X Tunnels through SSH

By Layne Bro
November 30, 2000

Introduction

Securing a network requires many different pieces of a very large puzzle. Some security is provided at the firewall. Some security is provided by user authentication at the server. Some security is done by Intrusion Detection Systems. Still other security is created on the network hardware itself.

In some instances, all the security in the world designed and built to prevent an attacker from entering your network is useless – he is already inside your walls. It is because of this that encrypting data on the wire is another good choice in designing your "Defense in Depth" security solution.

In my office, we have employees, customers, and contractors all accessing the same servers for development projects. They all come into the network from different directions but they are all mixed together on the segment where the server resides. It is because of this that we have chosen to encrypt traffic as it crosses over the network.

Some of the tools used to access the servers are easy to encrypt (telnet becomes a Secure Shell connection, HTTP becomes HTTPS) but one of the tools takes a little more work – X-Windows.

SecureShell

SecureShell (SSH) is a replacement for telnet, rlogin, rsh, and rcp. SSH has the ability to encrypt the entire session, including the authentication phase. This prevents eavesdroppers from learning your userid and password by sniffing the network. Now you may ask, "Why should I care if someone discovers my userid and password?" Accounts that have been compromised can be used to store illegal software, used as jumping points for attacks against other systems, or even used to forge email. Ever wanted to find out what would happen if your boss received an email from your account telling him what an idiot you thought he was? In this day of electronic communication, a large part of your professional presence is tied into your electronic presence – including email.

SSH software is available from several vendors. Recently the server portion has become a free package from openssh.com. There are two pieces to an SSH connection – the client and the server. In our environment, we use sshd on all of our servers and the windows desktops use SecureCRT from VanDyke Technologies. It is possible to have SSH connections from Windows to Unix or from Unix system to Unix system.

So, for simple text based connections between systems, simply using SSH connections will encrypt all of your data, including userid and password during the authentication phase. But, for a graphical or window based connection, we need to work a little harder to encrypt our data.

X Traffic

X Windows is the primary graphics/windowing system in use on Unix workstations today. However, X Windows has some serious security problems.

First, authentication to the server is done with clear text. In other words, anyone sniffing the network could discover your userid and password.

Secondly, X Windows actually initiates multiple connections, and not all from the client. Some of the connections used in an X Session are initiated by the server to your desktop (in a typical setup). This means that some of the connections are typical three way handshakes (syn, syn/ack, ack [Figure 1]) starting at your desktop and going to the server.

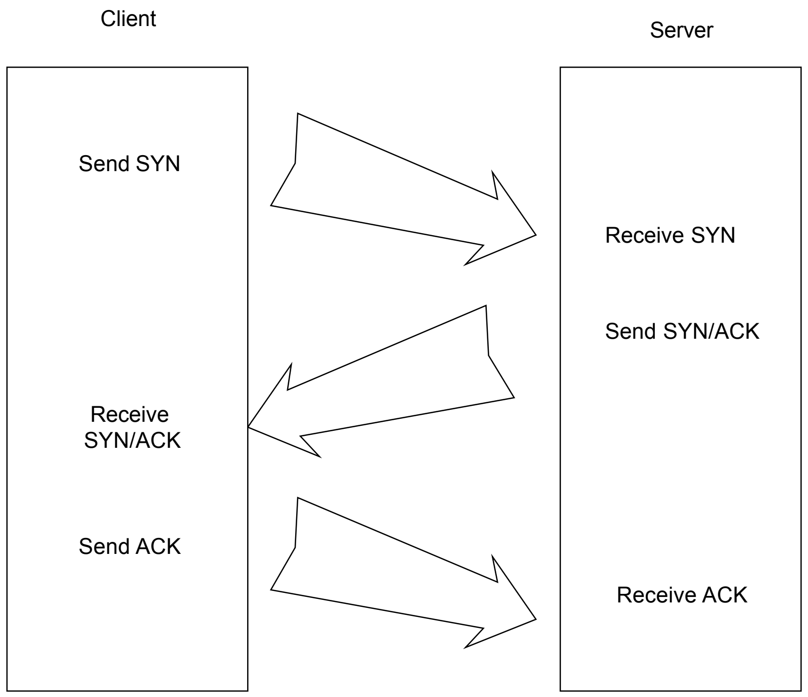

Figure 1

Some of the connections are three way handshakes starting at the server and going towards your desktop[Figure 2].

