Chapter 6

SYSTEM EVALUATION

6.1 SECURITY ASSURANCE

Proving that a security policy is absolutely secure is impossible. Formal methods, such as BAN logic, can find bugs in a security protocol design as they force the designer to take everything into consideration. These methods are too complicated to prove larger protocols and are also based on educated assumptions. Qualitative and intuitive methods therefore proves to be the most successful when deciding on the effectiveness of a policy. It must also be accepted that no security policy is perfectly secure. Attackers have infinite resources, human errors might create additional vulnerabilities and assumptions made during the implementation might no longer be valid after some time has passed. There are two different concepts that needs to be mentioned [25]:

- Assurance: An estimate of the likelihood that a system will not fail in a practical way. This is based on the process used to develop the system, the experience of the designers and technical assessments. Assurance looks at functionality, strength of mechanisms, implementation and usability. In recent years assurance has placed less emphasis on the product and started to concentrate more on the method used to develop the product.

- Evaluation: The process of assembling evidence that a system meets, or fails to meet, a prescribed assurance target. This is needed when the party relying on the system, and is therefore taking risks using the system, is not the party implementing or designing the system. Evaluation is either performed by the relying party until it is satisfied or by a third party (CLEF). CLEF's audit and test the technical aspects of the system according to a set standard. An evaluation certificate is then issued stating the evaluation assurance level of the policy. There are a number of these standards [25]:

  1. FIPS 140-1: Tamper resistance of cryptographic processors.
  2. IVV: Standards for nuclear weapon systems and NASA manned space flights.
  3. Orange book: Used by the government of the USA
  4. CTPEC: Accepted standard in Canada.
  5. ITSEC: Accepted standard in European countries.
  6. Common Criteria: Single standard now accepted in the US, Canada and Europe. Also see section 2.2.4

To evaluate the system proposed in this report would not be possible taken that the resources are not available to submit it to a CLEF. The best way to determine the effectiveness of the policy would be to use an intuitive and qualitative approach. Assurance will be given that the policy meets a number of criteria defined by the designer. The following criteria must be adhered to:

1. The design methodology followed must adhere to accepted standards or specifications.

2. The implemented policy must assure that the functional requirement of the network model identified in section 3.2.2 will be upheld.

3. The implemented policy must assure that all the threats identified in section 3.2.3 are addressed.

6.2 PERFORMANCE OF SECURITY MECHANISMS

Different security policies were described in section 5.3. Each of these policies specified alternative mechanisms to implement some security services. Some security services can be implemented using different mechanisms, e.g. authentication can be accomplished by asymmetric and symmetric cryptography. The implementation of the mechanisms has an impact on the performance of the system and the effectiveness of the particular policy. The embedded system's main constraint is storage space and not processing power. A policy with an excessive storage requirement cannot be implemented while a powerful workstation and an 8-bit controller can both implement cryptographic mechanisms, the one is just faster. Apart from storage and processing resources a mechanism also requires careful management depending on its complexity and requirements, e.g. symmetric key encryption requires an extensive key distribution system The following aspects of the security mechanisms must therefore be determined (given in order of importance):

1. Storage requirements: Memory is an expensive commodity and can easily raise the cost of a system a great deal. The variables and storage space required to implement the mechanism must be evaluated.

2. Processing requirements: The time taken to execute the operations needed to implement the mechanism must be evaluated.

3. Other requirements: Additional steps might be needed in the security policy if certain mechanisms are used. Complexity of the mechanism's implementation and management must be taken into account.

