International Conference on Cyber Security and Computer Science (ICONCS'18),
Oct 18-20, 2018 Safranbolu, Turkey

The Management and Configuration System for Low Cost Portable Crypto Device in an Embedded System

İ.M. ORAK[1] and O.YILDIZ[1]

[1] Karabuk University, Karabuk/Turkey, imorak@karabuk.edu.tr
[1]Duzce University, Duzce/Turkey, omer71173@ogr.duzce.edu.tr

***Abstract* - In this work, we propose an architecture and interface that enables the management and configuration of a portable crypto device running on an embedded system. The developed system is also designed to be capable of performing management tasks on any embedded system. Since it is not a language dependent architecture, programming language can be changed according to platform requirements. The management system uses a database on the GNU/Linux operating system and runs the necessary commands on the embedded system via an RPC scheme. Measures have been taken for security threats in the developed system using secure transport layer. The system is designed for client and server architecture. The C++ programming language is close to the machine language. For this reason, it runs faster than other common languages. So, it is used on the server side of the management system. Since the Java isolates operating system incompatibilities, it is used on the client side. Since the desktop application uses Java in the interface, it was also developed using Java SWT library.**

***Keywords* – management system, configuration system, embedded, RPC, secure management**

I. Introduction

ALONG with the rapid development of computer and communication technologies, network-supported embedded devices have taken their place in daily life. Therefore, Applications that manage and configure embedded system devices have become more important. Management and configuration are generally carried out over the network. The safety of the managed embedded device directly affects the security of the network. While embedded devices affect the security of the network, the network also affects the security of the devices. By reason of the fact that, embedded devices are often considered to be private network devices, management interfaces are not safe enough. Even the encryption devices are inadequate in terms of security. There are even hardware security modules that make the entire plain text network communication over TCP / IP [1]. As the time changes, attack methods are changing and evolving. For this reason, even in the private network, devices should communicate securely.

In this study, an easy-to-use design model is proposed to ensure the safe management and configuration of an embedded device, even in the private network.

II. Background Knowledge

A. Network Management

TCP / IP is a protocol that enables devices to communicate with each other on the Internet. It has four layers and the layers are shown in the figure.

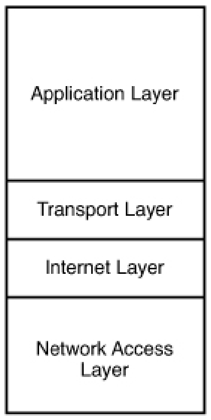


Figure 1: TCP/IP layers.

This protocol works in client / server architecture. A server application that will respond to requests runs continuously. Clients send requests to the running server and receive answers. TCP / IP is a stateless protocol. The session information of the clients is not kept. Each incoming connection is a new connection.

The protocol is widely used because it is compatible with all systems.

B. Embedded System

Embedded system composed of microprocessor, micro controller, some hardware and software to act as an operating system (OS) in PC with advantage in cheap price and high performance [2]. It has been widely used in the controlling kernel of mobile phone, PDA, and other electronic products. In the nearly future, embedded system will become the critical kernel of the intelligent digital home, mobile, and other intelligent devices to perform like PC [2].

In summary, the embedded system is a dedicated system for specific jobs.

C. Secure Layer

TCP / IP is a protocol that sends and receives plain data without encrypting. Therefore, it has a structure open to abuse. To close this gap, using the public key infrastructure, TLS (Transport Layer Security) was created, where the shared key was created, and the communication was encrypted.

The steps of TLS are shown in the figure below:

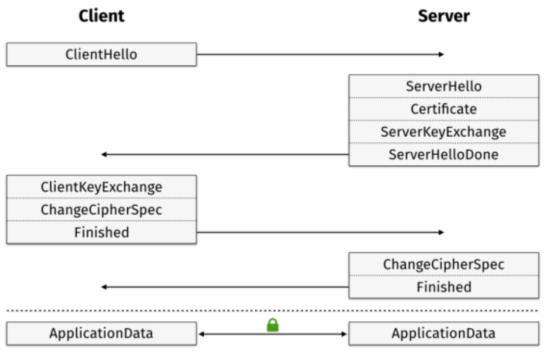


Figure 2: TLS handshaking [3].

III. Designing of Embedded Management and Configuration System

A. Secure Protocol for Embedded Device

Device management interface is written with Apache Thrift [4]. Apache Thrift has TLS support and security of the device is provided by this protocol. There are client and server in the system.

The client requests a connection from the crypto device for secure connection. The crypto device sends the certificate and public key information received from the same root authority to the client. The client-side library checks whether the certificate is trusted and whether the server has a certificate from the certification authority.

The server also checks the client for certificate validation. A key is created, and a session key is created. After the session key occurs, the data is symmetrically encrypted. The secure channel is created using TLS. Attackers who listen to the network cannot access plain text.

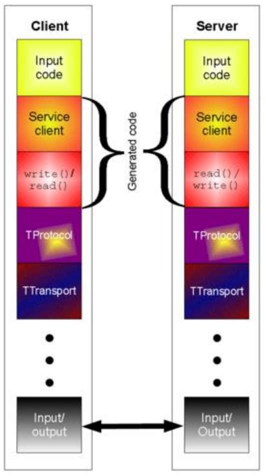


Figure 3: The Apache Thrift API client/server architecture [4].

