help the attacker inject the malfunctioning remote-code to break the web server or application security.

Another major type of the web server security can also be called the direct attacking methodology, in which the attacker directly penetrates or breach the security of the web server, by either means of SQL Injection, Brute Force, and XSS.

Upon looking at the past top web server security attack statistics from the WHID data, below image shows the most common and top category based attacks on web servers (Figure 1).

As we can see, SQL Injection, *Denial of Service* (DoS), and *Cross Site Scripting* (XSS) are the most common and top attacks for the web servers security, which are most widely found as the web application or web server (service) vulnerability, instead of the based operating system of the web server. These attacks can cause serious damage to the web server or web site availability, and this can also lead to the back-end database security breach.

Top Application Weaknesses

Another set of statistics will show us the top application weaknesses, which direct our attention to the vulnerabilities and risks commonly available in the web applications. These can be exploited to gain access of a web server, and thus attacker tries to attack on the data or database associated to the web server application (Figure 2).

As we can see in the above graph, which tells the top application weaknesses, where we can see that the major part of the top application weaknesses is unknown. It means that there is number of vulnerabilities and risks associated to the applications, which can also be called as bugs of the application. They are not discovered yet. Thus, they are threats, which are subject to be exploited.

Usually, attackers are highly intellectual people, they know the system or application from ins and out, and they try to exploit the application functionality or its characteristics, which help them to find the unknown vulnerability of the application. This is the reason why we say in the security that there is no 100% security. There is also no protection from the zero day attack. When a vulnerability is discovered and exploited, an attacker can perform their mali-

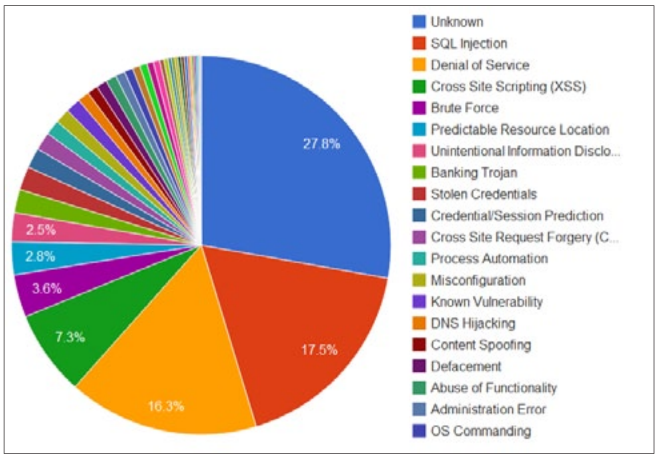


Figure 1. *The most common and top category based attacks on web servers*

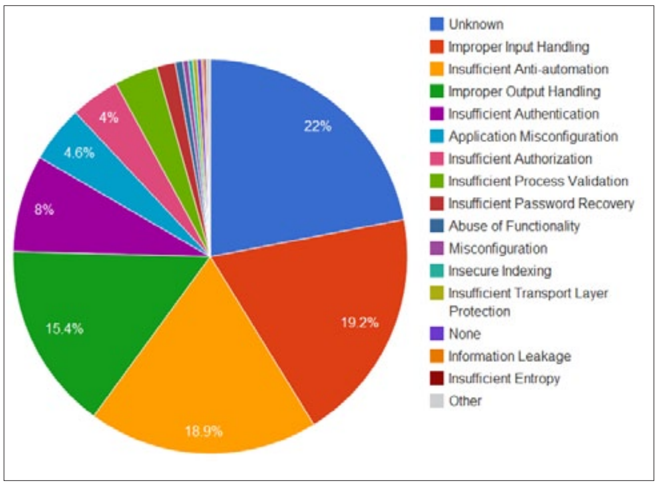


Figure 2. *Top application weaknesses*

cious activities since it is without the web administrator or the application or OS vendor knowledge.

The rest of the other application weaknesses, as per the above image for the top application weaknesses, are traditional weaknesses, which we have been seeing over a period of time. These other top application weaknesses can be improper input handling, insufficient anti-automation, and last but least unavailability of secure coding.

Top Impacts and Outcomes of Web Server Security Attacks

The last statistical figures, which we will discuss here are about the impacts and the outcomes of these past web security breaches. Since web servers are also called the info-hub of the organizations. For example, an online shopping web portal, would be connected to a database server, which will contain the customer private information, such as their credit card, purchase history and etc… In the healthcare sector, a hospital website would allow the patients to make online appointments, these databases would have the patients private information about their appointments with doctors, their current diseases and etc…

So, if this type of highly sensitive website information of an organization gets hacked then, it is not only the company's reputation on the line, but it may also cause huge financial losses.