6.3 TEST SYSTEM

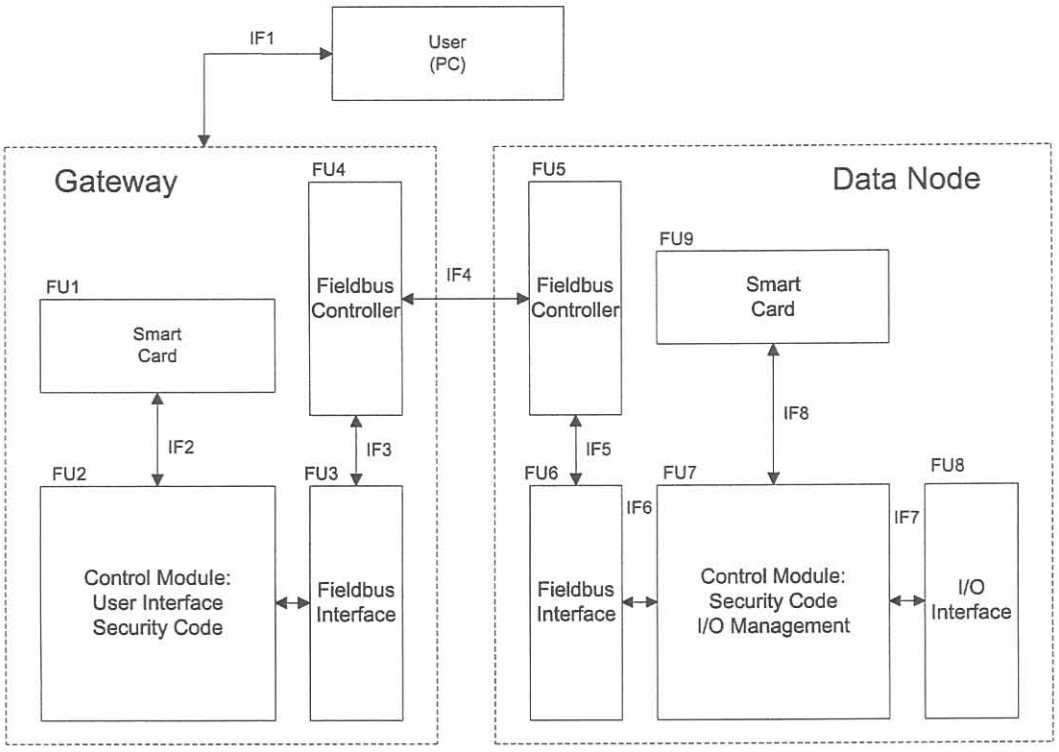

Figure 6.1:
Funcional analysis of the test system

A test system was implemented in order to test the performance and viability of the security policies. The functional analysis of the system is shown in figure 6.1. The security module (FU1 and FU9) incorporates a smart card and the support circuitry to connect it to the control modules (FU2 and FU7). The user interface shows statistics and data gained from nodes on the network. It also allows the user to issue commands for various tasks that the node can perform. The PC interface (IF1) physically connects the PC to the rest of the system. The gateway's control module (FU2) regulates information flow between the PC, security module and the network interface (FU3). The network interface and controllers (FU3, FU4, FU5 and FU6) allow the control modules physical access to

the network. The network interface (IF4) interconnects the PC gateway and all the nodes of the system together and allows for information to flow between them. The external interface (FU8) allows the node's control module (FU7) to gain access to external processes. The control module (FU7) interacts with the external interface (FU8) and controls traffic flow to the security module (FU9). It also responds to commands sent over the network interface (IF4) from the rest of the system. The hardware implementation is shown in figure 6.2.

Brief description of system components:

- The CAN protocol was developed in the 1980s for automobile applications. The CAN protocol implements the two lower layers (physical and data link) of the OSI network layer model. CAN is a real-time serial bus protocol offering high reliability, both physically and with error correction [56]. Messages are not addressed to specific stations but a unique identifier labels each message. This identifier describes the message contents and the priority of the message. Only data that is applicable to a station's own function will be processed. This means that only data with an identification header that realises a filter hit on the node will be processed. The CAN protocol is easy to implement as a wide range of IC manufacturers offer CAN bus drivers with DSPs and MCUs being released that have onboard CAN capabilities. The low bandwidth, bus architecture and message based communication models the behaviour of field area networks in general. The system's CAN network is configured at 125Kb per second.

- Microchips's PIC16F876 microcontroller is a 8-bit processor (200ns instruction cycle) with 256 bytes of EEPROM and 768 bytes of data RAM. It contains the peripherals to connect to the smart card, the user PC and CAN interface.

- The smart card used is the STARCOS 2.3 system by Giesecke and Devrient [57]. The smart card communicates at 9600 bps and uses the T=1 communication protocol (see ISO 7816 [37] for details). The cards used had 32Kbytes of data memory. It provides public and symmetric key cryptographic mechanisms:

  - Encryption: DES, 3DES, RSA
  - Signatures: DSA and RSA
  - Hash: 3DES, MD5, SHA-1

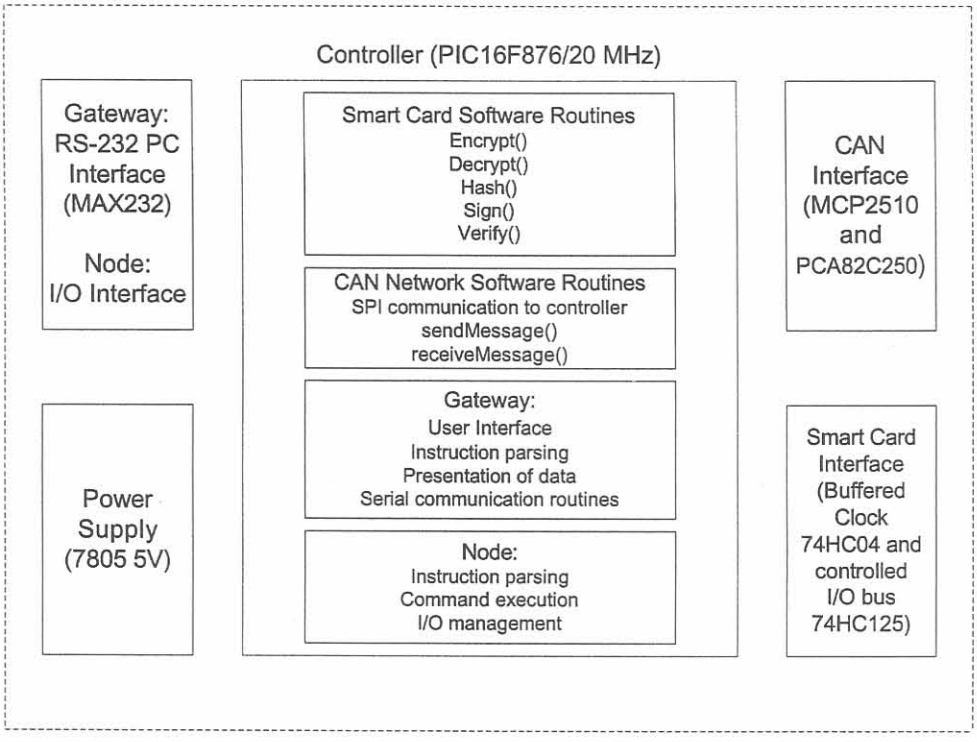

Figure 6.2:
Hardware implementation of the system