RemotelyAnywhere

Security Overview

Contents

Author Márton Anka, Chief Technical Officer, LogMeIn, Inc.

Abstract This paper provides a look at the security features of LogMeIn's RemotelyAnywhere. We at LogMeIn do not believe in security through obscurity, nor do we expect our customers to blindly accept our claims to important security features, such as end-to-end encryption. By publishing the details on how the security mechanisms work and interoperate in our products, we are also inviting the public to scrutinize our efforts.

Audience This document is technical in nature and is aimed at network engineers or network designers. Reading this paper can help the reader perform the necessary threat analysis before deploying our product.

Terminology In the RemotelyAnywhere architecture, there are two or three entities that take part in every remote access session.

The "client" or the "user" is the person or software accessing a remote resource.

The "host" or the "server" is the computer being accessed, or the RemotelyAnywhere host software on this computer.

The optional "gateway" is the RemotelyAnywhere SSH service that mediates traffic between the client and the host. It is possible to create direct connections to the host, or connecting via the "gateway".

Design Fundamentals RemotelyAnywhere was designed to allow secure remote access to critical resources over an untrusted network. During the development of the software, security considerations always prevailed. The following security design objectives, listed in order of priority, guided the decision-making process:

- Security and attack mitigation
- Authentication and authorization of users to the target resource
- Authentication of the target resource to users
- Data confidentiality
- Authentication and authorization of users within the target resource

Remote Access Axioms

Everything is a Target

With the penetration of broadband Internet connections, more and more computers are online 24/7. Most of these computers are operated by home users and have gaping security holes such as unpatched vulnerabilities and a lack of proper passwords.

The greatest weakness is, however, the user himself. The extremely quick penetration of so-called email viruses illustrates the lack of security-consciousness and the gullible nature of most Internet users. Email viruses, of course, are email attachments that are better classified as Trojan horses. They spread so quickly because users are surprisingly willing to violate fundamental rules when handling untrusted content. If the users themselves are responsible for infecting their computers with Trojans, how can you trust them to properly secure their systems against direct attacks?

Even competent network administrators can slip up and forget to install a patch or two, which, in the worst-case scenario, can allow attackers to run arbitrary code on the affected systems. Nothing demonstrates this better than the rapid spread of the Microsoft SQL Server worms in 2003. Both MSBlaster and Slammer infected a great number of computers, and generated such an excessive amount of network traffic - with the help of the exponentially growing number of infected hosts - that users could perceive a slowdown in Internet access even on uncompromised networks.

Worms like MSBlaster or Slammer were poorly written. Their spread rate was far from optimal, and they did not cause data loss or theft on a significant scale. Their creators exploited a widely known vulnerability in Microsoft SQL Server – a fix was available for several weeks before the first worm attack struck. In other words, they qualify as a very tasteless joke; one that undoubtedly caused many problems, but whose effect was nowhere near as disastrous as it could have been. One can only imagine what skilled hackers with malicious intent are capable of when they have a lucrative target.

Remote Access and Security

It is easy to see that many computers connected to the Internet are extremely vulnerable, even without installing a remote access product. Remote access products are perceived as risk factors, but mainly for psychological reasons. When a user first sees a remote access solution in action, their first negative reaction is usually with regard to the security implications. This is perfectly normal – even desirable. The real problem is that users do not immediately see the

threat inherent in other network-enabled applications, such as an email client, a web server or the operating system itself.

All modern operating systems include some sort of remote access solution by default. Windows, for example, ships with Microsoft's Remote Desktop as a simple remote administration interface. Even OpenBSD, the UNIX variant which is usually regarded as the most secure operating system available, includes SSH, a simple and secure application that allows command-line access over a network connection to the remote computer.

In essence, a well-chosen and well-configured remote access solution does not carry any additional risks. If a network manager can keep a network secure using a reliable remote access software package, such as RemotelyAnywhere, productivity can be increased and costs reduced without any adverse effects on network security.