Let's take a look at the below image, which shows the top and most common outcomes and impacts on the businesses after web server security breaches (Figure 3).

As we can see, the pie chart above shows how the most common impacts, organization see after the web server or web application security breach. Among these all impacts of hacking or security breaching, the most dangerous ones are the leakage of the information, which can even lead to a situation, where a business has come to its end or a down-time for a longer period along with the corruption of all the data.

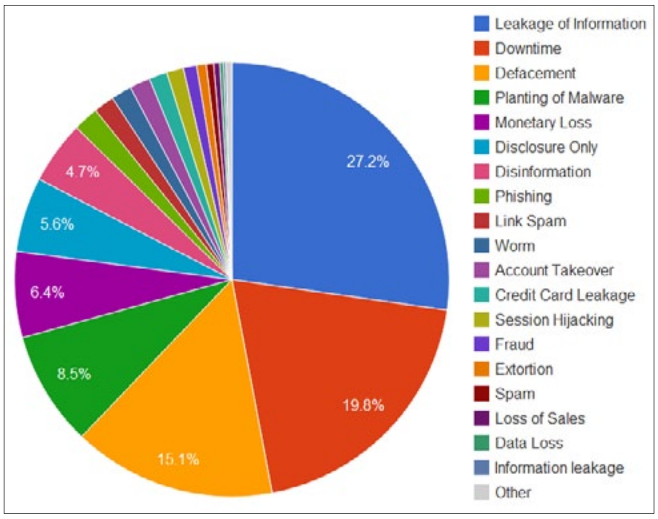


Figure 3. *The top and most common outcomes and impacts on businesses after a security breach*

In most cases, the impact doesn't only come from these types of security breach incidents, and always there are multiple business impacts and outcomes are associated to these attacks, which makes it more harder for the organizations to recover from the attack state to a normal business running state.

How to identify vulnerabilities in your Web Server?

This is an interesting topic, where we will discuss the methods for identifying the vulnerabilities in the web servers and web applications. Most of the organization keeps their web servers co-located in the managed services provider data centres, and few others keep their corporate website instance in the managed service provider, and their rest of the PHP and ASP.NET based application resides in their internal web server farm. Depending on your web server and application infrastructure, your preferred method of scanning these vulnerabilities will be constructed.

Before we go deeper with the web server vulnerabilities scanning tools and their details, let's first understand the web server infrastructure related components, which we will scan for our web infrastructure vulnerabilities. Following are the common components of any web server environment, which are recommended to be scanned in the web server and web application vulnerabilities scanning.

- Parameter Injection,
- SQL Injection,
- Cross-Site Scripting,
- Directory Traversal,
- Parameter Overflow,
- Buffer Overflow,
- Parameter Addition,
- Path Manipulation,
- Character Encoding,
- Site Search,
- SSL Strength,
- Sensitive Developer Comments,
- Web Server / Web Package Identification,
- Permissions Assessment,
- Brute Force Authentication attacks,
- remote code execution,
- format string vulnerabilities,
- Cross Site Scripting (XSS),
- known and default user name and passwords.

The above list provides the common vulnerabilities in all types web server technologies. This can be exploited by an attacker by injecting malicious code into the web server application. The attacker could even hack the website or web server.

Okay, since now we know about the common vulnerabilities of a web server, we will now go ahead with the methods and tools to identify these common vulnerabilities of web servers.

Methods and Tools for Identifying Web Server Vulnerabilities

If we compare the knowledge, tools and recommendations available in today's IT industry for securing corporate IT infrastructure and other related components from internal and external threats, is more speared and widely adopted by the organizations, than they were known and adopted ten or twenty years ago. After 9/11 security has become the utmost important agenda of business. But as we know there is no 100% protection, and there is some amount of residual risk always be there. Plenty of tools and frameworks are available to make sure that your network and servers are protected from past and future threats. To be specific about web servers, for identifying web server vulnerabilities, we can implement following tools and best practices:

The Tools and Best Practices for Identifying Web Server Vulnerabilities

The following best practices and tools are recommended to be use by the internal and external personnel for identifying the web server related vulnerabilities.

