



*Data delivery over the Internet can be simple, convenient, and cost-effective - but it must also be secure. Traditional file transfer methods do not provide adequate security, transmitting account information like user names, passwords, and data in the clear where they can be easily intercepted. The current version of Secure Shell (SSH2) provides a secure alternative. This paper explains the Secure Shell File Transfer Protocol (SFTP) and presents specific applications for system administration, finance, health care, and business-to-business. VanDyke Software clients and servers provide secure file transfer capabilities for Windows and are interoperable with SSH software on other platforms.*

## **Transferring Files Safely with Secure Shell**

Schools, hospitals, government agencies, and enterprises – today, all of these organizations are tapping the power of the Internet to access and distribute mission-critical information. Readily-available, low-cost connectivity makes it possible to deliver files to customers, business partners, and employees, inexpensively and immediately. Why rush that report over to FedEx when you could post it on a file server right now? Why courier patient medical records from diagnostic lab to hospital when they could be transferred immediately to the doctor who needs them?

Leveraging the Internet for data delivery is simple, convenient, and cost-effective – as long as file access, confidentiality, and integrity can be protected. Fortunately, cryptographic techniques like encryption, public-key authentication, and hashed-message authentication codes can prevent unauthorized disclosure or modification of private data. The trick is to provide this protection without slowing deployment, inhibiting ease-of-use, or running up the cost.

Many organizations have found that they can meet these objectives by transferring files safely with Secure Shell (often referred to as SecSH or SSH®), using products like VanDyke Software's VShell® server for Windows® and UNIX®, SecureFX®, and SecureCRT®. This paper describes how secure file transfer works, where it can be used, and the support provided by these products.

### **Secure Shell Safeguards File Transfer**

Secure Shell is an Internet standard originally designed to enable secure remote logon. Secure Shell employs state-of-the-art cryptographic technology to safeguard bits in transit and adds port forwarding to securely “tunnel” data between a client and server, over an otherwise unsecured network like the public Internet.

Secure Shell begins with strong authentication, using a combination of encrypted passwords and/or RSA/DSA public-keys to verify the identity of the client and server. With Secure Shell, organizations don't have to share easily-compromised text passwords with business partners and suppliers – they can rely on identifiers that are unique and secure, yet easily generated and distributed.

This authentication is combined with flexible access controls that ensure only authorized parties have access to sensitive files. Using Secure Shell, organizations don't need to compartmentalize files on different servers – for example, by dedicating a file server to each business partner. Instead, Windows or UNIX file servers running VanDyke Software's VShell combine strong authentication with file access privileges. Individual groups and users can be given access to SFTP without granting shell or port-forwarding privileges. Once a user has logged into SFTP, VShell enforces security permissions for read/write access for each file and folder.

Secure Shell preserves the confidentiality of all transferred data, including usernames and passwords, directory listings, and file contents. Symmetric ciphers like DES, 3DES, RC4, Twofish, Blowfish or AES can be used to encrypt data sent over a Secure Shell session. Rather than rely on manually-configured keys, Secure Shell employs public-key encryption to generate a random key for each session, used only until the session ends or the key is refreshed. These measures provide very strong protection against eavesdropping when files are transferred over the Internet.

For added protection against modification in transit, the most recent version of Secure Shell (referred to as SSH2) applies keyed Message Authentication Codes (MACs), based on SHA1 and MD5. These data integrity measures eliminate both accidental corruption and malicious tampering of messages exchanged over a Secure Shell session.

To learn more about the Secure Shell standards, protocols, and the cryptographic technologies employed by VanDyke's file transfer products, refer to our [Secure Shell Overview](#).

### ***Traditional Methods Provide Inadequate Security***

The traditional UNIX command-line utility for copying named files and directories is remote copy (RCP). In heterogeneous networks, the Internet file transfer protocol (FTP) is commonly used for interactive directory listing and file copy. FTP and RCP are very useful file transfer tools, but they are not secure. Sniffers can easily capture usernames, passwords, directory listings, and file content.

