Data Synchronization Specification

Shigeya Suzuki, Mark Harrison
Auto-ID Lab Japan, Keio University, Japan,
Auto-ID Lab, University of Cambridge, UK

Auto-ID Labs White Paper WP-BIZAPP-034

***Report Abstract**: Generalized synchronization of tag data and networked database is a challenging topic, but once synchronization-based data distribution is provided, operators such as maintenance mechanics can update information without connectivity to the network or networked databases.*

This report presents an overview of requirements and an initial proposal of data synchronization. This report also covers topics that need to be discussed with partners, as a vehicle to share current thoughts and issues around this research.

Business Processes & Applications

Contents

1. Introduction

In the RFID (or Intelligent-ID) enabled world, since RFID can carry more data than old ID technologies, it is quite natural to store a subset of useful information from some online database onto a tag. Since information in a tag is always available with the object to which the tag is attached, such information is always available regardless of the ability to access a central database. This is one of the major advantages of the use of a recently-developed ID technology. For applications in areas such as the airline industry, one of the candidate applications to apply a tag is inclusion of data about aircraft parts. Information such as the history of the part may be very useful in maintenance operations.

In this scenario, in technical terms, the tag data is treated as a "cache" of related information available on server(s). If someone wants to update the "cache" data on the tag, the entity will have to refer to the online database to perform an update. Thus, effectively, if there is no access to the database (or master information), the tag cache data is read-only.

In another scenario, for example, updating data on the tag without access to related information has different characteristics. If a tag is accessible from an updating entity and the updating entity has access to related information on the database, it is possible to update it by going through a proper procedure. However, if the tag has to be updated without access to the online database, and the updated data needs to be "synchronized" with an online database later, some protocol is required. This protocol is referred to as "data synchronization".

To get the most out of an ID technology, data synchronization is one of the important techniques to be developed. This report will describe early findings and some proposals with a list of further areas of study to complete this research.

2. Systems Architecture and Degree of Synchronization

2.1. Different architectures according to network availability

2.1.1. Architecture assuming network availability

In the current networked computing environment, a typical system architecture assumes that everything participating in a transaction is connected to the network by default. This does not mean that they are always connected — there may occasionally be network outages, unavailability of a particular computer, and so on. However, the system is designed to recover from such situations, or even mitigate disruption, by making use of techniques to increase system availability, such as data replication, stand-by systems and redundant connectivity.

For example, it has been learnt that the operation system at United is designed in a way that it assumes that the network is always available. Interviews with them have revealed that they will even bring satellite-based network equipment when necessary, to ensure connectivity to their networked databases.

2.1.2. Architecture allowing disconnected operation

In contrast, there are some situations where disconnected operations may increase overall operational value. For instance, information regarding parts maintenance history is stored on an online database. Maintenance staff typically access these online databases via computer terminals. Thus, if they need to access these data while in a disconnected environment, it will be very difficult, or sometimes impossible to refer to the maintenance history.

If the tags attached to the parts can hold part of the important information such as parts lifetime, the maintenance staff can safely avoid the use of outdated parts, without requiring access to the online database.

It is also possible to update tag data on a disconnected site. However, if the data on a tag is updated, the data should also be merged into the online database once connectivity is available.

2.1.3. Dependable system requires disconnected operation

It is currently possible to build a system with reasonable connectivity, but to achieve higher resilience, it is better if the system can also handle disconnected operations. To allow for disconnected operations — providing a way to update part of the data — is a challenging task. However, by restricting the type of update, it is possible to realize this.

2.2. Degree of synchronization

From a technology point of view — not from workflow analysis — there seems to be several levels or degrees of tag and database synchronization. Since each level of synchronization technology has different properties, we need to match the technique to use for each kind of data.

2.2.1. Case 0: Always-on environment without synchronization

Scenario: As discussed in the previous section, everything is networked and the network is always available.

