
# NAVAL POSTGRADUATE SCHOOL

**MONTEREY, CALIFORNIA**

# THESIS

**SECURING EMERGENCY STATE DATA IN A TACTICAL COMPUTING ENVIRONMENT**

by

Raymond Quah

December 2010

Thesis Co-Advisors: Cynthia E. Irvine
Timothy E. Levin

**Approved for public release; distribution is unlimited**

THIS PAGE INTENTIONALLY LEFT BLANK

| REPORT DOCUMENTATION PAGE | | | Form Approved OMB No. 0704-0188 |
|---|---|---|---|
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503. | | | |
| **1. AGENCY USE ONLY *(Leave blank)*** | **2. REPORT DATE** December 2010 | **3. REPORT TYPE AND DATES COVERED** Master's Thesis | |
| **4. TITLE AND SUBTITLE** Securing Emergency State Data in a Tactical Computing Environment | | **5. FUNDING NUMBERS** | |
| **6. AUTHOR(S)** Raymond Quah | | | |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School Monterey, CA 93943-5000 | | **8. PERFORMING ORGANIZATION REPORT NUMBER** | |
| **9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)** N/A | | **10. SPONSORING/MONITORING AGENCY REPORT NUMBER** | |
| **11. SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number _____NA_____. | | | |
| **12a. DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited | | **12b. DISTRIBUTION CODE** | |

**13. ABSTRACT (maximum 200 words)**

Allowing first responders to access emergency-support information for which they are normally not authorized can save lives and property. A trusted emergency management solution was developed at the Naval Postgraduate School to provide transient access to sensitive information during an emergency. This document provides a high-level design analysis for performing distributed emergency signaling for this system over a tactical IP network. We describe how the protocol supports the emergency management concept of operations. The document specifies requirements pertaining to emergency signaling and includes an analysis that describes the system response to all possible emergency state transitions. We also review various communication protocols and evaluate their merits with respect to their suitability for use on emergency networks. Detailed design analysis supports the commencement of an emergency management implementation phase.

| **14. SUBJECT TERMS** Emergency signaling, emergency state, transient access, heartbeat message | | | **15. NUMBER OF PAGES** 109 |
|---|---|---|---|
| | | | **16. PRICE CODE** |
| **17. SECURITY CLASSIFICATION OF REPORT** Unclassified | **18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified | **19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified | **20. LIMITATION OF ABSTRACT** UU |

NSN 7540-01-280-5500 Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

# SECURING EMERGENCY STATE DATA IN A TACTICAL COMPUTING ENVIRONMENT

Raymond Quah
Defence Science and Technology Agency, Singapore
B.S. (Comp), National University of Singapore, 2000
M.Tech. (SE), Institute of Systems Science (NUS), 2005

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

from the

**NAVAL POSTGRADUATE SCHOOL**
**December 2010**

Author: Raymond Quah

Approved by: Dr. Cynthia E. Irvine
Thesis Co-Advisor

Timothy E. Levin
Thesis Co-Advisor

Dr. Peter J. Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

Allowing first responders to access emergency-support information for which they are normally not authorized can save lives and property. A trusted emergency management solution was developed at the Naval Postgraduate School to provide transient access to sensitive information during an emergency. This document provides a high-level design analysis for performing distributed emergency signaling for this system over a tactical IP network. We describe how the protocol supports the emergency management concept of operations. The document specifies requirements pertaining to emergency signaling and includes an analysis that describes the system response to all possible emergency state transitions. We also review various communication protocols and evaluate their merits with respect to their suitability for use on emergency networks. Detailed design analysis supports the commencement of an emergency management implementation phase.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

# LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF ACRONYMS AND ABBREVIATIONS

| | |
|---|---|
| AES | Advanced Encryption Standard |
| AH | Authentication Header |
| BITS | Bump-In-The-Stack |
| BITW | Bump-In-The-Wire |
| CA | Certification Authority |
| CBC | Cipher Block Chaining |
| CIARN | Confidentiality, Integrity, Authenticity, Replay protection, and Non-Repudiation |
| CISR | Center for Information Systems Security Studies and Research |
| CONOPS | Concept of Operations |
| CRL | Certificate Revocation List |
| DH | Diffie-Hellman |
| ECDH | Elliptic Curve Diffie Hellman |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| ESM | Emergency State Message |
| ESP | Encapsulating Security Payload |
| FIPS | Federal Information Processing Standards |
| FTP | File Transfer Protocol |
| GPS | Global Positioning System |
| HMAC | Hash Based Message Authentication Code |
| IAN | Incident Area Network |
| IANA | Internet Assigned Numbers Authority |
| IKE | Internet Key Exchange |
| IP | Internet Protocol |
| IPComp | IP Payload Compression Protocol |
| IPSec | Internet Protocol Security |
| ISP | Internet Service Provider |
| LDAP | Lightweight Directory Access Protocol |
| LPSK | Least Privilege Separation Kernel |
| MLS | Multilevel System |
| NAT | Network Address Translator |
| NAT-T | NAT Traversal |
| NIST | National Institute of Standards and Technology |
| NPS | Naval Postgraduate School |
| NSA | National Security Agency |

| | |
|---|---|
| NTP | Network Time Protocol |
| OS | Operating System |
| PKI | Public Key Infrastructure |
| PL | Privilege Level |
| PPP | Point-to-Point Protocol |
| RA | Registration Authority |
| SAK | Secure Attention Key |
| SHA | Secure Hash Algorithm |
| SKPP | Separation Kernel Protection Profile |
| SLIP | Serial Line Internet Protocol |
| T-TASI | Transient Tactical Access to Sensitive Information |
| TCB | Trusted Computing Base |
| TCP | Transmission Control Protocol |
| TE | Trusted Executive |
| TOC | Tactical Operation Centre |
| TPA | Trusted Path Application |
| TSL | Trusted Services Layer |
| UDP | User Datagram Protocol |
| VPN | Virtual Private Network |

# ACKNOWLEDGMENTS

I would like to express my gratitude to all those who have assisted and supported me. Without them, I could not have completed the thesis.

I will start with my biggest supporters, my wife and daughter. My wife, Eileen, tirelessly handled most of the domestic chores and took care of our daughter day in and day out, while I spent countless hours on the computer. I would not have completed this endeavor without her. I would also like to applaud my daughter, Chanel, who continually amazes me with her childlike wonder.

For their professional guidance and patience, I would like to thank my two advisors, Professor Cynthia Irvine and Timothy E Levin. Both of them have allocated much time and energy in reviewing my thesis materials and ensuring that my progress is on track.

I am also grateful to my sponsor Defence Science and Technology Agency (DSTA) for providing me with the opportunity to study at the Naval Postgraduate School.

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

This thesis documents the analysis and design of performing emergency signaling within a tactical IP network. It provides high-level recommendations for the choice of communication protocol, and provides concrete design requirements pertaining to the signaling mechanism.

## A. PROBLEM DEFINITION

Crises can occur anywhere and at any time without warning. The emergency team members whose job is to deal with such situations may be geographically dispersed. These first responders may include medical services personnel, firefighters, law enforcement officers and resource coordinators. To be able to efficiently handle a crisis, the emergency team requires timely access to information and must be coordinated in its actions. The combination of communication and coordination is the key to an effective response to help save lives and limit damage [1].

Crisis response may involve occasional access to privileged or sensitive information that is beyond the security clearance of an individual. Such information may include floor plans, infrastructure schematics, city transit system plans, or even the medical records of disaster victims. Government policy makers and third-party data providers may be unwilling to release this information if they lack confidence in the ability of emergency systems to protect their data relative to its confidentiality, sensitivity, and privacy. As such, access to privileged information is often denied to first responders.

Many proposals were put forth to overcome such hurdles. We could temporarily elevate the security clearance of the first responder, or temporarily downgrade the security labels of the information. Both approaches require human intervention. Elevating an individual security clearance typically requires extensive background checks which prevents timely access to information. This

approach may also provide the individual with excessive access resulting in other security implications. Downgrading of security labels is likely to be disallowed due to the need to provide global and persistent policy enforcement. Also, it might provide too little access unless it is performed on a large scale basis.

A better solution involves developing a Transient Tactical Access to Sensitive Information (T-TASI) system that can process, store, and distribute information of different security classifications. Privileged information provided by third party data providers is kept in secure emergency partitions on the first responders' devices. The T-TASI system, also known as the E-Device, will enforce stringent access control such that the confidential data is accessible only in extraordinary situations with authorization from the authority. In these situations, the data is made available in a controlled and confined manner such that when the emergency ends, the T-TASI system is able to lockdown the emergency partitions and revoke all privileged information that was accessed during the emergency.

For this approach to be feasible, the authority needs to maintain a global emergency state. This state shall be synchronized throughout the emergency network via the use of a secure emergency signaling mechanism. A detailed design analysis is required to determine the implementation approach.

## B. OBJECTIVE

The thesis should help answer the following questions:

What are the requirements for secure emergency signaling?

Which communication protocol should be used to meet these requirements?

How should the signaling mechanism be designed and integrated into the existing tactical T-TASI system framework?

The thesis's contribution is in providing design guidance for secure transmission and a detailed design analysis for the processing of the emergency signals. The analysis shall provide sufficient details to commence the implementation phase.

## C. THESIS ORGANIZATION

The thesis is organized as followed.

The first chapter presents the problem definition and highlights the research questions that the thesis will address.

The second chapter provides background information that is relevant to understanding the problem domain. We will describe the concept of operations (CONOPS) in which the T-TASI system operates and provide a brief overview of prerequisite knowledge that will help the reader in appreciating the technical portions of the thesis.

The third chapter presents the conceptual requirements of the emergency signaling and the high-level security objectives. We review the communication protocols and evaluate them based on how well they support the CONOPS.

The fourth chapter discusses the high-level design of each of the major subsystems. It will cover state transitions and message semantics.

The fifth chapter presents a detailed analysis of the subsystems by describing the control flow and call parameters. It includes the required data structures and will provide sufficient details to commence the implementation phase.

The final chapter reviews related work, provides direction for subsequent research, and closes with a concluding section.

THIS PAGE INTENTIONALLY LEFT BLANK

# II. BACKGROUND

This chapter provides background information relevant to understanding the problem domain for signaling emergency status to tactical emergency devices. We will describe the CONOPS from the perspective of how the E-Device will be used. The T-TASI system implementation, or E-Device, is a small form factor device that serves as the target platform to validate the concept of transient security in the field. The security architecture implemented in the E-Device will be elaborated in this chapter. We also provide an overview of the internet protocol security (IPSec) and public key infrastructure (PKI) for readers who may not be familiar with them.

## A. CONCEPT OF OPERATIONS

The proposed concept of operations relies on having emergency teams comprised of individuals from various public safety organizations. These individuals may include members from the medical services, firefighters, law enforcement officers and resource coordinators who are required to communicate and share information as authorized during emergencies. Each of them will be equipped with an E-Device, which is a handheld computer that provides a mobile emergency response capability with which sensitive information can be securely processed, stored and distributed.

The E-Device is configured at the depot before being issued to the users. It will be installed with trusted software that ensures sensitive information is confined to partitions associated with sensitivity equivalence classes. There will be no flows between partitions, with the exception of inter-partition flows allowed by the security policy. The cryptographic keys that are used to verify user identity and encrypt messages will also be installed. As the E-Device is intended to serve as a dual-use terminal, day-to-day applications that are frequently used will also

be installed. The partitioning and confinement of applications and data establishes the security foundation with which the E-Device provides a secure computing environment.

A central authority known as the Tactical Operation Center (TOC) will be responsible for coordinating the emergency response. Prior to or during an emergency, information may be distributed to the E-Devices. The E-Device relies on a robust communication infrastructure in order to receive information from the TOC. If such an infrastructure is unavailable at the scene, an incident area network (IAN) to provide temporary coverage will be setup. An IAN is a network that is created for a specific incident and is typically provided by a wireless access point attached to the first responder's vehicle or a specific IAN node [2].

If it is deemed that the response team requires access to sensitive information, the TOC, with pre-approval from the relevant data provider(s), will declare an emergency state. This state will trigger temporary but controlled access to an emergency partition maintained on the E-Device where sensitive information is stored, thus allowing the user transient access to sensitive information, as seen in Figure 1. This information may include floor plans, infrastructure schematics, city transit system plans, or medical records of disaster victims for which tactical team members are normally not authorized.

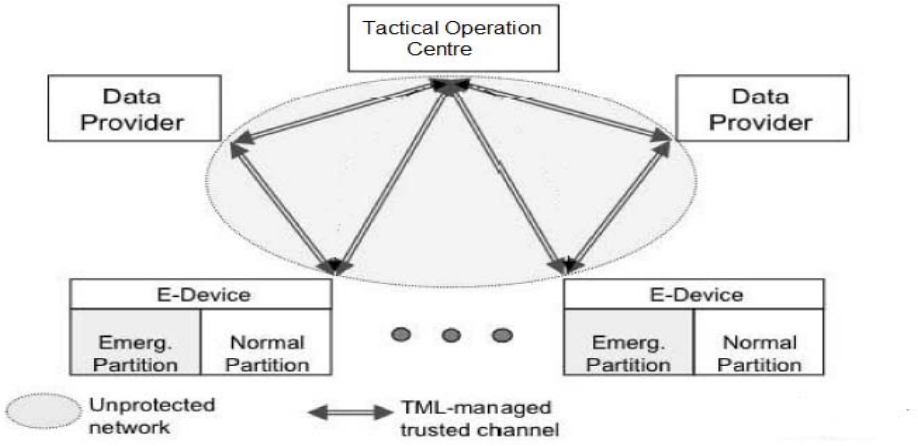



Figure 1. Emergency Network Architecture (After [3])

When the transient access is no longer required, the TOC will declare the end of the emergency. This will trigger a lockdown of the emergency partition. Before the lockdown occurs, users will be provided with a grace period to save their work within the emergency partition. The duration of the grace period is configurable and can be disabled depending on policy requirements. Once the emergency partition is locked down, the user is prevented from accessing the sensitive information held within that partition.

## B. TRUST MODELS

The secure use of E-Devices in the field is based upon a trust model [3] where the TOC is an entity that owns the E-Devices. The TOC installs its own trusted software and communication keys in these devices so that they can be trusted to enforce security policies. This establishes a trust relationship between the TOC and its E-Devices from which the TOC is able to securely distribute and receive sensitive information. In our emergency scenario, third party data providers have ownership of the privileged information that is used within the E-Devices. Hence, they too will need to establish cryptographic key-based trust relationships with the TOC and jointly agree on the security policies used by the E-Devices to access the privileged information. The TOC enforces these policies by means of its trusted software installed within the E-Devices. This results in a transitive trust relationship formed from the third party data providers to the E-Devices via the TOC.

In our emergency scenario, the E-Device may allow “extraordinary access” by which privileged information that the user is not normally authorized may be accessed. The protection of such emergency information is two-fold: the security policy allows extraordinary access only when the user is in the emergency partition and only during an emergency declared by the TOC.

## C. TRANSIENT TACTICAL ACCESS TO SENSITIVE INFORMATION SYSTEM

The E-Device was developed to support the transient tactical access to sensitive information (T-TASI) system that is a research project at the Naval Postgraduate School, Center for Information Systems Security Studies and Research (NPS CISR).

The T-TASI system was developed as a security architectural blueprint for resource constrained devices. It supports high assurance devices that can be relied upon to process, store, and distribute sensitive data and handle user interactions. The E-Device is a prototype to validate this concept. It is able to switch between three runtime modes – *normal*, *trusted*, and *emergency* – where one or more dedicated partitions are associated with each mode. Normal mode is the environment that will be most familiar to the user. In this mode, he is able to download, install, and run common workplace productivity applications accessed through the normal partition. Trusted mode allows the user to interact with the device in a high assurance manner and access the trusted path partition. This mode is accessible by means of a Secure Attention Key (SAK). Emergency mode is accessible only during an emergency that is declared by the TOC. This emergency state is triggered by means of an emergency signal (i.e. emergency state message) from the TOC that is sent over a secure channel. When the system is in this mode, the user is allowed to enter the emergency partition in order to access to privileged information from third party data providers during the duration of the emergency state. The T-TASI security architecture is displayed in Figure 2.