The first version of Secure Shell (SSH1) reduced security risks by "port forwarding" RCP and FTP, tunneled over a Secure Shell session. These now-legacy methods are commonly referred to as secure copy (SCP) and FTP over Secure Shell, respectively. A client running RCP or FTP and Secure Shell software (for example, VanDyke Software's SecureCRT) encrypts and tunnels traffic to a Secure Shell server (for example, VanDyke Software's VShell). At the Secure Shell server, the tunneled stream is decrypted. The file server can be on the Secure Shell server itself. Alternatively (see Figure 1), the cleartext stream can be "port forwarded" from the Secure Shell server to a target file server located somewhere in the private network.

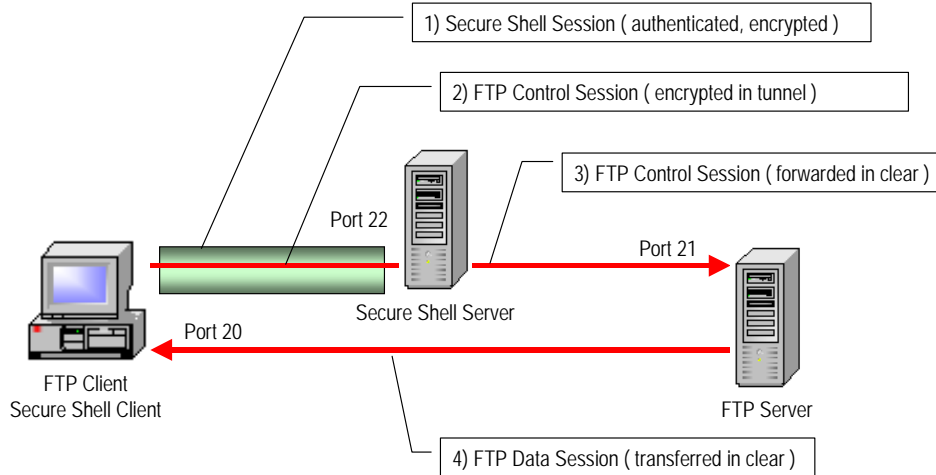


Figure 1: Port-Forwarded FTP Control Session (tunneled in SSH1 session)

Unfortunately, standard FTP uses separate TCP connections for control and data. FTP servers listen to port 21 for incoming requests. FTP clients authenticate themselves by connecting to this control port. Data connections are established as needed to get or put files, initiated from arbitrary ports. FTP control traffic can be port-forwarded over Secure Shell, preventing username and password sniffing. However, file content must be transferred outside Secure Shell, over an unprotected cleartext data connection.

### **Secure FTP Is A Better Answer**

SSH2 introduced a more robust method of secure file transfer: Secure Shell File Transfer Protocol. SFTP leverages Secure Shell for authenticated, encrypted file transfer *without requiring an Internet FTP server*. FTP servers (ftpd daemons) are a common target for exploits that can compromise the entire system. SFTP provides the functionality of regular FTP without the risks associated with running unprotected FTP daemons. Replacing FTP with SFTP can significantly reduce a file server's vulnerability. Furthermore, SFTP is not hampered by FTP's multi-connection architecture. As shown in Figure 2, SFTP protects every bit – usernames, passwords, listings, and file data – exchanged between an SFTP client and server.

