1 Table of content

1 TABLE OF CONTENT ................................................................. 4

2 ABBREVIATIONS .................................................................. 7

3 INTRODUCTION ..................................................................... 8

3.1 Objectives of the project ...................................................... 8
  3.1.1 Confidentiality ............................................................. 8
  3.1.2 Integrity ..................................................................... 8
  3.1.3 Availability ................................................................. 8
  3.1.4 Non-reputability .......................................................... 8
  3.1.5 Privacy ..................................................................... 8

3.2 Introduction to Set Top Boxes ............................................... 9
  3.2.1 Functionality .............................................................. 9
  3.2.2 Operating principles ...................................................... 9

4 OVERVIEW OF MHP ............................................................. 11

4.1 Definition ....................................................................... 11

4.2 Basic architecture ............................................................ 11
  4.2.1 Context .................................................................... 11
  4.2.2 Architecture .............................................................. 12

4.3 Profiles ........................................................................... 14
  4.3.1 Enhanced Broadcast Profile ......................................... 14
  4.3.2 Interactive TV profile .................................................. 14
  4.3.3 Internet access profile .................................................. 15

5 MHP SECURITY REQUIREMENTS ......................................... 16

5.1 Users of MHP ................................................................ 16
  5.1.1 Consumers ............................................................... 16
  5.1.2 Broadcasters ............................................................. 16
  5.1.3 Service providers ....................................................... 16

5.2 General security requirements in MHP ................................. 16

5.3 Security requirements for consumers .................................... 17

5.4 Security requirements for broadcasters ................................. 18

5.5 Security requirements for service providers ............................ 18

6 SECURITY MECHANISM IN MHP ......................................... 19
6.1 Authentication of applications ................................................................. 19
   6.1.1 The security framework for applications ........................................... 19
   6.1.2 Authentication of a hierarchical file systems ................................. 20
   6.1.3 Hash codes ................................................................................. 20
   6.1.4 Signature files ............................................................................ 21
   6.1.5 Certificate files ........................................................................... 22
   6.1.6 File authentication steps ............................................................ 22
   6.1.7 Example of a signed MHP application ........................................... 23

6.2 Security policies for applications .......................................................... 24
   6.2.1 File access ................................................................................. 25
   6.2.2 Application lifecycle control policy ................................................ 26
   6.2.3 Return channel access policy ....................................................... 26
   6.2.4 Tuning access policy ...................................................................... 26
   6.2.5 Service selection policy ............................................................... 26
   6.2.6 User setting and preferences access policy ...................................... 26
   6.2.7 Network permissions ..................................................................... 26
   6.2.8 Privileged runtime code extension permission .................................. 27
   6.2.9 Application storage .................................................................. 27
   6.2.10 Smart card access .................................................................... 27
   6.2.11 Provider management ............................................................... 27
   6.2.12 Privileged application access .................................................... 27
   6.2.13 Credentials .............................................................................. 27

6.3 Security of the return channel ................................................................ 28
   6.3.1 Basic information ..................................................................... 28
   6.3.2 TLS description and security ......................................................... 29
       6.3.2.1 Description and operating principles ..................................... 29
       6.3.2.2 Security ............................................................................. 31
       6.3.2.3 Downloading of certificates for TLS in MHP ....................... 32

6.4 DVB MHP public key infrastructure ...................................................... 32
   6.4.1 DVB MHP PKI for MHP terminal Manufacturers .......................... 34
   6.4.2 DVB MHP PKI for Application Developers ................................... 34
   6.4.3 DVB MHP PKI for Broadcasters ................................................ 35
   6.4.4 Certificate management ............................................................... 35

6.5 Application information table ................................................................ 35

6.6 MHP minimum security requirements .................................................. 36

7 DVB-J SECURITY ....................................................................................... 37
   7.1 Why Java? ...................................................................................... 37
   7.2 Java’s ‘Sandbox’ principle ............................................................... 38
   7.3 Java permission request file ............................................................. 38

8 MHP SECURITY ANALYSIS BASED ON THE SECURITY CRITERIAS ........................................... 41

9 CONCLUSION ......................................................................................... 43
10 REFERENCES ................................................................. 44

11 APPENDIXES ................................................................. 45

11.1 MHP end-to-end reference model ................................ 45
2 Abbreviations

In order of appearance:

MHP- multimedia home platform
STB- set top box
IP- internet protocol
VOD- video on demand
EPG- electronic program guide
DRM- digital rights management
PC- personal computer
IM- instant messaging
VoIP- voice over IP
RGB- red, green, blue mode
DVI- digital visual interface
HDMI- high definition multimedia interface
HDTV- high definition television
MPEG- moving picture experts group
DVB- digital video broadcast
API- application programming interface
HTML- hypertext markup language
XML- extensible markup language
ECMA- European computer manufacturers association
DOM- document object model
PIN- personal identification number
DVB-J- DVB Java
PSTN- public switched telephone network
TLS- transport layer security
SSL- secure socket layer
PKI- public key infrastructure
CA- certificate authority
MAC- media access control
DNS- dynamic name service
RCMM- root certificate management messages
CRL- certificate revocation list
USB- universal serial bus
UI- user interface
AIT- information allocation table
3 Introduction

3.1 Objectives of the project

The main objective of the project is to analyze the security of STB communication standard: the MHP as it is very important from the point of security. The above mentioned standard will be analyzed in accordance of these criteria:

- Confidentiality
- Integrity
- Availability
- Privacy
- Non-reputability

3.1.1 Confidentiality

The protection of information so that someone not authorized to access the information cannot read the information even though the unauthorized person might see the information’s container.

3.1.2 Integrity

Integrity is given when a certain piece of information is not tampered in an unauthorized manner. In general it can be said that any modification that happens against the intention of the owner of the information violates its integrity.

3.1.3 Availability

The property that a product’s services are accessible when needed and without undue delay

3.1.4 Non-reputability

To create evidence that data has been sent or received, so that the sender (or receiver) cannot later falsely deny this fact.