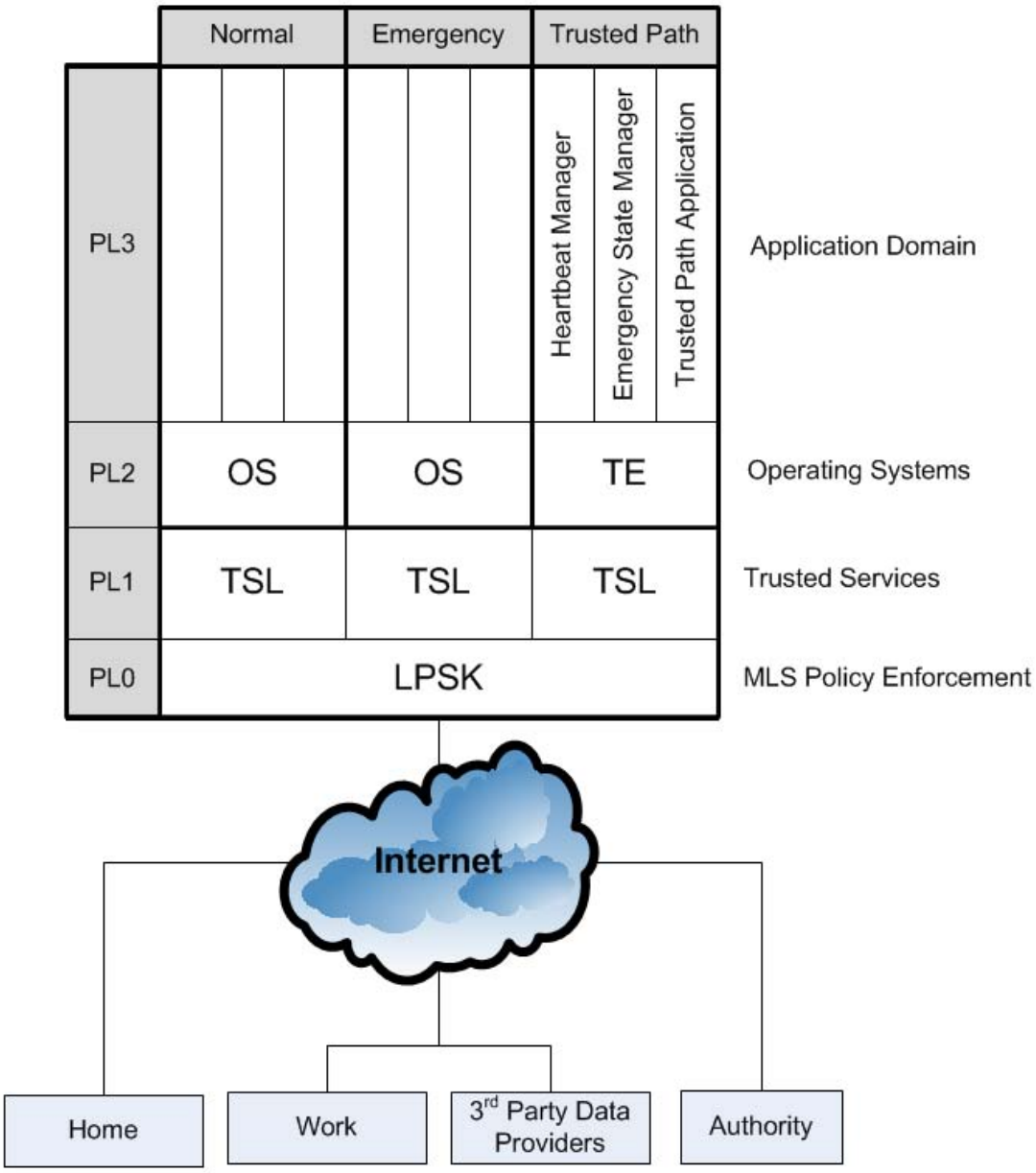



Figure 2. Security Architecture (After [4])

The following provides key highlights of the E-Device [4] and describes an overview of its major subsystems.

### 1. Least Privilege Separation Kernel

The Least Privilege Separation Kernel (LPSK) creates exported resources from the platform's physical resources, partitions the exported resources and controls interactions between the partitions. It allocates its exported resources on the E-device into discrete partitions. Users can only interact with one partition at a time.

### 2. Trusted Services Layer

The Trusted Services Layer (TSL) virtualizes certain LPSK resources for the use of partition applications (e.g. commercial OSs), and associates security labels with the kernel's exported resources. The TSL supports emergency management. In addition, the TSL manages the emergency partition as a protected staging area where emergency information is confined, and where users are provided with transient access to the information. The TSL ensures that information does not leave the Emergency Partition (except for writing to the third party data providers via a trusted network channel). The TSL ensures that only authorized users may access the Emergency (or any other) Partition. The net result is that the only way a user can ever access emergency information is through an Emergency Partition during an emergency.

### 3. Trusted Executive

The Trusted Executive (TE) provides process services for applications in the trusted partition. They include operating system services, application management services, session management, and document sealing.

### 4. Trusted Partition

The Trusted Partition hosts high integrity applications and data that provide trustworthy functionality. Trusted Partition services include the Trusted Path Application (TPA), which acts as a gatekeeper for user access to partitions, as directed by the TSL. This partition contains an Emergency Manager that is

responsible for initiating emergency state, and a heartbeat manager that periodically transmits keep-alive messages to the TOC for providing situational awareness.

#### 5. Emergency Partition

This partition hosts applications and data that have been provided by the third party data providers. The data is accessible only when an emergency state is declared. The application may run on top of commercial operating systems. Depending on established security policies, there can be multiple emergency partitions each belonging to a data provider, or one emergency partition shared by a set of data providers.

#### 6. Normal Partition

Each Normal Partition hosts user applications and data that are used during day-to-day activities. Typically, these applications run on commercial operating systems. The services in the Normal Partition are those most familiar to the users and offer functionally rich user interfaces.

### D. INTERNET PROTOCOL SECURITY

This section provides a brief overview of IPSec. The aim of this section is to provide a basic overview of what the protocol does and its implementation options. Knowledge of this topic is useful in allowing the reader to appreciate the rationale of our design recommendations for the use of IPSec (see Chapter IV).

IPSec is a framework of open standards for ensuring private communications over public networks [5]. It is a network layer security protocol that is typically used to create Virtual Private Networks (VPNs). A VPN is built on top of existing physical networks and is able to provide several types of data protection. These include confidentiality, integrity, authenticity, and replay protection. IPSec has the following components:

- Two security protocols, Authentication Header (AH) [6] and Encapsulating Security Payload (ESP) [7]
- Internet Key Exchange (IKE) Protocol
- IP Payload Compression Protocol (IPComp)

AH and ESP can be used in either transport or tunnel mode (see following figures).

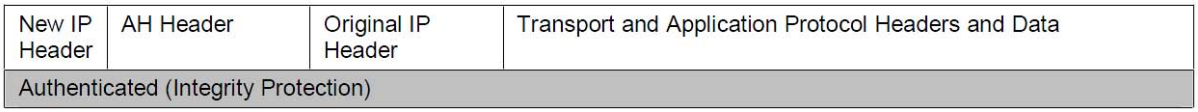



Figure 3. AH Tunnel Mode (From [5])

Figure 3 shows AH in tunnel mode. In tunnel mode, AH creates a new IP header and encapsulates the original IP packet. This setup provides integrity protection to both the outermost IP header (excluding mutable fields such as Time-To-Live) and the original IP packet.

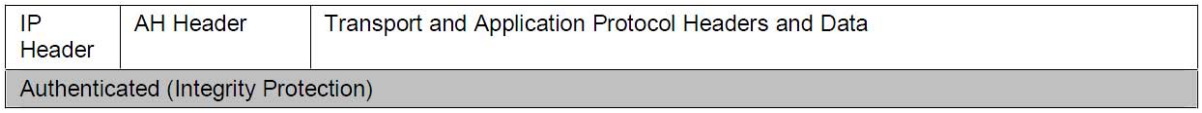



Figure 4. AH Transport Mode (From [5])

Figure 4 shows AH in transport mode. Under this mode, AH does not add a new IP header; instead, it relies on the original IP header. This setup provides integrity protection to the entire IP packet (excluding mutable fields in the header).

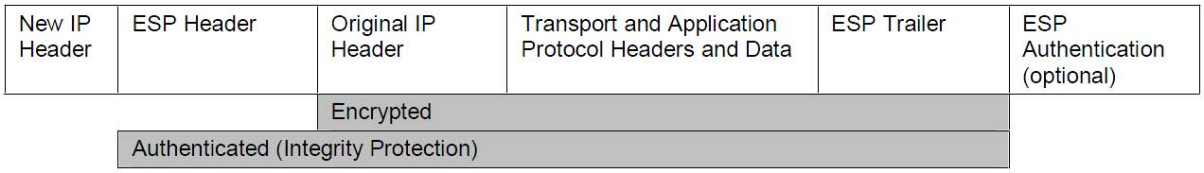



Figure 5. ESP Tunnel Mode (From [5])

Figure 5 shows ESP in tunnel mode. ESP in tunnel mode creates a new IP header and encapsulates the original IP packet. This setup provides both encryption and integrity protection for the original IP message. However, there is no integrity protection for the outermost IP header.

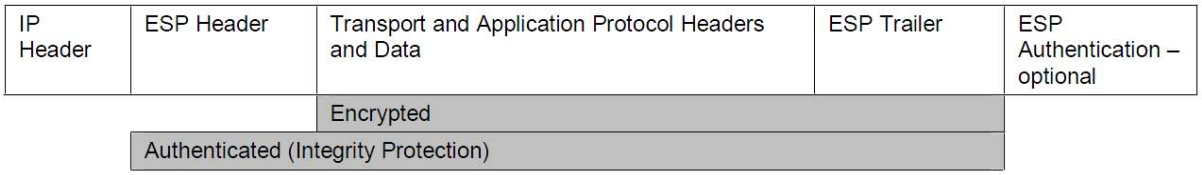



Figure 6. ESP Transport Mode (From [5])

Figure 6 shows ESP in transport mode. Under this mode, ESP does not create a new IP header; instead, it relies on the original IP header. This setup provides encryption only to the original IP payload and that there is no integrity protection for the IP header.

The other components of IPSec are Internet Key Exchange (IKE) and IP Payload Compression Protocol (IPComp). IPSec uses IKE to negotiate connection parameters between two endpoints. It defines the encryption algorithm, integrity protection algorithm, authentication method, and Diffie-Hellman (DH) groups. The use of IPComp is optional in IPSec. It is used to specify whether IPSec should compress packet payloads before encrypting them.

## E. PUBLIC KEY INFRASTRUCTURE

A public key infrastructure (PKI) is a set of policies, processes, server platforms, software and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, and revoke public key certificates [8]. Such an infrastructure is essential in binding public keys to entities, and providing key management in a distributed environment such as the one used by our emergency networks. Functional elements of a PKI encompass a certification authority (CA), registration authority

(RA), certificate revocation list (CRL) and PKI users. Like the IPSec section, this section is provided as background to support design recommendations in later chapters.

### 1. Digital Certificate

Digital certificates provide a means of proving user identity in electronic transmission. It contains information that identifies the CA issuing it, the subscriber names, the subscriber's public key, and the validity period. In addition, it must also be digitally signed by the CA issuing it.

### 2. Certification Authority

A certificate authority (CA) is an authority that is trusted to issue and manage public key certificates and the certificate revocation list (see below). It is the critical building block of a PKI and is a collection of hardware, software and people who operates it. It has four main responsibilities:

- Creates, signs, and issues certificates
- Maintain certificate status information and issues CRL
- Publishes current certificates and CRL
- Maintains archives of status information about expired certificates that are issued

### 3. Registration Authority

The registration authority (RA) is responsible for identification and authentication of certificate subjects, but it does not sign or issue certificates [8]. Its responsibility is to verify certificate contents for the CA.

### 4. Certificate Revocation List

A certificate revocation list (CRL) is a list maintained by the CA listing all the certificates that were issued but revoked prior to their stated expiration date

[8]. Best practices dictate that the CRL should be consulted whenever one relies on a digital certificate. Failing to do so may mean that a revoked certificate may be incorrectly accepted as valid. Of course, such an approach requires one to have an updated copy of the CRL.

## F. SUMMARY

This chapter has provided the background information for understanding the problem domain. In the next chapter, we proceed with the requirements-gathering based on the CONOPS that was described in this chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

# III. REQUIREMENTS ANALYSIS

This section discusses the results of our requirements-gathering process for signaling emergency status to the E-Devices. Military organizations, such as the Marine Corps, have many emergency situations similar in urgency to those faced by civilian disaster response. References to emergencies in this document are intended to apply to both civilian and military applications. The findings from this phase will influence the network protocol selection, the Emergency State Message (ESM) messaging format, as well as the design of the ESM module. Note that the communications requirements in this chapter provide high-level recommendations regarding the choice of protocol, whereas the Heartbeat and ESM Manager sections provide concrete design requirements to form the basis of the high-level design to be presented in Chapter IV.

## A. CONCEPTUAL REQUIREMENTS

The declaration of emergency state through the transmission of an ESM is a central theme in the CONOPS. The TOC maintains a binary global emergency state and notifies the E-Devices of any state change by sending an ESM. Upon receiving this message, the emergency partition within the E-Device will be unlocked thus providing the authorized user with temporary access to its privileged data. In order to provide accountability to the data providers, the TOC is required to track the extent to which privileged data is shared. To achieve this, the E-Device will need to acknowledge receipt of the ESM by replying to the TOC. Hence, we can derive the following two requirements:

Conceptual Requirement 1

The TOC, and no one else, shall be able to send an authentic ESM to E-Devices

Conceptual Requirement 2

Upon successful validation of a received ESM, the E-Device shall send an acknowledgment of its receipt to the TOC

Conceptual Requirement 3

The E-Device shall allow access to the emergency partition by authorized personnel when the emergency has started

Conceptual Requirement 4

The E-Device shall prohibit access to the emergency partition when the emergency has ended.

All of the above shall be performed in a secure manner using trusted mechanisms. The rest of this chapter will define what is required to create such a mechanism.

## B. HIGH-LEVEL SECURITY OBJECTIVES

This section expands on the conceptual requirements by defining them in terms of confidentiality, integrity, availability, authentication, and non-repudiation aspects. According to [9], there shall be technical controls that enforce these five measures on wireless information systems.

Confidentiality The ESM shall be kept secret so that only the intended recipients are able to understand its content. There is little reason why an emergency state declaration should be made known to the public. Transmitting such information in the clear may allow enemies and other malicious parties to intentionally interfere with emergency response.

Integrity The ESM receiver shall be able to detect modifications to the message that have occurred since its transmission by the TOC. Such modifications can fall into two groups. The first group is intentional modifications that are performed by malicious attackers. These include man-in-the-middle attacks where the ESM might be modified to indicate emergency start instead of

emergency end, thereby allowing unauthorized access to privileged data. The second group falls under unintended modifications such as accidental changes to data as a result of communication channel errors.

Authenticity The receiver shall be able to validate the sender of the received ESM. This is required as the policy dictates that only the TOC can declare emergency states. If there is no authenticity, malicious entities can impersonate the TOC and allow access to privileged data within the emergency partition.

Replay Protection The receiver shall be able to prevent replay attacks so that an earlier ESM, when retransmitted, does not initiate another emergency state declaration.

Non-Repudiation The TOC shall be provided with a proof of ESM acceptance that allows the TOC to identify those able to access privileged information during a particular emergency. This provides accountability to third party data providers and the assurance that the emergency responders have the necessary data access for efficient crisis handling.

The above items reflect the security requirements essential for signaling emergency status from the TOC to the E-Devices. The next step is to determine the technologies that are used to fulfill them.

## C. COMPARISON OF COMMUNICATION ALTERNATIVES

This section examines key communication protocols and analyzes them based on the network protocol layer at which they function. A TCP/IP model [10] is as depicted in the following diagram.

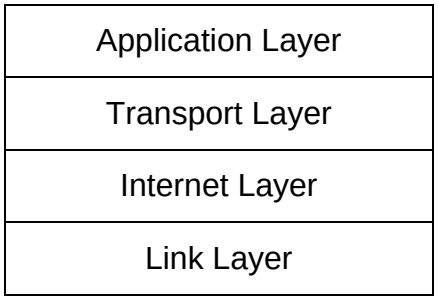



Figure 7. TCP/IP Model

The motivation for a layer-based analysis is that many of the protocols' characteristics are based on the layer from which they originate. The analysis focuses on how protocols associated with each layer can meet the CONOPS requirements with respect to its confidentiality, integrity, authenticity, replay protection, and non-repudiation. We will use the term CIARN to collectively represent these policy components. In addition, we will also judge the layers based on the ease with which applications can be implemented. Note that this comparison only looks at services offered by the reviewed layer and not in conjunction with other layers. The analysis shows the basic capabilities of each layer, which is sufficient for the design recommendations. A combinatorial analysis of the capabilities of multiple layers is beyond the scope of this thesis, although a gross combination is shown in Table 1.

### 1. Link Layer

Communication protocols operating in the link layer focus on point-to-point security. The controls at this layer are used to protect the information in the IP header and its payload that are transmitted over a specific physical link.

Link layer controls provide good confidentiality over point to point links and are suitable for communication that takes place between nodes that have a direct physical link to each other. Our CONOPS requires the TOC to communicate with E-Devices over public networks that may be operated by many different Internet

Service Providers (ISP). To deliver a message to intended recipients requires the data to go through insecure intermediate nodes, where the IP header (and its payload) would need to be decrypted to know the destination IP address that is residing in the encrypted IP header. Hence, the entire packet will be decrypted in order to determine the next hop and subsequently re-encrypted for transmission. There is also a key distribution problem in equipping intermediate nodes with the necessary keys for decrypting the ESM.

Since the packets would have to be decrypted at each intermediate node, this protection choice increases the possibility of exposure of sensitive information to malicious attacks. The attacker may control one of the intermediate nodes, thus allowing him to modify the packets before encrypting them for their next transmission. This modification cannot be detected with a link layer solution. Since this solution lacks integrity protection, it is also impossible to meet the other CIARN requirements.