RemotelyAnywhere Security Mechanisms

When users think of Internet data security, they are usually concerned about data encryption – to the point where security is measured in the length of the encryption key used. However, encryption and decryption, while being very important, are fairly trivial tasks compared to the other challenges faced by designers of secure systems. As you will see, data encryption is just one of the main goals set forth by the designers of RemotelyAnywhere.

The major design objectives outlined below are listed and explained in the order of achievement during the establishment of a remote access session, and not necessarily in the order of importance.

Authentication of the Target Resource to Users

RemotelyAnywhere offers several layers of authentication during an access process. These authentication methods may be User authentication or Resource authentication. Through the access session, data encryption is enabled for maximum security.

First and foremost, when a user connects to a RemotelyAnywhere installation – the "server" – they need to be 100% positive that the computer they are about to exchange data with is really the one to which they intended to connect.

Suppose that an attacker poses as the server towards the user, and it poses as the user towards the server. The attacker, in this case, can sit between the two parties while reading, or possibly modifying, the data in transit. This is known as a "Man in the Middle", or MITM attack and is especially hard to protect against. RemotelyAnywhere utilizes SSL/TLS certificates to verify Server identities and thus protect against MITM attacks. When a connection is made, the Server's certificate is verified. If the certificate was not issued by a certifying authority the user has chosen to trust, a warning will be presented. If the certificate was issued by a trusted certifying authority, but the host-name in the URL does not match the hostname included in the certificate, a different warning will be presented.

If the Server passes these verifications, then the User's browser generates a "Pre-Master Secret" or PMS, encrypts it with the Server's public key contained within its certificate, and sends it to the Server. As ensured by public key cryptography, only the Server who holds the corresponding private key can decrypt the PMS. The PMS is then used to derive the Master Secret by both the User and the Server, which, in turn, will be used to derive initialization vectors and session keys for the duration of the secure session.

In short, the above ensures that the User is establishing the connection with the Server, and not with a third entity. Should a MITM attack be attempted, either one of the security warnings will be triggered or the PMS will be unknown to the MITM, effectively rendering the attack impossible.

RemotelyAnywhere supports different implementations of SSL certificates.

- Self-signed SSL certificates for Server verification are generated with pre-existing or newly created CA certificate and can be assigned to a RemotelyAnywhere host.
- You can also utilize existing Certificate Services and CA certificates to generate a Server certificate for your RemotelyAnywhere installation.

Both types can be created, configured, and applied in the browser certificate store under **Security** > **SSL Setup**. In addition to web interface management of SSL certificates, RemotelyAnywhere also offers Command line parameters for SSL certificate management. For more information on Certificate related command line options, type remotelyanywhere.exe cert into a command prompt.

Data Encryption

The SSL/TLS standard defines a wide choice of cipher suites such as RC4 and 3DES, and some implementations offer more advanced suites such as AES as well. RC4 operates on 128 bit keys, 3DES uses 168 bit keys. AES can support 128 or 256 bit keys. The User and the Server will agree on the strongest cipher possible. This is done by the User sending the Server a list of ciphers it is willing to use, and the Server choosing the one it prefers from this list. The SSL/TLS standard does not define how the Server should choose the final cipher. In RemotelyAnywhere, the Server simply selects the strongest available cipher suite that the client has offered.

This method allows both the Client and the Server to decline the use of specific data-encryption algorithms without the need to updating both components, should an algorithm be deemed as broken or insecure. The available cipher suites are determined by the local environment; most recent versions include SSLv3 RSA AES(256), SSLv3 RSA 3DES(168) or SSLv3 DSS Camellia(256) ciphers.

Note: RemotelyAnywhere optionally allows the connection over untrusted HTTP channels instead of utilizing SSL. LogMeIn strongly recommends the use of SSL for any connection. If you still want to enable HTTP-only connections, you can do so under **Preferences** > **Network**.

Intrusion Detection

RemotelyAnywhere provides two layers to detect intrusion attempts.

