Server Security

Contents

Is Rumpus Secure?	2
Use Care When Creating User Accounts	2
Managing Passwords	3
Watch Out For Symbolic Links	4
Deploy A Firewall	5
Minimize Running Applications And Processes	5
Manage Physical Access To The Server	5
Set Rumpus Security Settings Properly	5
SSL: Trusted, Encrypted Transfers	6
IP Address Restrictions	6

Is Rumpus Secure?

This is a common question, but unfortunately there isn't a simple answer. Rumpus can certainly be used as part of a strategy in maintaining a well secured server, but it is crucial to understand that the security of your server depends not only on the software, but on how it is used. Like a hammer, for example, Rumpus is a tool that can be used safely, but needs to be used properly and with care.

It is also important to understand that there is no such thing as an unconditionally secure server. Stating unequivocally that a server is "secure" is simply impossible. Instead, server administrators need to take reasonable security precautions to achieve a level of security necessitated by the content of the server.

In other words, first consider the importance of the data on your server and the value it might have to an unauthorized user or the cost that would be incurred if the data were destroyed. The effort you spend securing your server should reflect the nature of the data you are protecting. This is no different than in the physical world, where greater security measures are employed by banks than by grocery stores, for example. Unfortunately, this common sense concept is often overlooked due to sensationalized stories of Internet theft and a general lack of understanding about how the Internet works.

The key to avoiding problems is to use common sense and remain diligent as you administer the server. Presented below are a few tips that will help you maintain reasonable security. Of course, this isn't an exhaustive list of every security issue you should consider. Again, common sense and an understanding of your actual security needs should guide your overall strategy to keeping your server safe.

Use Care When Creating User Accounts

One big advantage of Rumpus over some other FTP servers is that you don't have to create system user accounts in order to grant new users access to the Rumpus server. Right away, this prevents users from using their FTP login account to access the server via SSH, File Sharing, or other protocols that may be enabled on your server. However, be sure to set new user account Home Folders to well contained areas of the system, and limit FTP privileges to those capabilities required by the user. When creating new accounts, use common sense to assign access rights based on the trustworthiness of the person that will be using the account.

Each person who requires access to your server should be assigned their own user account. Even when multiple people are part of the same organization, each individual should be assigned a unique name and password, rather than defining a single account for the organization. Allowing users to share accounts creates access problems (for example, when one person leaves the organization) and makes it impossible to distinguish the actions of different people, should you ever have the need to review activity logs or track server usage.

Managing Passwords

Passwords are, by far, the most obvious and important mechanism used to secure your server, so it is important to assign and manage them carefully.

When assigning passwords, be sure to create non-trivial passwords of reasonable length. We recommend passwords of 6 to 12 characters that contain a mixed combination of uppercase letters, lowercase letters and numbers. Never assign common passwords like "password", "admin", or "1234". The Rumpus password generator (activated using the "new" button next to the password field on the Define Users window) generates random 8 character passwords with a good mix of upper and lower case letters and numbers.

System administrators may have access to user account passwords, so we don't recommend allowing people to use the same password they use on any other system or Web site.

Rumpus provides three different choices for storing user passwords on the server. You'll find this option on the "Preferences" tab of the "Network Settings" window in Rumpus.

Strong Password Encryption

Using strong password encryption, Rumpus uses a one-way secure hash (SHA-1, with salt) to store user account passwords in such a way that they can't be recovered. In this case, once a password is set, it is encrypted so that even a system administrator will be unable to view or in any way retrieve a user's password. This option offers a high level of security; even if a knowledgable hacker were to gain access to the Rumpus user account database (the "Rumpus.users" file in the Rumpus configuration folder), they would be prevented from discovering user account passwords or logging in to the Rumpus server.

This method of password storage is by far the most secure option and is mandatory in any strictly secured environment. The drawback of this option is that it is impossible for an administrator or legitimate user to recover a forgotten password. In the case of a forgotten password, a new password must be assigned.

Weak Password Encryption

When weak password encryption is used, a simple two-way algorithm is used to encode passwords when they are stored in the Rumpus user database. This prevents casual users who may have access to the Rumpus.users file from viewing passwords. It is, however, fairly trivial to extract passwords and it should be assumed that anyone who gains access to the Rumpus.users file has full access to every user account name and password defined in Rumpus. It is therefore important to carefully protect the Rumpus.users file when this option is used.

Use of weak password authentication is not suitable for strictly secured environments, but does offer the benefit of being able to retrieve and view user account passwords. In environments where security is not a primary concern and convenience is important, this option may be used to prevent casual users who have physical or extended remote access to the server from accidentally revealing passwords. When this option is used, it is particularly important to discourage users from using the same password on multiple systems, as a breach of security on the Rumpus server may lead to compromised security of those other systems.