3.1.5 Privacy

The protection of personal data so, that unauthorized person could not access it.
3.2 Introduction to Set Top Boxes

3.2.1 Functionality

An IP set-top box is a dedicated computing device that serves as an interface between a television set and a broadband network. In addition to decoding and rendering broadcast live TV signals, a set-top box provides functionality that includes video-on-demand (VOD), electronic program guide (EPG), digital rights management (DRM), and a variety of interactive and multimedia services. Set-top boxes can support additional features such as Web browsing, e-mail and viewing e-mail attachments, advanced multimedia codecs, home networking and PC connectivity including playback and rendering of content stored on the PC (photos, music, and personal videos), gateway functionality, instant messaging (IM), and real-time voice over IP (VoIP). These types of advanced functionality are in demand by end-users, enable incremental network operator service opportunities, and allow set-top box manufacturers to easily offer a large range of differentiated devices. Current set-top box development is driven by service provider requirements and customer demand for new features. Priorities for service providers include the capacity to deploy, using minimal capital expenditures, new revenue-generating services and multimedia and entertainment-oriented applications on a set-top box to meet changing customer requirements over time. Service providers also need to ensure that copyrighted content is protected from unauthorized distribution. To accommodate these expectations, the set-top box operating system platform must be extensible and remotely upgradeable, and include both rich multimedia technologies and fundamental security features, such as access control. [3]

![Figure 1: Example of STB, AmiNET 110](image)

3.2.2 Operating principles

STB can have different video outs to the TV-Set: Composite, Component (YPbPr), S-Video, RGB or the digital ones like DVI, HDMI (with possibility to receive HDTV); audio outs: Stereo, Dolby Digital 5.1surround and later versions. STB connect to IP network by the Ethernet port. In fact it is a small computer with its own operating system, Web-browser and MPEG decoder. Some STB devices do not have their own operating system and every time they connect to the IP network they download it’s core and configuration files.

STB operates in several directions. From the user’s perspective STB decodes data stream and converts it to the analog signal for the TV-Set. It also contains the general...
information about the channels and is used to choose the particular channel to watch. From the network perspective STB uses Internet Group Management Protocol to send and receive the service information to the network. During the start of the session STB enables the initialization process. During this process it downloads configuration information from the administrative system. This information includes digital guide and IGMP information which shows to which multicast group the particular channel refers. When the user switches the channel STB tells the network that the user no longer needs the current multicast stream and wants to connect to the different multicast group. Then STB receives new MPEG-2 stream, decodes it and sends the analog video to the TV-Set. [2]
4 Overview of MHP

4.1 Definition

**Multimedia Home Platform** (DVB-MHP) is an open middleware system standard designed by the DVB project for interactive digital television. The MHP enables the reception and execution of interactive, Java-based applications on a TV-set. Applications can be delivered over the broadcast channel, together with audio and video streams. These applications can be for example information services, games, interactive voting, e-mail, SMS or shopping. For all interactive applications an additional return channel is needed. [1]

At the moment different MHP versions are available: MHP 1.0.x, MHP 1.1.x, MHP 1.2.x and MHP/GEM different standards.

4.2 Basic architecture

4.2.1 Context

At its simplest level, the MHP is set in the following context (see Figure 2). The software of the MHP has access to flows of streams and data, and may write some data to storage. The platform may be able to route streams and data outwards to a sink or store.

![Diagram](image.png)

**Figure 2:** MHP context

The platform will receive inputs from Viewer input devices and output communications through a screen or other outputs like loudspeakers to present to the viewer. The platform may have access to communications with remote entities.
The diagram in Figure 3 shows a possible set of external interfaces between an MHP and the outside world.

![Diagram showing external interfaces between MHP and outside world]

**Figure 3:** External interfaces between the MHP and outside world

The resources of the MHP, accessible by an application, may be contained in a series of different connected physical entities. The local cluster may connect a number of MHP terminals and resources. A cluster may also include resources which are not part of the MHP infrastructure and are not available to the application.

### 4.2.2 Architecture

The Architecture describes how the MHP software elements are organized. The MHP model considers 3 layers (Figure 4):

- Resources
- System software
- Applications

The API lies between the Applications and the System Software seen from the perspective of an application.

![Diagram showing basic architecture]

**Figure 4:** Basic architecture

**Resources:** The hardware entities in the platform include a number of functions. They are represented by hardware or software resources. There is no assumption about how they are grouped. The model considers that there can be more than one hardware entity in the total Platform.
From an abstract point of view it makes no difference if the logical resources are mapped into one or several hardware entities. What is important is that resources are provided to the MHP transparently. An application should be able to access all locally connected resources as if they were elements of a single entity.

**System software:** Applications will not directly address resources. The system software brings an abstract view of such resources. This middle layer isolates the application from the hardware, enabling portability of the application. The implementations of the Resources and System software are not specified in this document.

**Application manager:** The system software includes an application management function, which is responsible for managing the lifecycle of all applications, including Interoperable ones.

**Application:** Applications implement interactive services as software running in one or more hardware entities. The interface for MHP applications is a top view from application to the system software.

**Figure 4** illustrates an idealized architecture model of the processes which will occur in an MHP. A hierarchy of control is assumed in which each layer controls the processes in adjacent layers. The top layer is responsible for the control of the operation via interactive applications. The Application Manager is part of the System Software and as such is implementation specific. It interacts with all applications.

The System Software implements the API by presenting an abstract model of:

- Streams played from different sources and pipes for conducting them
- Commands and events
- Data records or files
- The hardware resources

[5]
4.3 Profiles

**Figure 5: MHP profiles**

### 4.3.1 Enhanced Broadcast Profile

MHP's "Enhanced Broadcast Profile" was designed to mirror in many ways the functionality of existing middleware systems and the applications that run on them.

As suggested by the title, this profile calls for a set-top-box with no or limited return channel capabilities and represents the lowest of the three profiles in terms of set-top-box performance. [6]

### 4.3.2 Interactive TV profile

The Interactive TV profile calls for a set-top-box with a more significant interactive channel. One of the differences between the two profiles is that applications can be downloaded via the return channel in Profile 2, whereas this is only possible via the broadcast channel in Profile 1. There is also greater support of the interactive channel with appropriate APIs, etc. [6]
4.3.3 Internet access profile

The Internet Access Profile targets a more sophisticated set-top-box, with greater processing power and memory than in profiles 1 & 2. This profile is consistent with the accessing of Internet content on the set-top-box.

Profile 3 contains an optional HTML element, called DVB-HTML. [6]
5 MHP security requirements

5.1 Users of MHP

There are three different groups of users in MHP:

- Consumers
- Broadcasters
- Service providers

Each group have their own interests and based upon those different needs the security problems vary.

5.1.1 Consumers

Consumer is the person in the very end of the digital video broadcasting chain. He actually watches, listens to or interacts with the content of the DVB.