SSL/TLS

The first layer is provided by SSL/TLS to ensure that the data was not changed in transit. This is achieved via various techniques:

- **Record Sequence Numbering**: SSL/TLS records are numbered by the sender and the order is checked by the receiver. This ensures that an attacker cannot remove or insert arbitrary records into the data stream.
- **Message Authentication Codes (MACs):** Every SSL/TLS record ends with a message authentication code that is derived from the session key (known only to the two communicating parties) and the data contained within the record. If MAC verification fails it is assumed that the data was modified in transit.
- **Cipher Block Chaining (CBC mode):** The cipher suites preferred by RemotelyAnywhere utilize Cipher Block Chaining, meaning that every SSL/TLS record will depend on the contents of the previous record. In this mode, the input to the cipher is not only the current plaintext record but the previous one as well. This again ensures that packets cannot be inserted or removed from the data stream.

RemotelyAnywhere Filters

The second layer consists of three filters provided by RemotelyAnywhere itself:

- **IP Address Filter**: When RemotelyAnywhere receives a connection request from a User, it first checks its list of trusted or untrusted IP addresses and possibly denies the connection. An administrator can set up a list of known-good or known-bad IP addresses within RemotelyAnywhere. For example, he can designate the internal network and another administrator's home IP address as a known good address. IP Address filters can be set under **Security** > **IP Filtering**. You can also set filter profiles for the network adapters available on the Server under **Preferences** > **Network**.
- **Denial of Service Filter**: A Denial of Service filter will reject connections if the IP address the request is coming from has made an excessive number of requests without authentication within the observation time window. This is done to protect against someone overloading the host computer by, for example, automatically and repeatedly requesting the login page. You can customize the DoS filter behavior under **Security** > **IP Address Lockout**.

- **Authentication Filter**: If the user has made an excessive number of failed login attempts, the Authentication Filter will reject the connection. The Authentication Filter is in place to prevent a potential intruder from guessing an account name and password. You can customize the authentication attack filter behavior under **Preferences** > **IP Address Lockout**.

Authentication and Authorization of Users to the Target

After the user has been granted access by the previous layers, it is time for him to prove his identity to the Server. This is achieved by a mandatory Windows authentication step. An optional RSA SecurID authentication can follow Windows Authentication.

Windows authentication requires that the user authenticates himself to the Server using his standard Windows username and password. The Server will usually pass this request on to the relevant domain controller. This step not only validates the user's identity but also ensures that network administrators can control when and who can log in to a specific Server.

To add an extra layer of security over the simple username / password authentication required by Windows, system administrators can also configure RemotelyAnywhere to require RSA SecurID authentication; a widely-used two-factor authentication method.

Enabling RSA SecurID in RemotelyAnywhere requires third party installation of ACE Client authentication present on the same computer RemotelyAnywhere is installed on. The presence of ACE client will be recognized by RemotelyAnywhere, and the product will display custom RSA related configuration under **Security** > **RSA SecurID**.

Authentication and Authorization of Users Within the Target

RemotelyAnywhere, once it verified the user's identity using the above methods, will check its own internal user database to see which internal modules the user is allowed to access.

System administrators can configure RemotelyAnywhere so that users with certain roles have access only to a subset of tools offered by RemotelyAnywhere; for example, the Helpdesk department can be configured to only view the computer's screen and performance data but not actually take over the mouse and the keyboard or make any changes to the system configuration. In the meantime, the Sales department might be given full remote control access to their respective computers, but features such as performance monitoring and remote administration would be made unavailable to them.

While RemotelyAnywhere operates using the System_Service profile, using the Windows access token obtained when the user was authenticated, RemotelyAnywhere impersonates the user towards the operating system while performing actions on their behalf. This ensures that RemotelyAnywhere adheres to the Windows security model, and users have access to the same files and network resources as if they were sitting in front of their computer. Those

resources that are not available to users in Windows – such as NTFS File system permissions, network and share access rights - will not be made available to them via RemotelyAnywhere.

Under **Security** > **Access Control**, you can choose several global options such as NTLM authentication, domain listing, and you can define custom access permissions to Windows User profiles, local- and domain User Groups.