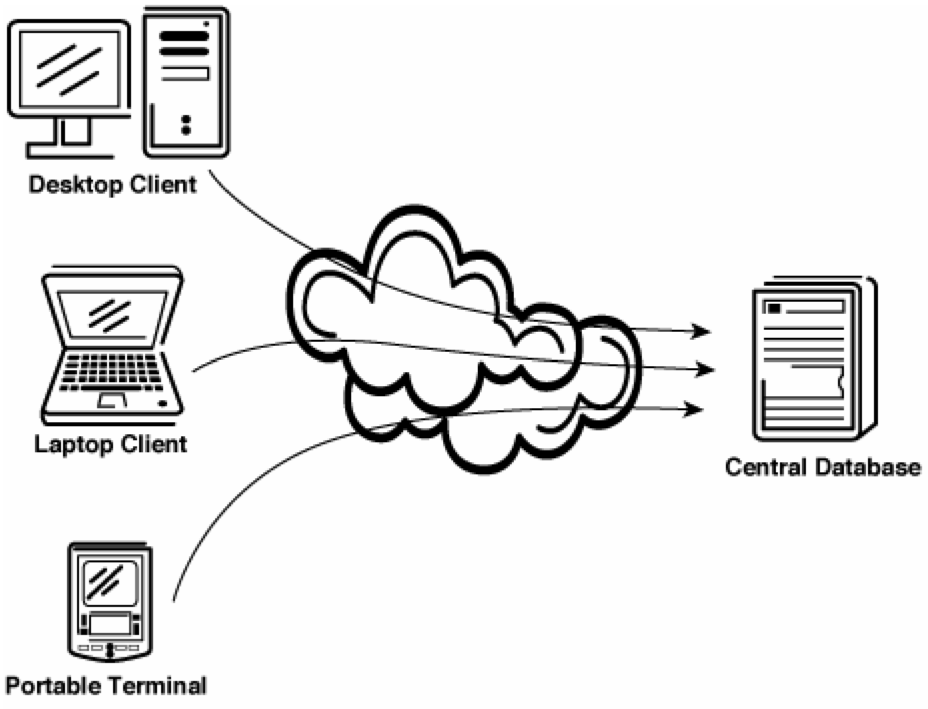

Figure 2.1 Always-on style, client-server access without synchronization

There is inter-system synchronization, but the client accesses online database or other systems as needed. Updates are always made first at the database and then data is also copied onto the tag if necessary.

2.2.2. Case 1: Always-on environment with synchronization, but treating tag data as a cache

Scenario: The master data (authoritative source) is always available in the online system at any time. The tag merely has a copy (cache) of the master data or a subset of it.