Usually the ordinary consumer will not care about the technical aspects of the MHP or any other part of the digital video broadcasting mechanism. The main goal for them is how to get more options and more sophisticated ones for less money. The most important aspect of security that consumers greatly care about is their privacy on the network. Consumers must be assured that no third party will ever get an access to their private information starting with account logins, various passwords, private contact details and ending with such important information as credit card numbers.

5.1.2 Broadcasters

Broadcaster is an entity that operates channels of popular one-to-many multicast content.

The broadcaster operates the channel carrying audio and video data to consumer or IP packet back to the service provider. The main goal for the broadcaster is to ensure that consumers pay for their services and that the information on the channel is legal.

5.1.3 Service providers

The service provider rent channels to the consumers. They must ensure that the consumer is able to get services and to encrypt channels only that he paid for. They also must protect against the users who are trying to do harm via the back channel.

5.2 General security requirements in MHP

The MHP security model should embrace all the functions and components like the hardware, operating system, applications and content data of the MHP. It must
maintain the integrity of the MHP terminal, the integrity and validity of the content, software and data held in the MHP or transmitted to or from it.

The MHP security model should guard against the following problems:

- Malicious damage of the MHP device by an application
- “Denial of service” through competing applications, malicious attacks or other means
- Unauthorized use of user data
- Unauthorized use or theft of content
- Maintaining integrity of the content in the content delivery chain
- Preventing unauthorized use of the return channel
- Unauthorized access to the communication of the return channel

In other words, the MHP should provide authentication and verification systems that validate incoming applications to the MHP. How to check the operation of application and their use of the MHP’s resources is also needed. A password system that allows access to secure applications or online sites is necessary. Furthermore, a copyright system that manages the storage of content within the MHP should be considered, an encryption system that guard against the theft of content is of course needed.

### 5.3 Security requirements for consumers

The main security problem for consumers is privacy. How to protect their private information and how to get service without intruding by unauthorized users are their main considerations. So the requirements defined here concentrated on privacy.

An application running on the MHP shall not be able to access private information such as Personal Identification Number (PIN) or credit card numbers that have been supplied to previous applications. An application shall not access private information without preliminary authorization from the consumer also. The MHP terminal shall not allow unauthorized access to private data, which is stored in the MHP terminal or other device connected to the home network by third party. An application running on the MHP platform shall be able to request and use security resources when available. A common internal, operational and API-related security model needs to be defined for data and applications. It shall be conditional access independent. MHP should have internal and operational security aspects including anti-piracy mechanism, secure transaction, value transfer and user authorization mechanism. Cross application data management, sharing of private sensitive information and secure time stamping should be included in MHP security too. The owner of the MHP should be able to change the resources that are granted to unauthorized applications. Sensitive user data stored in MHP terminals must be encrypted.
5.4 Security requirements for broadcasters

The main security problems for a service provider are information confidentiality, integrity and non-reputability. The service provider must keep almost all types of information in confidentially to enhance its competition in the market. Information should be integrity and consistent to make it usable. When some consumers want to deny the service the provider has provided, how to give evidence is also important for a service provider. For example, the service provider should store user information in secure and safe repository, including both physical security and information security storage. The application should not do harm operations on consumer’s equipment. The service provided and relative confirmation from consumer must be logged, so any malicious operation can be recorded to provide evidence for punishing the intruders. System crashes can be graceful recovered quickly, any user in the system will not be affected heavily. Of course it’s better if it does not affect the users. User must be able to escape from unaccepted application whenever he wants. Applications can terminate appropriately and resources are correctly released when the users want. Unauthorized user could not access system information via return channel. This is because the MHP applications can be downloaded from return channel. As an IP based connection, return channel must provide additional security mechanisms. Access rights to authorized applications must be carefully considered. This can ensure that users with different privileges can access different levels of resources. The virus checking is needed to consider from both data transmission directions. The service should detect viruses possible from the return channel. The same happened to the consumer. Their STBs, TVs or other instruments must be protected against the virus.

5.5 Security requirements for service providers

The broadcaster is responsible for operating channels. So how to keep the communication secure between the consumer and the service provider is the main task. One important task is that the sensitive information transferred must be encrypted. Because the radio signal is unavoidable to be listened by malicious users, the information transferred must be encrypted so that it is difficult for intruders to get useful information from the radio signals. The connection from the consumer to the service provider should be authenticated and encrypted including both the broadcast stream and return channel data. A secure connection could guard variety general attacks on the information system. The broadcaster should detect malicious tampering with application code and non-real-time data. Unauthorized application could not request return channel for interaction with service provider. Since return channel gives the consumer ability to interact with the servers, it is dangerous for an unauthorized user to access such servers.
6 Security mechanism in MHP

This section will cover the following areas of security:

- Authentication of applications
- Security policies for applications
- Security of the return channel
- DVB MHP Public key infrastructure
- Application information table
- Platform minimum security requirements

6.1 Authentication of applications

6.1.1 The security framework for applications

The security framework enables a receiver to authenticate the source of application code or other files. In the case of application code files, the authentication advises the receiver what access rights should be granted to an application for sensitive resources.

The system uses three different types of authentication messages:

- **Cryptographic hash codes** which provides a summary of a quantity of data - typically a subset of the total set of data under consideration.
- **Signatures** which deliver a master hash code (computed over all of the appropriate data) that has been "signed" by an authorizing organization. The signing process securely associates the master hash code with the signatory. The hash code process shows that the data has not been tampered with since it was signed by the signatory.
- **Certificates** which provide a "chain of trust" from the authorizing organization up to some trusted third party (the root certificate authority) that is well known to the receiver

The messages are delivered within files of the file system so this authentication scheme is applicable to any hierarchical file system whether operating over the broadcast or return channels. [5]

All applications come with an application identification number. Signed applications come with the identification number from signed application identification number range and unsigned applications from the unsigned applications number range. An application with an application_id from the signed applications range but that is not signed is considered to have failed authentication. An application with an application_id from the unsigned applications range is treated as unsigned even if the files might be transmitted with signatures.
The authentication messages are transported in files. In no cases shall a service transfer be required to access the file content.