By default, any user can login to RemotelyAnywhere who is a member of the local Administrator group or who inherits Administrator privileges from other domain sources.. Non-administrator users will be denied login completely, with no access rights. For details, see "How to Specify User Access Rights in RemotelyAnywhere" in the RemotelyAnywhere User Guide.

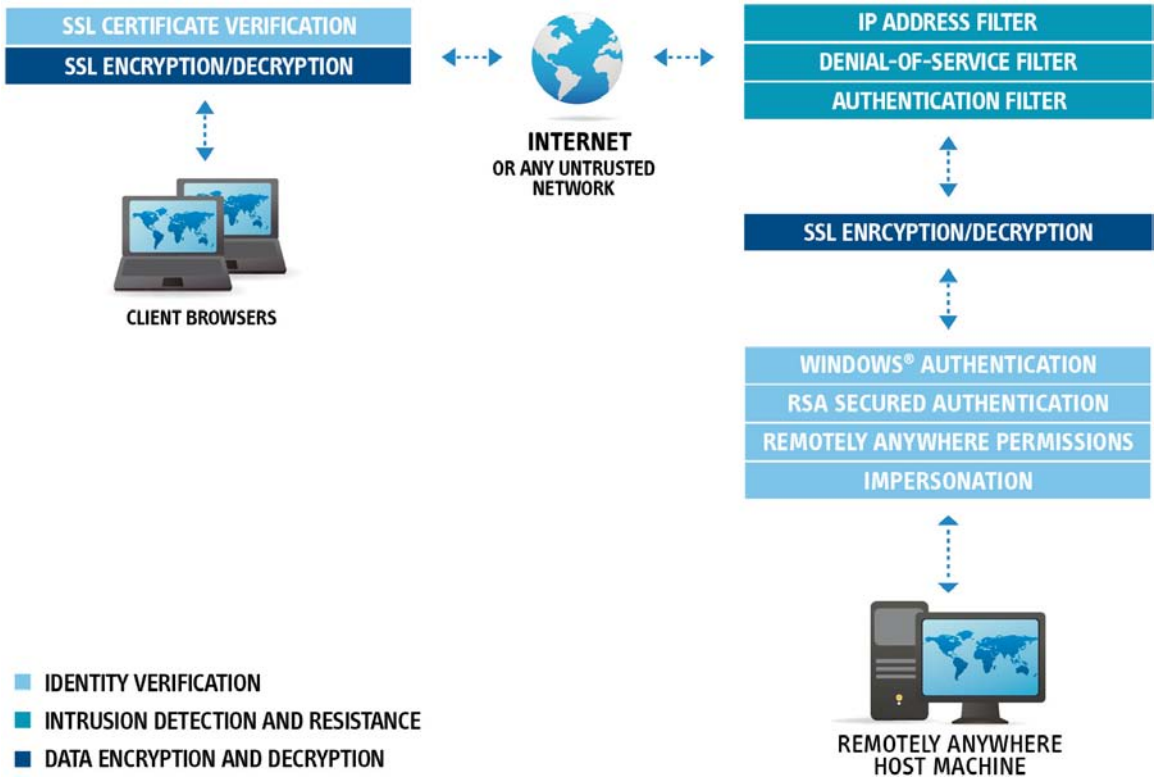

Figure 1. RemotelyAnywhere Security Building Blocks

Interactive User's Consent

RemotelyAnywhere allows sessions to unattended resources. However, when a workstation or server is logged in, an active user may be operating the RA Server's screen.

For this scenario, RemotelyAnywhere allows configuration of a Permission window, which is presented to the interactive user before the remote administrator gains screen control privilege.

Note: End user's permission protects the screen content of the server from being seen by remote administrator. Management functions and tools, such as Command Prompt or File manager are authenticated via login rather than permission.

By default, RemotelyAnywhere does not display permission notifications to interactive user because each remote user would be granted with Full Control. To configure a custom permission window and permission message go to **Preferences** > **Remote Control**.

Auditing and Logging

RemotelyAnywhere provides extensive logging capabilities. These audit features are explained in the table below.

