CALYPSO FUNCTIONAL SPECIFICATION

Card Application

AUTHOR

EDITOR

REVISION LIST

Version	*Date*	*Modifications*	*Author*
1.5	140403	Presentation of Calypso revisions. Modified according to Calypso Rev.3.2. AES algorithm. Linked applications: Stored Value, file sharing. References and glossary updated and moved to the end of the document. Minor improvements and corrections.	SDI
1.4	101208	Modified according to Calypso Rev.3. Updated references. Improved key identification description. Improved access modes description. Updated the session modification command list. Editorial improvements and corrections.	JCR/SDI
1.3	051025	Cryptography and files contents sections updated, and several minor modifications.	SDI
1.2	031124	Removed references to unnecessary documents.	FLV
1.1	010628	Minor corrections.	FLV
1.0	010608	First release	FLV

TABLE OF CONTENTS

1 OVERVIEW 4
1.1 Context 4
1.2 Calypso 4
1.3 Calypso Functional Specification: Card Application 5

2 THE CALYPSO SYSTEM 6
2.1 General Features 6
2.2 Card Characteristics 7

3 CARD DATA ORGANIZATION 9
3.1 Files 9
3.2 Calypso Application 9
3.3 Data Addressing 9

4 CALYPSO FUNCTIONALITY 11
4.1 Introduction to the Card Data Management Principles 11
4.2 Main Calypso Card Commands 12

5 EXAMPLE OF DEBIT TRANSACTION 13
5.1 General Description 13
5.2 Debit Transaction 13

6 SECURITY MECHANISMS 17
6.1 Introduction to the Calypso Security 17
6.2 Secret Keys 17
6.2.1 Types of Secret Keys 17
6.2.2 Key Identifier (KIF and KVC) 17
6.2.3 Cryptographic Algorithms 18
6.2.4 Security Modules (SAM) 18
6.3 Access Conditions 19
6.4 Session Description 19
6.4.1 Secure Session Security 19
6.4.2 Ratification 20
6.5 Memory Modification Management 21

7 INTEGRATION OF CALYPSO WITH ANOTHER APPLICATION 22
7.1 Presentation 22
7.2 Stored Value Integration Principle 22
7.3 File Sharing Principle 22

8 FILES CONTENT 24
8.1 Master File 24
8.2 Calypso Application 24

9 APPENDICES 26
9.1 Normative References 26
9.2 Glossary 27

1 OVERVIEW

1.1 Context

Today, the passenger is the center of attention of the public transport operators. The objectives of ensuring a good reception, offering more comfort and security and offering personalized services, all suppose a detailed knowledge of the clients to ensure a high quality of service.

The ticketing systems are also involved in this priority given to the customers: it is not enough to provide them with access to the transportation network, it must also allow building a new relationship with them.

For any operator interested by this global customer concern, Calypso brings a solution based upon the most recent contactless technologies.

However, Calypso is not limited to a simple technical evolution replacing the ticket by a contactless portable object (card, NFC phone, etc.): designed by the transport operators, for the transport operators, Calypso offers a solution open, modular and complete.

Note: a Calypso portable object is called a *card* in the present specification, whatever its physical format.

1.2 Calypso

Since 1990, a group of transport operators led by *Innovatron*, *RATP* and *SNCF* have set the goal of developing a non-proprietary technology for a secure, fast and flexible public transport ticketing system.

The ensuing fifteen years development program succeeded in creating the smartcard contactless technology adapted to the public transportation uses. The technology was made accessible to all industrial companies on a fair and non-discriminatory basis, to ensure:

- The birth of this new technology, suited to the public transport needs.
- A good products compatibility.
- A fair market concurrence.

Through the *Icare* and *Calypso* European projects, associating *Brussels* in Belgium, *Lisbon* in Portugal, *Konstanz* in Germany, *Paris* in France and *Venice* in Italy, it was ensured that the technology could be adapted to many public transport environments, and could be married to other services, such as electronic purse, fidelity, access control, etc.

These efforts resulted in the development of the CD97 family of contactless smartcards and their applications. It also leaded to the ISO 14443 type B standard, as well as many advances in the EN 1545 standard.

To better identify the technology, abstracted from a particular product, this “CD97” technology was renamed *Calypso* during the year 2000.

Following the large success of the Calypso technology and its widespread use in many parts of the world, during the year 2003, *Calypso Networks Association* (CNA) was created to let the users of the technology manage its evolutions.

More information may be found on the CNA web site: http://www.calypsonet-asso.org.

1.3 Calypso Functional Specification: Card Application

This document contains a functional description of the Calypso card application.

Calypso contactless portable objects comprise a secure microchip (smartcard) which hosts one or several applications complying with the Calypso specification, designed for payment or access to services such as public transport. They are used as part of a secure contactless system including a central system, reloading equipment, validators, and possible other equipment:

- The central system allows keeping track of the transactions, making statistics and verifying the system security and integrity.
- The reloading equipment loads tickets (one-way tickets, season tickets, etc) and value into the cards.
- The validators are used to validate entrance in (and optionally exit from) the network system.
- Possibly: some hand-held controlling equipment, personalization machines, etc..

All parts of the system must be designed with some of the major goals of a secure contactless system in mind, particularly the speed and security of transactions. The present manual therefore contains some guidelines on the card uses, some typical transactions descriptions and guidelines on the overall system design.

Calypso Revisions

The Calypso standard has evolved to include advances in security and standardization, and to improve flexibility and performances within interoperable networks:

Revision	*Issued*	*Main Evolutions*
3.2	2013	***Definition of features optional and additional to Revision 3.1***: AES cryptography Session with encryption, prior authentication, extended signatures
3.1	2006	***Improvement of compatibility between portable objects***: Full ISO/IEC 14443 (B or A) protocol TDES and DESX cryptography[1] Binary files, shared files, Stored Value application, PIN, improved key management, etc. Java Card compatibility improvements
2	2001	***Standard specification abstracted from specific products***: Full ISO/IEC 14443 type B protocol DESX cryptography[1] Several functional improvements: application selection, key version management, etc.
1	1997	***World first open specification for public transport ticketing***: ISO 7816 compliance Secure session with ratification (invented and patented by Calypso) Innovatron protocol (used as the basis for ISO/IEC 14443B)