Best Practices

- conducting Penetration Testing for Web Servers, Networks, Database Servers, and client computers,
- using vendor specific vulnerabilities and threat scanning software application,
- monitoring server and network performance and utilization,
- implementing IT management frameworks, like ISO 27002:27001, MOF, ITSM, ISO/IEC 20000, CoBIT, and etc…

Tools

- For Microsoft based IIS Web Servers, it is highly recommended to use MBSA (*Microsoft Baseline Security Analyzer*). MBSA is a free software, and allows the IIS web server administrator to check the following:
  - Check for Windows administrative vulnerabilities
  - Check for weak passwords
  - Check for IIS administrative vulner• abilities
  - Check for SQL administrative vulnerabilities
- There are other vendor specific scanning software are also available for the identifying of web server related threats and vulnerabilities.
- Common and open-source scanning software for web server vulnerabilities are the following:
  - Burp Suite
  - Nikto
  - W3af
  - Sqlmap
  - Netsparker
  - Firebug
  - Websecurify

Note

The list of above and all other web server vulnerabilities scanning software, are available on this site (*http://sectools.org/tag/web-scanners/*).

What are the steps to remove Web Server Vulnerabilities

After we had a look at the types of web server security threats and statistical data for the recent web security breaches in the year of 2012, which is quite overwhelming for most of us, as managing web server security by just putting a SSL certificate on our web server won't be enough to secure this from all the threats out there.

In this section of Managing Web Server Security, we will be visiting different types of strategies technical and administrative tactics, to make sure that our web server stays secure and healthy. Now let's move ahead with this administrative and technical method, which helps to maintain a secure web server environment from a small to medium size organization to a large enterprise level organization.

Securing base Operating System of a Web Server

To build a secure web services environment in our company, the first thing we would have to make sure is the safeguarding the base operating system of a web server. There is pretty fair amount of percentage of the threats can be minimized if the web server operating system can be configured appropriate, which can complement the organizational security boundaries and security policies. The most appropriate term for making sure that the operating system of the web server is appropriately configured in a secure manner, would be hardening the operating system of your web server.

Software vendors for operating system or web server software, mostly concentrate on the features and functionalities, their main objective is to provide as much as feature in their product to compete with the other products available in the market, while security is always the second best priority. Each of this operating system comes by default with less secure environment configured, so if we don't harden the operating system of the web server, then there are series of loop-holes and back-doors are opened for these threats to attack on your web server.

Below are the lists of best practices to secure the base operating system of a web server:

Update and Patch Management

Keep your web server operating system up-to-date with the latest security and operating system related patches. In case of Microsoft IIS Web Server, Microsoft every month releases patches for each of its operating system, so the organization can plan for a monthly maintenance, where these newly patches should be first tested on a NON-Production web servers, and then deployed them to the production, for keeping them updated.

Installing and Updating Antivirus

Updating only the server with operating system patches will not help, until and unless, we have sound mechanism of protecting our data and server from the viruses and malware. And therefore a good antivirus product should be installed and always be up-to-date with the latest signature of the viruses and malware for protecting data and server OS/binaries.

Disabling Unnecessary Operating System Services and Roles

It is always a best practice to make sure that only the relevant operating system role and services are running on the web server. Let's say if you are not using any Windows Server role or feature on a IIS web server, then it is more advisable to remove these unnecessary Windows Server role and feature from the server. This will make sure that you don't have any unsecured channel opened for any attack.

Removing Extra Administrative Rights from the Operating System (Server)

We should make sure that there are no extra administrative accounts are created on the server, who can access the server with full rights on the server. If you have multiple administrator changing stuff on your web server, you should validate that are these all valid administrators, and do they all need to have full rights (root) admin access, or if there could be a possibility, where delegation of administrative rights can be done with least access privileges.

Password Policy Hardening

As we said above to remove the unnecessary admin access from the web server, the associated hardening strategy to this is the hardening of the user and service account passwords. Since most of these web servers are located in the DMZ part of our networks, and public access are allowed, so keeping normal and weak passwords will put you in danger. Therefore it is highly recommended that all service and administrative passwords should be strong enough, so they cannot be guessed and brute-forced.

Cutting off Remote Administration, wherever it is possible

If this is a critical web server, then you should make sure that this server is not remotely administered at all. If there is a company policy that few of the senior web admins should have the remote administration access, then it is your duty to make sure that, it should only be done from the company's network, and some type of one-time-password and other high security policies should be given to ensure that only the authorized personnel can access it.

Security Testing of an Operating System

There is no end of security, so therefore we should always be testing the security of our web services, and if possible internal and external penetration testing should always be carried out to ensure that, the base operating system layer is secure of the web servers.