6.1.2 Authentication of a hierarchical file systems

The authentication of hierarchical file systems is based on the authentication of hierarchical structure of objects. Hash codes are computed systematically and accumulatively across some or all of the objects in the hierarchy. A signature at the top of the hierarchy identifies the source of the objects. The framework provides a method which enables the authentication of subtrees of a file system with a single signature. Since checking signature is far more time-consuming than checking hashcode values, this mechanism is more efficient than signing each object of a subtree. Further, only the objects that are loaded need real-time hashcode checking.

6.1.3 Hash codes

In MHP from the broadcasters side the integrity of the application files is verified and the hash values are created and stored in the hash files always having a name template ‘dvb.hashfile’. Every directory which has to be signed in the hierarchical tree finally contains the hash file. The hash computation considers the content and attributes of the objects rather than transport specific information. As a result, the authentication is independent of the underlying transport protocol. In the case of a directory the hash value depends on the hash values of the objects bound to it, and so provides a hash of all of the objects to be authenticated in the "tree" below it. The hash file itself contains one or more sections that are structured as follows:

- Hash algorithm identifier
- List of filenames
- Hash value

Example Syntax of the hash file:
Hashfile () {
    digest_count
    for( i=0; i<digest_count; i++ ) {
        digest_type
        name_count
        for( j=0; j<name_count; j++ ) {
            name_length
            for( k=0; k<name_length; k++ ) {
                name_byte
            }
        }
        for( j=0; j<digest_length; j++ ) {
            digest_byte
        }
    }
}

Figure 6: Syntax of the hash file. [5]

The HashFile lists all of the elements of the current directory except itself, and possibly except signature files. An application comprises files containing data that can be spread across various directories and is contained within a subtree of the file hierarchy.

For creation of the Hash values MHP allows the algorithms MD5 (128 bit) and SHA-1 (160 bit), whereas the SHA-1 algorithm has to be considered safer due to its significantly longer hash word.

6.1.4 Signature files

The signature files are stored in the root directory and have two functions:
- To protect the highest hash file (contained in the root directory)
- To specify the originator of the application

By signing, the signing entity assures that the signed application will not cause any damage and is therefore safe to run on the user terminals. In MHP all signature files have the name “dqb.signature.*”, whereas “*” stands for consecutive id-numbers in case of more than one signature files in the directory. The digital signatures contained in the signature files are created by hashing the “dqb.hashfile” in the root directory and then encrypting this hash value by using the RSA method: The broadcaster uses its private key for encryption and stores the public key (necessary to verify the signature) in the corresponding certificate file.

The SignatureFile is a File containing one digital signature. It contains the following structure:
Signature ::= SEQUENCE {
  certificateIdentifier AuthorityKeyIdentifier,
  hashSignatureAlgorithm OBJECT IDENTIFIER,
  signatureValue         BIT STRING }

**Figure 7:** Signature file structure [5]

The SignatureFile is located in the root directory of the subtree that it signs. There can be several SignatureFiles, as there can be several entities that sign the structure.

### 6.1.5 Certificate files

For every entity that signs an application (directory tree) two files have to be provided in the root directory: the signature file and the certificate file. The "*" in both filenames represent the id-number of the entity, so that it is clear which certificate file corresponds to which signature file. The certificate file contains the certificate whose public key allows for confirming the signature of the corresponding signature file. The MHP-standard uses X.509-certificates. The certificate that a broadcaster uses for signing is usually a leaf certificate standing at the end of a chain of trust. MHP defines, that all certificate files have to contain all certificates that are necessary to authenticate the leaf certificate used for signing: The complete chain from the leaf certificate up to the root certificate. The root certificate authority has to be well known to the receiver i.e. the root certificate has to be stored on the MHP terminal. Here is an example of a certificate file:

```c
Certificatefile () {
  certificate_count
  for( i=0; i<certificate_count; i++ ) {
    certificate_length
    certificate()
  }
}
```

**Figure 8:** Syntax of the certificate file [5]

Certificates are considered to be the same if they have bitwise identical contents. Where an MHP application is authenticated by multiple signature file / certificate file pairs, the MHP terminal shall check all the signature file / certificate file pairs before deciding that a file fails authentication. It is dependant on receiver policy whether additional signature file / certificate file pairs are checked once one satisfactory pair has been found.

Certificates and certificate management will be discussed more closely in the Certificate management section.

### 6.1.6 File authentication steps

Logically a file is authenticated as follows:
a) Confirm that the file is listed in the hash file located in the same directory as the file to be authenticated.
b) Verify that the file contents and the corresponding digest value are consistent.
c) Recursively ascend the directory hierarchy checking that the hash file in each directory is authenticated by the hash file in its parent directory until a directory is found that contains one or more signature files.
d) For a signature file locate the corresponding certificate file (where the x portion of the signature file's file name identifies the certificate file to be used).
e) If the root certificate in the corresponding certificate file is unknown to the MHP receiver, return to step (d) and repeat for the other signature files.
f) Use the corresponding certificate file to verify that the signature correctly signs the hash file.
g) Verify that the attributes of the Subject field include content that matches the signaling information.
h) Follow the certificate chain contained within the certificate file verifying each link in the chain until the link to the root certificate is found.
i) If the identified root certificate and all the intermediate certificates leading to it are "satisfactory"("satisfactory" condition depends on the receiver), accept the files as being authenticated.
j) If the identified root certificate or any of the intermediate certificates leading to it are not "satisfactory", return to step (d) and repeat for the other signature files.
k) Dependant on receiver policy return to step (d) and repeat for other signature files.

[5]

However this is only a logical order and it does not mean that implementations must perform these steps in this exact order.
A file system may contain several independent authenticated subtrees, each tree with its own subtree root directory.

6.1.7 Example of a signed MHP application

Figure 9 shows an example of a signed directory tree that contains a MHP application. The application itself consists of Java class files and data files.
The example shows that all directories in the tree contain the hash file whereas the hash file in the root directory checksums the complete directory tree. As explained in the sections above, the signature file secures the root hash file and specifies the originator of the application. The corresponding certificate file provides the chain of trust that connects the signing entity with the root certificate known to the MHP terminal. In this case only one entity has signed the application. In practice, signing of an application respectively a certain directory tree within an object carousel is accomplished by software tools.

6.2 Security policies for applications

This clause specifies the resource access policy for the downloaded applications. The resource access policy depends on two factors
- The access rights requested by the broadcaster through the signaling.
- The access rights granted by the user.

The ultimate access rights that are granted to the applications are the intersection of the access rights requested by the broadcaster and the access rights granted by the user.