1 Simple DES is still available for compatibility with existing systems (should not be used for new projects).

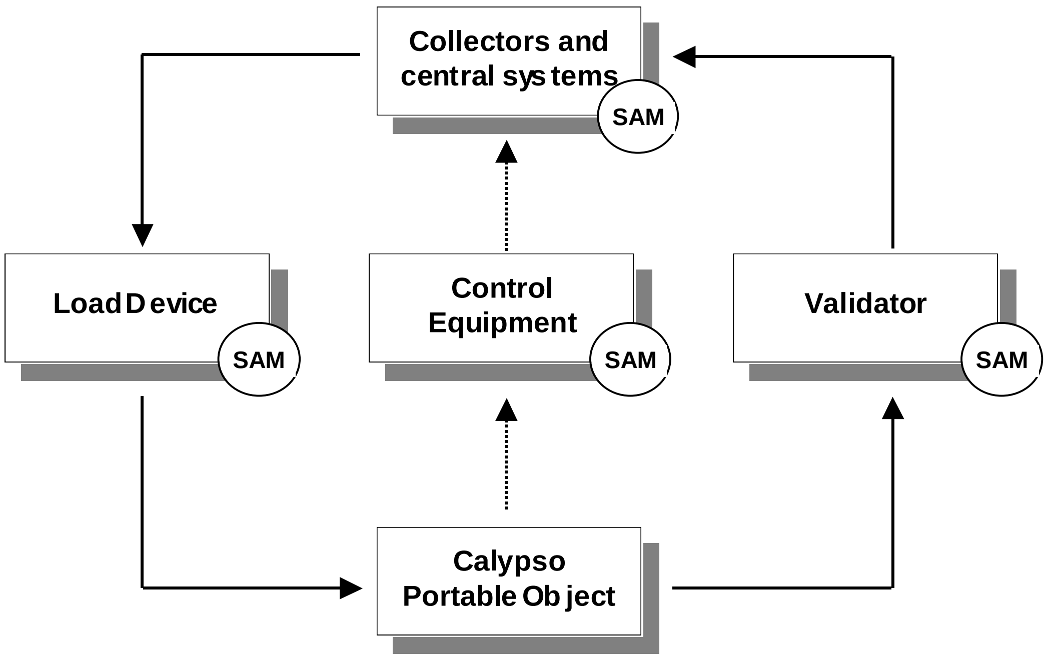
2 THE CALYPSO SYSTEM

2.1 General Features

A Calypso portable object ("card") is part of an information system architecture, allowing transfer and verification of transport rights. The card may contain other information, for example about the card holder, the last uses of the card, etc.

As the transport rights represent a monetary value, their utilization is protected by cryptographic algorithms, preventing defrauders from forging a false card, or reloading a card without proper authorization.

The cryptographic algorithms are based on secret keys, hidden in the cards and in the Secure Application Modules (SAM). A SAM is a smartcard present permanently in the equipment interacting with the cards (or remotely connected with the equipment).



After manufacturing, the card is personalized by writing in it: the secret keys and the application data.

During its life, the card will typically:

- Be loaded with new information, transport rights, transport tokens.
- Be used at a validator to access into the transport network or to gain other value, by showing the transport rights in the card, or debiting the transport tokens.
- Be controlled to check its information, or to display it to the user.

This manual will illustrate some typical transactions, and describe in detail the card data organization and commands.

A Calypso card may be used for other application than public transportation, and may have features extending its capabilities beyond the Calypso specification. This is outside the scope of the present document.

2.2 Card Characteristics

A Calypso portable object (“card”) includes a smartcard and contactless communication means allowing fast and secure transactions, as for example in public transport applications**.** The Calypso card may also contain other kind of applications and interfaces.

A Calypso portable object complies with ISO 14443 and includes a microprocessor based smartcard that:

- Complies with ISO 7816-3 if embedded in a smartcard format.
- Complies with ISO 7816-4.
- Allows coding of the EN 1545 transport data structures.
- Ensures a high security of transactions.
- Ensures fast contactless transactions.

Its internal structure can be described with layers, as follows:

Application Data (e.g. EN 1545)
Calypso Application Dedicated data management mechanisms (Anti-tearing, access control, file types, commands)
ISO 7816-4 Generic files types and commands
ISO 14443 Contactless communication

ISO 7816 and ISO 14443 Compliance

A Calypso card package may comply with the ISO 7810, ISO 7816-1 and ISO 7816-2 standards, or may be packaged differently: embedded in a wristwatch, in a paper ticket, in a mobile phone, etc. It always includes a radio antenna allowing contactless communication.

When packaged as an ISO 7816 card, it works in contact operations at 5V, according to the ISO 7816-3 T=0 standard. It may also support other operating voltages (e.g. 3V) and other protocols (e.g. T=1, I2C…).

A Calypso card works in contactless operations, at 13.56 MHz, according to the ISO 14443 parts 1, 2, 3 and 4. It may include other contactless protocols.

The card data is organized in files, according to the ISO 7816-4 standard.

EN 1545 Transport Data Structures

A Calypso card may be used for various applications.

For public transport applications, the card data is able to encode the EN 1545 transport data structures, if necessary.

The card never analyzes this data. The data may thus be coded differently for other applications. It is entirely up to the terminals using the cards (validators, reloading machines, etc.) to decide the data layout written in the card.

The Calypso application file structure typically allows storing at least the following information:

- Application and holder information,
- 4 or 8 contracts,
- 4 to 9 counters (0 to 16.777.215 units each),
- Log of at least the 3 latest events,
- Other applicative data,
- Card manufacturing information and secret keys.

High Security

Calypso cards ensure integrity of the whole transaction: in case of interruption of a transaction containing several data modification commands, the card ensure that all modifications are either cancelled or applied.

Regarding data access control, Calypso allows algorithms offering high security levels against false card manufacturing or unauthorized card reloading, as follows[2]:

- AES, with 128 bits secret keys.
- DESX (associated with a specific hash algorithm), with 120 bits secret keys.
- Triple-DES, with 112 bits secret keys.