Default text log	By default a very detailed log of the events that occur within the software is kept in the installation directory. These text based logs are kept for 90 days by default. The active log file is named RemotelyAnywhere.log. Older logs are stored with the naming convention RAYYYYMMDD.log. For example, the RemotelyAnywhere log file for June 1, 2012, would be called RA20120601.log. The text based log mechanisms will include every action performed via the Web interface of RemotelyAnywhere including any Chat content, files transferred, remote screen control sessions. You can modify the setting for these logs under **Preferences** > **Log Settings**.
SYSLOG Server Storage	The text-based detailed log can also be sent to a central SYSLOG server.

ODBC Logging	The text-based detailed log can be sent to a relational database over a pre-defined ODBC link.
Windows Event log	The most important events are also placed in the Windows application event log. Windows Event Log can only be accessed and cleared with appropriate, previously granted privileges. RemotelyAnywhere will always log the following events to the Windows Application Log: Service Start/Stop, LogIn/Logout, Remote Control Start/Stop.

Remote Control Screen Recording

The text based logs above do not include events which were performed using interactive mouse and keyboard operations using the Remote Control feature.

To provide full logging and audit track, RemotelyAnywhere also offers Session recording. The recording will be saved in a specified directory, in proprietary RemotelyAnywhere codec format, and can be played back after conversion. Session recording videos will be stored by a standard file naming convention, with the following file name:
`RCREC[HOSTNAME][YYYY-MM-DD_HHhMMmSSs][USERNAME].RCREC`

RemotelyAnywhere Access

Default Configuration

The default RemotelyAnywhere access configuration is as follows:

- Accepts Web and Console connections on port 2000 on all network adapters
- Accepts SSH connections on port 22 on all network adapters
- Accepts Telnet connections on port 23 on all network adapters
- All above connections must be authenticated with a username/password pair that identifies a user with Administrator rights
- Connections are accepted from any Internet address

Unused Services

Decide which RemotelyAnywhere services you want to use on the computer, and disable the rest. We recommend disabling Telnet and SSH if you do not need access to a command prompt. If you do, disable Telnet and leave SSH running.

Incoming Connections

You can change the IP address on which RemotelyAnywhere listens to incoming connections. Assume you have a web server on the Internet named www.MyWebServer.com and RemotelyAnywhere is installed on your web server on port 2000. The web server can now be remotely managed by typing `http://www.mywebserver.com:2000` in any web browser. However, by adding another IP address to the server for remote administration purposes and restricting RemotelyAnywhere to listen only to that particular address the server can only be accessed from this address. For example, if you add the address 177.246.27.91, the server can only be accessed by typing `http://177.246.27.91:2000`; any attempt to access RemotelyAnywhere using the original address (http://www.mywebserver.com:2000) will fail.

If possible, try to use an IP address that is on a different subnet than the IP addresses of the website or websites hosted on the computer. Most attacks begin with a port scan on the target computer. If the attacker sees port 2000 open on www.MyWebServer.com they will know that RemotelyAnywhere is installed on the computer. Using a different administrative IP address will hide this fact.

Default Port Numbers

Unlike the default Windows services, such as file sharing, the ports providing access to RemotelyAnywhere services can be changed to thwart potential intruders. You should, however, choose a port that is easy to remember.

SSH, Telnet, and FTP port numbers can all be configured under the relevant Preferences section of these services. The RemotelyAnywhere Service must be restarted for port changes to take effect, and you should take care to avoid port conflicts with other services when choosing new RemotelyAnywhere ports. Port conflicts may cause failure when starting the RemotelyAnywhere System service.

Local Software Firewall Protection

Local software firewalls, such as Windows Firewall, ZoneAlarm, or Norton 360, will filter applications from accessing local ports. For operation of RemotelyAnywhere access, these tools will need to be configured to allow communication over the RemotelyAnywhere service ports. By implementing additional controls in these third party tools, you can further restrict the access process to RemotelyAnywhere installations.

