TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Computer Engineering

Denis Konstantinov 111615 IASM

Embedded service oriented microcontroller architecture.

Extensible client-server communication architecture for small devices

Master thesis

Supervisor: Peeter Elervee
Associate Professor at the Department of Computer Engineering / Ph.D., Dipl.Eng.

Tallinn 2013

Author's Declaration

This work is composed by myself independently. All other authors' works, essential states from literary sources and facts from other origins, which were used during the composition of this work, are referenced.

Signature of candidate: Date:

Acronyms

ASIC Application-specific integrated circuit. 5

Annotation

Current work introduces conceptual approaches for implementing an extensible service oriented client-server application on a small microcontroller. This is a general-purpose transport and hardware independent embedded server that uses remote procedure calls as primary communication protocol. This server looks like remote service that could provide defined functions to the client. ...

Annotatsioon

Annotatsioon eesti keeles

Contents

1 Introduction

1.1 Outline

1.2 Contributions

This is a test of acronyms ASIC[1]

This is a test of code listings

```
32 public Response getResponse(long timeout) throws InterruptedException {
33         synchronized (lock) {
34                 if (response != null) {
35                         return response;
36                 }
37
38                 lock.wait(timeout);
39         }
40
41         reader.removeInputHandler(this);
42         return response;
43 }
44
45
46
```

Listing 1: Example of a listing.

At first here will be lots of words about this cruel world and how it was changed in resent years.
Speech about reusable components.
Speech about small mobile devices that are everywhere.
Communication.
Interaction.

[1] Application-specific integrated circuit

2 Preliminaries

2.1 Service oriented architecture

2.2 Data serialization

2.2.1 JSON

2.2.2 XML

2.2.3 Others

3 System architecture

This section will introduce you a main architecture of the system.

3.1 Introduction

Here will be about coffee machine example in general

3.2 Server architecture

3.3 Client architecture

4 Implementation

Here will be implementation report.

4.1 Implementation of the embedded server

Here will be STM32 server implementation.

4.1.1 Authentication and Authorization

Need of security Users are essential part of every system. System shold be designed with a requirement, that there will be at least one user. System without any users does not make sense. Usually information systems have lots of users with different roles. There should be an system administrator - the most authorized individual in the system, managers and normal users. System should distinct them all somehow.

Another requirement is system and information security. System may contain sensitive data, that should not be available to general users. In case of remote services there are some services that are not open. These services or some of their parts are require some identity to pass through.

Let's take a usual website as example. Common website has at least three different user roles: user or guest, content publisher and system administrator. Last two roles may be joined together, but in general content publishers do not do system maintenance, they just work with content of webpages. There may be more different roles, but these are the main ones. Imagine you open a web page and you see the content. You follow the links and surf the web site. If you want to change something, for example you do not like the design or some words on web page were misspelled, you need to find special place where you can input your **credentials** and get into the system. This will happen only if you have proper **permission** to do that. When you get inside you are still not able to do anything due to lack of privileges. For example you cannot turn of the webserver or disable the your website. There may be lots of different roles and responsibilities in the system and each role has limited access to system resources.

Embedded device as a service may be similar to system example above. Device may have some limited use cases, that are not available to not authorized service clients. This may be internal information retrieving, some device manipulation functions (turn on/off something, delete/remove something from the system, change of system preferences). Some device functionality may be available only for limited people, for example system owner. The real life example of such system is the wireless router. Router clients are other computers, they can send and receive network packets. Router uses wireless security protocols, which permit unauthorized access. Even if you are connected to a secure access point, you are not able to change system settings. You should have admin permission (password) to manage the system. This kind of system, like many embedded systems, is made for one purpose. Router purpose is to provide access for the network. You also can remember lots of similar systems, that use authorized access. Nowadays it is not new to get remotely into some device and to change internals, but embedded system integrations are still not so common. Imagine near future, you are sitting at work and thinking to go home. After a long day you became really hungry. You take your smartphone and connect to your remote wireless fridge service at home. You type your password and get list of all food in your fridge. Now you know what you need to buy and the real candidates to be throwed to the rubbish bin. You adjust power in some fridge area and your beer will be very cold when you get home. Is it just a dream? Is it really hard to realize using present time technologies?

The main problem here is the security. Nowadays lots of communication between different systems goes through the wireless channel. Radio link is also available to your neighbour behind the wall. Generally, you do not want to broadcast what is in your fridge or to give ability to connect to your air conditioning service. Therefore you need to use some authentication scheme for your service.