2 The simple DES algorithm (56 bits secret keys) is obsolete and not recommended for deployment.

3 CARD DATA ORGANIZATION

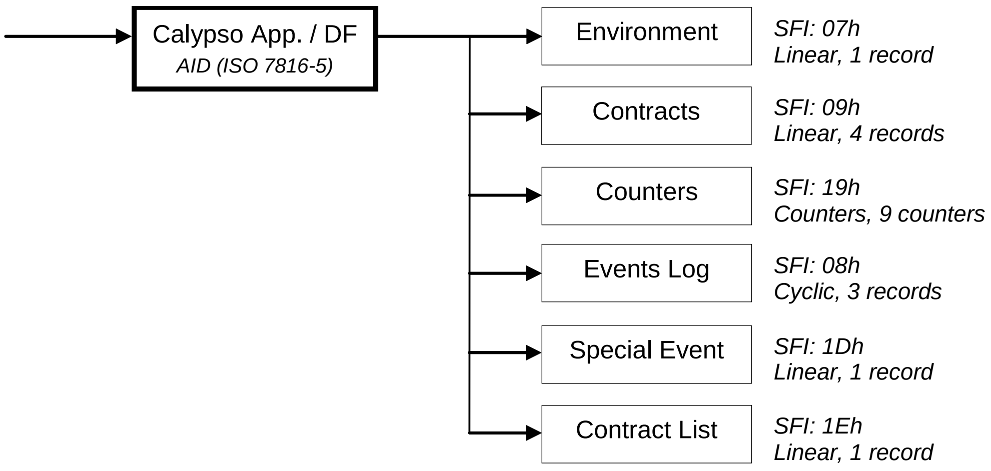
3.1 Files

The card data is organized in hierarchical files, and complies with ISO/IEC 7816-4.

The Calypso Specification does not define a fixed file structure layout of the card, but it does define a list of registered file structure layouts for Calypso card applications.

3.2 Calypso Application

A typical Calypso application may have the following file structure:



3.3 Data Addressing

There are two main types of files: Dedicated Files (DF, directories of files) and Elementary Files (EF, files containing data).

A DF is directory that may contain Elementary Files and other Dedicated Files. The root DF of a card is called the Master File (MF).

An EF is a file containing data: user data organized in linear or cyclic records, counters, keys, etc. The linear, cyclic and counter files are described in more details in the next sections.

All files are identified externally by their *Long Identifier* (LID). Additionally, most EFs are also identified by their *Short File Identifier* (SFI).

All Calypso applications may also be accessed by their name (*Application Identifier*, AID).

At any moment, one file of the card is the currently selected file, or the **current file**. This is the default file in which operations may occur, if not specified otherwise.

The selection of a file, to make it the current file, may be done directly by the Select command, or indirectly by using a command referencing this file. For example, the Read Record command (allowing reading data from a file), may be done on the currently selected file, or on a file whose SFI is specified.

Inside a file, the data is organized in **records** [4]. A file may have 1, or many records. All records exist after initialization of the application. A record that has never been written contains only zeroes (00h).

4 Calypso applications may also contain “transparent” files, not described in this functional presentation.

Access to the data, for reading or modifying, is subject to access rights described in the *Security Mechanism* chapter.

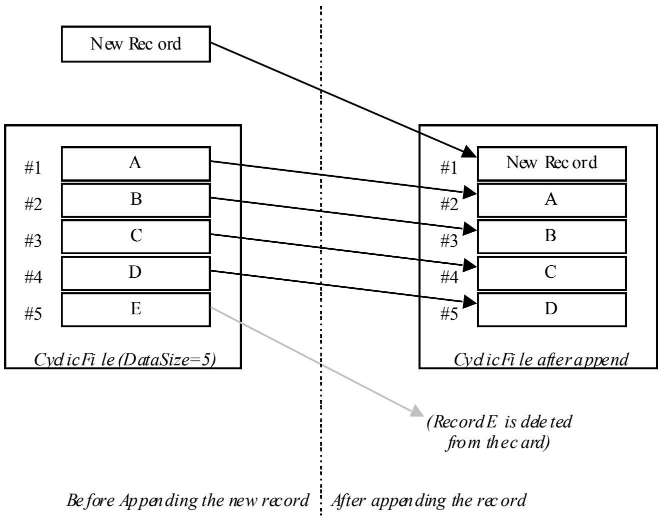
Linear Files

The records in a linear file are organized in sequence, from record #1 to record #*N* (the number of records in the file).

Any record may be accessed directly for reading of for modification.

Cyclic Files

The records in a cyclic file are organized in a cycle, from the most recent (#1) to the oldest (#*N*). Appending a record to the file makes it the number 1, while the others are renumbered, and the oldest one is removed.


New Record
#1
#2
#3
#4
#5
A
B
C
D
E
Cyclic File (DataSize=5)
New Record
A
B
C
D
Cyclic File after append
(Record E is deleted from the card)
Before Appending the new record
After appending the record

In a cyclic file, it is possible to read any record directly, but the only modifications possible are: appending a new record, and updating or "overwriting" the most recent record (overwriting means turning memory bits to "1").

Counters Files

Counters files are files upon which, in addition to the read and update operations, two counter-specific operations may be done:

- Increase: to add a value to one, or a few, counters.
- Decrease: to subtract a value from one, or a few, counters.

Each Counters file contains several counters, usually nine.

4 CALYPSO FUNCTIONALITY

4.1 Introduction to the Card Data Management Principles

When using a Calypso smartcard, the users may wish:

- To ensure that the card data is genuine. It must not be possible for defrauder to forge the data, or to modify it in the card.
- To ensure the integrity of the data written in the card, even if the card power supply is unexpectedly shut down during a single write operation, or during the synchronous update of related files in the card (for example, recording a new event in the card may be linked with a counter decrement).

Calypso Specification solves these needs by a single mechanism called the **Secure Session**[5].

A secure session begins by a specific command sent to the card to open the session (*Open Secure Session*), and ends by a specific command to close the session (*Close Secure Session*).

During the session, it is possible to read and write data into the card (the access may be restricted on some files by specific conditions, e.g. having presented a PIN code).

When the session closes, all the data exchanged is signed by the card, and by the SAM included in the terminal. This signature simultaneously:

- Proves the authenticity of the terminal to the card (authenticating the terminal),
- Proves the authenticity of the card to the terminal (authenticating the card),
- Certifies that the data exchanged is genuine and has not been tampered with by a defrauder.

Finally, it also proves to the terminal that the card has been correctly updated.

The details of the secure session mechanism, as well as a special feature called the **Ratification**[5], are explained in detail in the chapter *Security Mechanisms*.

5 Note that the Secure Session and the Ratification described in the Calypso Specification are patented. Use of these technologies is subject to the Calypso Application license. Please contact Innovatron (calypso@innovatron.fr) for further information.