Network Access Mapping

By default, RemotelyAnywhere is implemented in a LAN or WAN environment, allowing access only to internal resources. Typically there is no firewall between two LAN devices, so access to RemotelyAnywhere's management port is not blocked or restricted.

A user must be present in the LAN and behind the Internet facing firewalls to attempt access to a RemotelyAnywhere server. To gain access to a server from outside the company environment over highly untrusted networks such as the Internet, network engineers usually have to deal with one or more firewalls. Allowing access to RemotelyAnywhere installations from an outside network is slightly more complicated.

Firewall Port Mapping

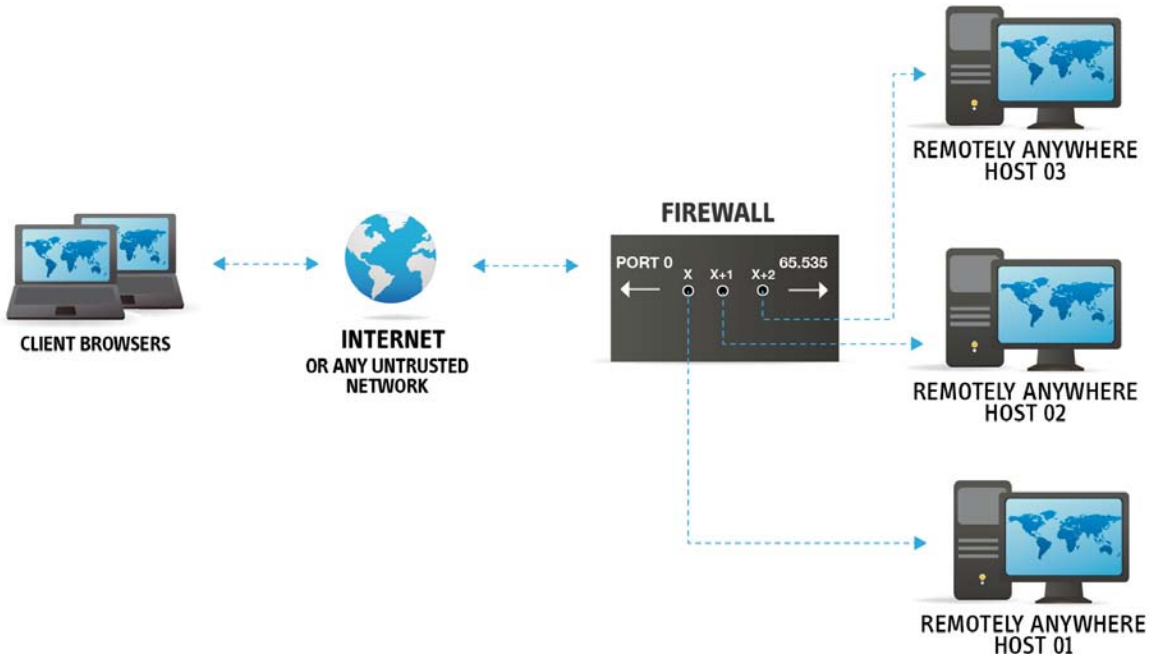

Figure 2. RemotelyAnywhere and Firewall Port Mapping

Figure 2 describes the basic case of several RemotelyAnywhere installations behind a firewall. System administrators can choose which computers should be available from the Internet and map a port on the firewall to each computer that needs to be accessed from outside the corporate security perimeter.

In the above example, assuming that the firewall's external IP address is 123.456.789.012, Host 01 can be accessed by going to 123.456.789.012:X, Host 02 can be accessed at 123.456.789.012:X+1, etc.

This scenario is very workable for a small number of hosts, but presents several issues when a large number of computers need to be made accessible. Most notably, the system administrator will need to keep track of and manage a number of port mappings, therefore introducing a human factor which is always a risk factor. Furthermore each computer that is accessible over the Internet will need to be kept secure, which can be a daunting task.