Plain Text

Storing passwords in plain text in the Rumpus.users file is to be avoided, as it offers no benefit within Rumpus over weak password encryption. The only reason to store passwords in plain text is when Rumpus is used in conjunction with third party applications or scripts that also access the Rumpus.users file. For example, if you maintain user accounts in an external database or spreadsheet and need to import and export the Rumpus user database as part of that more comprehensive system, then storing passwords in plain text may be required.

If you do not need to integrate Rumpus user account management with an external application or script, then choose either weak or strong password encryption.

As is the case with weak password encryption, special care should be taken to prevent anyone from obtaining unnecessary access to the Rumpus.users file, and it is particularly important to discourage users from re-using passwords from other Web sites or systems.

Watch Out For Symbolic Links

Mac OS aliases and Windows Shortcuts (symbolic links, or symlinks, for short) are extremely handy for giving users access to areas of the hard drive outside of their Home Folder as needed, but they need to be used carefully. One misplaced symlink can change a user's access from a tightly controlled home folder to free reign over your entire network. And remember that more than one symlink can be used in a path. For example, if the folder "Bob" includes a symlink to the top level of the server hard drive, and a symlink to Bob's folder is put in the Home Folder "Tom", then Tom will instantly be granted access not only to Bob's folder, but the entire server hard drive.

Deploy A Firewall

Both Windows and OS X include firewalls that are simple and effective, and enabling a firewall is perhaps the most basic security precaution you can take. When first getting started with Rumpus, it often makes sense to disable the built-in firewall to keep it from interfering with your server as you bring it on-line. However, once your server is running and tested, turn the firewall on. You'll need to make holes in the firewall, of course, to allow FTP and WFM traffic, usually on ports 21, 80 (or 8000) and 3000-3008. For full details, see the "Firewall Setup" article in the Rumpus package.

Minimize Running Applications And Processes

Many server attacks make use not of a single insecure server process, but several processes used together in unexpected ways. Reduce this risk by minimizing the number of programs running on your server. And don't think that Rumpus is the only process running... Open the system Activity Monitor and you'll find dozens of active programs running on your server.

Manage Physical Access To The Server

Physically securing a server is often overlooked, but all of the firewalls and software controls in the world won't stop someone from messing with your system if someone can sit down at the keyboard with system administrator privileges. If necessary, keep the server in a locked closet or server room.

Set Rumpus Security Settings Properly

Rumpus includes numerous security features, some of which prevent general forms of attack and others that block specific potential vulnerabilities. Control over most of these features is provided by the "Security" tab of the FTP Settings window. In general, the default settings for these options are recommended, but a quick review is a good idea to make sure that the Rumpus defaults make sense for your server. Be sure to see the help page for the FTP Settings options for details on each security option.

In particular, be sure to set the "Restrict Access" option to the most restrictive setting that is appropriate for your server. If possible, it is best to set the FTP Root folder to the highest level folder to which Rumpus users should ever be given access. For example, if the FTP content folder is "/Users/Shared/" or "C:\Rumpus\FTPRoot\", and no user should ever be granted access to any file or folder outside that top-level folder, then that folder should be made the FTP Root. The "Restrict Access" option can then be set to "FTP Root Folder", ensuring that attempts by unscrupulous users to gain access to content outside the FTP area will be denied.

SSL: Trusted, Encrypted Transfers

Standard FTP and HTTP are generally considered insecure because the commands and data are transferred in clear text between the client and the server. This means that any network device between the client and the server can read, capture and even alter the data being sent. While this is potentially a serious problem, keep in mind that not every computer on the Internet can "listen in" on every data transfer. A reasonable analogy is the global telephone system, where phones can be "tapped" by patching into the network at some point between the two ends of the connection. And just as the easiest way to listen in on a phone conversation is to pick up an extension phone in the same house, note that other computers on the same local network can usually be used to intercept communications fairly easily.

Making the fact that basic FTP and Web data streams are unencrypted even worse is that the visibility of the text includes the user's name and password. This makes it possible for a hacker to intercept just the initial login portion of an FTP session in order to discover a secure account password.

Encrypting FTP and HTTP sessions is, of course, possible. Web browsers typically include well-integrated SSL capability, and many FTP clients support SSL encrypted sessions as well. Rumpus WFM and FTP sessions can encrypted by enabling secure services, and these options are described in the "Secure Transfers" article in this package.

IP Address Restrictions

In addition to requiring users to authenticate themselves, Rumpus can also restrict access to specific computers. Rumpus allows you to specify, by IP address, troublesome computers and will deny them even the attempt at logging on to your server. In addition, Rumpus can actively track potential attacks and block offending computers automatically.

A list of "blacklisted" client IP addresses is managed using the "Blocked Clients" window.

Any connection attempt made from a computer with an IP address in this list will be immediately terminated. Subnets can also be specified simply by entering "0" for one or more parts of the address. For example, to block access from the address "192.168.1.33" you would enter the entire address, but to block all access from any address on the entire "192.168.1.XXX" subnet, you would add "192.168.1.0" to the list.

For additional information on managing the block list, see the "Blocked Clients" help page in Rumpus.