Unsigned applications have limited access to system resources.

Usually signed applications have the same access rights as unsigned applications. An application broadcaster can request additional permissions to access specific resources by providing a signed "Permission Request File" along with the application. The example syntax of the Permission Request File is defined in the Java security section. The permission request file may also contain a credential that indicates that a persistent file owned by another organization may be accessed. If the "Permission
Request File" is not correctly authenticated the application is not granted any
additional permissions but is not prevented from starting for this reason.
The way the user grants rights to the downloaded applications is implementation
dependant and is not discussed by the present document.

For DVB-J applications, accessing a resource consists of method calls. Each method
call that results in accessing the resource shall throw a security exception. For each
resource subject to access restriction, the application can test whether it has been
granted permissions to access it by using the corresponding java Permission class.
For a DVB-J application to be correctly authenticated, all the class files that the
application consists of need to be signed, the signatures need to verify and the
application_id needs to be from within the range allocated to signed applications. If,
during the loading or execution of the application the MHP detects a signed file
containing a class that failed to pass the authentication process (e.g. because its actual
hash value does not match the expected hash value), then the class shall be considered
as not available.

In order to be efficient, if a directory contains objects that are likely to frequently
change, it is advised to put a signature file in this directory and to mark the directory
as non authenticated in the hash file located in the parent directory of. By doing so, it
will limit the propagation of modifications to just one directory. [5]

The authentication of a file is evaluated each time that the file is loaded from a
transport connection. File version information in the transport system cannot be
assumed to be secure.

Applications authors should be aware that deciding whether to grant a permission or
not may, depending on the implementation, involve prompting and asking the end
user. The latest point in time when the implementation must decide if an application
has a permission or not is when the application either queries the presence of this
permission for the first time or when it invokes an action that requires the permission
for the first time. Application authors should be aware that in these situations, an
implementation may prompt and ask the user. Depending on the implementation, this
prompting (if necessary) can also happen at any point in time prior to this (e.g. at the
application start up time).

An MHP terminal is required to be able to operate in a mode where it grants
permission to provide access to all of the functionality required by the profiles and
options that it supports when appropriately requested (e.g. via the permission request
file). The mechanism for causing the terminal to operate in this mode is
implementation-dependent. The granting of permissions for accessing functionality
outside of the claimed MHP profile and options is not required.

6.2.1 File access

Unsigned applications have no access to the persistent storage.
Signed applications by default also cannot access the persistent storage unless
otherwise indicated in the permission request file.
6.2.2 Application lifecycle control policy

An unsigned broadcast application can launch any application visible in the listing API that is signaled in the currently selected service for the service context in which the application wishing to do the launching is running. An unsigned application can control (pause, stop, resume) the lifecycle of an application it has launched. An unsigned application cannot control the lifecycle of an application it has not launched.
By default signed application has the same rights as unsigned concerning the application lifecycle control policy although those rights could be changed by the permission request file.

6.2.3 Return channel access policy

An unsigned application must not use the return channel. Signed applications also cannot access the return channel unless otherwise specified in the permission request file.

6.2.4 Tuning access policy

An unsigned application may not tune using the Tuning API. By default, a signed application may not tune using the Tuning API. However, the right to tune can be requested with the Tuning permission that can be put in the permission request file.

6.2.5 Service selection policy

Unsigned applications cannot select a new service. By default, signed applications can select any new service, unless otherwise specified in the permission request file.

6.2.6 User setting and preferences access policy

Unsigned applications have read access to user language, parental rating, default font size and country code. It cannot read or write other preferences. By default, same as unsigned applications. The permission request file may include items that request read access to all user preferences and/or write access to all user preferences.

6.2.7 Network permissions

Unsigned applications can not have access to the return channel and therefore can not access remote network hosts. For signed applications, the permission request file can contain a set of permissions that specify the hosts and actions for which permissions are requested.
6.2.8 Privileged runtime code extension permission

Unsigned applications shall be permitted to do runtime code extension with string data from any source. Signed applications with no permission request file shall be permitted to do runtime code extension with string data from any source. By default, a signed application with a permission request file is prohibited from doing any runtime code extension from a string. The permission request file may include items that request the ability to do runtime code extension with internal strings only, or with both internal and external strings.

6.2.9 Application storage

Unsigned applications have no permissions to store applications. By default signed applications do not have permission to store any applications. Permission to store applications can be requested using the permission request file.

6.2.10 Smart card access

Unsigned applications cannot access smart cards. By default signed applications do not have permission to access smart cards. Permission to access smart cards can be requested using the permission request file.

6.2.11 Provider management

Unsigned applications have no permissions to install or remove providers. By default signed applications have the same rights as unsigned applications however this could be changed in the permission request file. An application shall only be granted permissions to insert or to remove providers that have the same organization identifier as the application.

6.2.12 Privileged application access

Unsigned applications shall not be able to access any privileged APIs. Only applications successfully authenticated as “privileged applications” shall be able to access privileged APIs.

6.2.13 Credentials

In addition to permissions, the permission request file may also request credentials. A credential is a right owned by a third entity and because of that has to be certified by this entity. Credentials are therefore necessary for example if an application wants to access data stored in the persistent storage that is owned by another application.
6.3 Security of the return channel

6.3.1 Basic information

To use the return channel, the interactive broadcast and internet access profile need to be supported by the terminal. The quality (speed) of the return channel depends on the type of return channel offered by the terminal and potentially also on the type of subscription for internet access a user has.

The return or interaction channel. It adds the real remote interactivity to the whole system. The return channel can be analysed based on the available technologies and based on how applications use the return channel.

An MHP application typically uses a return channel for having a bi-directional IP link between the MHP receiver and application servers. These servers are application specific and can have an interface to the billing system, which is operator specific.

With a PSTN modem it is also possible to make drop calls. This can be used by very basic applications, like voting, where only one-way interactivity is needed. In that situation, the MHP receiver just calls a number. The semantics of the user’s response is bound to the dialed phone number itself. On the server side in order to collect users’ responses, it is sufficient to answer the phone call and, immediately after, drop the call without setting up an IP connection. [4]

As was mentioned before neither unsigned nor signed applications (unless mentioned otherwise in the permission request file for the signed ones) have no access to the return channel.

