

PREVENTATIVE MEASURES

Over the past year, there has been a considerable spike in cyber attacks against the financial services and the online retail industry. There are a number of actions a firm can take in order to prevent or thwart the specific attacks and techniques used by these intruders. The following steps can be taken to reduce the likelihood of a similar compromise while improving an organization's ability to detect and respond to similar incidents quickly and thoroughly.

Attacker Methodology:

In general, the attackers perform the following activities on the networks they compromise:

1. They identify Web sites that are vulnerable to SQL injection. They appear to target MSSQL only.
2. They use "xp_cmdshell", an extended procedure installed by default on MSSQL, to download their hacker tools to the compromised MSSQL server.
3. They obtain valid Windows credentials by using fgdump or a similar tool.
4. They install network "sniffers" to identify card data and systems involved in processing credit card transactions.
5. They install backdoors that "beacon" periodically to their command and control servers, allowing surreptitious access to the compromised networks.
6. They target databases, Hardware Security Modules (HSMs), and processing applications in an effort to obtain credit card data or brute-force ATM PINs.
7. They use WinRAR to compress the information they pilfer from the compromised networks.

We are providing the following preventive measures. Performing these steps may not prevent the intruders from gaining access, but they will severely impact their effectiveness based on current attack methods.

1. **Recommendation 1:** Disable potentially harmful SQL stored procedure calls.

The xp_cmdshell, OPENROWSET, and OPENDATASOURCE stored procedures should be disabled on all databases unless they are explicitly serving a business need within the network.

The xp_cmdshell procedure allows someone to execute commands on a local system from the database, with the permissions of the service account used for the database. The OPENROWSET and OPENDATASOURCE procedures allow one to cause the database to transfer data from the local database to a remote database and vice versa.

The following two steps should be taken to remove the potentially harmful stored procedure calls.

1. Disable access to the xp_cmdshell functions within Microsoft SQL Server.

```
Microsoft SQL Server 2000
EXEC sp_dropextendedproc 'xp_cmdshell'

Microsoft SQL Server 2005
EXEC sp_configure 'xp_cmdshell', 0
```

2. Remove the "xplog70.dll" file from the server.

If it is necessary to use the potentially harmful stored procedure calls, limit the exposure by applying IP filters on the SQL servers. Assign explicit ALLOW rules to the interfaces for the application the SQL server is supporting. Disallow communication between SQL Server hosts unless an application necessitates otherwise.

2. **Recommendation 2:** Deny extended URLs.

Excessively long URLs can be sent to Microsoft IIS servers, causing the server to fail to log the complete request. Unless specific applications require long URLs, set a limit of 2048 characters. Microsoft IIS will process requests over 4096 bytes long, but will not place the contents of the request in the log files. This has become an effective means to evade detection while performing attacks.

1. Modify "%windir%\system32\inet_srv\urlscan\urlscan.ini"
 - i. Ensure "MaxQueryString=2048" is present
 - ii. Ensure "LogLongURLs=1" is present

3. **Recommendation 3:** Implement specific approaches to secure dynamic web site content.

Certain measures can be taken to mitigate the risk of these types of attacks by developing a secure code base. The steps below are a few of the best practices for secure coding that will help prevent the attack associated with this incident. Additional information can be found at <http://msdn2.microsoft.com/en-us/library/ms998271.aspx>.

1. Replace escape sequences

```
private string SafeSqlLiteral(string inputSQL)
{
    inputSQL.Replace("'", "''");
}
```

2. Use parameters with stored procedures

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    DataSet userDataset = new DataSet();
    SqlDataAdapter myDataAdapter = new SqlDataAdapter(
        "SELECT au_lname, au_fname FROM Authors WHERE au_id = @au_id",
        connection);
    myCommand.SelectCommand.Parameters.Add("@au_id", SqlDbType.VarChar, 11);
    myCommand.SelectCommand.Parameters["@au_id"].Value = SSN.Text;
    myDataAdapter.Fill(userDataset);
}
```

3. Constrain input in ASP.NET web pages

```
if (!Regex.IsMatch(userIDTxt.Text, @"^[a-zA-Z'.\s]{1,40}$"))
    throw new FormatException("Invalid name format");
```

4. **Recommendation 4:** Install and run authorized Microsoft SQL Server and IIS services under a non-privileged account.

Unless a specific application requires system or administrative level permissions, all instances of Microsoft SQL Server and IIS should run under accounts with restricted user permissions.

5. **Recommendation 5:** Apply the principle of 'least privilege' on all SQL machine accounts.

The attackers generally create tables into which they store malware or data collected from the enterprise. Unless specific applications dictate otherwise, restrict the capabilities of the accounts used to modify databases on the servers. In particular, remove the ability to create new tables, denying the attackers a means of transporting malware and stolen data.

6. **Recommendation 6:** Require the use of a password on Microsoft SQL Server administrator, user, and machine accounts.

Several SQL servers examined had an empty password on the "sa" SQL account. All accounts with access to resources should be protected with passwords or certificates.

7. **Recommendation 7:** Lock out accounts on the mainframes after several unsuccessful logon attempts.

Locking accounts and requiring IT support to restore service aids in protection against brute force attacks. This can serve as an early detection of potential security problems.

8. **Recommendation 8:** Run the minimum required applications and services on servers necessary to perform their intended function.

Several servers, to include Active Directory master servers, have unnecessary software installed (e.g. Microsoft Office). In addition, ensure that no unnecessary services are running. This includes SQL Server and SQL Server Express on support and other workstations. Should these services be necessary, restrict access through IP filters on Microsoft Windows or through third-party firewall software.

9. **Recommendation 9:** Deny access to the Internet except through proxies for Store and Enterprise servers and workstations.

Attacks on victim networks make extensive use of HTTP, HTTPS, and DNS network ports. Denying direct access to the Internet will frustrate and mislead an attacker.

10. **Recommendation 10:** Implement firewall rules to block or restrict Internet and intranet access for database systems.

Disallow all traffic outbound from servers harboring sensitive data. Communication to the SQL servers and data warehousing servers should be tightly controlled. Restrict traffic between data centers and stores to essential ports and services only.

11. **Recommendation 11:** Implement firewall rules to block known malicious IP addresses.

Firewall rule sets designed to block all ingress (incoming) and egress (outgoing) traffic to the known malicious IP addresses have been put in place. Note that traffic violating the rules should be logged and observed in near-real time.

12. **Recommendation 12:** Ensure your HSM systems are not responsive to any commands which generate encrypted pin blocks. More specifically, HSMs should not accept commands that allow plain text PINs as an argument and respond with encrypted PIN blocks.

HSMs are normally used to verify Personal Identification Numbers (PINs), generate PINs used with bank accounts and credit cards, generate encrypted Card Verification Values (CVVs), generate keys for Electronic Funds Transfer Point of Sale systems (EFTPOS), and generating and verifying Message Authorization Codes (MACs). These systems, if accessed by an unauthorized intruder, can provide the attacker the ability to discover the appropriate PIN number for a corresponding credit or debit card. Therefore, in an effort to prevent this, HSMs should be configured to disallow "in the clear" PINs as an argument for performing its tasks.