4.2 Main Calypso Card Commands

Security related commands:

Open Secure Session	Opens a secure session
Close Secure Session	Closes a secure session

Application access commands:

Select Application	Selects the indicated application
Invalidate	Invalidates the application

Data access commands:

Read Record	Reads a record from a file
Append Record	Append a new record to a cyclic file, erasing the oldest record of the file
Update Record	Writes data in a file record, replacing the existing data
Write Record	Writes data over a file record (OR operation with the given and existing data)
Decrease	Decreases the value of a file counter
Decrease Multiple	Decreases the value of one or more file counters
Increase	Increases the value of a file counter
Increase Multiple	Increases the value of one or more file counters

5 EXAMPLE OF DEBIT TRANSACTION

5.1 General Description

This chapter describes an example of debit transaction with a Calypso card.

The main uses of a card will typically be, for a user:

- Getting a new card.
- Buying transport rights, loaded in the card.
- Using the card to enter (and possibly exit) the transport network (debit transaction).

These actions correspond to different processes applied to the card:

Action	*Name*	*Description*
Getting a new card	Personalization	The card is prepared for use by having its secret keys loaded, as well as other information (discount rates, etc.).
Getting new rights	Reloading	New transport rights (contracts) are loaded in the card, in exchange for payment.
Entering network	Debit	The card content is checked, and a new transaction is recorded in the card, before allowing entrance in the network.

5.2 Debit Transaction

As any Calypso operation, the Debit Transaction may be done in contactless or in contact way.

In contactless mode, it is possible to proceed directly with the example protocol described hereafter, just after the radio link establishment.

<u>Objectives:</u>

The protocol below does a debit transaction at a validator point: a *validator transaction*.

The transaction described here is a theoretical transaction, given as an example of the transport commands workings. It is for a single point transport network entrance (the transaction for an entrance-exit transaction, where the price depends on the exit point, is not described here).

Operations:

- A Calypso application is selected and its serial number is read.
- A secure session is opened.
- The card is read to determine the rights of the card owner (environment data, holder information, latest event, contracts).
- The validator determines if the card has a contract allowing entrance.
- The validator writes the new information in the card, and asks the card for a confirmation.
- Upon confirmation receipt, the green light is issued to let the card holder enter the transport network.

The protocol below does not specify all the operations that would be necessary in a real application, for example:

- Handling of reduction rates.
- Description of the transporter policies (season tickets, token price computations, etc.).
- Details of the transport debit SAM handling.

In the protocol given, a special feature, the *ratification*, is described. It allows handling the case where a validator session is broken at the last step, once the card has recorded the transaction, but the validator has not received a correct confirmation. In this case, entrance is allowed for a few minutes if the card is presented again.

Remarks:

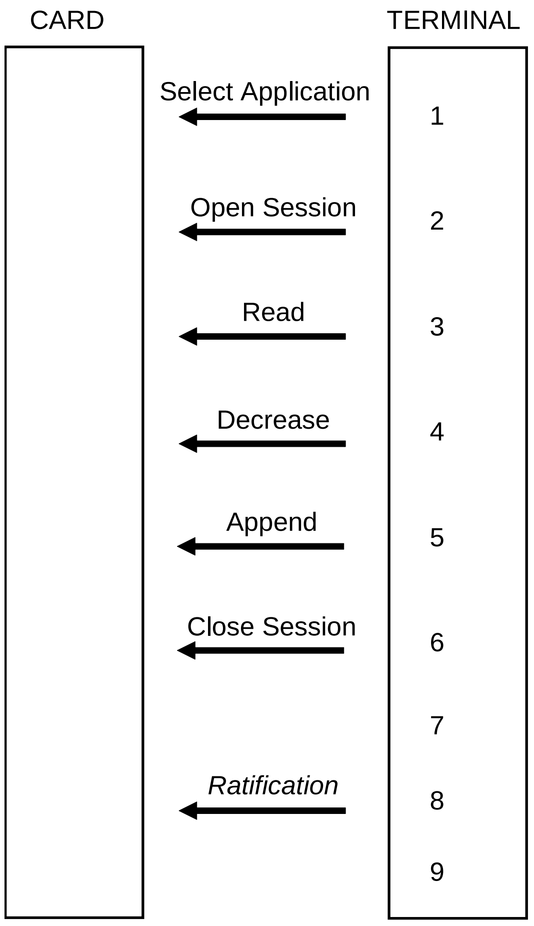
Speed of transaction is the major concern at a contactless validator. The validator operations must be optimized to the maximum in order to achieve the fastest possible transaction.

For example, in order to speed up the validator processing, and to limit the data read from the card, the events recorded in the card may contain the list of valid contracts (contract which are not virgin, nor expired).

Keys used for the example debit:

- Master Debit Key (in the validator SAM).

Protocol: Debit transaction



Protocol description:

1 The validator establishes the radio communication with the card. Then it selects the Calypso application by sending the **Select Application** command to the card with the Calypso application name, and gets back the application serial number with other information.

2 After having requested a validator challenge from the SAM, the terminal sends the **Open Secure Session** command to the card, with the validator challenge returned by the SAM, and requesting reading of the last event.

The validator receives the card challenge, the ratification data and the last transport event (which includes the list of valid contracts).

The terminal requests a session opening from the SAM, with selection of key #3 (Debit Key), and the data sent by the card (to be included in the MAC).

If the last transaction was made less than 7 minutes ago at the same entrance (indicated in the event data), then: if the ratification data indicates that the last transaction was not ratified, the validator continues directly to step 6 with a good status. If the ratification data indicates correct ratification, the validator continues to step 6 with a bad status (in the protocol described here, we refuse a second entrance with the same card before a 7 minutes lapse, but this is an applicative choice).

If one of the contracts of the contract list could authorize access to this entrance, the validator continues to step 3.

Else, the validator continues to step 6 with a bad status. (Other actions would be possible, depending on the network policy. For example, the card might contain a dedicated counter used as a “token purse” that could allow entrance if not empty, etc.)

3 The validator sends the **Read Record** command to the card to read a contract indicated in the contract list, and a **Read Record** command to read the Counters file (to read all the counters in one operation).

The validator receives the contract and the counter values. The validator determines if the contract authorizes entrance, under which conditions (possible decrement of the associated counter). If not, another contract may be read (depending of the available contracts in the contract list).

All the data exchanged are sent to the SAM to be included in the MAC.