### 2. Internet Layer

Internet layer controls are applied to all connections from the host and is not application-specific. For example, all network communications between two hosts or networks can be protected at this layer without modifying any applications on the clients or the servers.

Unlike the link layer approach where the ESM needs to be decrypted to determine the next hop, the routing information in this method is readily available in the IP header. An internet layer protocol provides end to end security through the use of shared secret keys among the communicating parties. Replay attacks are prevented through the use of nonces. Authenticity and non-repudiation are derived by signing outgoing data with a private key. Message authentication codes that are computed based on the ESM content and a shared secret key can accompany the ESM to provide integrity protection.

Another advantage of using an internet layer solution is its ability to hide the IP address of the destination node. When E-Devices generate acknowledgments for an ESM, the destination IP field contains the IP address of the TOC. This provides attackers with another vector of attack and should be avoided. Instead, this packet can be tunneled and encrypted within another IP packet. Traffic analysis will only reveal that the packet is directed to a VPN gateway without compromising the TOC.

### 3. Transport Layer

Transport layer controls are used to protect the data in a single communication session between two hosts. It cannot protect internet layer information such as the IP information as it resides above that layer. In addition, applications typically need to be modified to use transport layer controls. A transport layer solution will also provide end-to-end security through the use of shared secret keys among the communicating parties. Similar to an internet layer solution, it will employ the use of nonces and message authentication codes to protect the ESM.

This solution requires a transport layer protocol such as TCP or UDP to be developed in the TML. Unlike the internet layer solution where the implementation is transparent to the applications, the ESM Manager will need to specifically work with the transport layer solution. This imposes unnecessary implementation requirements, increases the complexity of the Trusted Computing Base (TCB), and thus, makes formal verification difficult. In addition, the IP address of the TOC is readily observable to traffic analysis.

### 4. Application Layer

Application layer controls are used to protect the data across communication sessions between two applications. This approach provides a very high degree of flexibility and customization over the application’s security.

However, the need to establish separate controls for each application often incurs significant overhead. It is also difficult to design and implement secure application controls.

It is technically possible to apply application layer controls to the ESM Manager. However, unless it is using proven and tested security protocols, it could be extremely difficult to verify any new design for its cryptographic soundness. It is also challenging to ensure that they are implemented properly. Doing it incorrectly may create more vulnerabilities instead. The problem is exacerbated when every new application has to develop its own security controls. This leads to duplication of effort resulting in a larger TCB. Furthermore, if the solution entails the use of keys, there will be logistic and management overhead in distributing multiple sets of keys to every E-Device. This creates additional overhead and unnecessary implementation complexity in the TCB and should be avoided.

Lastly, application layer solution has the same disadvantage of not being able to protect TCP/IP information because this information is at a lower level. This subjects the TOC to unnecessary exposure.

## 5. Results

Table 1 briefly summaries the suitability of each communication solution based on the CONOPS requirements. It is assumed that the security controls are implemented and configured correctly. An X in the cell indicates that the requirements are met for a specific category.

| | Confidentiality | Integrity | Authenticity | Replay Protection | Non-Repudiation | Protection from Traffic Analysis | Ease of Implementation |
|---|---|---|---|---|---|---|---|
| Link Layer | | | | | | X | X |
| Internet Layer | X | X | X | X | X | X | X |
| Transport Layer | X | X | X | X | X | | |
| Application Layer | X | X | X | X | X | | |

Table 1. Comparison of Alternatives for Communication Protection

We select a solution by considering both CIARN requirements and overall impact to the system. Specifically, the solution must be able fulfill its requirements in a manner that does not add considerable complexity to the TML.

The findings showed that internet, transport, and application layers are generally able to achieve the CIARN requirements. However, the internet layer solution also provides protection against traffic analysis. This helps to minimize exposure to the TOC by hiding its IP address.

In terms of system complexity, in general, higher level controls provide greater flexibility at the cost of increased complexity. The complexity comes from additional dependencies on other components, and having many of them performing similar functionalities. Such an approach represents a major hurdle when performing either formal verification or informal analysis of the TCB. It is generally agreed that complexity is the enemy of security [11, 12, 13]. Hence, a simpler design that is able to fulfill the requirements is preferred.

In summary, we compared the merits of each design by considering the extent that it meets the CIARN requirements and the ease of its implementation. The internet layer solution was the obvious choice based on the below differentiators:

- Ability to protect TOC's IP address from traffic analysis
- Security implementation is transparent to the ESM module
- Security implementation can be reused by other applications
- Can be implemented without significant TCB footprint
- End to end security as opposed to point to point

The current standard for securing IP communications is Internet Protocol Security (IPSec). The rest of the report will discuss what is required for secure communication of ESM and will assume the use of IPSec within the T-TASI system framework.

## D. HIGH-LEVEL REQUIREMENTS FOR EMERGENCY STATE PROCESSING

This section highlights the high-level requirements pertaining to the ESM module on the E-Device. The requirements are described in terms of "shall" and "should." The "shall" requirements describe mandatory requirements that are essential in supporting the CONOPS. These requirements must be incorporated in the final design. Requirements described using "should" are not directly tied to the CONOPS but are nonetheless beneficial to the system. They comprise of best practices and useful attributes that enhance the system in non-tangible ways such as making the system more scalable or modular.

### 1. Emergency State Management

It is imperative that applications first check the validity of their data prior to processing. Incorrect data may be malicious or accidental. In either case, processing invalid data is likely to result in unintended system behavior and may potentially induce a breach in security. We have made it mandatory for the E-Device to validate the ESM before processing it. The requirements are shown in Table 2.

| Group | Requirements | Identifier |
|---|---|---|
| ESM | The E-Device shall verify the correctness of the ESM prior processing it. | 1.1 |
| | The E-Device shall verify that the ESM is from the TOC prior processing it. | 1.2 |
| | The E-Device shall acknowledge receipt of an ESM to the sender of the ESM upon successful validation of a received ESM | 1.3 |
| | The E-Device shall allow access by authorized users to the emergency partition when the emergency has started | 1.4 |
| | The E-Device shall prohibit access to the emergency partition when the emergency has ended | 1.5 |

Table 2. ESM Requirements

### 2. Communication

The communication subsystem provides a means to transfer data packets among E-Devices or to the TOC. It includes IPSec and operates at a lower layer than the ESM. As of this writing, the ESM and Heartbeat Managers are the only users of these services. However, it is very likely that additional applications will use the communication subsystem in the future and it is essential that the design allow that, because a redesign to support a broader set of applications would be costly. The requirements are shown in Table 3.

| Group | Requirements | Identifier |
|---|---|---|
| Communication | Interface should be modular and usable by higher layer applications. | 2.1 |
| | All communicating parties shall support a common set of IPSec protocols. | 2.2 |
| | All communicating parties shall support a common set of IPSec modes. | 2.3 |
| | All communicating parties shall support a common set of cipher suites. | 2.4 |
| | All communicating parties shall standardize on whether the use of compression is required. | 2.5 |

Table 3. IPSec Requirements

### 3. Emergency Applications

This section incorporates an additional heartbeat application to the E-Device. The purpose of this application helps the TOC to better perform crisis management by knowing the location of the E-Devices. These requirements are described in Table 4.

| Group | Requirements | Identifier |
|---|---|---|
| Application | TOC should be able to estimate the number of emergency responders that are near the emergency scene. | 3.1 |

Table 4. Emergency Application

### 4. Testing

All of the above requirements are assigned an identifier. The purpose of this identifier is two-fold: First, it serves as a traceability mechanism so that mapping between the requirements to the design can be ensured. Second, it is used to reference the test cases used to validate that the system meets all its requirements. Note that the focus of this thesis is on formulating high-level requirements and design and it will not address testing activities further.

THIS PAGE INTENTIONALLY LEFT BLANK

# IV. HIGH-LEVEL DESIGN

Our CONOPS requires the TOC to perform secure signaling of emergency status to the E-Devices. In the previous chapter, we broke down this broad set of requirement into its constituents. These constituents can be categorized under each of the three subsystems as shown in Table 5.

| Subsystem | Description | Requirement Identifier(s) | Design Section |
|---|---|---|---|
| Communication | Provides secure channel to and from the TOC for purpose of transmitting heartbeat messages and receiving emergency signals | 2.1 – 2.5 | IV.A, IV.B |
| Heartbeat Manager | Provides situational awareness to the TOC by publishing the location of the E-Device | 3.1 | IV.C |
| Emergency State Manager | Processing emergency signals from the TOC and activating emergency state on the E-Device | 1.1 – 1.3 | IV.D |

Table 5. Overview of New Subsystems

These subsystems have been identified as new additions to the T-TASI system framework and Figure 8 shows the envisaged system architecture that includes these new subsystems.

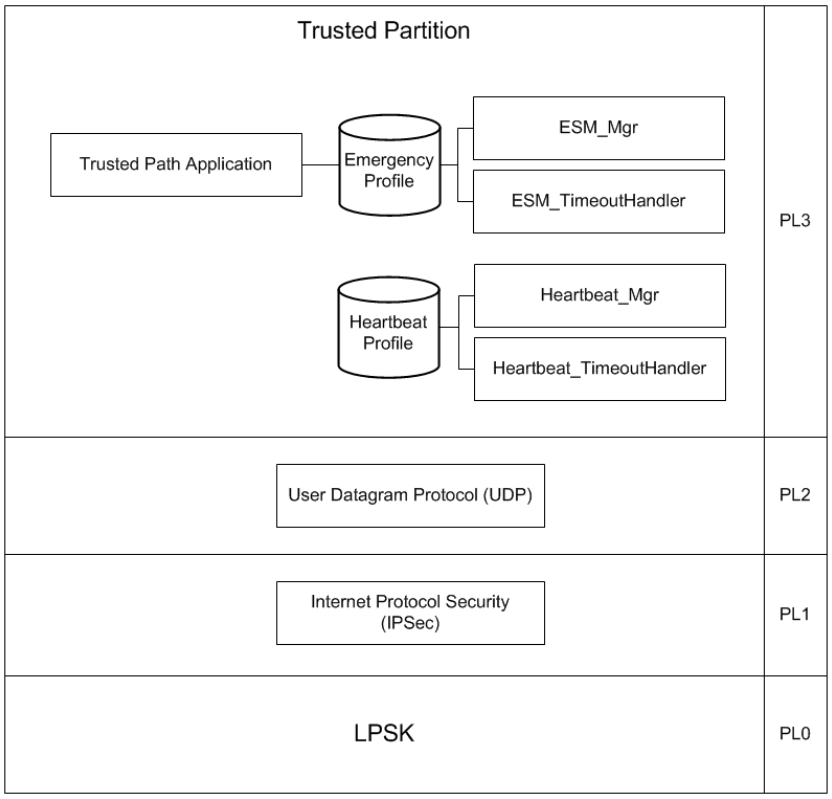



Figure 8. Proposed System Architecture

The E-Device is being built using the Least Privilege Separation Kernel [14], which utilizes the four hardware privilege levels (PLs) available in the Intel x86 processor [15, 16]. These PLs are protection domains that are arranged in a hierarchy ranging from most privileged (PL0) to least privileged (PL3). Components that play a key role in enforcing system security will be in a more privileged hardware domain. The concept of emergency signaling requires a vertical slice implementation that will span the various hardware PLs. The rest of this chapter discusses the high-level design for each of the new subsystems, starting from IPSec and working up through UDP, Heartbeat Manager, and ESM Manager. The IPSec and UDP sections provide high-level design recommendations, whereas the Heartbeat and ESM Manager sections provide enough detail to form the basis of a proof of concept design analysis in Chapter V.

## A. INTERNET PROTOCOL SECURITY

IPSec can be used to establish a trusted channel between two parties. However, there are multiple ways of accomplishing this task given its flexible nature. In this section, we highlight the deployment and implementation issues and propose a way to use IPSec within the T-TASI system.

### 1. Protocol and Mode

Combining both AH and ESP appears to maximize confidentiality and integrity protection. However, such an approach has some drawbacks. First, as AH provides integrity protection to the outermost IP header, any change in this header will fail the integrity checks. This results in AH being incompatible with network address translation (NAT) protocols. In NAT, the source IP address is often changed from a private to a public address. Hence, AH integrity protection at the destination host will not match. Although there are NAT Traversal (NAT-T) techniques to overcome this limitation [5, 17, 18], they cannot cope with protocols such as File Transfer Protocol (FTP) and Lightweight Directory Access Protocol (LDAP) [19]. The reason being these protocols embed the peer IP addresses within the application packet. In addition, NAT-T adds complexity to existing implementations by requiring NAT-T discovery among the edge devices and additional UDP encapsulation [20]. Second, AH does not provide confidentiality and needs to be deployed with either ESP or used in conjunction with application level encryption.

The use of ESP is essential as we are not provisioning for application level encryption. The use of AH is required if integrity protection is required for the outermost IP header. In our case, it is unnecessary, as the successful authentication of the original IP payload already proves that it came from someone with the corresponding private key. That by itself should provide adequate assurance. The outermost IP header is only used to address the packet to the recipient, and should not affect the interpretation of the packet. Hence, the use of ESP is recommended.

The use of tunneling over transport mode is recommended as tunneling will hide the destination IP address of the message. In the context of the ESM, this address is the private IP address of the TOC. This increases protection against traffic analysis and reduces exposure for the TOC host.

The use of AH and transport mode should be removed from our IPSec implementation as they will not be used. The removal of unused code results in a smaller memory footprint, which can be useful for resource constrained devices. Following the Reference Monitor Concept [21], the IP module is kept small so that it can be subjected to analysis and tests to assure its correctness.

### 2. Cipher Suite

The interface to the encryption suite contains security parameters that can be used to select encryption schemes to protect IPSec communication channels. They include the choice of encryption algorithm, integrity protection algorithm, authentication method, and DH groups. It is beyond the scope of the thesis to analyze each of the possible choices. Instead, the design shall rely on recommendations put forth by National Institute of Standards and Technology (NIST) [5], National Security Agency (NSA) [22], and the separation kernel protection profile (SKPP) [23].

At the heart of the E-Device is the LPSK which is being constructed to be conformant with the SKPP. The protection profile states explicitly that the target of evaluation shall use NSA approved cryptographic mechanisms. It further states that this includes NIST FIPS-validated cryptography with additional NSA-approved methods for key management and for cryptographic operations. Hence, the NSA Suite B Cryptography package [22] was selected for the cryptographic primitives. It can serve as an interoperable cryptographic base to protect information up to the SECRET level.

The use of cryptography places a demand for higher computing resource. The choice of cryptographic algorithms is often a tradeoff between the need for

security and computational limitations. In the case of the ESM, we can afford to use the strongest available cryptographic mechanism in Suite B, as the declaration of emergency states does not take place frequently. Processing the cryptographic primitives will not incur significant computational overhead in comparison with a mechanism that must use cryptography often. Based on the intended use of IPSec, the selected cipher suite is *Suite-B-GCM-256* as defined in [24] was chosen. The suite is comprised of:

- Encryption Algorithm: 256-Bit AES CBC
- Integrity Protection Algorithm: HMAC-SHA-384
- Authentication Method: ECDSA-384
- Diffie-Hellman Group: ECDH-384

Note that a key aspect of Suite B cryptography is its use of elliptic curve technology instead of classic public key technology. In order to facilitate adoption of Suite B by industry, NSA has licensed the rights to twenty six patents held by Certicom, Inc. covering a variety of elliptic curve technologies. Under the license, NSA has the right to grant sublicenses to vendors building certain types of products or components that can be used for protecting national security information [22]. The final choice of cryptographic suite will be made later in the project.

### 3. IP Payload Compression Protocol

The use of IP Payload Compression Protocol shall be disabled for two reasons. First, the size of ESM is small and applying compression on small messages does not provide significant size reduction. This also wastes computing resources and drains the batteries on the E-Device. Second, it helps to keep the TCB lean by not adding unnecessary modules.

## 4. Recommendations

There are three ways in which IPSec can be implemented within a host [25].

The first method involves integrating IPSec into the native IP stack. The native implementation allows all IPSec capabilities to be provided as easily as regular IP support. This is efficient but requires access to the IP source code.

The second method is referred to as a bump-in-the-stack (BITS) implementation (see Figure 9). This approach makes IPSec a separate architectural layer between the native IP layer and the link layer. IPSec will intercept the IP datagrams as they are passed down the protocol stack. It adds security protection to these datagrams before handling them to the link layer. Availability of source code for IP is not required, thus making this implementation appropriate for legacy systems.