Authentication protocols Authentication is any process by which you verify that someone is who they claim they are. (https://httpd.apache.org/docs/2.2/howto/auth.html)

Humanity has already invented a lot of different authentication techniques.

The ways in which someone may be authenticated fall into three categories: (http://en.wikipedia.org/wiki/Aut

- the ownership factors: Something the user has (e.g., wrist band, ID card, security token, software token, phone, or cell phone)
- the knowledge factors: Something the user knows (e.g., a password, pass phrase, or personal identification number (PIN), challenge response (the user must answer a question), pattern)
- the inherence factors: Something the user is or does (e.g., fingerprint, retinal pattern, DNA sequence (there are assorted definitions of what is sufficient), signature, face, voice, unique bio-electric signals, or other biometric identifier).

Authentication may be one way (only client is checked for validity) and two way (both client and server check each other). Some systems may requre to use different security factors together: you say password, provide ID card and show you fingerprint There are also available many standart authentication protocols. If you start searching you will probably find similar list:

- Transport Layer Security (TLS)
- Extensible Authentication Protocol (EAP)
- Password authentication protocol (PAP)
- Challenge-Handshake Authentication Protocol (CHAP)
- Password-authenticated key agreement
- Remote Authentication Dial In User Service (RADIUS)
- Kerberos
- Lightweight Extensible Authentication Protocol (LEAP)

Choosing suitable protocol is not trivial problem. There is no any case general protocol. Most of them are designed to interconnect big computers inside network. Mostly hey operate on transport and application level and use TCP/IP protocol stack. Protocols could be devided into these groups:

- Protocols that transmit the secret over the network. (For example Password authentication protocol). These protocols are not secure.
- Protocols that not send secrets and providing authentification sending messages. (CHAP and Password-authenticated key agreement).
- Protocols that require a trusted third party.

One solution is to use closed encrypted proprietary protocol and be calm, but as it was mentioned earlier, it limits the possibility of integration between other embedded systems. It this case all of your devices should support that protocol and you choise of different hardware is limited. Proprietary protocols are often vendor-specific, code is closed, documentation is not free and all it works only with the proprietary devices from the manufacturer.

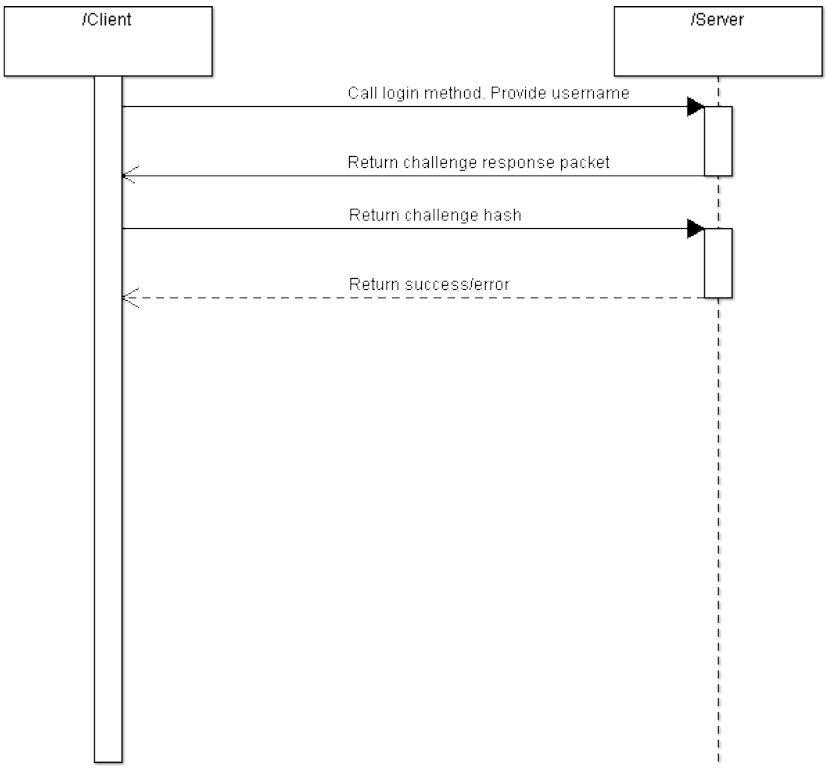

Figure 1: Client authentification process

4.2 General purpose service library implementation

Here will be general purpose library implementation report.

4.3 Implementation of android client

Here will be android java client implementation report.

Android development Some words about development under Android platform

5 Conclusions

5.1 Future work

List of Figures

List of Tables

References