Mobile Remote Control Architecture

George Stefan BOGDAN
IT&C Security Master
Department of Economic Informatics and Cybernetics
Bucharest University of Economic Studies
ROMANIA
bogdan.george.stefan@gmail.com

Abstract: When mobile device become more and more popular the need to have a software bridge between them and the old fashioned computers became evident. Many different solutions have appeared to fill the void but few offer more than file sinking or remote SSH (secure shell) connections. For computer professionals and other alike a tool that can do more regarding this connectivity and control gap became necessary. With this paper it is described an architecture and an implementation for creating such a system, underlining the difficulties, challenges and the many choices regarding overall system design, security and implementation that need to be made in order to provide the user with a secure, reliable and professional solution.

Key-Words: remote, mobile, secure, server, system, network

1. Introduction

When mobile device become more and more popular the need to have a software bridge between them and the old fashioned computers became evident. Many different solutions have appeared to fill the void but few offer more than file sinking or remote SSH (secure shell) connections. For computer professionals and other alike a tool that can do more regarding this connectivity and control gap became necessary. With this paper it is described an architecture and an implementation for creating such a system, underlining the difficulties, challenges and the many choices regarding overall system design, security and implementation that need to be made in order to provide the user with a secure, reliable and professional solution.

The remote desktop architecture involves establishing a connection over the internet between a mobile device and a computer. Through this connection the mobile device should have the possibility to control certain functions and tasks that run on the computer. Because of the specifics of mobile devices and the connectivity issues that arise from different computer networks architectures a third component appears to complete the system: the server that acts like a proxy between the other components. Given the nature of the software described, security represents a major issue that was taken into consideration early in the design phase. A poor understanding of these issues may cause data loss, data theft or even provide an attacker with complete control over the target system. By security, I mean here more than the custom data connection encryption. Solutions for different areas of the entire system were provided so that in the end, the outcome may be a coherent security solution that targets not only individual components but the entire system.

Providing a strategy for all parties involved means approaching the problem in a way that focuses both on how users interact with applications and how different parts of the system interact with each other. The part that influences the system in a new, unique way is the mobile device platform. As the software paradigm changes from desktop computers, servers and communication between them to mobile devices and the communication with traditional architectures and solutions, so does the way we approach such as system, especially through the security perspective. Starting from connectivity problems that may arise from using untrusted, public wireless networks, ease of physical access that mobile devices are subject to and ending with the
changes in user behavior when it comes to approaching these new generation devices, new solutions must be found and implemented in order to offer the same level of protection the user became accustomed to experience in the personal computer world.

The system provides an application for the Android operating system, an application for a desktop computer running Microsoft Windows and a server component that will act as a proxy between the two applications mentioned here. The server will be known through this paper as the proxy. The system is based on communication between the three data components. Its role is to allow a user with a mobile device running the Android operating system access the data from a remote computer by using an application. The data that is passed around in the system can contain any information that the user chooses to access. Also access to installed and running processes and services is provided through the application. Furthermore, messages that are exchanged between these software modules contain information and commands that can give easy access to the system. The entire communication between the three software components requires security as to allow secure communication and data transfer no matter of the access point used by the mobile device.

2. Overall architecture

By definition the software system has two components: the client which resides on a mobile device and the server which is the computer. The role of the computer is to allow the client to connect and issue a set of commands that can be interpreted and executed. From this basic model a few characteristics appear: the client and the server must both be reachable over the internet, the server must run in the background since its purpose is to serve information and does not require local manipulation by the user, the client should be running on demand to save the devices battery power. The third system component is the server (both hardware and software). It has a crucial role in the entire architecture because it is the common link between the computer and the client. Note: to avoid confusion between the attributes and names of the three components that make up the system I will refer to the software running on the computer as the server and to the software running on a remote machine (which is always up and running) as the proxy.

2.1. The server

The server is a software component that runs in the background on the target computer as a system service. It does not have a user interface. It is designed with an install once – run forever philosophy. It was not designed to require any user intervention for any of its functions. The server receives commands from the client through the proxy over the internet and responds with data without requesting permission from the user. The main requests to which the server responds are:
- Request for a list of all running programs and the amount of time each one has been running for.
- Request for a list of files starting from the root (the drive letters) or the files contained in a folder specified by the client.