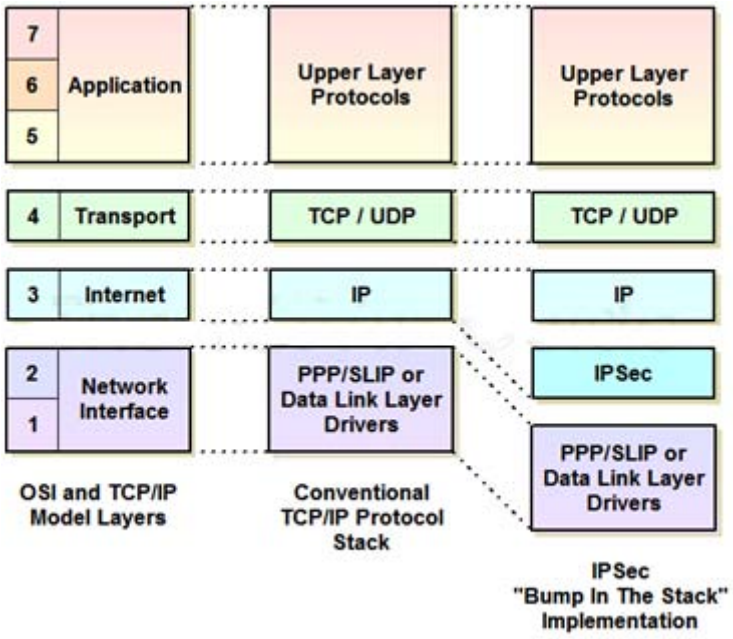



Figure 9. IPSec BITS Implementation (From [26])

The third method is referred to as a bump-in-the-wire (BITW) implementation. This implementation involves the use of dedicated and inline security devices that are usually IP addressable and provide IPSec services. In the below example (Figure 10), both routers are incapable of IPSec functions and rely on the IPSec devices. These devices intercept outgoing datagrams and add IPSec protection to them. These devices also remove IPSec information from incoming datagrams before forwarding them to the routers.

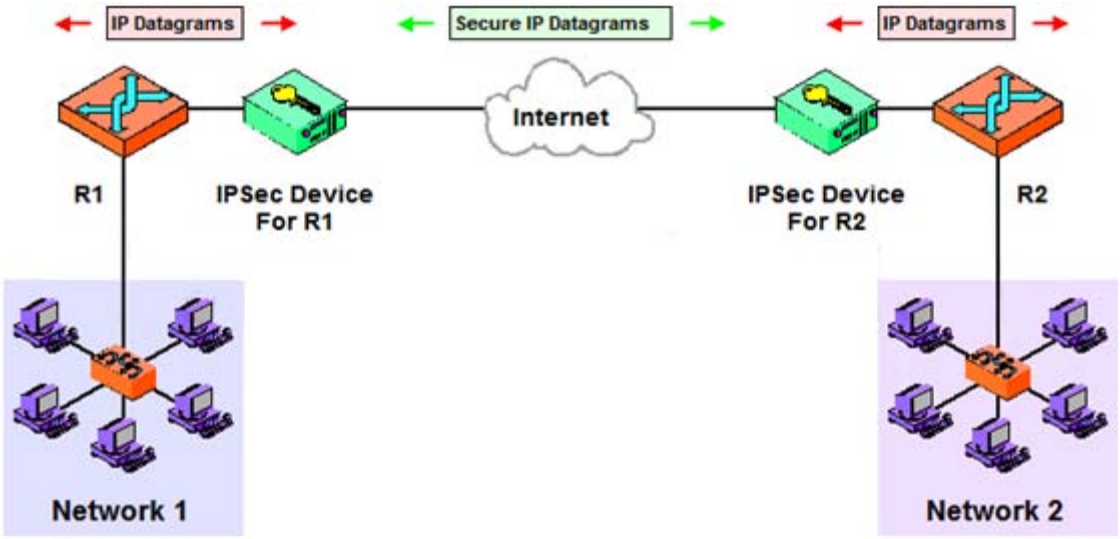



Figure 10. IPSec BITW Implementation (From [26])

As mentioned earlier, the native implementation of IPSec within the IP layer offers the best performance at reduced complexity and costs. This is the proposed approach as the IP layer within the E-Device was developed in-house.

### 5. Summary

This section reviews the requirements for IPSec and determines whether they have been met.

| Identifier | Requirements | Remarks |
|---|---|---|
| 2.2 | All communicating parties shall support a common set of IPSec protocol. | ESP shall be supported |
| 2.3 | All communicating parties shall support a common set of IPSec modes. | Tunnel mode shall be supported |
| 2.4 | All communicating parties shall support a common set of cipher suites. | Suite-B-GCM-256 shall be supported |
| 2.5 | All communicating parties shall standardize on whether the use of compression is required. | Disable IPComp |

Table 6. IPSec Requirements

As shown in Table 6, all the IPSec requirements have been satisfied.

## B. TRANSPORT LAYER MECHANISM

This section discusses the selection of a suitable transport layer mechanism for use by the ESM and the heartbeat manager in the trusted partition. This mechanism resides in PL2 and serves as an intermediary between IPSec and the applications that require its services. We use a requirements-driven approach that analyzes the messaging needs of each application and determines whether its needs can be met by the IPSec layer. Any unmet needs that are non application specific become a requirement for the transport layer mechanism. Following that, we analyze how those requirements can be met. The chosen transport layer implementation is the one that can fulfill these needs in a manner that is efficient and does not violate the layering concept.

### 1. Messaging Requirements

This section discusses application-specific messaging requirements by listing the essential messaging features that are required by the applications and determine whether they can be met by IPSec.

The messaging features are as followed:

Acknowledgement This refers to end-to-end acknowledgement that takes place when a message is delivered to the transport layer of the receiving host. If the sender does not receive the acknowledgement by a certain timeout, it resends the message subjected to a maximum number of retry and interval.

Error Detection This refers to the use of a checksum to detect unintended modifications to the packet that may be caused by environmental factors.

The messaging requirements for each application are tabulated in Table 7. The checkmarks under both ESM and Heartbeat Managers indicate a required feature. The checkmark under IPSec indicates that the feature is provided by IPSec.

| | ESM Manager | Heartbeat Manager | Provided by IPSec |
|---|---|---|---|
| Acknowledgement | √ | | |
| Error Detection | √ | √ | √ |

Table 7. Application Specific Messaging Requirements

**2. Analysis**

Figure 6 indicates that there is an acknowledgement requirement for the ESM Manager. This is because the CONOPS requires the E-Device to send an acknowledgement back to the TOC when the emergency partition has been unlocked. Consequently, the acknowledgement is best implemented within the ESM Manager rather than the transport layer because it requires application knowledge that is transparent to the transport layer. Hence, the ESM Manager will perform its own acknowledgement and does not require this service from the transport layer.

Moreover, it is evident that both the ESM and heartbeat managers require error detection services. This feature is already provided by IPSec in its integrity

protection mechanisms. As such, the applications do not require a transport layer mechanism and are able to send their payloads directly to IPSec. However, this is not an optimal solution.

At the IP layer of the receiving end, the protocol field in the IP header is referenced. This field will indicate which protocol should subsequently process the packet. In the case of messages targeted for the ESM and heartbeat managers, a unique protocol number needs to be specified for each application. IPSec will forward the packet to the application specified in the incoming packet. This seems to work in theory but has three drawbacks:

- Protocol numbers are assigned by IANA [27] and are reserved for protocol use instead of applications. Assigning them to ESM and heartbeat messages violates this convention.
- Another approach is to bypass IANA assigned protocol numbers. This choice assumes that the use of the protocol numbers is kept internal to the emergency network. However, there is a possibility that these numbers may be assigned by IANA and we may need to switch to another unused number.
- Whenever a new application that does not require a transport layer mechanism is added, the IP layer needs to be updated to accommodate its protocol number. This incurs additional testing and formal verification overhead.

### 3. TCP and UDP Comparison

We list the functions for implementation for both TCP and UDP. TCP [28] requires the following:

- 3-way Handshake
- Sliding Window Protocol
- Retransmission

- Acknowledgement
- Flow Control
- Checksum Computation

In contrast, UDP [29] requires only checksum computation and its use is not mandated. From the brief comparison, it is evident that UDP implementation is much simpler.

### 4. Recommendations

We propose using UDP as the transport layer mechanism because it meets the current messaging requirements and has slightly less implementation complexity. Receiving applications are identified based on a port number instead of protocol numbers. Adding new applications will not require change either the IP or UDP layers. The use of TCP will become necessary only when there are applications that require messaging functionalities that are not found in the IP layer. These applications will be highlighted in chapter VI (see Future Work).

### 5. Summary

This section reviews the requirements for messaging and determines whether they have been met.

| Identifier | Requirements | Remarks |
|---|---|---|
| 2.1 | Protocol should be usable by higher layer applications. | UDP is a well known transport layer protocol with established interfaces |

Table 8. Messaging Requirements

As shown in Table 8, the communication requirement pertaining to the transport layer has been satisfied.

## C. HEARTBEAT MANAGER

The Heartbeat Manager resides in the trusted partition and is responsible for providing the TOC with situational awareness regarding the E-Device. This allows the TOC to determine the number of emergency responders that are reachable across the network during a crisis. It also allows the TOC to determine the responders who are already in the vicinity of the incident. This is accomplished by having the E-Device transmit a heartbeat message to the TOC at periodic intervals. The sending interval is configured at the depot and is determined based on consideration of the number of E-Devices and the capacity of the networking infrastructure.

### 1. Message Format

Table 9 shows the format of the heartbeat message. The last row indicates the field size in number of bytes.

| Heartbeat Message | | | | | | |
|---|---|---|---|---|---|---|
| Type | Date/Time | Sender Device ID | Receiver Device ID | Heartbeat Counter | Longitude | Latitude |
| 1 | 4 | 8 | 8 | 2 | 8 | 8 |

Table 9. Heartbeat Message Format

Each field is as explained as followed:

Type This indicates a message type field. The E-Device will support multiple applications and the use of a type field will differentiate them from one another. A 1-byte field allows up to 256 different message types to be supported.

Date/Time This is the date and time when the heartbeat message was transmitted. The granularity should be in terms of number of seconds from an agreed upon epoch date reference. A 4-byte field allows up to 136 years from the epoch date.

Sender Device ID This 8-byte field uniquely identifies the sending E-Device

Receiver Device ID This 8-byte field uniquely identifies the hardware device of the receiving TOC

Heartbeat Counter This is a running numbering sequence that is tagged to each transmitted heartbeat message. The purpose is to allow the TOC to determine the ordering sequence of the messages.

Longitude & Latitude This indicates the location of the E-Device. If there is no location information available, use *0xFFFFFFFF* for this field. This could imply that there is no GPS coverage (i.e. the device is under shelter) or that the GPS is disabled due to privacy reasons. Both coordinates are expressed in terms of degrees with decimal notation [30] to keep the values within the 8-byte constraint.

The total size of a heartbeat message is 39 bytes.

### 2. Heartbeat Interval

There are some considerations in determining the size of the heartbeat interval. Setting a long interval may not provide an accurate locations status. Setting a short interval may potentially induce a denial-of-service on the TOC. This may be highly dependent on the operational context and there may be no optimal value for all situations. We have provided a rudimentary means to derive the interval based on a simple scenario. The purpose is not to provide an authoritative computation method, but to highlight the factors that are considered when deriving such a value.

One of the purposes of the heartbeat message is to provide an approximate location of the E-Device. We deem that an accuracy of +/- 300 meters to be suitable for our approximation. Studies have found pedestrian

walking speeds ranging from 1.25 m/s for older individuals to 1.5 m/s for younger individuals [31, 32]. Assuming the faster walking speed, this suggests a location update every 200 seconds.

Let us assume that there are 500 E-Devices reporting to a TOC. This suggests that the TOC will receive an average of 2.5 location updates per second. This seemed to be rather expensive, as the creation of IPSec channels and the associated key exchange mechanisms may unnecessarily overload the TOC. We feel that an average of 1 location update every 3 seconds is more appropriate considering that there are other data transmissions taking place that involve the TOC. To achieve this average for 500 E-Devices, we require that each device transmit no more than once every 1,500 seconds (i.e. 25 minutes).

The implication of an extended interval from 200 to 1500 seconds causes the location granularity to reduce to +/- 2250 meters. This simplified case study shows that necessary tradeoffs must be made between location accuracy and network infrastructure loading.

## D. EMERGENCY STATE MESSAGE MANAGER

The Emergency State Message Manager resides in the trusted partition and is responsible for two functions: (1) Processing the received emergency signal from the TOC, (2) Transitioning the E-Device between emergency states as a result of receiving an emergency signal or a timer interrupt.

### 1. Message Semantics

In our design, the emergency signals are represented as emergency state messages (ESM). Each message will contain the emergency parameters shown in Figure 11.

| Parameter | Description |
|---|---|
| MissionID | Unique Emergency Identifier |
| StartDate | When to start the emergency |
| EndDate | When to stop the emergency |
| GracePeriod | Time provided for users to save/finish their work in the emergency partition. |

Figure 11. List of Emergency Parameters

While the date fields are self-explanatory, the *MissionID* and *GracePeriod* fields require further elaboration. Having a *MissionID* identifier allows the TOC to plan for multiple non-overlapping emergency declarations scheduled in the future. This also allows the E-Devices to differentiate between different emergencies.

The *GracePeriod* field has dual functionality. Under normal situations when the emergency ends at a predefined end date, the user is notified sometime before it actually ends. This notification is based on the *GracePeriod* field. Under exceptional situations, the end date may be brought forward. In this case the user may no longer have sufficient time for a full grace period. In these cases, the emergency end date will be extended to permit a full grace period instead. For example, if an emergency is scheduled to end at 1500 hrs with grace period set to 300 seconds, then under normal situations, the user will be prompted to save his work at 1455 hrs. However, if an ESM was received at 1440 hrs indicating that the end date was brought forward to 1430hrs. The user will be provided with a 300-sec grace period even if the end date has been exceeded. In our example, the new end date will be 1445 hrs. The TOC also has the option of disallowing the grace period. For such a scenario, the emergency will end immediately.

We define an emergency (as stored on the E-Device) by *Mission = (MissionID, StartDate, EndDate, GracePeriod)*. An emergency as defined by an incoming ESM will be given by *Input = (MissionID, StartDate, EndDate, GracePeriod)*. This convention will be used in subsequent sections.

## 2. State Transitions

We have identified four possible states that the system can be in at any one time, based on the relationship of the current time to a current or pending emergency. Changes to these states result from receiving and processing an ESM. The states are illustrated in Figure 12. The *GraceStart* label on the timeline indicates the commencement of the grace period and is derived by deducting *GracePeriod* from *EndDate* (i.e., *GraceStart = EndDate – GracePeriod*). The *Current* label indicates the current date and time.

As highlighted in the earlier section, there are exceptional situations when the end date is brought forward. This results in the grace period commencing immediately for a specified duration (i.e., .*GraceStart = Current*, *EndDate = Current + GracePeriod*). Hence, the condition where (*GraceStart = EndDate – GracePeriod*) still holds. This state is represented by State 3.

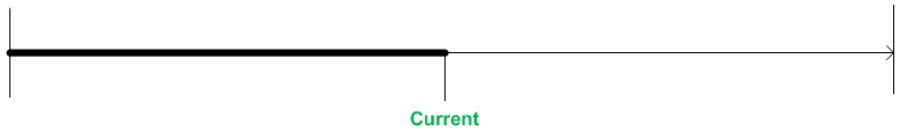
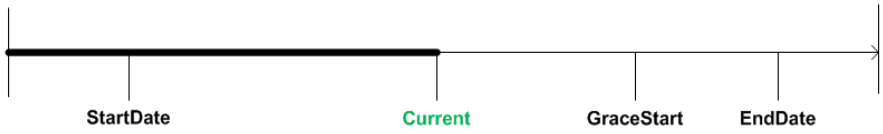
| State | Timeline |
|---|---|
| State 0 (OFF) |   |
| State 1 (OFF-Pending) |   |
| State 2 (ON-B4Grace) |   |
| State 3 (ON-InGrace) |   |

Figure 12. List of Allowable System States

In State 0 (OFF), the emergency state is off and there are no on-going or pending emergencies. The E-Device would be in this state when it was initially booted up.

In State 1 (OFF-Pending), the emergency state is off and a future emergency has been scheduled as a result of a previously received ESM. The emergency state will be activated once the emergency start date is reached.

In State 2 (ON-B4Grace), the emergency state is on and the user is in the midst of an emergency. However, the grace period has not commenced yet.

In State 3 (ON-InGrace), the emergency state is on and the user is in the midst of an emergency. The grace period has commenced and the emergency is expected to end soon.

To preserve the integrity of the state transitions, we define conditions that must be satisfied for all transitions.

*[**AXIOM 1**] For a given MissionID, StartDate < GraceStart < EndDate*

> For a given emergency, the *StartDate* must precede the *GraceStart*, and the *GraceStart* must precede the *EndDate*. This prevents scenarios whereby an emergency ends before it was actually started.

*[**AXIOM 2**] For a given MissionID, once the emergency has commenced, the StartDate shall never be changed.*

> For a given emergency, the TOC is allowed to make changes to the *StartDate* only if the emergency has not yet commenced. This prevents an E-device from having to perform multiple emergency starts and stops which may result in state de-synchronization.

*[**AXIOM 3**] Every ESM that is to be processed shall have an emergency counter value that is higher than the previously processed ESM.*

This axiom prevents the effects of a later ESM from being overwritten by an earlier ESM. This anomaly is possible when ESMs arrive out of sequence.

*[**AXIOM 4**] For a given MissionID, once the emergency is over, none of the emergency parameters shall be changed.*

For a given emergency that has already ended, the TOC is prohibited from making changes to any of its parameters. This prevents an emergency from having to start and stop repeatedly resulting in state de-synchronization.