When implementing the return channel security MHP should:
- Implement the cipher suites for TLS which for MHP (minimum requirements) are as follows:
  1. RSA
  2. MD5
  3. SHA-1
  4. DES
  5. AES
The MHP however is not required to implement:
- The functionality of being a server for the TLS protocol.
- Compliance with the SSL 3.0 [5]

6.3.2 TLS description and security

6.3.2.1 Description and operating principles
Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols which provide secure communications on the Internet for such things as web browsing, e-mail, Internet faxing, instant messaging and other data transfers. There are slight differences between SSL 3.0 and TLS 1.0, but the protocol remains mainly the same.
[9]
The TLS protocol allows applications to communicate across a network in a way designed to prevent eavesdropping, tampering, and message forgery. TLS provides endpoint authentication and communications privacy over the Internet using cryptography. Typically, only the server is authenticated (i.e., its identity is ensured) while the client remains unauthenticated; this means that the end user (whether an individual or an application, such as a Web browser) can be sure with whom they are communicating. The next level of security—in which both ends of the "conversation" are sure with whom they are communicating—is known as mutual authentication. Mutual authentication requires public key infrastructure (PKI) deployment to clients which model in MHP will be discussed later.

TLS involves three basic phases:

1. Peer negotiation for algorithm support
2. Public key exchange and certificate-based authentication
3. Symmetric cipher encryption [9]

A TLS client and server negotiate a stateful connection by using a handshaking procedure. During this handshake, the client and server agree on various parameters used to establish the connection's security.

- The handshake begins when a client connects to a TLS-enabled server requesting a secure connection, and presents a list of ciphers and hash functions.
- From this list, the server picks the strongest cipher and hash function that it also supports and notifies the client of the decision.
The server sends back its identification in the form of a digital certificate. The certificate usually contains the server name, the trusted certificate authority (CA), and the server's public encryption key.

The client may contact the server that issued the certificate (the trusted CA as above) and confirm that the certificate is authentic before proceeding.

- In order to generate the session keys used for the secure connection, the client encrypts a random number with the server's public key, and sends the result to the server. Only the server can decrypt it (with its private key): this is the one fact that makes the keys hidden from third parties, since only the server and the client have access to this data.

- Both parties generate key material for encryption and decryption.

This concludes the handshake and begins the secured connection, which is encrypted and decrypted with the key material until the connection closes.

If any one of the above steps fails, the TLS handshake fails, and the connection is not created. The handshake mechanism is displayed in Figure 11.
6.3.2.2 Security

TLS/SSL have a variety of security measures:

- The client may use the CA's public key to validate the CA's digital signature on the server certificate. If the digital signature can be verified, the client accepts the server certificate as a valid certificate issued by a trusted CA.
- The client verifies that the issuing Certificate Authority (CA) is on its list of trusted CAs.
- The client checks the server's certificate validity period. The authentication process stops if the current date and time fall outside of the validity period.
To protect against Man-in-the-Middle attacks, the client compares the actual DNS name of the server to the DNS name on the certificate.

Protection against a downgrade of the protocol to a previous (less secure) version or a weaker cipher suite.

Numbering all the Application records with a sequence number, and using this sequence number in the MACs.

Using a message digest enhanced with a key (so only a key-holder can check the MAC).

The message that ends the handshake ("Finished") sends a hash of all the exchanged handshake messages seen by both parties.

The pseudorandom function splits the input data in half and processes each one with a different hashing algorithm (MD5 and SHA-1), then XORs them together. This provides protection if one of these algorithms is found to be vulnerable.

SSL v3 improved upon SSL v2 by adding SHA-1 based ciphers, and support for certificate authentication. Additional improvements in SSL v3 include better handshake protocol flow and increased resistance to man-in-the-middle attacks.

[9]

6.3.2.3 Downloading of certificates for TLS in MHP

Before the TLS connection can be established, the MHP has to ensure that the certificate list sent by a server contains at least one trusted certificate. In computer environment, this is simply done by checking the list of certificates against one certificate that is resident in the computer. In the MHP environment, a downloadable application can establish a TLS session. This can be used for e.g. sensitive transactions. In such a scenario, the application knows which server to connect to, and also knows one certificate against which it can check that a given certificate chain contains the expected certificate that it knows and trusts.

6.4 DVB MHP public key infrastructure

Public key infrastructure - A system that enables users of a public network to exchange data securely and privately through the use of a public and private cryptographic key pair that is obtained and shared through a trusted authority. Figure 12 shows the MHP public key infrastructure hierarchy.
Actors of the MHP PKI and their roles:

**DVB services SARL** - implements the DVB-MHP PKI, issues certificates and keys in particular root certificates, provides the CRL and RCMM management messages.

**Certification authority (CA)** - creates, issues and manages the certificates. Root CAs are part of the DVB-MHP PKI.

**Application developer** – (DVB-MHP PKI subscriber). Application developers can sign their applications using the key and certificates obtained at DVB Services.

**Broadcaster** – (DVB-MHP PKI subscriber). A broadcaster can sign the applications delivered over its network using the key and certificates obtained with DVB Services. It broadcasts management messages (CRL and RCMM) provided by DVB Services.

**MHP terminal vendor** – (DVB-MHP PKI Root Embedding). The vendor embeds the set of root certificates, provided by DVB Services, in the MHP terminal. The vendor processes the management messages. [4]
6.4.1 DVB MHP PKI for MHP terminal Manufacturers

Manufacturers need to install a set of root certificates into their terminals in order provide them with the possibility to authenticate the certificate hierarchy files that are delivered with applications. Without these root certificates the receiver is unable to run signed applications.

Currently there are three certificates that need to be installed, namely:

- An active root certificate currently used to authenticate certificates.
- A substitute root certificate that could be used to authenticate certificates at a future date.
- A special root certificate used to authenticate changes to the set of root certificates stored in the receiver.

The process of updating these root certificates is triggered by information delivered in a broadcast stream and the manufacturer should not need to be involved in the update process.

Manufacturers need to obtain the set of root certificates from DVB Services SARL by entering into the DVB MHP PKI Root Embedding Agreement. [8]

6.4.2 DVB MHP PKI for Application Developers

Application developers need to sign applications that access restricted resources in MHP terminals or which need to verify the integrity of the application file tree for any other reason. The signing process for an application requires the use of an asymmetric key pair (also known as a public/private key pair) where the private key is securely managed under the control of the authenticating end-entity. The DVB MHP PKI uses cryptographic hardware tokens (usually a USB device or a SmartCard) to protect the private key and these keys are normally generated by DVB Services SARL as part of the certificate subscription process. This certificate subscription process also associates the identity of the private key owner with a certificate that contains the corresponding public key part.