The main actions that the server implements are:
- System restart
- System shut down
- Process kill (closes a process identified by the process ID)

The server is installed as a windows service. It implements a communication module that handles the connections over TCP/IP with the remote proxy. Besides that, it monitors the system for events and for counting the time elapsed since an application started.

2.2. The client
The client is a software component running on a mobile device powered by Android operating system [1]. It has an interface through which the user can manipulate the entire system. It is the only component that presents a user interface. All the other components run in the background and do not require user intervention. Its main function is to display information that comes from the remote computer which has the server side software installed. The main tasks that can be performed with the client are:

- Authentication
- Request for running processes list from the server (the remote computer)
- Request for the drives list from the server
- Request for a file from a folder selected from the interface of the client
- Request to shutdown, restart the system
- Request to close a running process.

2.3. The proxy

The proxy is a software component that runs on a dedicated machine which is connected to the internet and has a public IP address. Its role is to handle all the connections between clients and servers and to dispatch appropriate messages between two connected parties. The main tasks accomplished by this component are:

- Accept and verify connections both from clients and servers.
- Send messages from clients to servers in real time.
- Keep a database of users
- Authenticate the other 2 components if requested
- Dispatch all incoming traffic to the right component based on identification and.

3. Communication

The three components that compose the system communicate over HTTP with special formatted XML messages. These messages are encrypted using the open-source blowfish algorithm. Message encryption was necessary because the messages send back and forth between the different components can contain critical information about the user its system and can even provide enough information to take down the user's machine. The structure of the message is common through the system. All components must be able to understand a message that is received even though it was not intended for that specific component. This was accomplished by following a general rule for message formatting. Thus, all messages contain a message description tag which represents a system wide global value that identifies a unique message. No two messages have the same identification number. Each message also contains message body tag. That tag contains all the useful information for that specific message. The data in the message body tag is interpreted according to the description for that message.

Not all components are interested in all the messages received. For example, the proxy is only interested of information such as login data. The rest is ignored and passed to the component that is intended to receive it. For determining which data goes where, the proxy looks at every piece of information it receives. That involves decrypting each message and analyzing its content.

As shown, the user is responsible for installing the software both on the mobile device and the personal computer. The proxy is build to interact with many components at the same time and it is not subject to direct manipulation by the user. The proxy is a third party component here. The only thing the user must do is to create an account and provide some credentials. The mobile device issues the requests and the proxy dispatches them to the computer. When a response message...
arrives, the proxy sends it to the device that formed the initial data request.

4. System security

The user can, from the mobile device, access data from a personal computer through a link over the Internet, managed by a proxy. This setup brings up several security issues involved in the entire data transfer process. The most important security issue of the entire system is the way the three components connect and exchange information. Many factors influence this process. Among them:
- Mobile network security (wireless networks);
- Mobile device security (physical);
- Proxy security;
- Lack of control over communication media;
- Mobile operating system security.

Many types of attacks can be deployed on the system. All though it is impossible to offer protection against all of them, a reasonable amount of attacks can be stopped or rendered useless by choosing the appropriate security model and protection level. Among the most popular types of attacks on such systems we have
- Denial of service (DOS)
- Man in the middle
- Packet replay

The solution provided for this system took into accounts all of the above and manages to offer protection against known threats.

This was achieved through industry-standard algorithms and protocols, such as Diffie-Hellman [2] and Blowfish [3] algorithms, and by deploying other security measures related to binary package protection and identity theft protection.

Because of the unique system characteristics, some traditional security measures were left out and other combined. Also, advances in cryptanalysis pushed out a number of security measures usually deploy in systems that perform remote communication or sensitive data transfers through insecure channels. A few examples of such technologies include SSL [4] (Secure Sockets Layer) and MD5 [5].

Several other algorithms were taken into consideration but were left out due to lack of rigorous mathematical proofs for safety or support in the programming languages used to create the system. An example is Password-Authenticated Key Agreement [6]. Details about the technology will be presented later.