B. Roles

Roles in the PKCS11 document are used for the device's management interface. These roles are Security Officer and User roles. The SO role from these roles is only used to initialize the device. It is not authorized to use any other functions. The normal user of the admin slot can use all other administrative functions.

C. Functions

Since PKCS11 is used in the device, the management interface also includes slot operations [5]. Functions designed to be used on any embedded device are as follows:

Table 1: Management functions.

Slot initialization	OC_InitToken()
Blank Slot Initialization	OC_InitFreeToken()
SO PIN Specify / Change	OC_SetPIN()
Initializing the User PIN	OC_SetPIN()
Network Settings	OC_SetNetworkConfiguration()
Time and Date	OC_SetDate(), OC_SetTime()
Backup	OC_GetRecoveryFile(), OC_SetRecoveryFile()
Factory Reset	OC_DeviceReset()

IV. General Design of The Configuration and Management System

In the general structure, an Apache Thrift identification structure was created. Skeleton was created for server service using Thrift's library generation program. The inside of this skeleton was filled with the algorithm and the server was made operational. The client is again constructed from the same structure. Unlike any other language supported by Thrift library production can be done. Finally, library written application and client communicates with the server.

In the study, the Java programming language is used for the client.

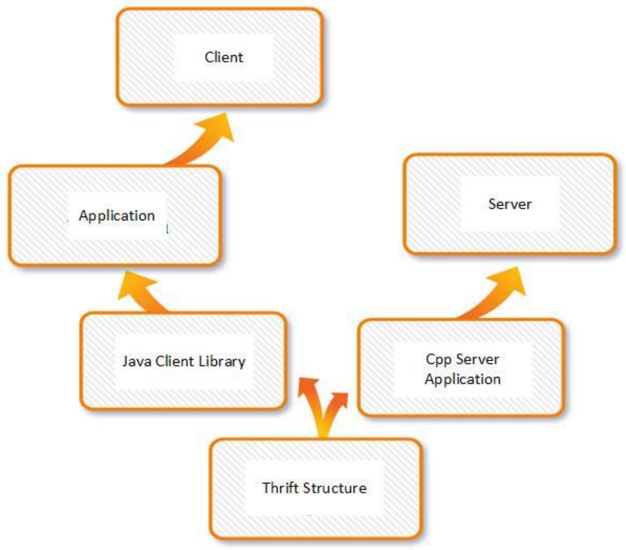


Figure 4: General structure of client/server applications.

A. *Management Service Structure*

The management service is a service program that responds to server-side requests. This service only responds to commands from the API.

It is necessary to design the data structure of the middleware to provide communication between the client and the server. For this purpose, C++ types and their functions were defined by means of a definition language. Apache Thrift was used as the definition language.

Table 2: C++ structure and Thrift structure.

C++ Structure	Thrift
bool OC_SetDate(std::string ocDate)	bool OC_SetDate(1: string ocDate) throws (1:ErrorCode rev)

The identification structure used to capture non-PKCS11 structure errors is as Table 3.

.

Table 3: Thrift exception structure.

```
exception ErrorCode {
1: i32 revoke;
}
```

Represented data structures are common to both the server and the client.

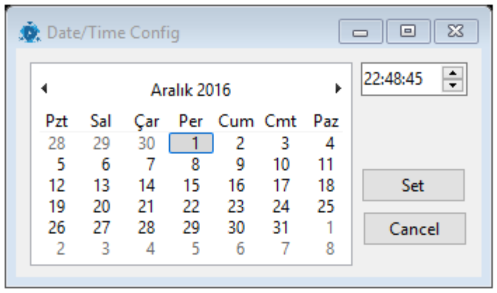
The network program that will run on the server side makes the network exchange with Apache Thrift. To achieve this, the data structures we have mentioned previously must be defined in a common way and the files should be produced according to the desired language. On the crypto device, after the operations were done, the functions on the server side, which meet and respond to the request, were prepared.

In remote procedure functions, all operations are performed according to the following algorithm:

- Define local data types.
- Set the data received by the client to local data types.
- Call the corresponding function with local data types.
- If the result is returned correctly, set the data type to be sent with local variables.
- Return the desired data type as the return value.

B. *Client Structure*

The management application uses a definition language. Therefore, the client can be coded independently of the programming language. Using Apache Thrift, the client library is created in the desired language. The client application is created using the Management Application API. In Figure 5, the sample client window of the time set function mentioned earlier was written using Java Swing:



Pzt	Sal	Çar	Per	Cum	Cmt	Paz
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Figure 5: Date/Time config window.

V. Conclusion

A management and configuration system have been developed that provides a secure connection for cryptographic devices. The lack of language dependency on the client will be easier for the applications to be written.

References

[1] N. A. Ivarsson Johan, "A Review of Hardware Security Modules," 2010.

[2] X. Wu, Y. Zhu, and X. Deng, "Design and Implementation of Embedded SNMP Network Management Manager in Web-Based Mode," in *2008 IEEE Asia-Pacific Services Computing Conference*, 2008, pp. 1512–1516.

[3] D. Evans, "The First Few Milliseconds of an TLS 1.2 Connection · TLSeminar." [Online]. Available: https://tlseminar.github.io/first-few-milliseconds/. [Accessed: 14-Oct-2018].

[4] A. Prunicki, *Apache Thrift: Introduction*. http://www.ociweb.com/: Object Computing Inc. %93 An Open Solutions Company.

[5] RSA, *PKCS #11: Cryptographic Token Interface Standard*. .