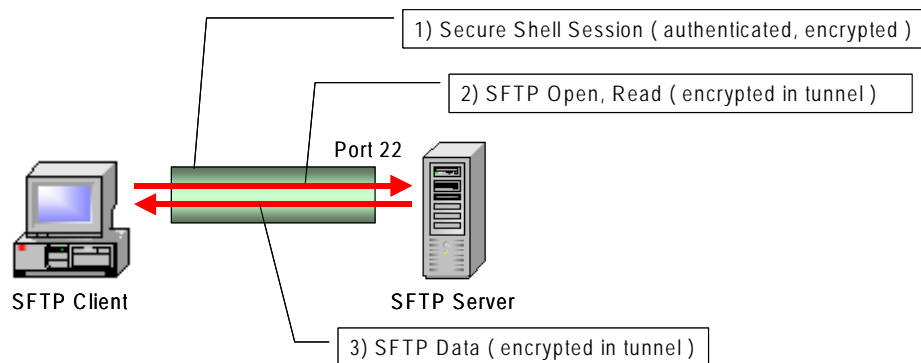


Figure 2: SFTP Open, Read Commands (tunneled in SSH2 session)

SFTP does not use port forwarding. Instead, SFTP operates as a subsystem, integrated with SSH2. An SFTP client like VanDyke Software's SecureFX initiates a Secure Shell session to a target SFTP server like VanDyke Software's VShell. The SFTP protocol consists of remote file system commands like open and read; these commands are tunneled directly through the existing Secure Shell session. A subset of SFTP also provides the basis for SCP(2), a replacement for port-forwarded SCP.

To the end user, SFTP and SCP(2) appear quite similar to the legacy file transfer methods they replace. But, when it comes to security, it's what's inside that counts. For maximum interoperability, VanDyke Software's SecureFX supports both secure and legacy file transfer methods. Whenever possible, organizations should use SFTP, the most robust method available for transferring files safely over Secure Shell.

### **Secure File Transfer In System Administration**

Secure Shell can be used for secure system administration, as shown in Figure 3. Instead of cleartext Telnet, many administrators prefer using a Secure Shell client like SecureCRT for remote logon or command-shell access. Secure Shell daemons are usually present on UNIX servers and increasingly found on network devices like routers, switches, and firewalls. Secure administration of Windows and UNIX servers can be greatly enhanced using VanDyke Software's VShell server which provides a wide range of authentication options and fine-tune control over user and group privileges.

Many administration tasks are interactive, but a complete solution also requires secure file transfer. System administrators must transfer software, configuration files, user account data, and usage records. FTP over Secure Shell protects the root password – essential for after-hours remote administration over the public Internet. SFTP goes a step further by protecting valuable and sensitive file content. For example, transferring account records over SFTP prevents unauthorized disclosure of credit card numbers, permissions, and passwords. Furthermore, doing so proves that you've taken steps to ensure privacy, potentially limiting liability.