If entrance is possible, the validator continues to step 4 if the associated counter must be decreased, or to step 5, if no counter must be decreased.

Else, and continues to step 6 with a bad status.

4 The validator sends a **Decrease** command to the card with the value to subtract.

All the data exchanged are sent to the SAM to be included in the MAC.

5 The validator sends an **Append Record** command to the card to add an event in the Event Log file, filling the parameters with the date and time, location, and other information about the transaction.

All the data exchanged are sent to the SAM to be included in the MAC.

The validator continues to step 6 with a good status.

6 After having sent the command closing the session with the SAM, the terminal sends the **Close Secure Session** command to the card with the four bytes signature returned by the SAM.

The validator receives the card four bytes signature as answer and as soon as possible sends it to the SAM for verification.

7 If the current status indicates a bad transaction (no good contract, an error occurred, etc), or if the SAM indicates that the signature is not correct, the validator will issue a red light.

If the SAM indicates that the signature is correct, then the card was genuine, and all the data exchanged is proved, as well as the card data updating. In this case, the validator will issue a green light.

8 As soon as possible since step 6, the validator sends a command (e.g. incorrect Get Challenge), to make the card ratify the session.

9 The validator issues the green or red light, and begins searching again for a card in the radio field.

Whenever the validator is connected to a host system, the data exchanged during the transactions, including the signatures which prove the debit transactions, are uploaded to the host for statistical analysis and possible money clearing among different transport operators.

6 SECURITY MECHANISMS

6.1 Introduction to the Calypso Security

This chapter describes the Calypso Specification security architecture.

The cryptographic algorithms used are DESX[6], Triple-DES and AES (see below).

The access to data in a Calypso card is submitted to a number of rules that may require that specific access rights be granted. These rules depend upon file access conditions specific to every file, and upon cryptographic computations using secret keys stored in the card.

A specific security mechanism is also used to change the value of these keys.

Furthermore, to handle the specific ergonomics of the contactless link, two special security features called “secure session” and “ratification” are used (and described in this chapter).

6.2 Secret Keys

6.2.1 Types of Secret Keys

The following cryptographic key types are defined:

Key 1 Issuer key	Issuer Key. It allows changing the other keys, and may be used to authorize the modification, or to verify the value, of some files data. It may also be used in place of the Load and Debit keys. This key is typically used to authorize modifying the data global to the application.
Key 2 Load key	It may be used to authorize the modification, or to verify the value, of some files data. It may also be used in place of the Debit key. This key is typically used as a reloading key.
Key 3 Debit key	It may be used to authorize the modification, or to verify the value, of some files data. This key is typically used as a validation, debit or control key.

All keys are 16 bytes keys, and are used as DESX, Triple-DES or AES keys (the cryptographic algorithm to use is implicit, it is known by the card and by the SAM).

6.2.2 Key Identifier (KIF and KVC)

The keys written in a Calypso application are identified by a public parameter: the Key Identifier, containing the key type, *KIF*, and the key version, *KVC*.

The KIF is a value on one byte, used to identify the type of the key (transport issuer key, stored value debit key, etc.). KIF=00h may be used only for a key with a null value, and KIF=FFh is reserved to indicate that the KIF is unknown.

The KVC is a value on one byte, allowing distinguishing between different keys for the same key type (debit keys for two interoperable networks, two versions of the issuer key, etc.). KVC=00h is reserved for keys with a null value.

6 The simple DES algorithm is now obsolete and is not recommended for deployment.

A key identifier should be unique within one interoperable system. Keys derived from the same master key share the same KIF and KVC as their master key.

In order for the terminal to choose which secret key to use, the terminal needs to know the key identifier made of the KIF and KVC (to choose the correct master key), and the Calypso serial number of the portable object application (for the derivation).

Terminals get the Calypso serial number with the Select Application command. The Open Secure Session commands may return the key identifier of the key selected for the session. (The Select File command also allows reading all the key identifiers of a Calypso application.)

Furthermore, in the card each key of a given application is referenced by its order number, from #1 to #3. The Issuer Key is always number 1, the Load Key is number 2 and the Debit Key is number 3.

6.2.3 Cryptographic Algorithms

Remark: The cryptographic algorithms are confidential. They are not detailed in the present document.

DESX and Triple-DES

A cryptographic algorithm is associated with every key for diversification and use.

The DESX was introduced in international papers to prevent a brute force attack on the DES, by enumerating all the possible keys. It increases the key size by 8 bytes with a simple (and fast) operation.

The Triple-DES is a series of three DES operations. Several modes are defined in the ISO 18033-3 and ISO 9797-1 standards.

Diversification

To increase the security of the system, the keys usually written in a card are *diversified*, so that every card has a different key value. Thus, if the keys of one card become known, the other cards keys are still secret.

To simplify this diversification, the keys of each card are computed from a *master key*. Generally, the card key is computed by a cryptographic operation on the application serial number, using the master key.

MAC Computation Algorithm

The MAC uses a hash function, which updates an 8 bytes digest. The MAC is the result of the last step of the digest.

Key Writing

A special mechanism allows writing the value of the keys in a Calypso application (see the command *Change Key*). The key is securely transmitted to the card, as computed by a SAM, thus ensuring the message confidentiality (it is encrypted), integrity (it contains a signature) and replay prevention (it uses a challenge).

6.2.4 Security Modules (SAM)

The different keys are stored in the terminal in a SAM (Security Application Module). The SAM is a smartcard that can authenticate a card and the data received from it, and prove to the card the authenticity of the terminal.

The SAM performs all the cryptographic computations needed to manage the Calypso applications (keys diversification, MAC computation, etc.).

A SAM contains the master keys corresponding to the terminal functions (personalization, loading or debiting).

There will typically be different kinds of SAM corresponding to the different types of terminals.

6.3 Access Conditions

For a specific file, the access conditions define the access mode of each data access command. For the access modes controlled by a key, they also define which key applies among the three keys of the card.

Calypso defines the following access modes:

Access mode	*Description*
Always	Free access: access rights are always granted.
Never	Access forbidden: access rights are never granted.
Session	Access is possible only if inside a secure session, using the corresponding key. If the session is not closed before shut down, or if an error occurs during the close, the data modifications done during the session will be canceled. This access mode may only be applied to modification commands (not to the Read command).
PIN	Access for reading is granted only if the PIN code has been previously successfully verified by the portable object. The PIN management is optional in a Calypso application.