Table 10 provides a summary of our discussion.

| State | | |
|---|---|---|
| Symbols | Prequisite Condition | Description |
| State 0 (OFF) | For all MissionID where (MissionID > 0), EndDate < Current | Emergency state ON, no ongoing emergency and no pending emergency |
| State 1 (OFF-Pending) | StartDate > Current | Emergency state OFF, pending emergency sequence (i.e. will happen sometime in future) |
| State 2 (ON-B4Grace) | (StartDate <= Current) & (EndDate > Current) & ((EndDate – Current) >= GracePeriod) | Emergency state ON, grace period has not commenced |
| State 3 (ON-InGrace) | (StartDate <= Current) & (EndDate > Current) & ((EndDate – Current) < GracePeriod) & (GracePeriod > 0) | Emergency state ON, grace period has commenced |

Table 10. Summary of System States

### 3. State Notifications

State transitions typically change emergency state from OFF to ON or vice versa. Occasionally, they will also involve commencement of the grace period. These events are of interest to the user and are communicated via the

*setMsgArea* interface in LPSK. This interface allows the ESM manager to display a message to the user via the PL1 display area. Displayed messages are specific to each state transition. Five message templates necessary during the state transitions are shown in Table 11.

| Template Name | Description |
|---|---|
| EMERGENCY_FUTURE | Display “[OFF] Pending emergency state at hhmm hrs.” |
| EMERGENCY_INITIATE | Display “[ON] Emergency state is currently ACTIVE. Ending at hhmm hrs.” |
| EMERGENCY_CANCEL | Display “[OFF] Pending emergency state has been cancelled.” |
| EMERGENCY_WARN | Display “[ON] Emergency state ending at hhm hrs. Please ensure your work in the emergency partition is saved.” |
| NORMAL | Display “[OFF] Normal mode. Not in emergency state.” |

Table 11. List of Message Display Templates

The choice of which template to use will be highlighted in the state transition chart in subsequent sections.

### 4. Shorthand Notations

Given four allowable system states, there are potentially sixteen possible state transitions among them. We utilize 14 different transitions. To simplify and improve the readability of these transitions, a shorthand notation is used to describe them. This section highlights their notation and how they are interpreted.

| Shorthand Notation | Description |
|---|---|
| *StartDate* | Original *StartDate* value as stored in the system |
| *in.StartDate* | *StartDate* value as stored in the incoming ESM message |
| *StartDate'* | Updated *StartDate* value in the system |

Table 12. Notations for describing field

Table 12 shows an example of the notation used to describe the same field but when it is assigned different values at different times. In our example, we have a *StartDate* field in the E-Device. The original value of this field is referred to as *StartDate*. This field was updated as a result of an incoming ESM. The *StartDate* value in the ESM is referred to as *in.StartDate* (i.e. *in* for input). After this field was updated, the new value as stored on the E-Device is referred to as *StartDate'*.

| Shorthand Notation | Expanded Notation | Description |
|---|---|---|
| in.E-past | (in.EndDate < Current) | The emergency has ended |
| in.E-started | (in.StartDate <= Current) & (in.EndDate > Current) | In the midst of the emergency |
| in.E-future | (in.StartDate > Current) | Pending emergency |
| in.E-cancel | (in.StartDate = in.EndDate) | Cancelling an emergency that has not commenced yet |
| in.is-room4grace | (in.EndDate – Current) >= in.GracePeriod | The current emergency has not commenced grace period yet |
| in.no-room4grace | (in.EndDate – Current) < in.GracePeriod | The current emergency is already in its grace period |
| in.allow_grace_period | (in.GracePeriod > 0) | Provision for grace period |
| in.not_allow_grace_period | (in.GracePeriod = 0) | Grace period is not allowed |

Table 13. Notations for describing conditions

When describing state transitions, we specify the necessary conditions for the transition to take place. The shorthand notation for these is shown in Table 13.

### 5. State Transition Chart

A state transition chart as shown in Table 14 describes the transitions. A transition occurs when either of two events occurs. The first event is the receipt of an ESM from the TOC. The ESM could signify the beginning or end of an emergency. It could also indicate an update in the emergency parameters. The ESM manager will determine the resultant state and perform the required actions.

The second event is the occurrence of a timer interrupt to the ESM manager. This timer is used as a means to inform the ESM manager that a certain amount of time has elapsed and a state change or notification is required. An example of its use could be to end the emergency in thirty minutes. The LPSK interface, *setTimer*, is invoked so that the kernel will inform the ESM manager once thirty minutes has elapsed.

The state transition chart shows all the possible state transitions and is made up of sixteen transition cells with the following information:

- Transition Identifier The identifier is to uniquely identify each transition cell
- Critical Input This lists the emergency parameters that were referenced within the condition (see critical condition) for this particular transition
- Critical Condition This lists the conditions that must exist (e.g., in the incoming ESM or timer interrupt) in order for the transition to occur
- Actions This lists the operation(s) that take place when the transition occurs

We will walk through a sample cell depicting the transition from state 0 to state 2, as listed in Figure 13, to understand how to interpret the notation.

| C. in.StartDate, in.EndDate, in.GracePeriod<br><br>**(in.E-started) & (in.is-room4grace)**<br><br>Mission' := Input<br>setMsgArea(EMERGENCY_INITIATE),<br>setFocus(TRUSTED) |
|---|

Figure 13. Sample Transition Cell

According to the cell description, the transition identifier is C and the emergency parameters that play a part in the transition are the *StartDate*, *EndDate*, and *GracePeriod* values within the incoming ESM. For the transition to take place, the *(in.E-started)* and *(in.is-room4grace)* conditions must be present in the message. The *(in.E-started)* notation expands to *(in.StartDate <= Current) & (in.EndDate > Current)* (see Table 13). This condition implies that the emergency has already started but has not yet ended. The *(in.is-room4grace)* notation expands to *(in.EndDate – Current) >= in.GracePeriod*. This condition implies that the grace period has not commenced yet. If both conditions are present, the transition from state 0 to state 2 occurs. For the transition to be successful, the three actions must be completed. The *Mission' = Input* notation indicates that the emergency parameters on the E-Device will be overwritten by those of the incoming ESM (see Table 12). The *setMsgArea(EMERGENCY_INITIATE)* notation denotes that this function call will be activated based on the *EMERGENCY_INITIATE* message template as illustrated in Table 11. Lastly, the *setFocus(TRUSTED)* notation specifies a *setFocus* interface call to the trusted partition.

While it may not be obvious on the state transition chart, it should be noted that receiving an invalid ESM does not trigger a state transition. The following are examples of invalid ESM:

- Receiving an ESM from another entity other than the TOC
- Receiving an ESM that has a *StartDate* that is later than the EndDate
- Receiving an ESM that has an emergency counter value that is lesser than the previously processed ESM
- Receiving an ESM that has incorrect values such as non-matching Device ID, or negative grace period.

| | | Starting State | | | |
|---|---|---|---|---|---|
| | | **State 0 (OFF)** | **State 1 (OFF-Pending)** | **State 2 (ON-B4Grace)** | **State 3 (ON-InGrace)** |
| | | For all MissionID where (MissionID > 0), endDate < Current | StartDate > Current | (StartDate <= Current) & (EndDate > Current) & ((EndDate – Current) >= GracePeriod) | (StartDate <= Current) & (EndDate > Current) & ((EndDate – Current) < GracePeriod) & (GracePeriod > 0) |
| **Target State** | **State 0 (OFF)** | A. in.EndDate<br>**(in.E-past)** | E. in.StartDate, in.EndDate<br>**(in.E-cancel) or (in.E-past)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_CANCEL) | I. in.EndDate, in.GracePeriod<br>**(in.E-past) & (in.not_allow_grace_period)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_END)<br>setFocus(TRUSTED) | M. in.EndDate, in.GracePeriod<br>**((in.E-past) & (in.not_allow_grace_period)) OR (in.timer_interrupt)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_END)<br>setFocus(TRUSTED) |
| | **State 1 (OFF-Pending)** | B. in.StartDate<br>**(in.E-future)**<br>Mission' := Input<br>setMsgArea(EMERGENCY_FUTURE) | F. in.StartDate<br>**(in.E-future)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_FUTURE) | J. Disallowed due to violation of Axiom 2 | N. Disallowed due to violation of Axiom 2 |
| | **State 2 (ON-B4Grace)** | C. in.StartDate, in.EndDate, in.GracePeriod<br>**(in.E-started) & (in.is-room4grace)**<br>Mission' := Input<br>setMsgArea(EMERGENCY_INITIATE),<br>setFocus(TRUSTED) | G. in.StartDate, in.EndDate, in.GracePeriod<br>**((in.E-started) & (in.is-room4grace)) OR (in.timer_interrupt)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_INITITATE)<br>setFocus(TRUSTED) | K. in.EndDate, in.GracePeriod<br>**(in.is-room4grace)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_INITITATE) | O. in.EndDate, in.GracePeriod<br>**(in.is-room4grace)**<br>Mission' = Input<br>setMsgArea(EMERGENCY_INITIATE) |

| | | | | | |
|---|---|---|---|---|---|
| | **State 3 (ON-InGrace)** | D. in.StartDate, in.EndDate, in.GracePeriod<br><br>**(in.E-started) & (in.no-room4grace) & (in.allow_grace_period)**<br><br>MissionID' = in.MissionID<br>GracePeriod' = in.GracePeriod<br>StartDate' = in.StartDate<br>EndDate' = Current + in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE),<br>setFocus(TRUSTED) | H. in.StartDate, in.EndDate, in.GracePeriod<br><br>**(in.E-started) & (in.no-room4grace) & (in.allow_grace_period)**<br><br>MissionID' = in.MissionID<br>GracePeriod' = in.GracePeriod<br>StartDate' = in.StartDate<br>EndDate' = Current + in.GracePeriod<br>setMsgArea(EMERGENCY_<br>INITITATE)<br>setFocus(TRUSTED) | L. in.EndDate, in.GracePeriod<br><br>**( (in.no-room4grace) & (in.allow_grace_period)) OR (in.timer_interrupt)**<br><br>MissionID' = in.MissionID<br>GracePeriod' = in.GracePeriod<br>StartDate' = in.StartDate<br>EndDate' = Current + in.GracePeriod<br>setMsgArea(EMERGENCY_WARN) | P. in.EndDate, in.GracePeriod<br><br>**(in.no-room4grace) & (in.allow_grace_period)**<br><br>MissionID' = in.MissionID<br>GracePeriod' = in.GracePeriod<br>StartDate' = in.StartDate<br>EndDate' = Current + in.GracePeriod<br>setMsgArea(EMERGENCY_WARN) |

Table 14. State Transition Chart

## 6. State Transition Diagram

Figure 14 shows another representation of the state transitions, in pictorial format.

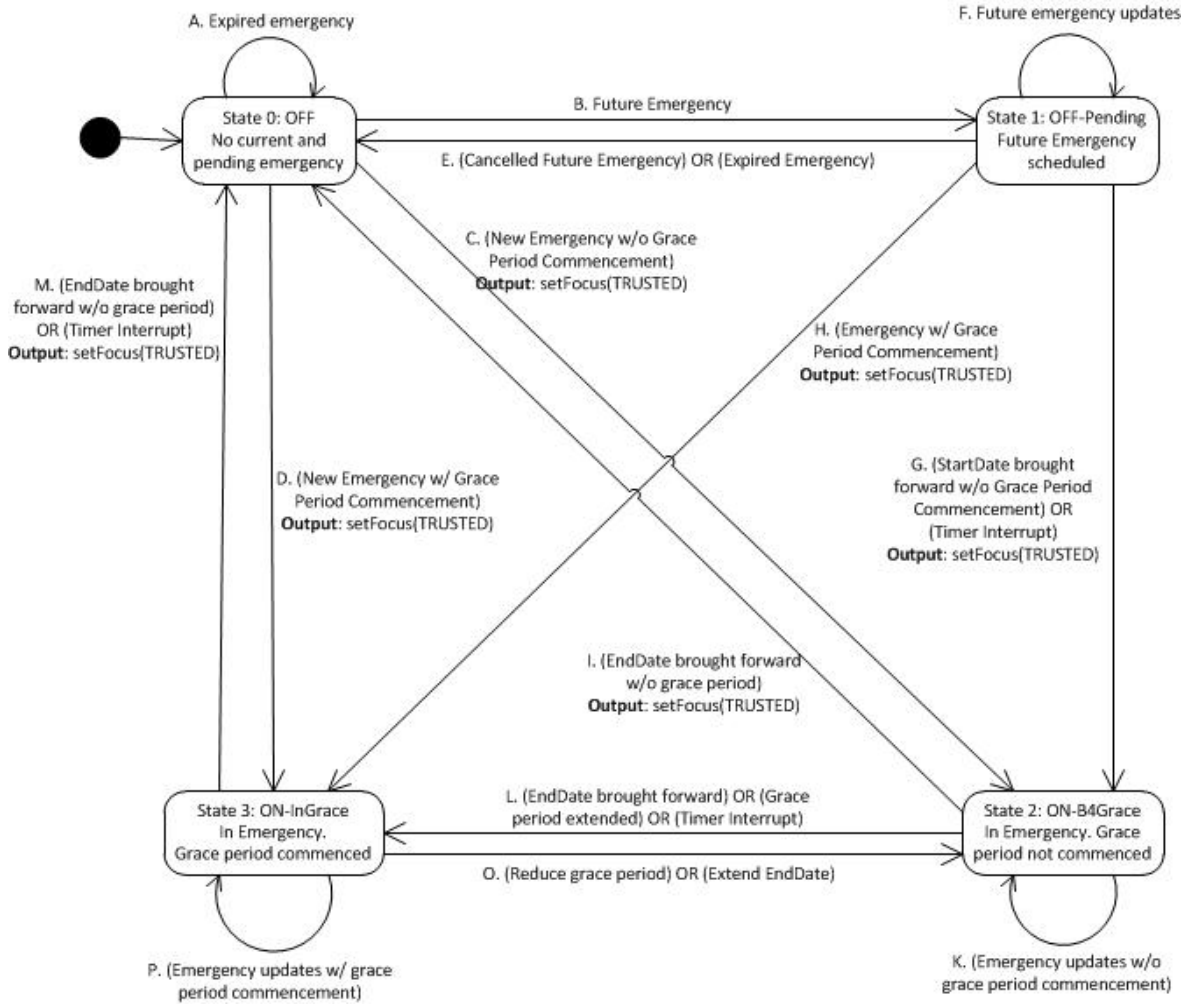



Figure 14. State Transition Diagram

## 7. State Transition Table

Table 15 provides the complete details of the state transitions in tabular format without the use of shorthand notation.

| Transition Name | A:  0 -> 0 |
|---|---|
| Critical Inputs | in.EndDate |
| Critical Relationships | (in.EndDate < Current) |
| Actions | - |

| Transition Name | B: 0 -> 1 |
|---|---|
| Critical Inputs | in.StartDate |
| Critical Relationships | (in.StartDate > Current) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_FUTURE) |

| Transition Name | C: 0 -> 2 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | (in.StartDate <= Current) & (in.EndDate > Current) &<br>((in.EndDate – Current) >= in.GracePeriod) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE),<br>setFocus(TRUSTED) |

| Transition Name | D: 0 -> 3 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | (in.StartDate <= Current) & (in.EndDate > Current) &<br>((in.EndDate – Current) < in.GracePeriod) &<br>(in.GracePeriod > 0) |
| Actions | StartDate' = in.startDate<br>EndDate' = Current + in.GracePeriod<br>GracePeriod' = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE)<br>setFocus(TRUSTED) |

| Transition Name | E: 1 -> 0 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate |
| Critical Relationships | (in.StartDate = in.EndDate) or (in.EndDate < Current) |
| Actions | StartDate' = in.startDate<br>EndDate' = in.endDate<br>GracePeriod' = in.GracePeriod<br>setMsgArea(EMERGENCY_CANCEL) |

| Transition Name | F: 1 -> 1 |
|---|---|
| Critical Inputs | in.StartDate |
| Critical Relationships | (in.StartDate > Current) |
| Actions | StartDate' = in.startDate<br>EndDate' = in.endDate<br>GracePeriod' = in.GracePeriod<br>setMsgArea(EMERGENCY_FUTURE) |

| Transition Name | G: 1 -> 2 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | ((in.StartDate <= Current) & (in.EndDate > Current) &<br>((in.EndDate – Current) >= in.GracePeriod)) OR<br>in.timer_interrupt |
| Actions | StartDate' = in.startDate<br>EndDate' = in.endDate<br>GracePeriod' = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE)<br>setFocus(TRUSTED) |

| Transition Name | H: 1 -> 3 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | (in.StartDate <= Current) & (in.EndDate > Current) &<br>((in.EndDate – Current) < in.GracePeriod) &<br>(in.GracePeriod > 0) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = Current + in.GracePeriod<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE)<br>setFocus(TRUSTED) |

| Transition Name | I: 2 -> 0 |
|---|---|
| Critical Inputs | in.EndDate, in.GracePeriod |
| Critical Relationships | (in.EndDate < Current) & (in.GracePeriod = 0) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_END)<br>setFocus(TRUSTED) |

| Transition Name | J: 2 -> 1 |
|---|---|
| Critical Inputs | - |
| Critical Relationships | Disallowed due to violation of Axiom 2: once the emergency has commenced, the StartDate shall never be changed |
| Actions | - |

| Transition Name | K: 2 -> 2 |
|---|---|
| Critical Inputs | in.EndDate, in.GracePeriod |
| Critical Relationships | (in.EndDate – Current) >= (in.GracePeriod) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE) |

| Transition Name | L: 2 -> 3 |
|---|---|
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | ((in.StartDate <= Current) & (in.EndDate > Current) & ((in.EndDate – Current) < in.GracePeriod) & (in.GracePeriod > 0)) OR in.timer_interrupt |
| Actions | StartDate’ = in.startDate<br>EndDate’ = Current + in.GracePeriod<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_WARN) |

| Transition Name | M: 3 -> 0 |
|---|---|
| Critical Inputs | in.EndDate, in.GracePeriod |
| Critical Relationships | ((in.EndDate < Current) & (in.GracePeriod = 0)) OR in.timer_interrupt |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_END)<br>setFocus(TRUSTED) |

| Transition Name | N: 3 -> 1 |
|---|---|
| Critical Inputs | - |
| Critical Relationships | Disallowed due to violation of Axiom 2: once the emergency has commenced, the StartDate shall never be changed |
| Actions | - |

| Transition Name | O: 3 -> 2 |
|---|---|
| Critical Inputs | in.EndDate, in.GracePeriod |
| Critical Relationships | (in.EndDate – Current) >= (in.GracePeriod) |
| Actions | StartDate’ = in.startDate<br>EndDate’ = in.endDate<br>GracePeriod’ = in.GracePeriod<br>setMsgArea(EMERGENCY_INITIATE) |

| | |
|---|---|
| Transition Name | P: 3 -> 3 |
| Critical Inputs | in.StartDate, in.EndDate, in.GracePeriod |
| Critical Relationships | (in.StartDate <= Current) & (in.EndDate > Current) & ((in.EndDate – Current) < in.GracePeriod) & (in.GracePeriod > 0) |
| Actions | StartDate' = in.startDate<br>EndDate' = in.endDate<br>GracePeriod' = in.GracePeriod<br>setMsgArea(EMERGENCY_WARN) |