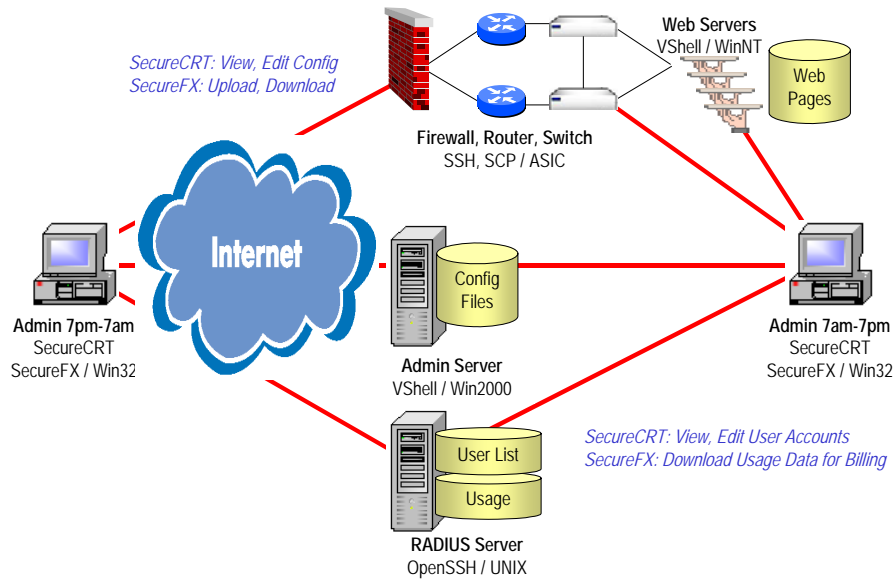


Figure 3: Secure System Administration

### **Secure File Transfer For Business-to-Business**

SFTP can be used to transfer files securely within and between businesses, as shown in Figure 4. Deploying SFTP servers at strategic intranet and extranet locations creates a cross-platform file sharing infrastructure for interacting with and delivering work products to business units, customers, and partners. In this example, the accounting department uses SFTP to deliver financial spreadsheets to an outside auditor and purchase orders to a manufacturer. Online delivery increases business efficiency, but only authorized parties must be permitted to access these files. By combining password and public-key authentication, this company verifies recipient identity before sending any file. By using MACs to detect modification, recipients are assured that copied files remain authentic.

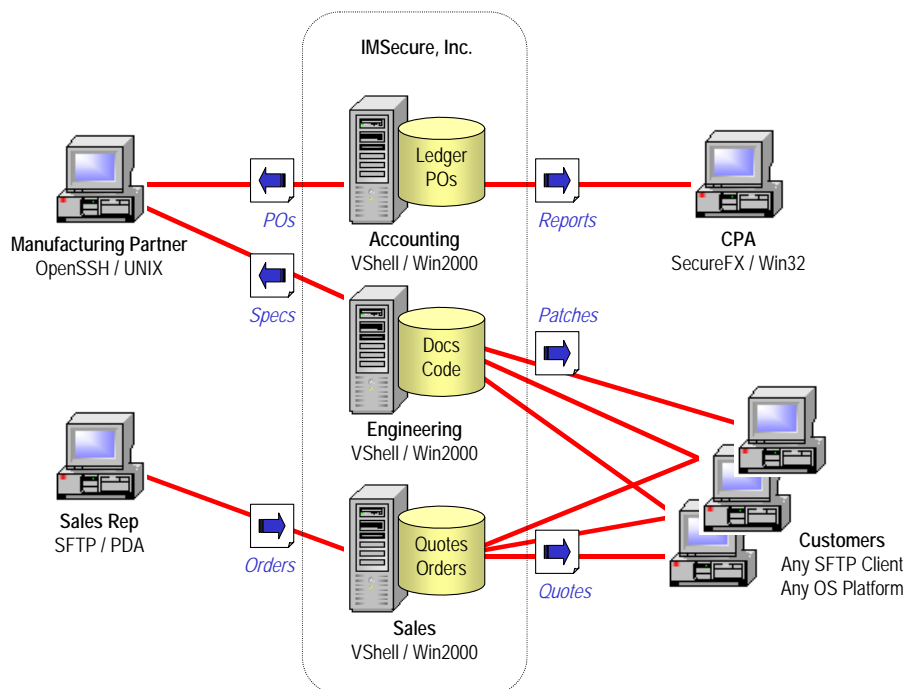


Figure 4: Secure Business-to-Business

In business-to-business transactions, organizations are usually unable to dictate the operating system, server, or client software employed by others. For example, consider consultants delivering confidential reports and IT service companies delivering software patches to customers. These situations require a platform-independent solution that can be deployed quickly, with minimal investment, accommodating any customer. File transfer based on Secure Shell is well suited because low or no-cost software is readily available for nearly every OS, and interoperability issues are relatively uncommon.

While there are many business motivations to protect the confidentiality of transferred files, legislation is a factor of increasing importance. In the US, federal, state, and local governments have enacted privacy legislation, requiring businesses to define policies that limit disclosure of personally-identifiable information. The European Union Data Protection Directive requires that information transfer to a third country only take place if certain conditions are met. This directive applies not just within the EU, but to any company doing business with European nationals.