Once application developers have received their private key token and associated certificate, they can use these in conjunction with an appropriate application-signing tool to create the additional security files that authenticate their application on the receiver.

Application developers who need to sign applications need to obtain keys and certificates from DVB Services SARL by entering into the DVB MHP PKI Certificate Subscriber Agreement. As part of this subscription process, the application developers need to provide proof of their identity and of their right to broadcast applications over their chosen network(s). The subscription agreement also requires the key owner to take appropriate precautions to ensure the security of their keys and to inform DVB Services SARL if the key is lost, stolen or otherwise compromised.

Note that only organizations that have been assigned MHP Organization IDs may apply for subscription to DVB MHP PKI. [8]
6.4.3 DVB MHP PKI for Broadcasters

Broadcasters take responsibility for the content that they deliver over their network and, as such, may decide to take responsibility for signing applications that they broadcast. In this case a broadcaster may need a key pair and associated certificate of their own to sign applications in the same manner as described for an application developer.

Broadcasters also have a responsibility to include management messages to MHP receivers with each application that is delivered. These management messages include lists of certificates that are considered invalid (Certificate Revocation Lists) and changes to the root certificates stored in receivers (Root Certificate Management Messages). The files that contain these messages are available to broadcasters from DVB Services SARL. [8]

6.4.4 Certificate management

X.509 (MHP uses X.509 certificate standard) certificates may be revoked by the issuing certificate authority prior to their regular expiration date. Possible reasons for that are that a broadcaster’s private key has been compromised or a broadcaster stops using a certificate authority. Each certificate authority publishes a list of revoked certificates, called a Certificate Revocation List (CRL).

The CRLs are transmitted to the MHP terminals usually within the object carousel. The possibility of using the interaction channel for the distribution of the CRL files is also given but is in most cases not favorable due to the limited availability of return channels in the user terminals.

When authenticating an application, MHP-terminals have to inspect the set of CRL files periodically and cache the revocation information for future use. During the validation process of a certificate chain, the CRL of each certificate authority on the certification path is checked.

Root certificate management is another important issue in MHP security framework. Every compliant MHP terminal will have to maintain a set of X.509 root certificates in persistent storage. When the root certificates are updated, a standard mechanism using messages called Root Certificate Management Messages (RCMM) is used to finish this task. RCMM is authenticated by multiple signatures. An RCMM message will be accepted by an MHP terminal if and only if it has at least N signatures. The N is 2-12 currently, and the initial value is expected to be 2. The use of multiple signatures guarantees that the set of root certificates can be updated securely even if one of the root certificates has been compromised.

6.5 Application information table

The Application Information Table (AIT) plays also an important role in MHP security mechanism. The AIT provides full information on the data broadcast, the required state of the application carried by it.
Data in the AIT allows the broadcaster to request that the receiver change the
activation state of an application. If errors found in AITs, they will be processed
accordingly. Processing of the application or the AIT will continue. Applications shall
not launch broadcast applications that are not signaled in the AIT of the same service.
[5]

6.6 MHP minimum security requirements

MHP platform hardware is required to support the following minimum sizes to
support the MHP security model:
• A value of 2 for the number of signatures in RCMM
• A minimum depth of 5 levels in the certificate chain.
• A minimum of 8 CRLs.
• A minimum of 10 entries per CRL.
• A minimum of 3 root certificates (regardless of the number of underlying root
certificate authorities).
• A minimum of 3 root certificates for testing.
[5]
7 DVB-J security

As a part of the security model of the entire MHP the brief description of the security of the DVB-J should be considered.

7.1 Why Java?

The MHP platform and its programming interfaces are based on Java, thus the developer must take Java’s security features into account as well as its restrictions. Java is object-oriented programming language and from the beginning was designed for the development of networked applications and to provide a secure way of downloading applications over insecure networks. Nonetheless, this has not fully come true in Java implementations; vulnerabilities that can cause applications to break its security policy come up from time to time. The main components of the Java are byte code and virtual machine, where the code is executed. Java’s virtual machine hides the operating system and hardware from the Java application, so it makes sense to talk about Java as a platform rather than a programming language. For networked applications, Java is a better option than the traditional platforms based on C-programming language because Java’s inbuilt security model has been developed since the first version of Java. Java’s application programming interfaces provide support for cryptographic algorithms and public key infrastructure, including a certificate-based X.509 authentication framework that is also utilized in the MHP platform to ensure the origin of the application.

DVB has adopted the Java programming language for MHP interactive applications and has created a lightweight version suitable for broadcast applications called DVB-J or DVB-Java. Consensus was reached by all partners involved in DVB to adopt Java as language for MHP interactive applications due to its maturity, quality of tools and development skills available.

Whereas MHP is based on a subset of Personal Java 1.2, several major elements were removed; some to save space, others because their functionality is not needed in the TV context. Below the main changes in MHP are listed:

- Several major parts have been added in MHP, such as additional APIs for STB-specific functions (DVB MHP API).
- Where the functionality needed was too different from standard Java the code was modified.
- The UI model reflects the consumer, TV-centric model rather than the PC/workstation model.
- Changes in the core Java classes were made to save memory space.
7.2 Java’s ‘Sandbox’ principle

Figure 4 shows Java’s security model, which provides variable rights depending on the application’s security policy and origin. Based on these, trust in the application is evaluated and how much of the application gets rights in the system. At one end is an application with 100% trust, whose execution is not limited in any way, and at the other end is a fully untrusted application, whose execution is not permitted at all. Java uses the term ‘sandbox’ when speaking of restricting an application’s execution rights in the system. It is possible to define different kinds of sandboxes for different kinds of service developers (device manufacturer, operator, third party).