Table 15. State Transition Tables

## 8. Message Format

Table 16 shows an implementation of the ESM. The last row indicates the field size in number of bytes.

| Emergency State Message | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Type | Date/Time | Sender Device ID | Receiver Device ID | Mission ID | Emergency Counter | Start Date | End Date | Grace Period | Partition Number |
| 1 | 4 | 8 | 8 | 4 | 2 | 4 | 4 | 2 | 1 |

Table 16. ESM Message Format

Each field is explained as followed:

Type This indicates a message type field. The E-Device will support multiple applications and the use of a type field will differentiate them from one another. A 1-byte field allows up to 256 different message types to be supported.

Date/Time This is the date and time when the ESM was transmitted by the TOC. The granularity should be in terms of number of seconds from an agreed upon epoch date reference. A 4-byte field allows up to 136 years from the epoch date.

Sender Device ID This 8-byte field uniquely identifies the hardware device of the TOC

Receiver Device ID This 8-byte field uniquely identifies the E-Device that is the intended recipient of the ESM

Mission ID This is a unique identifier for each emergency under a specific TOC. A 4-byte field allows up to 232 different emergencies before a rollover is required.

Emergency Counter This is a running numbering sequence that is tagged to each ESM transmitted by the TOC. The purpose is to allow the receivers to determine the most recent message.

Start Date The date and time when the emergency will start. It is based on the same granularity as the Date/Time field.

End Date The date and time when the emergency will end. It is based on the same granularity as the Date/Time field.

Grace Period The amount of time (in seconds) provided to the user to save or finish his work in the emergency partition before the emergency state terminates. A 2-byte field allows a maximum of up to 18 hours. Use 0 for this field if the grace period is disallowed.

Partition Number This field is reserved for future use when multiple emergency partitions could be installed on the E-Device. Each emergency will require different sets of partitions to be unlocked. A 1-byte field supports up to 8 emergency partitions.

The total size of an ESM is 38 bytes. Table 17 shows an implementation of the acknowledgement message that is sent to the TOC when the E-Device has successful processed the ESM.

| Emergency State Acknowledgement Message | | | | | | | |
|---|---|---|---|---|---|---|---|
| Type | Date/Time | Sender Device ID | Receiver Device ID | Mission ID | Acknowledging Emergency Counter | New End Date | Current Emergency State |
| 1 | 4 | 8 | 8 | 4 | 2 | 4 | 1 |

Table 17. ESM Acknowledgement Message Format

The total size of an ESM acknowledgement is 32 bytes. The fields are largely similar to the ESM message except for the following:

Acknowledging Emergency Counter This contains the emergency counter value for which this acknowledgement is intended.

New End Date Certain state transitions require the end date to be extended. This field contains the updated end date as computed by the E-Device. If no extension is required, it will contain the same *EndDate* value as stored in the originating ESM.

Current Emergency State This indicates the resultant emergency state of the E-Device as a result of the previously received ESM

## E. MAINTAINING TIMING CONSISTENCY

A challenge in distributed computing is the ability to be able to synchronize timing across all nodes. The timely declaration of emergency states relies on both TOC and E-Devices having a common timing reference.

A time keeping mechanism such as the Network Time Protocol (NTP) [33] should be used to ensure timing consistency. However, in the absence of such a mechanism at this phase of the project, we require that the correct date, time and time zone to be correctly configured at the depot. In addition, with the exception of the normal partitions, users are prohibited from adjusting the system time on the E-Device.

## F. DEPOT

The depot will be the entity responsible for the preparation and issuance of the E-Devices. It must be able to identify the users in order to provide E-Device customization of the following:

- Setting up the T-TASI system software
- One-time synchronization with time server
- E-Device Customization
    - Creating normal, trusted, and emergency partitions
    - Creating user account on the E-Device
    - Setup own Device ID
- ESM Manager Configuration
    - Setup Device IDs of authorized TOC
    - Setup default emergency partition number
- Heartbeat Manager Configuration
    - Setup Device ID of heartbeat recipient
    - Setup IP address of heartbeat recipient
    - Configure heartbeat transmission interval
    - Configure whether to enable location information within heartbeat message
- Installing PKI Materials
    - Public key certificates belonging to root CA, issuing CA, and other PKI users
    - Certificate Revocation List
    - Private key

The user will sign out the E-Device and the root CA will be notified. Root CA will subsequently update the repository to reflect the new certificate status. The proposed PKI process is as depicted in Figure 15.

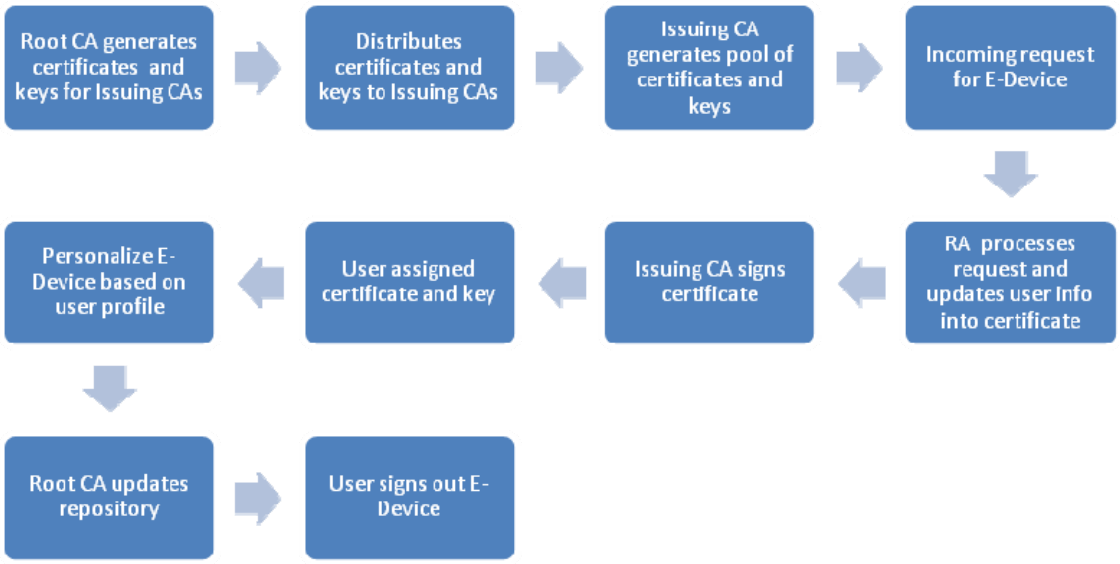



Figure 15. PKI Process

Venturing into the details of creating a PKI is beyond the scope of this thesis. However, such an infrastructure is necessary for the purpose of administering public-private keys, certificates, and the CRL.

THIS PAGE INTENTIONALLY LEFT BLANK

# V. DESIGN ANALYSIS

This chapter outlines the design analysis of the ESM and Heartbeat Managers. The analysis is a proof of concept design that is described using sequence diagrams depicting the control flow and call parameters. It also includes deriving the data structures required to support the operations. The aim of the analysis is to present technical details sufficient to inform the implementation of both ESM and Heartbeat Managers in the next project phase.

## A. HEARTBEAT MANAGER

This section introduces the two components of the heartbeat manager. They comprise of the *Heartbeat_Mgr* (V.A.1) and the *HeartbeatProfile* (V.A.2) modules. The *Heartbeat_Mgr* is responsible for transmitting heartbeat messages at regular intervals. It queries *HeartbeatProfile* for recipient information and also the location information of the E-Device. After every transmission, it increments the heartbeat counter in *HeartbeatProfile*.

### 1. Heartbeat_Mgr

Description This module is responsible for processing LPSK timer interrupts. It also handles the transmission of the heartbeat message as a result of these interrupts as shown in Figure 16.

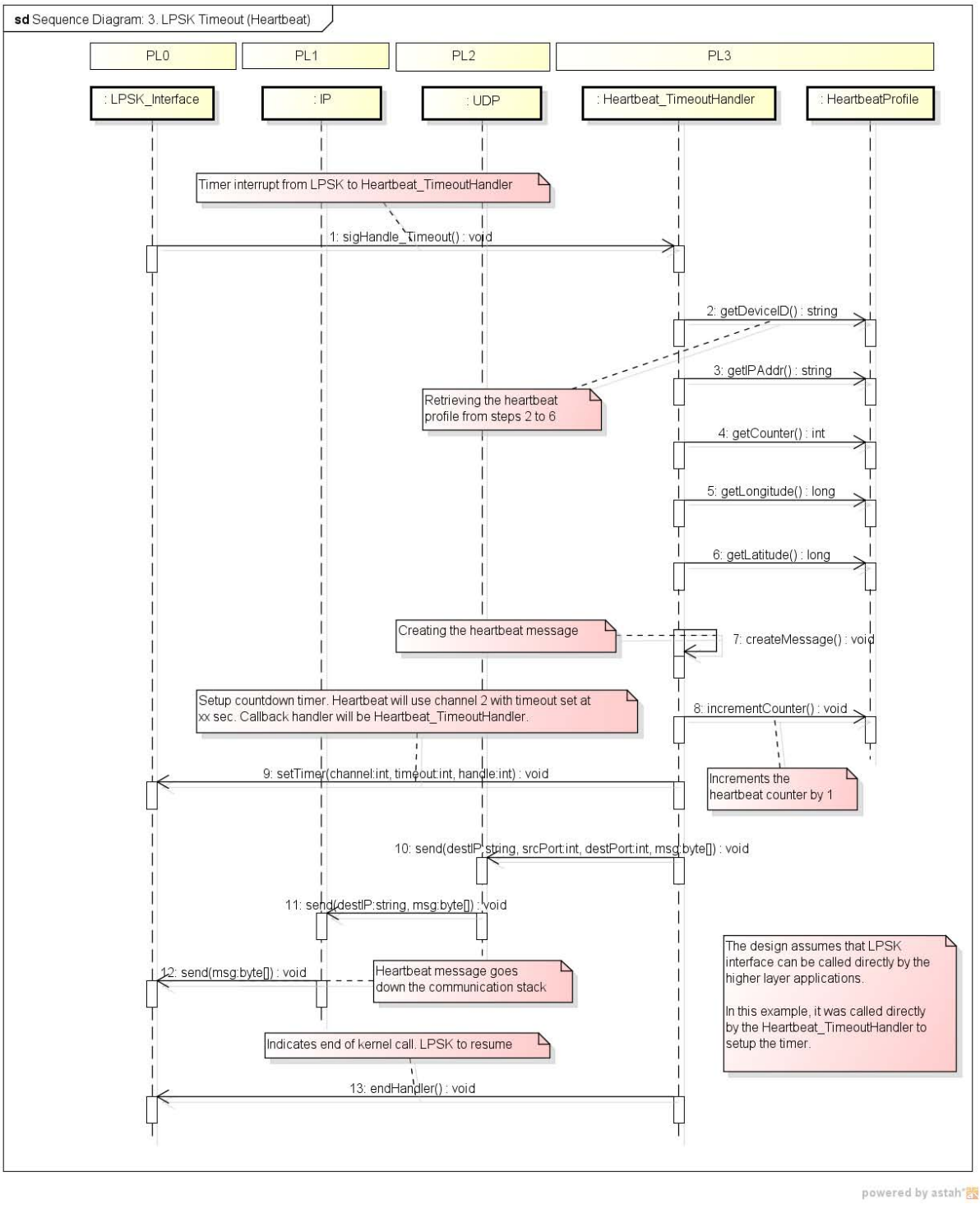



Figure 16. Heartbeat Timer Interrupt

Interfaces

***a. Public void sigHandle_Timeout ()***

This interface initializes heartbeat processing. It is called by the LPSK when a timer interrupt occurs. At the end of its processing, it notifies the LPSK via the *endHandler* interface.

**2. HeartbeatProfile**

Description This module is responsible for storing heartbeat parameters. Interfaces are provided for external components to query these parameters.

Attributes

*a. string dest_device_id = 0;*

This attribute stores the device ID of the heartbeat message recipient which typically is the TOC. This field is configured at the depot.

*b. string dest_ip_addr;*

This attribute stores the IP address of the heartbeat message recipient which typically is the TOC. This field is configured at the depot.

*c. int tx_interval = 0;*

This attribute stores the transmission interval (in seconds) of the heartbeat message. If it is set to 0, the heartbeat message will be disabled. This field is configured at the depot.

*d. int heartbeat_counter = 0;*

This attribute stores the current counter of the heartbeat message. Every message transmission will increment the counter by 1. This allows the

receiver to determine the message sequence and allow it to determine how many of the messages were missed. The rollover will happen when the counter reaches $2^{16}$ - 1.

*e.* *bool allowLocationBroadcast = TRUE;*

This attribute indicates whether location information should be embedded in the heartbeat message. If set to *FALSE*, both *getLongitude()* and *getLatitude()* interfaces will return *0xFFFFFFFF* regardless of availability of location information. This attribute is configured at the depot.

*f.* *long longitude = 0xFFFFFFFF;*

This attribute stores the longitude coordinate of the E-Device. The value is derived from the location provider of the E-Device (typically the GPS or WiFi). This field is initialized to *0xFFFFFFFF* if location is unavailable.

*g.* *long latitude = 0xFFFFFFFF;*

This attribute stores the latitude coordinate of the E-Device. The value is derived from the location provider of the E-Device (typically the GPS or WiFi). This field is initialized to *0xFFFFFFFF* if location is unavailable.

Interfaces

***a.*** ***Public int getDeviceID ()***

This interface allows querying of the *dest_device_id* field.

***b.*** ***Public string getIPAddr ()***

This interface allows querying of the *dest_ip_addr* field.

***c.*** ***Public int getInterval ()***

This interface allows querying of the *tx_interval* field.

***d.*** ***Public int getCounter ()***

This interface allows querying of the *heartbeat_counter* field.

***e. Public long getLongitude ()***

This interface allows querying of the *longitude* field. It will return *0xFFFFFFFF* if the *allowLocationBroadcast* attribute is *FALSE*.

***f. Public long getLatitude ()***

This interface allows querying of the *latitude* field. It will return *0xFFFFFFFF* if the *allowLocationBroadcast* attribute is *FALSE*.

***g. Public void incrementCounter ()***

This interface increments the *heartbeat_counter* by one. It shall be called after each heartbeat message transmission. If the next increment will exceed the limit of an integer data type (i.e. $2^{32}$ - 1), the counter will reset to 1.

## B. EMERGENCY STATE MESSAGE MANAGER

This section introduces the three components of the ESM Manager. They are: the *ESM_Mgr* (V.B.1), *ESM_TimeoutHandler* (V.B.2), and the *EmergencyProfile* (V.B.3) modules. The *ESM_Mgr* is responsible for processing all incoming ESM, whereas *ESM_TimeoutHandler* is responsible for handling timeouts. Both modules will query and update *EmergencyProfile* for the emergency parameters.