5. General security design aspects

Two approaches become evident in securing a system that performs communications over the internet.

5.1. Local authentication

The first assumes local authentication based on a local generated key and the user credentials. This process would take place on both the mobile device and the remote desktop machine - the user is trying to communicate with. The algorithms used could provide keys that follow a logic known only by the proxy. Not even the user would have knowledge on the implementation. Each time the user starts using the application, a key would be created to communicate with the server. All messages would be encrypted using that key based on the user credentials. Authentication would be made only once, using secure networks. The user password would only travel once between the three system components. The user credentials would be stored on the local device.

This model brings some immediate benefits to the system, such as:
- The username and password do not have to travel on the internet each time;
Without access to the key generation algorithms, an attacker cannot reproduce messages. This model also comes with some major drawbacks:

- User credentials are stored on a local machine. By gaining physical access to the machine, an attacker could obtain that information.
- The key generation algorithms may be deduced through cryptanalysis.
- The data is as secure as the systems used to store them.

The greatest security risk in this case is the mobile operating system and physical access to the device. Unlike desktop computers, mobile devices bring along a new set of difficulties when it comes to securing information and data. Among the differences between the two types of devices are: Mobile devices are easier to steal. While computers have a secure, stable location, a mobile device can easily be stolen or lost.

- Mobile devices have new operating systems that have not passed the test of time when it comes to proving their security. Traditional operating systems for computers like Microsoft Windows have been proven to be safe or have changed over a long period of time to meet security requirements of the higher standard.
- Mobile devices connect to wireless networks without knowing how safe they actually are. Opposed to this, traditional computers are part of networks with a high trust level.

The mobile operating system used in implementing this system is Android. Android is an open source operating system created and maintained by Google. All data stored on it is as safe as the system itself.

Android offers a security model based on the Linux operating system and enhanced with features specific to mobile platforms. Thus, each application that is installed requests access to different areas of the system as well as to different data. Based on the requested permissions, the user is presented with the choice of installing or not the application. To bypass this system, all that is needed to be done is trick the user into accepting an application that presents itself as safe and instead try to take control of the system.

Being based on the Linux security, each application runs in its own user group and cannot be elevated to root user. Although this is the case for the vast majority of application, some have managed to beat the security module and gain full access to the entire system as well as to other applications that are installed on it. The private data stored by those applications becomes readable for the malware with elevated rights. For this reason, a model which is based on local authentication is considered to be unsafe for mobile devices (running the Android operating system).

### 5.2. Remote Authentication

The second method of securing communication over the internet in this kind of system means securing the communication by encrypting the data being send to and from the server. Traditionally such an encryption could be made using SSL (Secure Socket Layer) with certificates issued by a trusted authority, password-authenticated key agreement or MD5 (Message Digest 5). Even if all of these technologies are proven technologies, they all come with drawbacks that make them unsuitable for the kind of system presented here.

The SSL protocol was originally developed by Netscape, to ensure security of data transported and routed through HTTP, LDAP or POP3 application layers [11]. SSL is designed to make use of TCP as a communication layer to provide a reliable end-to-end secure and authenticated connection between two points over a network (for example between the service client and the server). SSL can be used for protection...
of data in transit in situations related to any network service. It is used mostly in HTTP server and client applications.

The main objectives for SSL are [11]:

- authenticating the client and server to each other. The SSL protocol supports the use of standard key cryptographic techniques (public key encryption) to authenticate the communicating parties to each other. Though the most frequent application consists in authenticating the service client on the basis of a certificate, SSL may also use the same methods to authenticate the client.
- ensuring data integrity: during a session, data cannot be either intentionally or unintentionally tampered with.
- securing data privacy: data in transport between the client and the server must be protected from interception and be readable only by the intended recipient. This prerequisite is necessary for both the data associated with the protocol itself (securing traffic during negotiations) and the application data that is sent during the session itself.