### **Secure File Transfer Between Financial Institutions**

Some privacy laws single out a specific industry; in the US, one example is the Gramm Leach Billey (GLB) act. Intended to enhance competition in the financial services industry, GLB includes a provision requiring consumer privacy protection. The Federal Reserve System, national banks, and savings associations are not the only organizations impacted; mortgage companies and insurance underwriters are also included. Under GLB, financial institutions must establish appropriate security and confidentiality measures for customer records – specifically, preventing unauthorized disclosure of non-public personal information. GLB compliance starts with policy definition; SFTP is one tool available for implementing those policies.

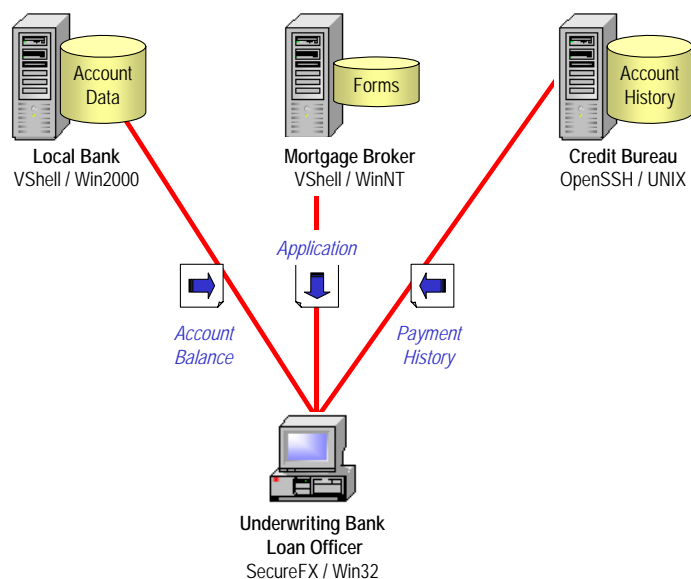


Figure 5: Securing Files Shared Between Financial Institutions

For example, SFTP can provide strong authentication and role-based access to private data involved in mortgage approval when several companies are involved. As shown in Figure 5, an underwriting bank uses SFTP to pull loan applications from a mortgage broker's database, obtain history from a credit bureau, and verify account balances. In this example, SFTP ensures the integrity and confidentiality of non-public personal information in transit between cooperating financial institutions. Server event logs provide an audit trail, identifying who accessed what and when. Of course, SFTP must be combined with additional enterprise security measures, protecting data stored at each financial institution.

### **Secure File Transfer In Healthcare**

Healthcare is another industry significantly impacted by new privacy legislation. The Health Insurance Portability and Accountability Act (HIPAA) of 1996 was created to facilitate the flow of healthcare information while protecting confidential patient data from inappropriate access, disclosure, and use. HIPAA regulations define transaction codes and forms, privacy rights, information security, and identifiers for patients, providers, plans, and employers. HIPAA security requirements cover administrative policies and procedures, physical safeguards, technical services, and technical mechanisms. Technical services cover “data at rest”. Technical mechanisms cover “data in motion”, requiring entity authentication, access control, encryption, data integrity, event reporting, and alarms. SFTP is clearly a useful tool for implementing policies that comply with HIPAA security requirements.