The Revision 3.2 of the Calypso specification contains additional access rights, for confidentiality management and prior authentication. For more information, please refer to this specification.

6.4 Session Description

Notes:

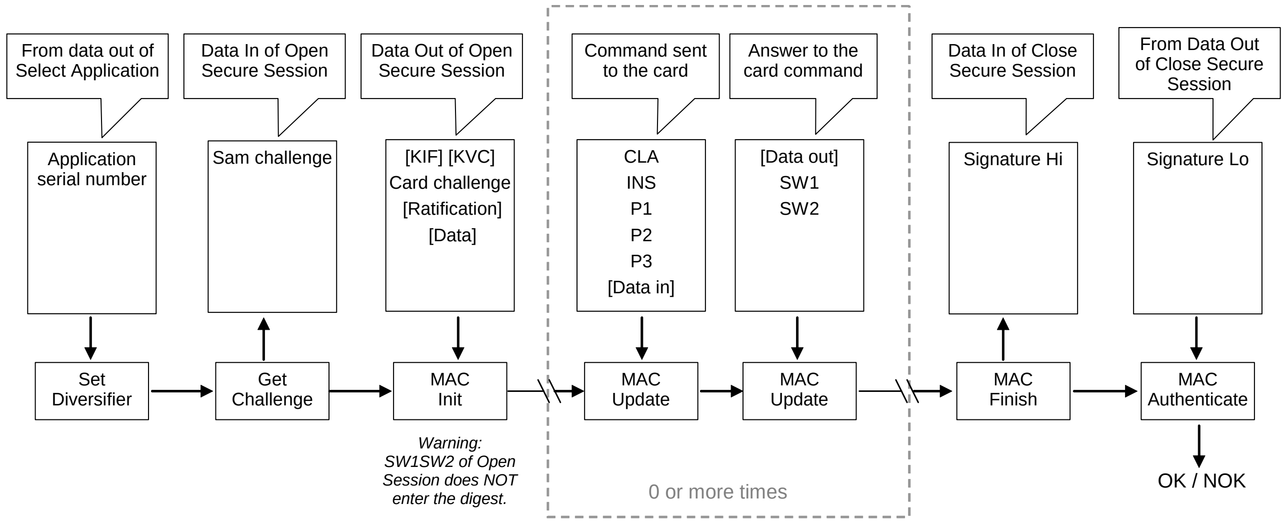
- *The secure session and the ratification mechanisms described below are patented, and their use is subject to a Calypso Application License.*
- *The confidentiality management and prior authentication mechanisms of the Revision 3.2 of the Calypso specification are not in the scope of this document.*

6.4.1 Secure Session Security

The secure session performs simultaneously:

- the authentication of the card,
- the authentication of the terminal (actually, of the SAM used by the terminal),
- the authentication of all the data exchanged during the session,
- the proof that the card modifications have been correctly done,
- the integrity of the data modifications (whole transaction anti-tearing).

The session MAC ensures these authentications and proof: It is computed by the terminal (with its SAM) as described in the following diagram:


From data out of Select Application
Data In of Open Secure Session
Data Out of Open Secure Session
Command sent to the card
Answer to the card command
Data In of Close Secure Session
From Data Out of Close Secure Session
Application serial number
Sam challenge
[KIF] [KVC] Card challenge [Ratification] [Data]
CLA INS P1 P2 P3 [Data in]
[Data out] SW1 SW2
Signature Hi
Signature Lo
Set Diversifier
Get Challenge
MAC Init
MAC Update
MAC Update
MAC Finish
MAC Authenticate
OK / NOK
Warning: SW1SW2 of Open Session does NOT enter the digest.
0 or more times

These operations are done with a high-speed algorithm to allow a very quick transaction. This is particularly important when using the card with a contactless validator.

All the data modification commands given during the session are automatically canceled by the card if the verification of the SAM signature fails, or is not done. The data modifications commands are: , Append Record, Decrease, Decrease Multiple, Increase, Increase Multiple, Update Binary, Update Record, Write Binary, Write Record, Put Data, Invalidate, Rehabilitate and SV operations (SV Load, SV Debit, SV Undebit).

Thus, the session mechanism ensures that either the modifications of the card made during the session are *all completely and correctly done*, or that *none are done*. If the session is not successfully closed (because of a bad signature, a card error, an unexpected shut down, etc.), then all the modifications done during the session are canceled.

Furthermore, a special feature, named the “ratification”, allows the validator to handle gracefully a possible communication link problem (see the later ratification section).

These rules apply in exactly the same way in contactless and in contact modes.

6.4.2 Ratification

During any communication, it may happen that the link be broken unexpectedly. This is particularly true in contactless communication, where the card may be taken out of the validator radio field during normal use, and before the transaction completion.

The Calypso session is a very efficient mean to solve this problem, as an interruption before the session closing will cancel all the modifications done to the card, leaving it in the same state as it was before the session. For example, if a counter must be decreased and a network entrance event must be recorded at the same time in the card, the session mechanism will ensure that either both are completed or that none is done.

However, after the end of the session, and the validation of the changes by the card, the acknowledgement (including the card signature) must still reach the validator. If the communication link is broken between the session closing, and the good reception of its acknowledgement, the validator has no proof that the card is legitimate and that the transaction succeeded. In this case, the user might have paid, or have its transport rights decreased, and not be allowed entrance in the network.

The usual solution to this problem involves a complex mechanism in the validator, which must remember the cards that might fall in this case, and handle them properly if they are presented again soon after. The problem is even more complex in transport networks, where many validators may control the same network gate, and where the user might be tempted to try another validator if the previous one failed to open the gate.

To allow the user to enter the network without paying twice, while avoiding this very complex management in the validators of a network entrance or exit, the **ratification** mechanism was put in place.

6.5 Memory Modification Management

In order to ensure that the writing and erasing in non volatile memory cannot be corrupted by an unexpected shutdown, a Calypso card implements an automatic recovery mechanism. All data written during a secure session are either all completely and correctly written in the card, or not written at all.

7 INTEGRATION OF CALYPSO WITH ANOTHER APPLICATION

7.1 Presentation

The Calypso specification does not specify in detail the association of Calypso with another application, such as a banking application. The Calypso specification defines a set of card requirements which may be extended by various other applications, such as electronic purse, access control, fidelity, etc.