### 1. ESM_Mgr

Description This module is responsible for verifying and processing incoming ESM. It also handles the starting and stopping of emergency states within the E-Device as a result of the received ESM. The sequence of actions that take place when an ESM is received is shown in Figure 17.

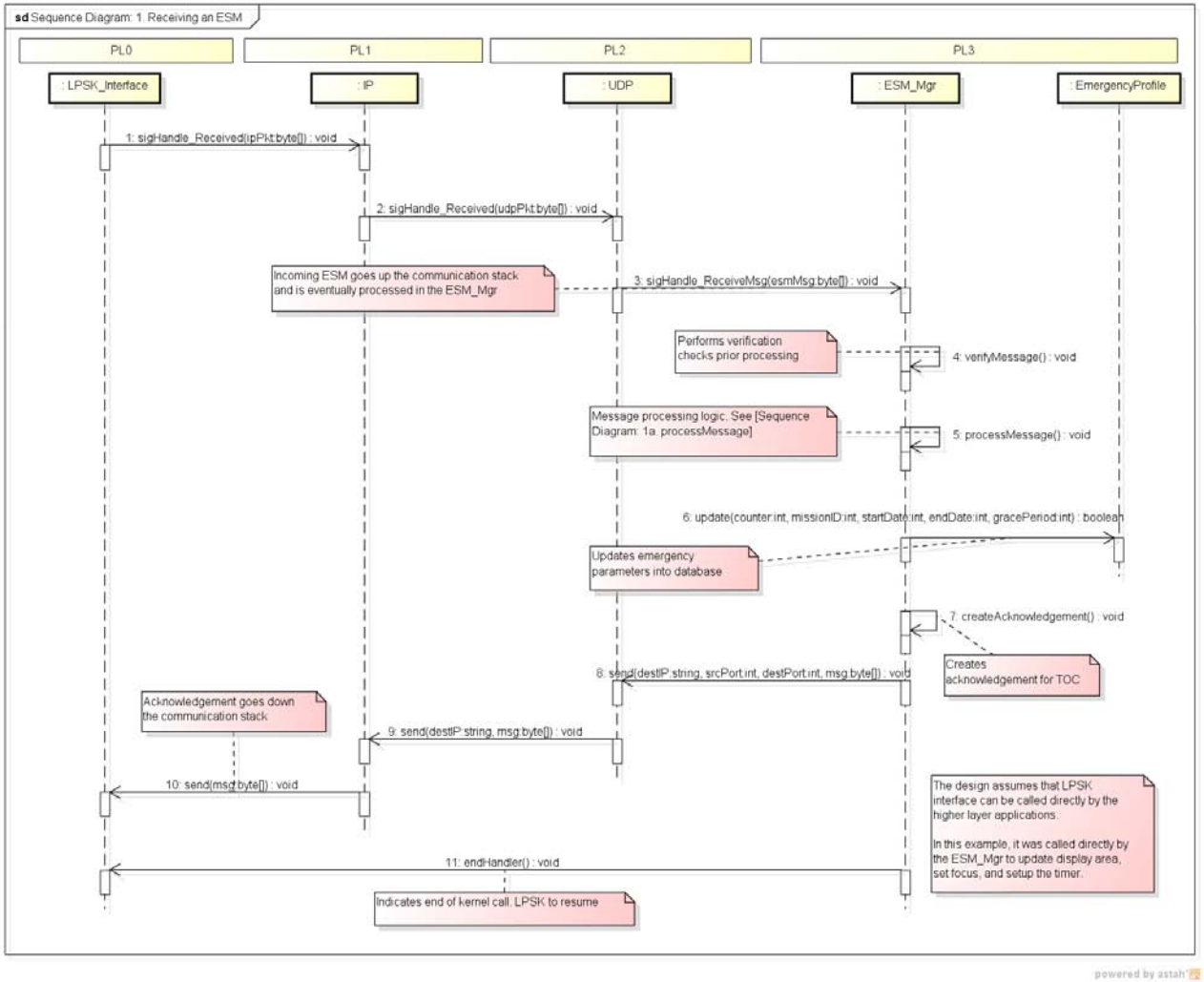



Figure 17. Receiving an ESM

Interfaces

***a. Public void sigHandle_ReceiveMsg (byte[] esmMsg)***

This signal handler is called by the UDP module when a new ESM is received. It initiates the processing of a message (see Figure 18), including message verification, emergency state transition, and generating an acknowledgement back to the sender. At the end of its processing, it notifies the LPSK via the *endHandler* interface.

***b. Private void verifyMessage ()***

This private interface is called by *sigHandle_ReceiveMsg* after a new ESM is received. It runs through a list of verification checks to ensure the validity of the message. The checks ensure that:

- Ensure the ESM is from an authorized TOC
- The emergency counter value is higher than the previously processed ESM (see Axiom 3)
- The *StartDate*, *EndDate*, and *GracePeriod* values are well-formed (see Axiom 1)
- The E-Device is the intended recipient of the ESM

***c. Private void createAcknolwdgement ()***

This private interface is called by *sigHandle_ReceiveMsg* upon the successful processing of the ESM. An acknowledgement message is generated and sent to the TOC.

***d. Private void processMessage ()***

This private interface is called by *sigHandle_ReceiveMsg* and handles the processing of the ESM including its emergency state transitions, querying emergency parameters, and handles the interfaces to other components. Figure 18 shows a continuation of the *processMessage* interface that was shown in Figure 17**Error! Reference source not found.** (step 5). It depicts the exact processing step performed on the ESM.

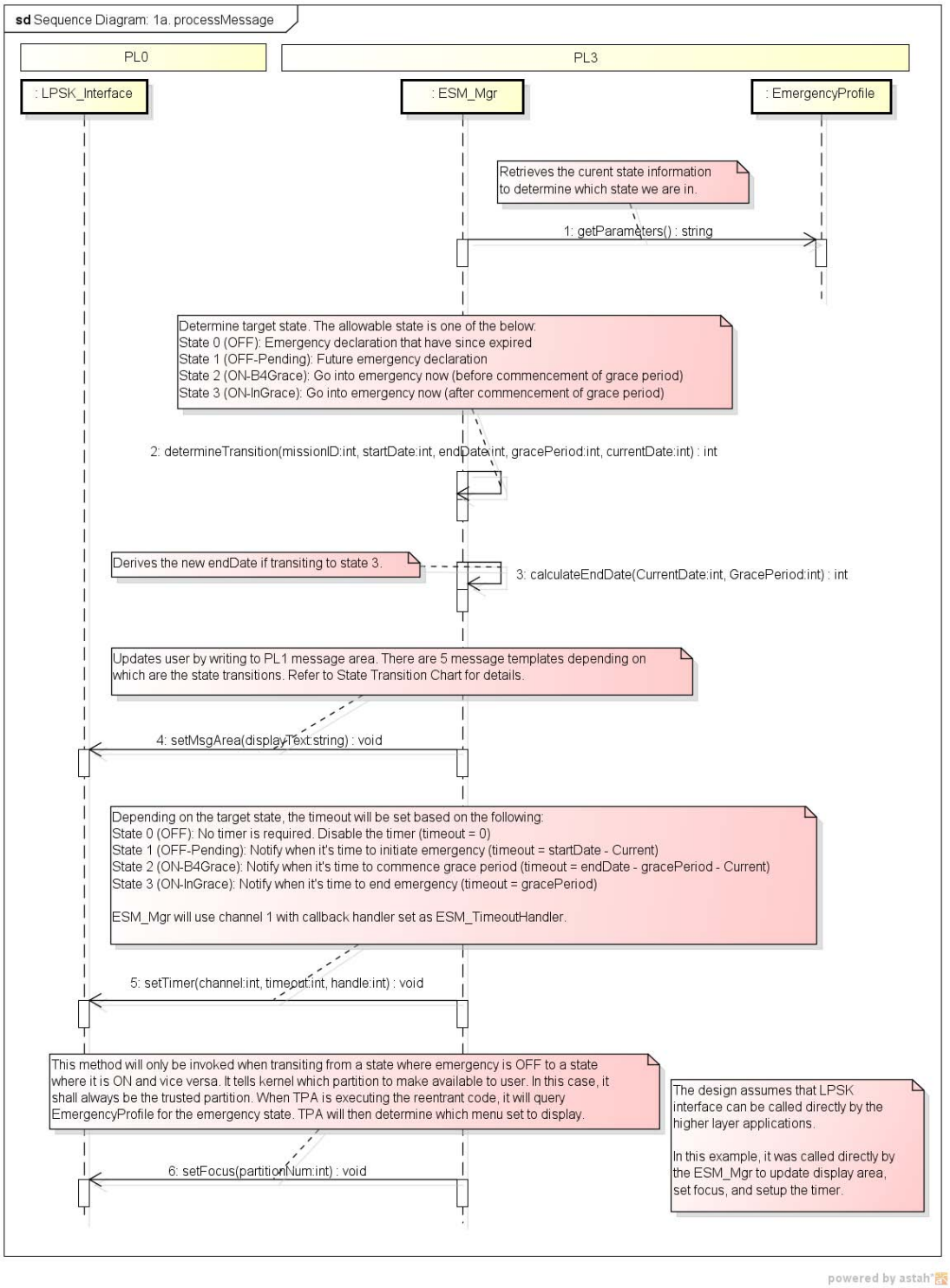



Figure 18. *processMessage* Interface

***e. Private int determineTransition (int missionID, int startDate, int endDate, int gracePeriod, int currentDate)***

This private interface is called by *processMessage* and is responsible for determining state transitions. It achieves this by reviewing incoming emergency parameters, analyzing their impact on existing parameters, and deriving the resultant state.

***f. Private int calculateEndDate (int currentDate, int gracePeriod)***

This private interface is called by *processMessage* and is responsible for re-computing the end date when transiting to state 3 (ON-InGrace).

**2. ESM_TimeoutHandler**

Description This module is responsible for processing LPSK timer interrupts. It also handles the starting and stopping of emergency states within the E-Device as a result of these interrupts. At the end of its processing, it notifies the LPSK via the *endHandler* interface. Figure 19 depicts the sequence of actions taking place when a LPSK timer interrupt (i.e., .timeout) was received.

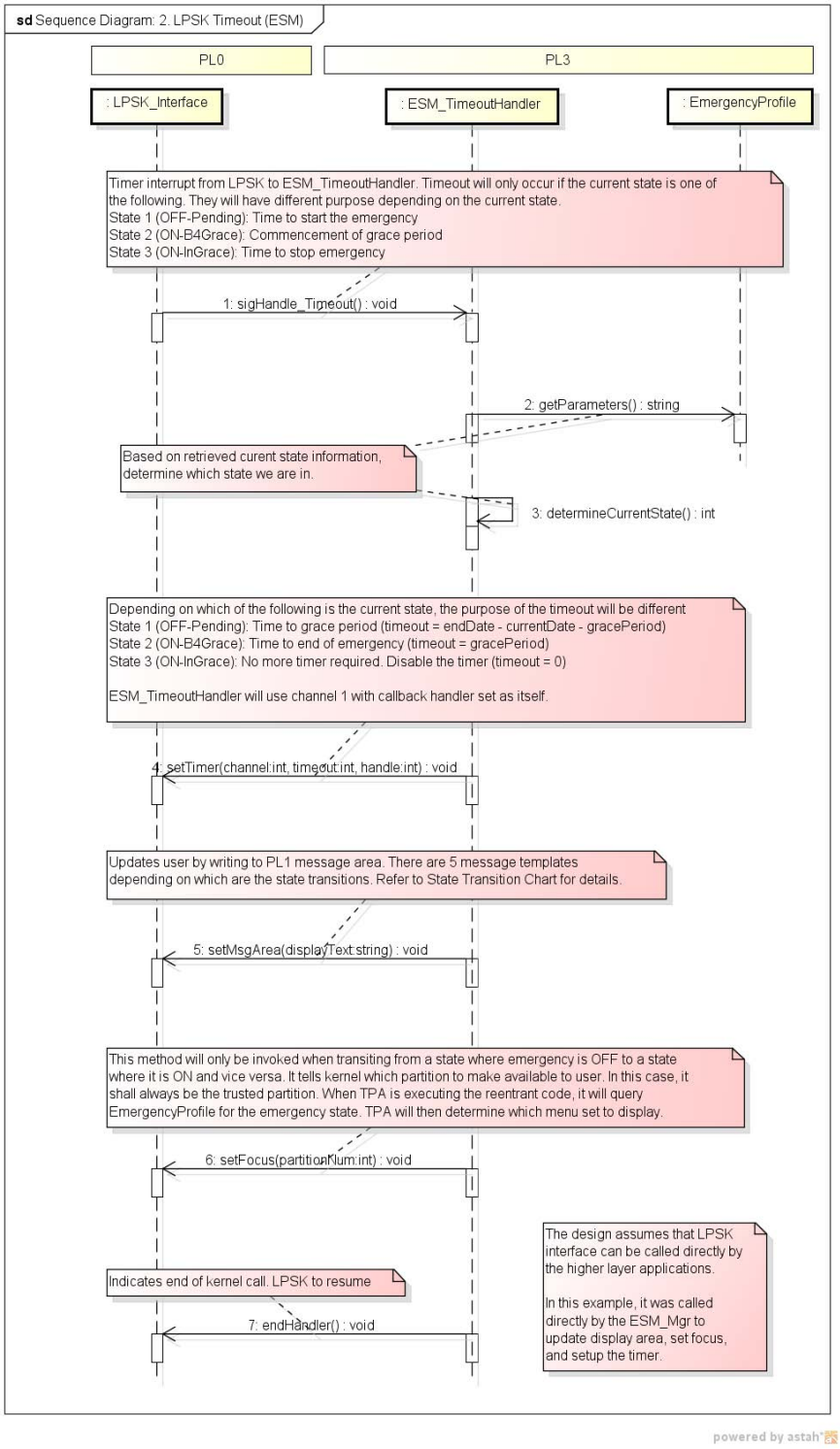



Figure 19. ESM Timer Interrupt

Interfaces

***a. Public void sigHandle_Timeout ()***

This interface initializes the timeout handling. It is called by the LPSK whenever a timer interrupt occurs. This timeout occurs as a result of a previously executed *setTimer* interface.

***b. Private int determineCurrentState ()***

This private interface is called by *sigHandle_Timeout* during the interrupt handling process. It is used to identify the current system state to determine the significance of the timeout. For example, if the current state is 1 (OFF-Pending), the timeout would indicate the start of an emergency. However, if the current state is 2 (On-B4Grace), the timeout would indicate the commencement of the grace period.

**3. EmergencyProfile**

Description This module is responsible for maintaining the emergency parameters. It handles all parameter read and write requests and provides interfaces for external entities to query them. It is called by *sigHandle_ReceiveMsg* via the *update* interface. It is also called by both *processMessage* and *sigHandle_Timeout* via the *getParameters* interface.

Attributes

*a. int emergency_counter = 0;*

This attribute stores the emergency counter of the previously processed ESM. Based on Axiom 3, the newer value shall always be higher than the current value. The rollover will happen when the counter reaches $2^{16}$ - 1.

b. *string authorizedSender = “TOC”;*

This attribute stores the device ID of all parties that have been authorized to send ESM to the current E-Device. It shall contain only TOC entries as they are the only ones allowed to declare emergency states.

c. *struct MISSION {*

*int missionID = 0;*

*int startDate = 0;*

*int endDate = 0;*

*int gracePeriod = 0;*

This structure contains parameters pertaining to a specific emergency. Each emergency is identified by *MissionID* and the E-Device may contain multiple scheduled emergencies at any one time.

Interfaces

***a. Public int getCounter ()***

This interface allows querying of the emergency_*counter* field.

***b. Public bool verifySender (string sender)***

Given a device ID, this interface will return a boolean value indicating whether the sender associated with the ID is authorized to send ESMs. This is accomplished by matching the input string with the *authorizedSender* field in *EmergencyProfile*.

***c. Public bool update (int counter, int missionID, int startDate, int endDate, int gracePeriod)***

This interface overwrites existing emergency parameters within *EmergencyProfile* with new values that are passed in as arguments.

***d. Public string getParameters ()***

This interface returns a list of emergency parameter values within *EmergencyProfile.* The parameters is returned as *a* comma delimited string.

## C. OTHERS

This section discusses the expected interfaces of the supporting infrastructure. However, instead of providing an exhaustive review of all their interfaces, we will focus on those that directly support the operations of both ESM and Heartbeat Managers. These supporting infrastructure functions are comprised of the LPSK, IP, and UDP modules. They are responsible for the receiving and sending of ESM and heartbeat messages. In addition, the LPSK is also responsible for timer services.

### 1. LPSK_Interface

Description The LPSK creates exported resources from the platform's physical resources, partitions the exported resources and controls interactions between the partitions.

Interfaces

***a. Public void setMsgArea (string displayText)***

This interface provides a means to display application messages to the user. By using this interface, the application can print a string to the PL1 display area which is a trusted area that is managed by the kernel.

***b. Public void setFocus (int partitionNum)***

This interface signals the LPSK to switch focus to a specific partition. In the case of the ESM Manager, this interface is activated whenever the emergency state goes from OFF to ON or vice versa. The ESM Manager

will set focus to the trusted partition for purpose of obtaining the user's input regarding which partition, emergency or normal, to toggle to.