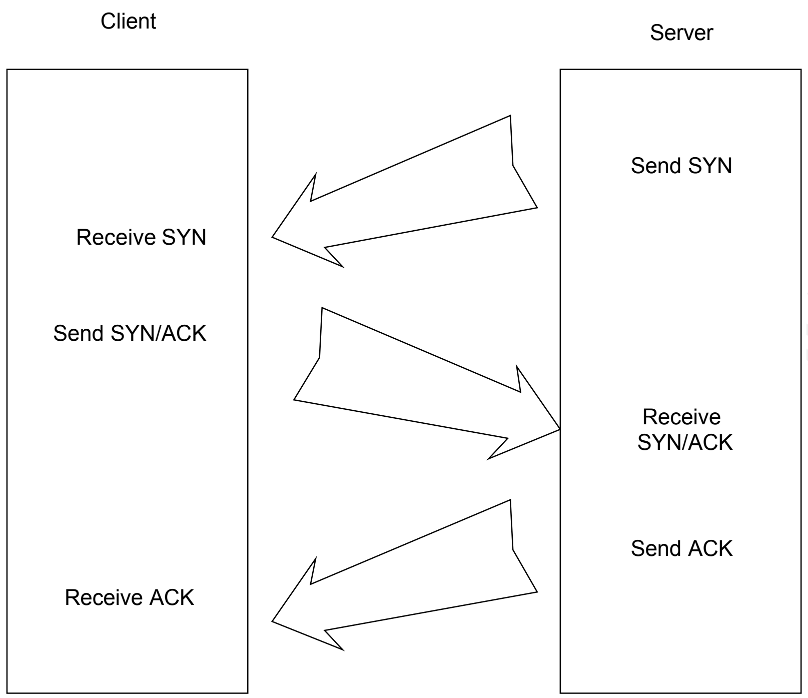

Figure 2

Therefore, even if we were to place a firewall between our employees and the server they are trying access, we would need to leave a large range of ports open for the server initiated three way handshakes back to the employee desktops. This means that other users could pass through the firewall to initiate connections [Figure 3] just like the X Windows Server.