Setting up Web Server Security

Once we have completed the first part of securing the operating system of a web server, the second step in this series is setting up the web server security. This means choosing the correct web server product, which can fit your web application and organizational security needs best. Other steps included in setting up the secure web servers are the following:

Securing Initial Web Server Setup

Once we have chosen the web server application for our web server environment, we should securely install it on the servers. And any unnecessary applications or services should be removed from it before taking it to production.

Web Server Application Specific Patch Management

As we discussed the patch management for the base operating system, we should also keep our web server application up to date with the latest security and application level patches. This helps to make sure the web server application is always protected from any malicious attack on the server.

Limit the Access to the Web Server Admin side

Just like limiting the access to the operating system of the web server, we should also consider how to secure the access to the web server administration. We should always provide least admin rights to the web server administrator to fulfill the required duties.

Web Server Activity Logging

One of the essential parts of securing the web server, is logging the web server related activities, whether it is administrative level or user level. This is needed because in case of breach of the security or normal day to day inspection, if the logs are not available then carrying out these tasks becomes impossible.

Securing File System and Web Server Directory

As for securing the web server application, we should also be looking at the file system for the web server content and application directories, which should only be allowed to be read / written by the authorized personnel. Web server application content directory should not be placed on the system partition of the web server, because in case of an operating system corruption data on the system partition should be threatened by this.

Employing Server Authentication and Encryption

After we looked at the administrative level security practices, now let's shift towards the "technological stuff," which helps to maintain high level of security. The following are the best practices to employ the server authentication and encryption practices in-place to secure web servers.

Assessment of Authentication Methods In-Place

In the first place, organization should assess the users and the data that which type of users will be accessing which sort of data and until when this access should be maintain. Once organization has this data, then they should consider the appropriate authentication and encryption methods to secure this data from the unauthorized access. An organization can employ a single or a combination of the authentication mechanism to make sure that only authorized people are granted to access the data, these different authentication methods could be Address-Based Authentication, Basic Authentication, Digest Authentication, and OS Integrated Authentication.

Configuring SSL/TLS for Client Server Authentication and Encryption

This is one of the most common scenario for configuring SSL and TLS authentication and encryption for web servers. SSL/TLS allows a Web client to confirm a Web server's identity. SSL/TLS-enabled Web clients (browsers) can employ standard techniques of public key cryptography to check that a server's name and public key are contained in a valid certificate issued by a CA listed in the client's list of trusted CAs. From the client-side authentication prospective, a SSL/TLS method of authentication and encryption allows a Web server to confirm a user's identity using the same techniques as those used for server authentication by reversing the roles. The certificate you choose for your SSL, it should be drown from the appropriate CA. In case of intranet it could be your internal CA (*Microsoft Certificate Services*), whereas in the case of public web server, your SSL certificate should be generated by a public Certification Authority. Last but not least, your SSL certificate should also have the required level of encryption and type for your web application.

Securing your Web Application User Account and Password

The user accounts and the passwords your application or users uses, should be strong enough so the dictionary or brute force cannot breach your web application security. We should employ the following types of user and password policies to make our web application access stronger:

- using of hardware tokens, one-time passwords, biometric authentication, and SSL/TLS client certificates.
- time-out and user account locked out policies should be in-place
- strong password policy should be set on all users
- enforcing password history to minimum
- enabling Password Complexity

Securing Core Network Infrastructure

At the end of the day, it is all about the physical network layout. If your network is engineered in such way that it was built upon a basic security principle, then it will provide an in-depth-security mechanism for securing all the server roles and services running on it. Now let's see a few of the following practices we should follow to ensure that your physical network infrastructure is secure.

Network Structure built from Security In-depth Concept

The network structure which is built up on the security in-depth concept, where the physical network layout makes it either easier or harder for an attacker to breach the security of your web server. For an example if your web server is located in the internal network, and if this gets compromised then, it will become handy for the attacker to takes over to the other resources of the internal network. On the other side, if you place your web server in DMZ part of the network, then upon compromising this server, your other internal sources will be safe, because other security mechanism will stop the attacker to take over to these internal resources.

Securing Web Server with Firewall and IPS & IDS

A good network layout design should also employ the proper allocation and placement of the different network and security devices in the design to provide necessary security to a web server. A good design can have external firewall in-front of a DMZ and an internal firewall placed in-front of the internal network, so this double layer of security ensures that, in any case, any unauthorized user should not access any service or data. IPS (Intrusion Prevention System) and IDS (Intrusion Detection System) can also be used to ensure the maximum security of a web server.