Digital certificates [12] are digital files that certify the identity of an individual or institution seeking access to computer-based information. The combination of standards, protocols, and software that support digital certificates is called a public key infrastructure, or PKI. The software that supports this infrastructure generates sets of public-private key pairs. Public-private key pairs are codes that are related to one another through a complex mathematical algorithm. Digital certificates are issued by certificate authorities. They are used for signing and authenticating users. SSL makes use of them when communicating with the server. Combined, they create a system very hard to break. Even if the protection is complete and provides authentication as well as data encryption, this solution has some drawbacks. They are related to the overhead implied by all of these extra checks and algorithms. For SSL to be powerful it also needs certificates to operate with. This is even more difficult when it comes to mobile platforms. Also the support offered by different hosting solutions may cause the system to be very specific and needy in respect to platform support. To issue and maintain a set of valid certificates for all mobile devices and to validate each one when a connection is established, means providing dedicated hardware than can magnify the cost of such an implementation by a factor. SSL is also much slower than transmitting data without encryption. Speed is at the essence for such a system. Physical limitations are in place due to weak performance offered by wireless networks. Adding to that complexity another layer could render the entire communication scheme useless.

6. Solution

In order to provide a reliable communication system, I have chosen an industry standard technique for this system: Diffie-Hellman key exchange algorithm. It is part of the SSL mechanism described above but provides only the needed functionality. The rest of the protection offered was duplicated by using build in mechanisms as described in the following paragraphs. Some were left out because I considered they only bring performance overheads and little or no protection.

The Diffie-Hellman key agreement protocol (also called exponential key agreement) was developed by Whitfield Diffie and Martin Hellman in 1976 and published in "New Directions in Cryptography." The protocol allows two users to exchange a secret key over an insecure medium without any prior secrets. This protocol is used in conjunction with the Blowfish algorithm.

Blowfish [13] is a keyed symmetric block cipher [14] designed in 1993 by Bruce
Schneier. It provides a good encryption rate in software and no effective cryptanalysis of it has been found. Blowfish is a general purpose, open source algorithm that came in a time where algorithms were mostly dedicated and proprietary (Including the - at the time - secret DES implementation).

First the system components obtain a Diffie-Hellman key and secure a communication channel with the server. This is done both by the mobile device and the personal computer. Next, for each message transmitted in the system, this key is used for encryption. This way, the user credentials as well as the rest of the data messages that get passed around are protected.

Even if this method assures the data is confidential, it does not provide a mechanism for eluding packet replay attacks. A replay attack is a form of network attack in which a valid data transmission is maliciously or fraudulently repeated or delayed. This is carried out either by the originator or by an adversary who intercepts the data and retransmits it. It is possible for an attacker to intercept key messages and perform on them offline cryptanalysis. It is also possible for an attacker to identify messages that perform certain crucial tasks on the remote computer such as: system shutdown or system restart. By resending these packets during an active connection the attacker could disrupt normal system functionality and cause data loss. To protect against such an attack, each packet has in it a number that describes its order. In this way, the proxy can check each message received and make sure that it is indeed the next one in line.

7. Conclusion

As technology evolves so do the threats and the security measures that must be taken to protect it. Providing safe communication in a system that involves a mobile party takes more than deploying some known algorithms. Even if solutions for client-server communications exist, in the mobile devices world I needed to keep track of other factors that do not represent a concern for traditional systems. A close analysis of the system can bring to surface new types of attacks and vulnerabilities not custom for traditional models involving personal computers. Letting out established technologies and implementing or redesigning some of them to better suit the needs of limited hardware performance or limited connectivity has been the major challenge of the entire design effort. The outcome is a system that has little performance penalties and a very high level of security. This model can be used to implement all kinds of solutions that involve a client-server model where the client is a mobile device and the information passed between the two is sensitive. It is not limited to Android platforms and can be ported to other devices powered by different operating systems.

References
[6] Taekyoung Kwon, Practical Authenticated Key Agreement using Passwords, School of Computer Engineering, Sejong University, Seoul 143-747, Korea

[14] Ivan I., Toma C., Secure Distributed Databases Using Cryptography, Informatica Economică, nr.3 (39), 2006, pp. 25 – 29, ISSN 1453-1305