SSH Tunneling

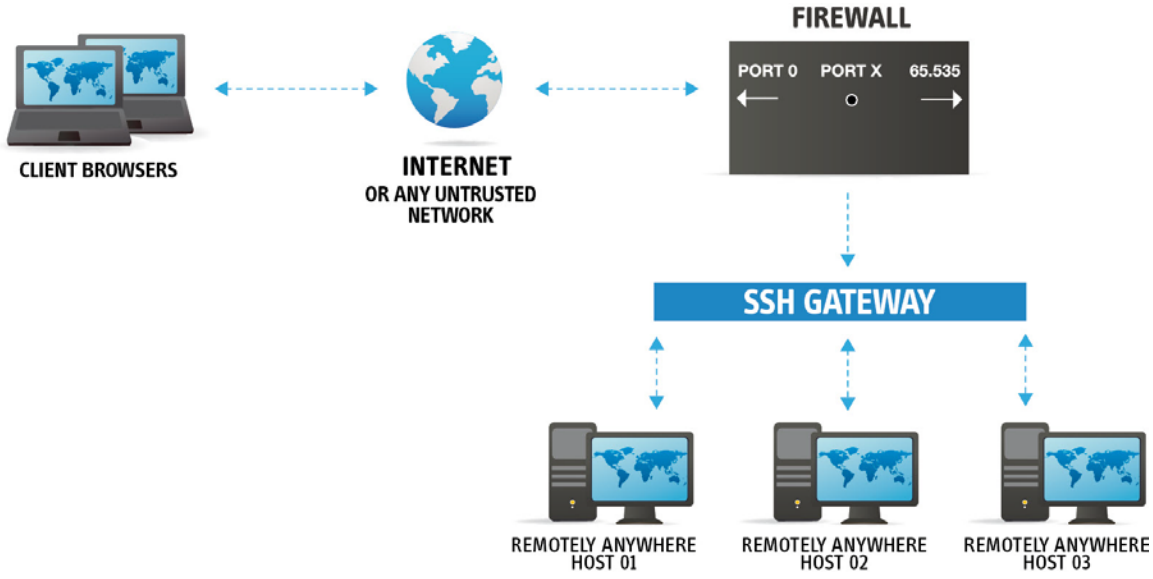

Figure 3. RemotelyAnywhere and SSH Tunneling

Figure 3 shows another scenario wherein a single port needs to be opened on the firewall. This requires the use of an SSH gateway. RemotelyAnywhere's Server edition includes a feature-complete SSH server, so a single Windows computer running RemotelyAnywhere can take this role.

The administrator will need to open a single port on the firewall and map it to the SSH port of the gateway. The client computers need to run SSH client software and establish an SSH2 session to the SSH gateway (there are many SSH client software implementations available, ranging from free to commercially supported ones, but RemotelyAnywhere's Server edition also includes an SSH Server module). Then the client's user will need to configure the SSH client to map a certain local port (x) to the RemotelyAnywhere port of the desired host on the internal network. Then the client can launch a browser and go to `localhost:x` to gain access to the RemotelyAnywhere login screen on the target computer.

This temporary tunnel is available while the SSH client is running and connected to the SSH gateway. The advantages of implementing this scenario are obvious: there's a single point of entry to the network and this is where the system administrator needs to concentrate his efforts. The disadvantage is that users will need to be educated on using an SSH client. To use RemotelyAnywhere's built-in SSH server and SSH Port forwarding features, go to **Preferences** > **SSH Server** and **Server Functions** > **Port Forwarding Config**.

Conclusion

With professional implementation of RemotelyAnywhere, the risk status of an existing networked system does not increase - compared to pre-RemotelyAnywhere levels.

A well-designed remote access solution can greatly increase productivity and provide a rapid return on investment. When deployment is done with care and RemotelyAnywhere's optional security features are utilized, the benefits greatly outweigh the risks.

References

SSL and TLS: Designing and Building Secure Systems by Eric A. Rescorla, Addison-Wesley Pub Co, 2001. ISBN: 0-201-61598-3

SSH, The Secure Shell: The Definitive Guide by Daniel J. Barrett, Ph. D., Richard E. Silverman, and Robert G. Byrnes, O'Reilly & Associates, 2001. ISBN: 0-596-00011-1