Figure 13: Java information security model [7]

7.3 Java permission request file

Java’s security policy is implemented using a permission request file. As was mentioned before MHP uses both Java’s own security policies and policies defined by MHP. Implementation of these varies from one receiver to another. The MHP standard has intentionally left undefined where authorization requests are handled by the user. In other words, this leaves the definition of a security policy to the market, the consumers and the regulating authorities.
Example of the permission request file:

```xml
<?xml version="1.0"?>
<!DOCTYPE permissionrequestfile PUBLIC "-//DVB//DTD Permission Request File 1.0//EN" "http://www.dvb.org/mhp/dtd/permissionrequestfile-1-0.dtd">

<permissionrequestfile
  orgid="0x000023d2"
  appid="0x0020">

  <file value="true"></file>

  <capermission>
    <caysystemid
      id="0x1111"
      messagepassing="true"
      entitlementquery="true"
      mmi="false">
    </caysystemid>
  </capermission>

  <applifecyclecontrol
    value="true">
  </applifecyclecontrol>

  <returnchannel>
    <defaultisp></defaultisp>
    <phonenumber>+3583111111</phonenumber>
    <phonenumber>+3583111112</phonenumber>
    <phonenumber></phonenumber>
  </returnchannel>

  <tuning value="false"></tuning>
  <servicesel value="true"></servicesel>
  <userpreferences
    read="true"
    write="false">
  </userpreferences>

  <network>
    <host action="connect">hostname</host>
  </network>

  <persistentfilecredential>
    <grantoridentifier id="0x02020300">
    </grantoridentifier>
    <expirationdate date="24/12/2032">
    </expirationdate>
    <filename read="true" write="false">
      5/15/dirl/scores
    </filename>
    <filename read="true" write="false">
      5/15/dirl/names
    </filename>
    <signature>
      0232032932932921493143929423943294239432
    </signature>
    <certchainfileid>3</certchainfileid>
  </persistentfilecredential>
```
The fact that an application requests these permissions is no guarantee that it will get them, however - the user must explicitly grant an application these permissions. Exactly how this is done will vary from receiver to receiver - some may set these values as part of the user preferences, some may ask the user every time, some may take yet another approach. In every case, however, the application should always be aware that it may only get some of the permissions that it requested.
8 MHP security analysis based on the security criterias

The authentication tree in MHP gives strong protection of the consumer and service provider. The MHP security model divides applications to signed application and unsigned application. Unsigned application has restricted access rights to resources and services. The hash codes in MHP ensure the data integrity and confidentiality, because hash codes are computed with the content and attributes of the objects. And a hash file indicating objects to be authenticated will be put in each directory containing such objects. This mechanism protects that only authenticated application can modify related objects. Moreover, an authenticated application could not modify the objects that not listed in the hash file it used to authenticate itself. Under strict authentication, the MHP resources could be accessed only by the authorized users.

The non-reputability is mostly ensured through the signatures. Digital signature in MHP is stored in a signature file located in the root directory of the sub tree that it signs. By generating this signature, a certification authority certifies the validity of the information, especially the binding between the public key material and the subject of the certificate. Since applications have to sign a digital signature, and this signature is stored in a file including the public key, the hash algorithm and the computed signature values. Any user that has signed for an application could not deny that he has done such operations. This protects against the non-repudiation. From other point of view, non-reputability exists also in the opposite direction. The service provider and broadcaster could not deny any kind of operations they have done to the consumer, especially when such operations may cause consumer’s STBs damaged.

The availability should be guaranteed by all the participants: consumers, service providers and broadcasters. The strict authentication rules can promise that unauthorized person will not misuse a user’s private information in the normal way. But the privacy could not be promised by the person who can access the information. For example, the network administrator in the service providers’ company can access many types of consumers’ private information like their PIN code, their credit card number. If such person misused these information, it is difficult to detect and prevent by the MHP security mechanism. The AIT table in some points provides the availability protection. The applications could not be launched without an AIT definition. This ensures that only signed application can access the MHP resources.

From another point, since Secure Socket Layer (SSL) is used in return channel, the detection of sensitive information on the transferring becomes difficult. The privacy is protected very well without considering personnel factor.

The availability needs co-operation of the service provider and broadcaster. The broadcaster must keep the channel working appropriate at any situation. The transmission rate should be kept stable and the quality should be promised. When
consider return channel, this issue becomes more complicated because the user can access resources via return channel.

The certificate file contains all of the certificates in the certificate chain up to, and including the root certificate. A certificate chain is a hierarchy of certificates that enable the implementation to verify the validity of the key used to check a signature. In the MHP environment, the root certificate is embedded in the MHP. So the file structure shall carry all of the certificate chain apart from the root certificate.

In the root directory of an authenticated sub tree there shall be one certificate file for each signature file. This certificate file ensures that the signature is the right one who signed it.

Since Java is used as the development platform in MHP, the security feature like sandbox, which can limit access to system resource can also be used in MHP applications. Actually, Java plays an important role in enforcing strong typing and preventing application from executing common attacks such as stack overflows.

Finally, the security requirements for the three profiles are certainly not identical. The more powerful service needs more security guards. The enhanced broadcast profile focuses more on broadcaster’s security issue. While keep at least the same level of enhanced broadcast on broadcaster, the Interactive broadcast must keep an eye on the return channel. The Internet Access should give more power on the return channel security issues than the other two profiles. At the same time, it should keep the same power on security issue of broadcaster and service provider as the two other profiles.
9 Conclusion

The object of the project was to analyze the security of the MHP platform for the set top boxes. The security framework was analyzed based on the data privacy, availability, non-reputability, integrity and confidentiality.

From this paper we can see that MHP provides quite strong security mechanism. It could authenticate incoming applications that are from valid sources, verify that the user is appropriately authorized to use the application. The application integrity can be verified. An application’s access to private data can be controlled. When one task is finished, the application can terminate and resources are correctly released.

In summary, the security model in MHP gives an efficient protection of the users’ data from technical point. The broadcasting program is protected well by the MHP security model. From the ETSI standard, the security mechanism is one of the essential parts in the MHP, especially for the broadcaster. The privacy should be the main point of the consumer. The data integrity and confidentiality are concerned by the service provider. The non-reputability should be cared by all the users in MHP.

The availability is related to capacity of the broadcast channel and IP network routes. This paper has not discussed it in detail.

Current paper has only analyzed the existing security mechanism. Due to the lack of the qualification I cannot give the improvement advises for the mechanism. The main focus in the future should be on the return channel security as nowadays more and more interactive services become popular and available. The reliable anti-piracy mechanism should be developed due to the increasing popularity of VODs and other online media sources.
10 References

[3] Internet protocol Set Top Boxes, Microsoft Corporation, May 2004, Olivier Fontana, Product Manager Mobile and Embedded Devices Group (MED)
11 Appendixes

11.1 MHP end-to-end reference model

This figure presents MHP system end-to-end model. It shows the components that contribute to architecture and infrastructure of the MHP end-to-end chain.