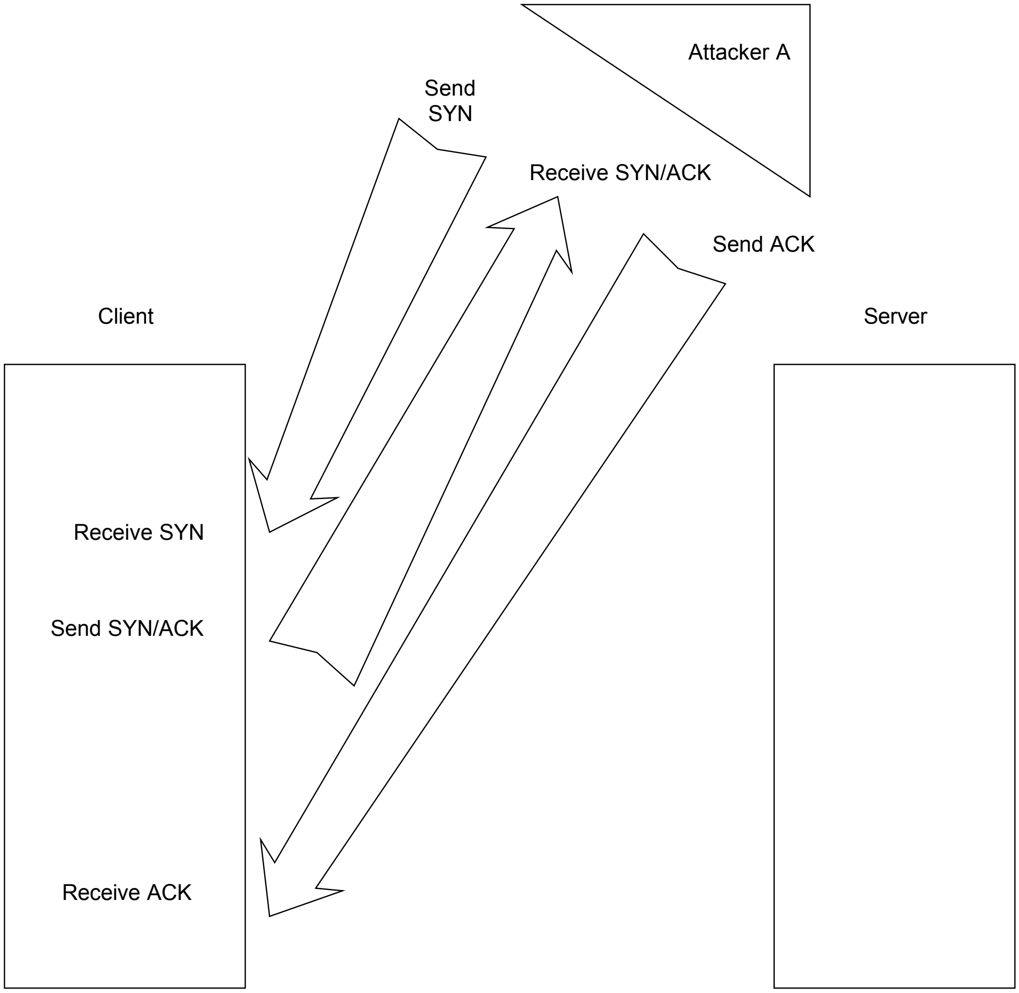

Figure 3

Therefore we would be putting ourselves at risk of attacks from the contractors and customers who also have access to the systems in the server segment. (In other words, we can't simply allow established connections through the firewall because the X Traffic isn't always established.)

Tunneling X-Traffic through your SSH Connection

In order to defend against attacks disguised to look like X Windows connections, we have chosen to lock down our firewall and require all X connections to be tunneled through a SSH Connection. This means we don't have to leave large sections of ports open on the firewall for x windows connections originating at the server. It also means we can encrypt the authentication phase of the connection to hide the userid and passwords from casual eavesdroppers. The following is a step by step guide to setting up the tunneling of X Windows through an SSH connection using SecureCRT on the client side of the connection. (Please note that for this example, Exceed is used as the X

Windows manager on a Microsoft Windows 2000 box).

SecureCRT setup –

To set up port forwarding, follow these steps.

1. Start SecureCRT
2. Click on File / Connect and select the SSH session for which you would like to use forwarded ports.
3. Click on the Session Options button or right-click on the session and select **Properties** from the pop-up menu to bring up the **Session Options** dialog.
4. In the **Connection** category, click on the **advanced** button and select the **Port Forwarding** tab.
5. Check the Forward X11 Packets option.
6. OK and exit

Exceed Setup –

1. Run Exceed's Xconfig
2. Choose Communications
3. Select passive for your mode – click ok
4. Choose Screen Definition
5. Under the Screen 0 choose multiple for your window mode
6. Exit

Put it to the test --

Run Exceed
Connect to your SSH session (as defined above)
At the prompt type xterm
Run an x window (i.e. admintool)

This will provide you with X Windows on your desktop intermixed with your desktop windows. Please note that this setup does not provide the logon screen as your initial connection. You are already connected through your SSH connection and only the subsequent X Windows will be displayed back to your desktop.

Words of Warning

While the above setup will certainly encrypt your data on the network for both text and gui based connections to a Unix server, it is not a magic bullet for securing your network. Please remember that this is one small piece in a very large puzzle. This setup does nothing to address the issue of a server that is already "owned" by a malicious user. I could establish an X connection to a server, but a malicious user who owned the server could theoretically launch rogue programs or commands on my system due to the nature of new connections created under X Windows from the server. This is a very serious security issue and therefore the use of X Windows should be limited to systems known to be relatively safe. These systems should also be under strict security monitoring and control to prevent them from becoming compromised and posing further risks to the systems connecting to them.

Configuring your X connections this way will allow you to pass through firewalls that, if properly locked down, would otherwise prevent the initiation of a new connection from the server to your local system. This is a double edged sword – what if you don't want to allow X Connections out through your firewall but you do want to allow SSH connections? Users could conceal their X Connections inside of their SSH packets and the firewall would never know.

1. “SSH FAQ Section 1: About Secure Shell (SSH)?” http://www.tigerlair.com/ssh/gaq/ssh-faq-1.html
2. “SecureShell (SSH) at SLAC” http://www.slac.stanford.edu/comp/unix/ssh.html
3. “X Windows Introduction” http://www.coe.uncc.edu/project_mosaic/mosaic_help/os/x.html
4. “OpenSSH Homepage” http://www.openssh.com
5. “SSH Overview” http://www.vandyke.com/products/securecrt/overview.html