However, it may be necessary for some applications to *link* a Calypso operation and an operation with another application. Calypso allows two types of such links:

- *Stored Value* application: to allow a purse debit linked to a public transport network entrance, Calypso defines a specific application, the Stored Value, with which other Calypso applications of the card may be linked. This link is presented in section 7.2.
- *File sharing*: several Calypso applications of a card may share data, seen from each application as a regular file. This mechanism is presented in section 7.3.

7.2 Stored Value Integration Principle

The Stored Value application (private electronic purse) link with a Calypso operation is made through the Session mechanism.

The debit of the Stored Value application is done during the session when other card data is modified. For example writing a transport event recording an entrance and debiting the stored value for the corresponding amount.

If the session is canceled, for example because the card was removed from the radio field during the transaction, all card modifications done during the session, including the Stored Value modification, are canceled. The card returns to its previous state.

If the session succeeds, all modifications are validated at the same time: the Stored Value debit and the data modifications.

The Stored Value operation is done normally, with its own security and signature computations, using the Stored Value keys. The Stored Value commands are also included in the Calypso MAC signature, as all commands done during the session.

A specific problem to be addressed concerns the Stored Value debit signature. The card delays the sending of the signature to the terminal until the end of the session, since the proof of buying (signature) must only be sent if the debit has not been cancelled in the card. Because the signature may be retrievable by other commands after the debit operation, it is recommended that the only command accepted after a Stored Value debit during a session be the Close Session command.

Finally, another important problem to address is the transaction time. In order to make the whole transaction, including the Stored Value debit, compatible with the contactless validation times, the Stored Value debit is optimized to be very fast (less than around 50 ms for the stored value operation).

7.3 File Sharing Principle

An optional mechanism allows accessing the same data from different EF of the same application, or of different applications of the same portable object.

Such EFs are named *Shared EF*.

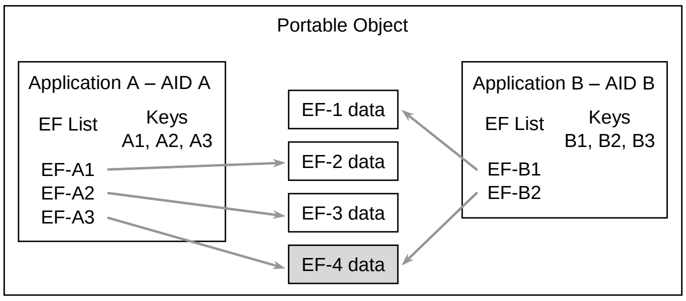
The access to the data is controlled by the access rights of each EF: for example, both EFs might have the same access rights, or one EF could be limited to reading only.

When files share their data, all the data of the file is shared and the files have the same number of records and record size.

This mechanism allows for example the following file structure characteristics:

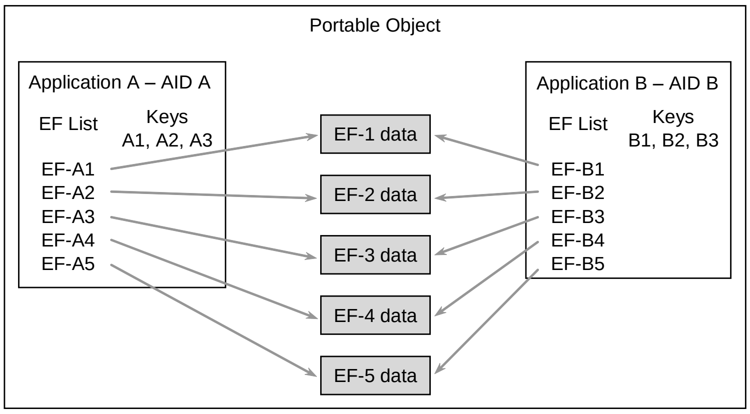
- Access from different applications to some data common to these applications (each application also containing data not shared):

Some EFs of different applications share their data



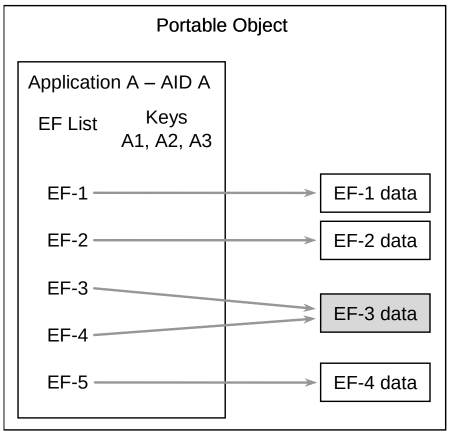
- Two applications sharing all their data but not their secret keys and their AID:

Different DFs accessing exactly the same data



- Access to the same data with different LID/SFI within the same application (for example accessing to the same file with LID 2050h and 2030h and with SFI 1Eh and 06h, as required by some Calypso Revision 2 file structures):

Some EFs within a DF share their data



8 FILES CONTENT

A Calypso card application may hold any kind of data, as it is mainly a fast and secure data holder.

This chapter describes an example of the Calypso files content, for a typical public transport application.

8.1 Master File

The Master File is the card root DF (root directory). Its management is not described by the Calypso Specification. It may be absent (e.g. Java Card).

8.2 Calypso Application

A Calypso application is a directory (DF) complying with the Calypso specifications (i.e. containing Calypso files and managing the Calypso commands).

An application identifier (AID, as defined by ISO/IEC 7816-5) must be chosen for the application.

The files described below are just examples of files usually present in Calypso applications (they are not mandatory since any file structure may be defined).

Environment and Holder File

The linear file *Environment* typically contains information about the transport application: its version number, the network identity (Brussels, Paris, Lisbon, etc.), a validity date, etc, and about the card holder (social profile, reduction rate, etc.).

Contracts File

The linear file *Contracts* typically contains up to four contracts.

For example, a contract may be used to store a specific set of transport rights:

- Contract type,
- Validity period,
- Geographical validity,
- Usage restrictions (holidays, access to specific transport means, etc.),
- Sales information.

A contract might, for example, be a season ticket valid for a month on a specific part of the transport network, or a season ticket allowing only a round trip everyday on a specific bus line.

Apart from the contract structure itself, a contract may be used with an associated *counter*. The counter is processed independently from the contract structure. The counter is a positive value (between 0 and 16,777,215).

Counters File

The simplest use of a counter is as an amount of units. The amount may be decreased during a debit, and increased for a reloading.