***c. Public void setTimer (int channel, int timeout, int handle)***

A timer service is provided by the LPSK and this interface allows that service to be activated. Applications are each assigned a specific channel for using the timer service. This allows the LPSK to differentiate between each caller. Upon timeout (based on seconds), the LPSK will raise an interrupt on the predefined timeout handler.

***d. Public void send (byte[] msg)***

This interface is called by the IP module to request LPSK to transmit an IP message.

***e. Public void endHandler ()***

This interface signals to the LPSK that interrupt handling for a particular interrupt has been completed. LPSK may proceed to free any resources that are allocated for the interrupt processing. It is called by *ESM_Mgr.sigHandle_ReceiveMsg*, *ESM_TimeoutHandler.sigHandle_Timeout*, and *Heartbeat_TimeoutHandler.sigHandle_Timeout*.

***f. Public string getCurrentDateTime ()***

This interface is called by *calculateEndDate* and returns the local system time. The time zone must be included in the returned string.

**2. Internet Protocol**

Description IP is the network layer protocol used for data communication within the emergency network.

Interfaces

***a. Public void sigHandle_Received (byte[] ipPkt)***

This interface is called by the LPSK when a new IP packet is received.

***b. Public void send (string destIP, byte[] msg)***

This interface is called by the UDP module to send a message via the IP module. The destination IP address is passed in as an argument.

### 3. User Datagram Protocol

Description UDP is the transport layer mechanism for both ESM and Heartbeat Managers.

Interfaces

***a. Public void sigHandle_Received (byte[] udpPkt)***

This interface is called by the IP module when a new UDP message is received.

***b. Public void send (string destIP, int srcPort, int destPort, byte[] msg)***

This interface is called by both *sigHandle_ReceiveMsg* and *sigHandle_Timeout* and requests that the UDP module transmit a message. The destination IP address, source and destination port numbers are passed in as arguments.

## D. SUMMARY

This section reviews the requirements for ESM and Heartbeat Managers and determines whether they have been met.

| Identifier | Requirements | Design |
|---|---|---|
| 1.1 | The E-Device shall verify the correctness of the ESM prior processing it. | *sigHandle_ReceiveMsg* to activate *verifyMessage* interface (see Figure 17 - Step 4) |
| 1.2 | The E-Device shall verify that the ESM is from the TOC prior processing it. | *verifyMessage* interface (See V.B.1b) |
| 1.3 | The E-Device shall acknowledge the sender of the ESM upon successful validation of a received ESM. | *sigHandle_ReceiveMsg* to activate *createAcknowledgement* interface (see Figure 17 - Step 7) |
| 1.4 | The E-Device shall allow access to the emergency partition when the emergency has started | *sigHandle_ReceiveMsg* to activate *update* interface (see Figure 17 - Step 6) Emergency state is set to TRUE. |
| 1.5 | The E-Device shall prohibit access to the emergency partition when the emergency has ended | *sigHandle_ReceiveMsg* to activate *update* interface (see Figure 17 - Step 6) Emergency state is set to FALSE. |
| 3.1 | TOC should be able to estimate the number of emergency responders that are near the emergency scene. | Heartbeat Manager to provide status update (See V.A.1) |

Table 18. ESM and Heartbeat Managers' Requirements

As shown in Table 18, the requirements of the ESM and Heartbeat Managers have been satisfied.

This chapter realizes the high-level design analysis described in Chapter IV and confirms that the design is consistent and complete so that detailed design is supported. The described data structures, flow control, and interface invocation provides a proof of concept and details sufficient to kick start the implementation phase.

# VI. SUMMARY

The summary chapter reviews related work in similar areas, and also provides direction for subsequent research. A concluding section provides a recap of our research efforts in designing the emergency signaling mechanism.

## A. RELATED WORK

We review other emergency access solutions that temporarily extend one's security privileges. One such approach is break-glass [34] and its purpose is to provide a quick means for extending a person's access rights in exceptional cases [35]. Implementing break-glass typically involves pre-staging "emergency accounts" that have greater access rights. The distribution of these accounts is carefully managed to provide timely access when needed. These accounts are monitored so that the security administrator is alerted when they are used. There are procedures in place to perform clean-up after the break-glass action is completed. The key difference with the T-TASI system is that the work we propose does not require the creation of "emergency accounts". It extends the break-glass approach by preventing the unwarranted dispersal of sensitive information due to the potential misuse of such accounts. In addition, within T-TASI, the privileged access rights are automatically revoked after the crisis.

Another approach is optimistic security [36] where it assumes that the risk of failure and recovery cost is low compared to the cost of not granting access. It depends on strong authentication and auditing processes so that users can be associated with given actions. It also assumes that for any transformation on data or security properties of that data such as confidentiality, integrity, etc., there is a compensating transaction that exists to reverse this transformation. The key difference with the T-TASI system is that the T-TASI system does not need to back out of transformations because unauthorized accesses are prevented. Also,

it is not necessary to incorporate compensating transactions for reversing an incorrect transformation. This makes T-TASI a much simpler architecture.

Ferreira et al. [37] propose a Break The Glass policy approach that implements the security policy in the application logic. Access control is defined based on the identity of the individual rather than the role he belongs to for better granularity. The key difference with the T-TASI system is in its scalability. In T-TASI, the same security policy for extraordinary access and data confinement can be applied to a variety of scenarios. Unlike the Break the Glass policy, the policy is built into the architectural framework and does not reside in the application.

A System-Of-Systems (SOS) Security approach proposed in [38] looks at constructing high assurance composite systems from existing systems with varying security properties. The key difference with the T-TASI system is that T-TASI deals with both inter- and intra-site data flows. Secure channels are setup to govern the communication among the partitions within an E-Device, and among partitions of different E-Devices (i.e., remote trusted channels). SOS Security focuses on modeling inter-site data flows based on a dynamic networking environment.

## B. FUTURE WORK

While the design was developed taking into account the security requirements, we have largely assumed a CONOPS environment where the E-Device is always in possession of its authorized owner. In some situations, the threat of hostile possession of an E-Device must be considered. Hence, we propose the following research areas as a follow-up to mitigate the risk of E-Devices falling into the wrong hands.

### 1. Storage Encryption

Storage encryption is used to encrypt data at rest (i.e .on secondary storage). Hence, these solutions can be used to safeguard the data in the emergency partitions so that the data are inaccessible even if the attacker has physical access to the E-Device. The study will look at whether the data associated with each emergency partition should be encrypted using different keys and that these keys be provided by the TOC instead of being stored within the E-Device.

### 2. Remote Purging

The objective of remote purging is to delete sensitive material such as privileged information within the emergency partitions, private key, certificates, and proprietary software residing in the E-Device. The purge operation will take place when the E-Device is no longer in the possession of its authorized users. When the loss of an E-Device is reported to the TOC, the TOC will issue a device revocation message to the affected E-Device which will proceed with the purge operation in the background. The study should examine alternate means of protecting the data, especially if the E-Device was lost after the emergency state was declared.

## C. CONCLUSION

The objective of this work was is to investigate and propose a means to perform emergency signaling within a tactical IP network. This provides a means to provide secure, transient access to privileged information and also supports revocation of access to it after the emergency. We embarked on our research using both top-down or bottom-up approaches.

The top-down approach reviews various communication protocols and evaluates their merits with respect to their suitability for use on the emergency network. The evaluation was conducted based on the protocols being able to

meet security requirements and the integration challenges with the existing TCB. After selecting the IPSec protocol, we proceeded with an analysis of how the protocol will be used in supporting our CONOPS. The analysis includes how IPSec should be configured, choice of cipher suite, and integration strategies to the TCB.

The bottom-up approach reviews specific requirements pertaining to both Emergency and Heartbeat Managers. We examined the possible system states and state transitions including the prerequisite conditions and actions taken after a transition. We proceeded by defining the essential data structures and message semantics for both emergency signal and heartbeat messages. We also propose how both modules can be integrated into existing the TCB architecture. Subsequently, we examined the choice of transport layer mechanism for transmitting the emergency signal.

In summary, we have presented the design for a vertical slice of the broader T-TASI device implementation, the emergency signaling mechanism, with sufficient granularity to commence the implementation phase.

# LIST OF REFERENCES


[1] Y. Yuan, B. Detlor, Intelligent mobile crisis response systems. Communications of the ACM 48, 2 (Feb 2005), 95–98.

[2] SAFECOM, Public Safety Statement of Requirements for Communications and Interoperability Version 1.2, Available from: http://www.safecomprogram.gov/SAFECOM/library/technology/1258_statementof.htm (accessed 25 Sep 2010).

[3] T. E. Levin, J. Dwoskin, G. Bhaskara, T. D. Nguyen, P. C. Clark, R. Lee, C. E. Irvine, T. Benzel, Securing the dissemination of emergency response data with an integrated hardware-software architecture. In International conference on the technical and socio-economic aspects of trusted computing (TRUST), Lecture Notes in Computer Science, University of Oxford, Springer, Apr 2009.

[4] T. E. Levin, C.E. Irvine, T. Benzel, T. D. Nguyen, P. C. Clark, G. Bhaskara, Trusted Emergency Management, NPS Technical Report NPS-CS-09-001, Naval Postgraduate School, Monterey, CA, September 2008.

[5] S. Frankel, K. Kent, R. Lewkowski, A. D. Orebaugh, R. W. Ritchey, S. R. Sharma, Guide to IPSec VPNs, NIST Special Publication 800–77, Dec 2005.

[6] S. Kent, IP Authentication Header, RFC 4302, Dec 2005.

[7] S. Kent, IP Encapsulating Security Payload (ESP), RFC 4303, Dec 2005.

[8] D. R. Kuhn, V. C. Hu, W. T. Polk, S. J. Chang, Introduction to Public Key Technology and the Federal PKI Infrastructure, NIST Special Publication 800-32, 26 Feb 2001.

[9] The Committee on National Security Systems, Policy on Wireless Communications: Protecting National Security Information, CNSSP No. 17, May 2010.

[10] IETF, Requirements for Internet Hosts – Communication Layers, RFC 1122, Oct 1989.

[11] B. Schneier, Software Complexity and Security, Crypto-Gram Newsletter, Mar 2000, Available from http://www.schneier.com/crypto-gram-0003.html#8 (accessed 8 Oct 2010).

[12] L. Singaravelu, C. Pu, H. Härtig, C. Helmuth, Reducing TCB complexity for security-sensitive applications: three case studies, In Proceedings of the 1st ACM Sigops/Eurosys European Conference on Computer Systems 2006 (Leuven, Belgium, April 18–21, 2006), EuroSys '06. ACM, New York, NY, 161-174, 2006.

[13] D. E. Geer, Complexity Is the Enemy, IEEE Security & Privacy, vol. 6, no. 6, 88–88, 2008.

[14] T. E. Levin, C. E. Irvine, C. Weissman, T. D. Nguyen, Analysis of Three Multilevel Security Architectures, Proc. of Computer Security Architecture Workshop, ACM. Nov 2, 2007, Fairfax, Virginia, USA.

[15] M. D. Schroeder, J. H. Saltzer, A hardware architecture for implementing protection rings, Communications of the ACM, vol. 15, no. 3, 157–170, Mar 1972.

[16] Intel Corp., Intel 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide, Part 1, Nov 2006. http://download.intel.com/design/processor/manuals/253668.pdf (accessed 9 Aug 2008).

[17] B. Aboba, W. Dixon, IPSec-Network Address Translation (NAT) Compatibility Requirements, RFC 3715, Mar 2004.

[18] T. Kivinen, B. Swander, A. Huttunen, V. Volpe, Negotiation of NAT-Traversal of in the IKE, RFC 3947, Jan 2005.

[19] Haluk Aydin, NAT Traversal: Peace Agreement between NAT and IPSec, SANS Institute InfoSec Reading Room, Available from: http://www.sans.org/reading_room/whitepapers/vpns/nat-traversal-peace-agreement-nat-ipsec_731 (accessed 28 Nov 2010).

[20] A. Huttunen, B. Swander, V. Volpe, L. DiBurro, M. Stenberg, UDP Encapsulation of IPsec ESP Packets, RFC 3948, Jan 2005.

[21] J. P. Anderson, Computer Security Technology Planning Study. Technical Report ESD-TR-73-51 Vol. II, Oct 1972.

[22] National Security Agency, NSA Suite B Cryptography, Available from http://www.nsa.gov/ia/programs/suiteb_cryptography (accessed 8 Oct 2010).

[23] Information Assurance Directorate: U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness. Version 1.03, Dated 29 Jun 2007.

[24] L. Law, J. Solinas, Suite B Cryptography Suites for IPSec, RFC 4869, May 2007.

[25] S. Kent, K. Seo, Security Architecture for the Internet Protocol, RFC 4301, Dec 2005.

[26] C. M. Kozierok, The TCP/IP Guide, Available from http://www.tcpipguide.com (accessed 9 Oct 2010).

[27] IANA Protocol Numbers, Available from http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml (accessed 26 Oct 2010).

[28] Information Sciences Institute, Transmission Control Protocol, RFC 793, Sep 1981.

[29] J. Postel, User Datagram Protocol, RFC 768, Aug 1980.

[30] National Geodetic Survey, NADCON Computations, Available from http://www.ngs.noaa.gov/cgi-bin/nadcon.prl (accessed 21 Nov 2010)

[31] Study Compares Older and Younger Pedestrian Walking Speeds, Available from http://www.usroads.com/journals/p/rej/9710/re971001.htm (accessed 21 Nov 2010)

[32] Karen Aspelin, Establishing Pedestrian Walking Speeds, Available from http://www.westernite.org/datacollectionfund/2005/psu_ped_summary.pdf (accessed 21 Nov 2010)

[33] D. Mills, U. Delaware, J. Martin, J. Burbank, W. Kasch, Network Time Protocol Version 4: Protocol and Algorithms Specification, RFC 5905, Jun 2010.

[34] Joint NEMA/COCIR/JIRA Security and Privacy Committee (SPC), Break-Glass Procedure – An Approach to Granting Emergency Access to Healthcare Systems, Dec 2004.

[35] A. D. Brucker, H. Petritsch, Extending access control models with break-glass, Proceedings of the 14th ACM symposium on Access control models and technologies (SACMAT '09). ACM, New York, NY, USA, 197–206.

[36] D. Povey, Optimistic security: a new access control paradigm, *Proceedings of the 1999 workshop on New security paradigms* (NSPW '99), ACM, New York, NY, USA, 40–45.

[37] A. Ferreira, R Cruz-Correia, L. Antunes, P. Farinha, E. Oliveira-Palhares, D. W. Chadwick, A. Costa-Pereira, How to Break Access Control in a Controlled Manner, 19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06), 847–854.

[38] K. Kifayat, D. Llewellyn-Jones, A. Arabo, O. Drew, M. Merabti, Q. Shi, A. Waller, R. Craddock, G. Jones, State-of-the-Art in System-of-Systems Security for Crisis Management, Fourth Annual Layered Assurance Workshop (LAW 2010), Dec 2010.

# INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Kris Britton
   National Security Agency
   Fort Meade, MD

4. John Campbell
   National Security Agency
   Fort Meade, MD

5. Deborah Cooper
   DC Associates, LLC
   Reston, VA

6. Grace Crowder
   NSA
   Fort Meade, MD

7. Louise Davidson
   National Geospatial Agency
   Bethesda, MD

8. Vincent J. DiMaria
   National Security Agency
   Fort Meade, MD

9. Rob Dobry
   NSA
   Fort Meade, MD

10. Jennifer Guild
    SPAWAR
    Charleston, SC

11. CDR Scott Heller
SPAWAR
Charleston, SC

12. Dr. Steven King
ODUSD
Washington, DC

13. Steve LaFountain
NSA
Fort Meade, MD

14. Dr. Greg Larson
IDA
Alexandria, VA

15. Dr. Carl Landwehr
National Science Foundation
Arlington, VA

16. Dr. John Monastra
Aerospace Corporation
Chantilly, VA

17. John Mildner
SPAWAR
Charleston, SC

18. Dr. Victor Piotrowski
National Science Foundation
Arlington VA

19. Jim Roberts
Central Intelligence Agency
Reston, VA

20. Ed Schneider
IDA
Alexandria, VA

21. Mark Schneider
NSA
Fort Meade, MD

22. Keith Schwalm
Good Harbor Consulting, LLC
Washington, DC

23. Ken Shotting
NSA
Fort Meade, MD

24. Dr. Ralph Wachter
ONR
Arlington, VA

25. Dr. Peter J. Denning
Naval Postgraduate School
Monterey, CA

26. Dr. Cynthia E. Irvine
Naval Postgraduate School
Monterey, CA

27. Timothy E. Levin
Naval Postgraduate School
Monterey, CA

28. Professor Yeo Tat Soon, Director
Temasek Defence Systems Institute
National University of Singapore
Republic of Singapore

29. Ms Tan Lai Poh, Assistant Manager
Temasek Defence Systems Institute
National University of Singapore
Republic of Singapore

30. Mr Raymond Quah
Defence Science and Technology Agency
Republic of Singapore