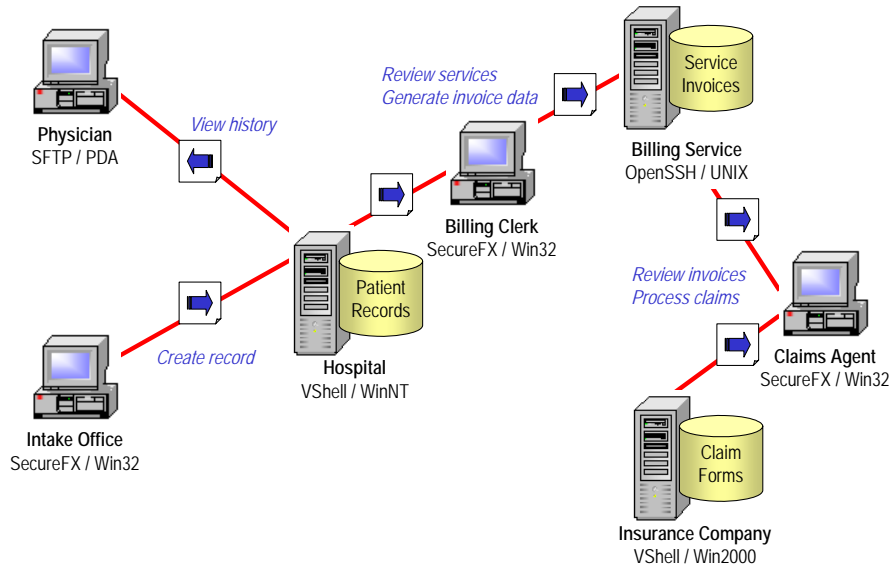


Figure 6: Securing Patient Records in Healthcare

Figure 6 illustrates how SFTP can be used as a technical mechanism, protecting data in motion within a distributed healthcare system. Organizations affected by HIPAA include healthcare providers (physicians, hospitals), health plans (insurance companies, HMOs, Medicare), clearinghouses (billing services, claims repricing companies), and any other business partner involved in the “chain of trust”. In this example, Secure Shell MACs prevent message alteration. Secure Shell passwords and public-key authentication control file access, at user and group levels. Secure Shell encryption ensures the confidentiality of healthcare information. Server event logs provide the information needed to facilitate a security audit. HIPAA also mandates that physicians have fast emergency access to patient records generated by others. Secure electronic access through a standard protocol like SFTP complies with this requirement.

### ***VanDyke's Solutions For Secure File Transfer***

VanDyke Software’s VShell, SecureCRT, SecureFX, and VanDyke ClientPack products provide broad support for secure file transfer. These products enable the applications illustrated in this paper and are deployed in a wide variety of industries, ranging from IT, financial, education, and business/consulting services to healthcare, internet, and telecommunications service providers.

**VanDyke ClientPack’s VCP** is a Windows command-line SCP(2) utility, perfect for scripting routine administrative tasks that might otherwise be accomplished over FTP without security. VCP brings the convenience and security of UNIX SCP(2) to any Microsoft Windows platform.

**VanDyke ClientPack’s VSFTP** is a Windows command-line SFTP utility, perfect for scripting routine administrative tasks that might otherwise be accomplished over FTP without security.

**SecureFX** is a Windows file transfer client that supports SFTP and FTP protocols, integrated under one easy-to-use graphical user interface. SecureFX is inexpensive and takes just a minute to install. It offers an extensive set of security options, including AES and Blowfish encryption and RSA/DSA public-key authentication. SecureFX supports multiple concurrent file transfers. Advanced features include convenient drag and drop file transfer, one-click folder synchronization, and network neighborhood integration. SecureFX includes `securefxcl.exe`, a command-line SFTP utility that can be used to create automated, unattended, secure file transfer sessions.

**VShell** is a full-featured Secure Shell server for Windows and UNIX. It supports SSH2 command shell and port-forwarded sessions, initiated from SecureCRT or any standard SSH2 client. In addition, it supports secure file transfer from any Windows or UNIX SFTP or SCP(2) client, including SecureFX SFTP and VanDyke ClientPack's VCP. VShell supports wide range of authentication methods including encrypted password, public-key, Kerberos, and keyboard interactive, and X.509 digital certificates, and enforces access restrictions based on user account, privilege and file permissions. Easy-to-define filters can permit or deny access to clients, identified by IP address, subnet, hostname, or domain. SFTP access can be limited to specific users or groups. A root folder can be specified to limit access to only a section of the file system. Replacing an unsecured FTP server with a secure VShell SFTP server can be accomplished in less than five minutes. VShell licenses are available for personal, workgroup, or enterprise use.