It is also possible to use the counter in more complex way, for example by dividing it in two parts, one to store a date, and the other one to store a number of units. In this way it is possible to have in a counter a date for the last use, with a number of trips remaining on that date. It would thus be possible to code the usual transport contracts allowing two trips per days on a certain route, or two trips every Sundays, etc., depending on the value of an associated contract.

For example: the 14 higher order bits might indicate the date of the next authorized travel, and the 10 lower bits might indicate the remaining number of travels possible at that date. Other coding of the counter bits are possible, depending on the transport network policy.

Although the counters are handled independently, a counter is usually used in association with the corresponding contract. In this way, the application would use counter #1 with contract #1.

Events Log File

The Event Log file is a cyclic file containing at least the three last events.

The event record is typically used to log a card transaction. For example, it may log a card debit and may contain:

- The transaction type (entrance, exit, selection, correspondence, reloading, etc.),
- The place of transaction (network, company, line, stop, terminal),
- The date and time of transaction,
- The amount debited and the contract that allowed entrance,
- etc.

Usually, the event record also contains the *ContractList*: the list of active contracts in the card. This list contains the status of the valid contracts in the card, with a one-byte contract information. This allows the validator to read only the relevant contract in the card, speeding up the transaction.

Special Event File

The linear file *Special Event* typically contains an event upon which special processing is done:

- Either events that must not be lost during normal usage of the card. For example, it might be used to store a parking entrance and be able to read it back after card usage in a transport network.
- Either special events like entrance refusal description.

A special event contains the same type of information than normal event (event type, place and time, etc.).

Contract List File

The linear file *Contracts* is used if the *ContractList* is too long to be stored in the Events Log file.

9 APPENDICES

9.1 Normative References

EN 1545-1:2006	Identification card systems - Surface transport applications - Part 1: General data elements
EN 1545-2:2006	Identification card systems - Surface transport applications - Part 2: Transport payment related data elements
ISO/IEC 7816-1:2011	Identification cards - Integrated circuit(s) cards with contacts - Part 1: Physical characteristics
ISO/IEC 7816-2:2007	Identification cards - Integrated circuit(s) cards with contacts - Part 2: Dimensions and location of contacts
ISO/IEC 7816-3:2006	Identification cards - Integrated circuit(s) cards with contacts - Part 3: Electronic signals and transmission protocols
ISO/IEC 7816-4:2013	Identification cards - Integrated circuit(s) cards with contacts - Part 4, Inter-industry commands for interchange
ISO/IEC 7816-5:2004	Identification cards - Integrated circuit(s) cards with contacts - Part 5: Numbering system and registration procedure for application identifiers
ISO/IEC 18033-3:2011	Information technology -- Security techniques -- Encryption algorithms -- Part 3: Block ciphers
ISO/IEC 9797-1:2011	Information technology -- Security techniques -- Message Authentication Codes (MACs) -- Part 1: Mechanisms using a block cipher
ISO/IEC 14443-1:2008 ISO/IEC 14443-1:2008/Amd 1:2012	Identification cards - Contactless IC cards - Proximity cards - Part 1: Physical characteristics
ISO/IEC 14443-2:2010 ISO/IEC 14443-2:2010/Amd 1:2011 ISO/IEC 14443-2:2010/Amd 2:2012 ISO/IEC 14443-2:2010/Amd 3:2012	Identification cards - Contactless IC cards - Proximity cards - Part 2: Radio frequency power and signal interface
ISO/IEC 14443-3:2011 ISO/IEC 14443-3:2011/Amd 1:2011 ISO/IEC 14443-3:2011/Amd 2:2012	Identification cards - Contactless IC cards - Proximity cards - Part 3: Initialization and anticollision
ISO/IEC 14443-4:2008 ISO/IEC 14443-4:2008/Amd 1:2012 ISO/IEC 14443-4:2008/Amd 2:2012 ISO/IEC 14443-4:2008/Amd 3:2013 ISO/IEC 14443-4:2008/Amd 4:2014	Identification cards - Contactless IC cards - Proximity cards - Part 4: Transmission protocol
ISBN 3-540-61512-1	DESX algorithm definition. Kilian & Rogaway, “How to Protect DES Against Exhaustive Key Search”, from *CRYPTO'96. Advances in cryptology 16th annual international conference*, springer verlag

9.2 Glossary

Card	A Calypso Card (or smartcard) is a typical example of a Calypso Portable Object. In the present specification the term C*ard* is used in place of Calypso Portable Object to ease reading.
CD97	Contact and contactless microprocessor smartcard upon which these specifications are based (Carte Déplacement 97)
AES	Symmetrical ciphering algorithm producing 16 bytes of data from 16 input bytes, using a 16, 24 or 32 bytes key (as defined in *ISO/IEC 18033-3*). In Calypso, only 16 bytes keys are used.
Dedicated File	(DF, also called “application”) Equivalent of a directory. A DF contains other files.
DES	Symmetrical ciphering algorithm producing 8 bytes of data from 8 input bytes, using a 7 bytes key (as defined in *ISO/IEC 18033-3*)
DESX	Symmetrical ciphering algorithm producing 8 bytes of data from 8 input bytes, using a 15 bytes key (as defined in *How to Protect DES Against Exhaustive Key Search* by Kilian & Rogaway)
DF	See *Dedicated File*
EF	See *Elementary File*
Elementary File	(EF) File containing data. There are three types of EF defined by Calypso: linear, cyclic and counter files.
LID,	
Long file Identifier	(LID) External unique number identifying a file (0000h to FFFEh, without 3FFFh). All files have an LID. The MF has an LID of 3F00h.
MAC	Message Authentication Cipher. Value allowing the computation of a signature of data.
Portable Object	Any media (e.g. a Calypso smartcard, a mobile phone, usb fob, etc.) which may contain a Calypso application.Record The data in the files are organized in records of at least 29 data bytes.
Record Number	A file may contain more than one record. The record number identifies one record in the file. Files have record number from 1 to the maximum number of records of the file. For cyclic files, record number 1 if the most recent record added to the file.
SAM	Security Application Module.
SFI,	
Short File Identifier	(SFI) External unique number identifying a file (1 to 30). Some files have no SFI.
TDES	Symmetrical ciphering algorithm producing 8 bytes of data chaining three successive applications of the DES algorithm on the same block of 8 bytes data, with a 14 or 21 bytes key (as defined in *ISO/IEC 18033-3*, also called Triple-DES, or 3DES). In Calypso, only 14 bytes keys are used.