How to make sure that your Web Server is secure and protected against security threats?

Security is not like a software or an application bug, which can be fixed by installing the hotfix once and for the rest of the time, we can just go and sleep. Security is like a life-cycle which should always be running, because if you install the security patches to your servers one month, but you don't do it for next five months, then you will not be protected from future threats, which may hurt your server availability and data in these coming five months.

Making sure that our web server and its data stays protected can only be achieved if the organization pays serious attention to the security needs on all the layers of networks and applications. If you only consider patching your servers, but don't pay attention on the physical network of your company, then you are vulnerable to the internal and external threats to your organization.

We can summarize this topic of ensuring about the health and the protection of web server, by mentioning the following best practices:

- periodic performing the disaster recovery drill for the various IT infrastructure components, including servers (web, database, applications) and other components of an IT infrastructure.
- implementing IT management frameworks, like CoBIT, ITSM:2000, ISO 27002:27001, MOF, ITIL, and etc…
- performing periodic Security Penetration Testing on all the areas and components of an IT infrastructure.
- making security at the top most requirements for the existing and up-coming future projects.
- performing health-check-ups of the system on the periodic basis.

Summary

In this article, we covered web server security threats and the security incidents reported for the data and web security breaches. After we covered these two elements, we moved on with the best practices for employing the web server security.

These best practices include safeguarding the operating system of the web server, setting up web server application security, implementing server authentication and encryption, and last but not least securing the core network infrastructure for the web server.

ZAHIR HUSSAIN SHAH

Zahir Hussain Shah, a Microsoft Most Valuable Professional, who has worked with businesses from Small-to-Medium size organizations to the Multi-National companies, for providing IT Consultancy and Solution Delivery. He has been with IT industry for over 7 years now. Currently he is working with UAE's prestigious Oil and Gas sector for providing solution designing and delivery for Microsoft Hyper-V, Clustering, Active Directory, Exchange Server, Lync Server, and System Center. A part from the daily office life, Zahir is an Author, Public Speaker, and a blogger, who owns a successful blog (http://zahirshah-blog.com) on Microsoft Private Cloud, Messaging, Unified Communications, and Systems Infrastructure solutions. He is also certified for CISSP, MCSE, MCITP, MCTS, and CCNA. In the year of 2011, he was honoured with Microsoft Most Valuable Professional (MVP) Award for Microsoft Exchange Server, for his excellent contribution in the Microsoft Exchange Server Technical Communities. As a CISSP, Zahir's specialized security domains are: Building Enterprise Security Program, Information Risk Management, Access Control, Operations Security, DRP/BCP.

Is your
MISSION-CRITICAL
security strong enough
to stop a
SKILLED ATTACKER?

Don't guess
Don't believe
Don't hope
KNOW!

acros

An ACROS Penetration Test is conducted exactly like a real attack by a skilled, motivated adversary – only without the damage. We will find the weakest links in your security and use all our knowledge, skills and capabilities to try to achieve exactly what your security measures and policies are there to prevent. If it sounds difficult, we're interested.

Experience the ultimate test of your security.
(After all, the only alternative is to wait for an actual attack.)

ACROS Security – http://www.acrossecurity.com – security@acrossecurity.com

Fusic

Fusion of Society, IT and Culture

Fusic Co.,Ltd.

Founded in 2003 in Fukuoka, Japan. Fusic provides several IT related services all around Japan. Among the services we provide are: web development, contract-based software development (such CMS and CRM), etc. We also developed our own web-based presentation service "Zenpre", and e-Commerce platform "Ureru-net-kokoku-tsukuru", and serve consumer through ASP. Currently, we also play a leading role in the mobile applications development in platforms such iPhone and Android.

浜崎陽一郎

Yoichiro Hamasaki
Vice-President
Co-Founder

納富貞嘉

Sadayoshi Noutomi
President
Founder

Fusic Co.,Ltd http://fusic.co.jp/ info@fusic.co.jp

Fukuoka Head Office
Shin-nihon build.9F, 2-4-22 Daimyo Chuo-ku, Fukuoka-shi,
810-0041, JAPAN
+81-92-737-2616 +81-92-737-2617

Fukuoka Laboratory
East Fukuoka General Office 4F, 1-17-1 Hakata Station East,Hakata-ku, Fukuoka-shi,
812-0013, JAPAN

Tokyo Branch
Okura build. 3F, 1-4-10, Shibadaimon, Minato-ku, Tokyo, 105-0012, JAPAN
+81-3-6450-1633 +81-3-6450-1634