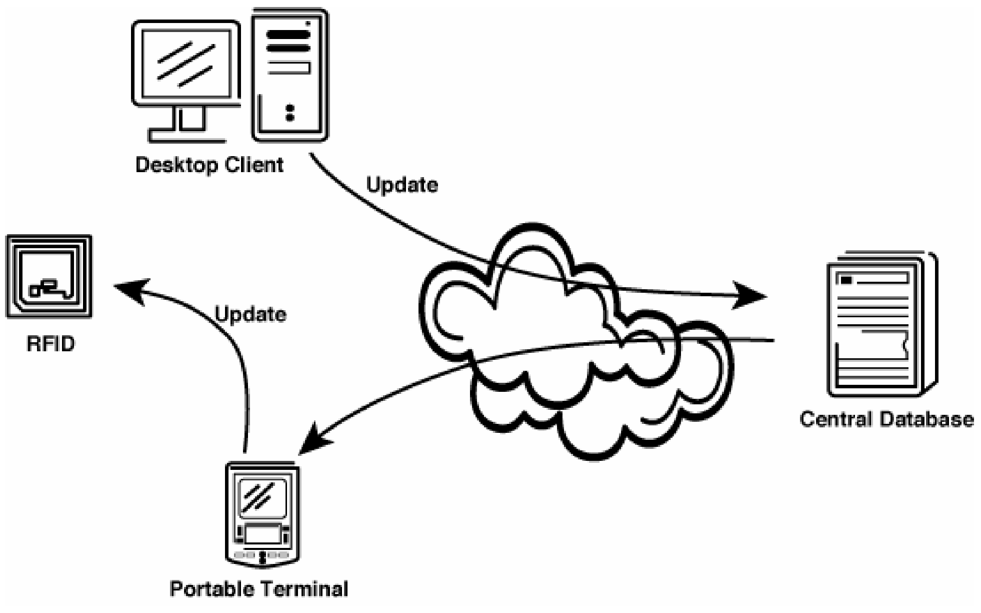

Figure 2.2 Using tag data as a cache of database

The client can access tag data conveniently, but it is always possible to check the authenticity of the data via the online database. Thus, it is not necessary to provide a way to verify the authenticity of the tag data.

Updates are always made first at the online database and then data is also copied onto the tag.

2.2.3. Case 2: Allows disconnected operation, but treating tag data as a cache (read-only)

Scenario: The master data (authoritative source) is always available in online system at anytime. Cache of the online information is available on a tag, together with a proof of its authenticity.

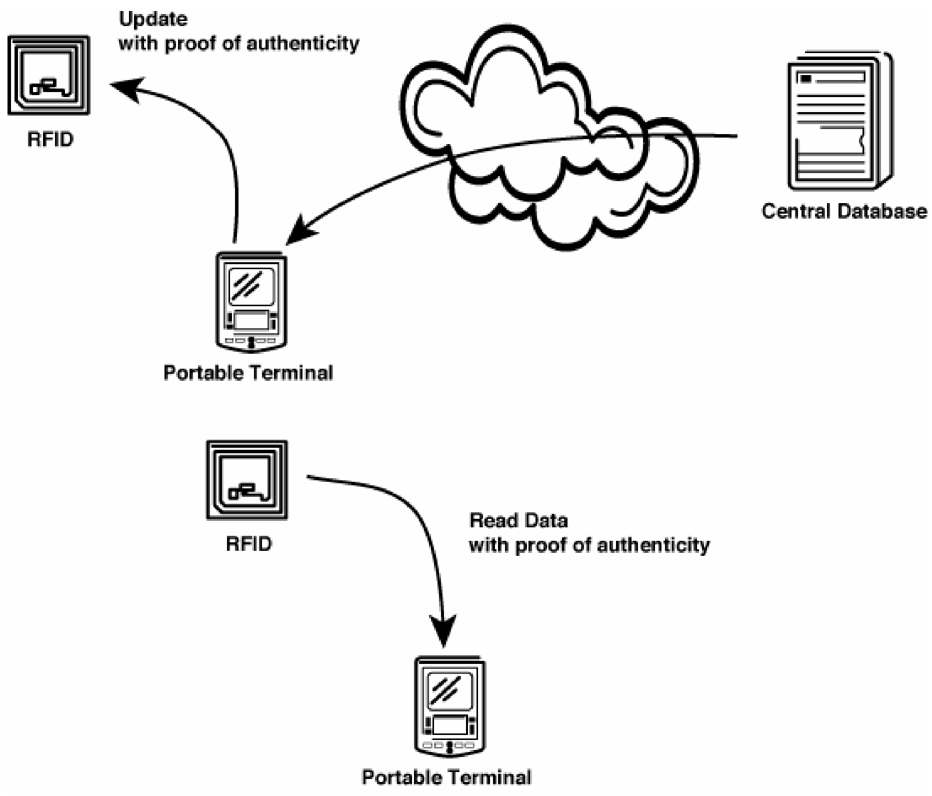

Figure 2.3 Allows disconnected operation, read only

Updates always occur first at the online database, and then data is copied onto the tag. We can treat some kind of certificate information like FAA 8130-3 using this level of synchronization.

2.2.4. Case 3: Allows disconnected operations, also allows updates from the tag. Online database will not be updated until there is a network connection

Scenario: In this case, the tag may contain some data that is more recent than the corresponding data on the online database. For this data, the tag acts as the master (authoritative source of data). Thus, the system allows for updating the data on a tag since it is the master. An example of such a scenario may be that the tag memory contains recent data from sensors attached to the tag.

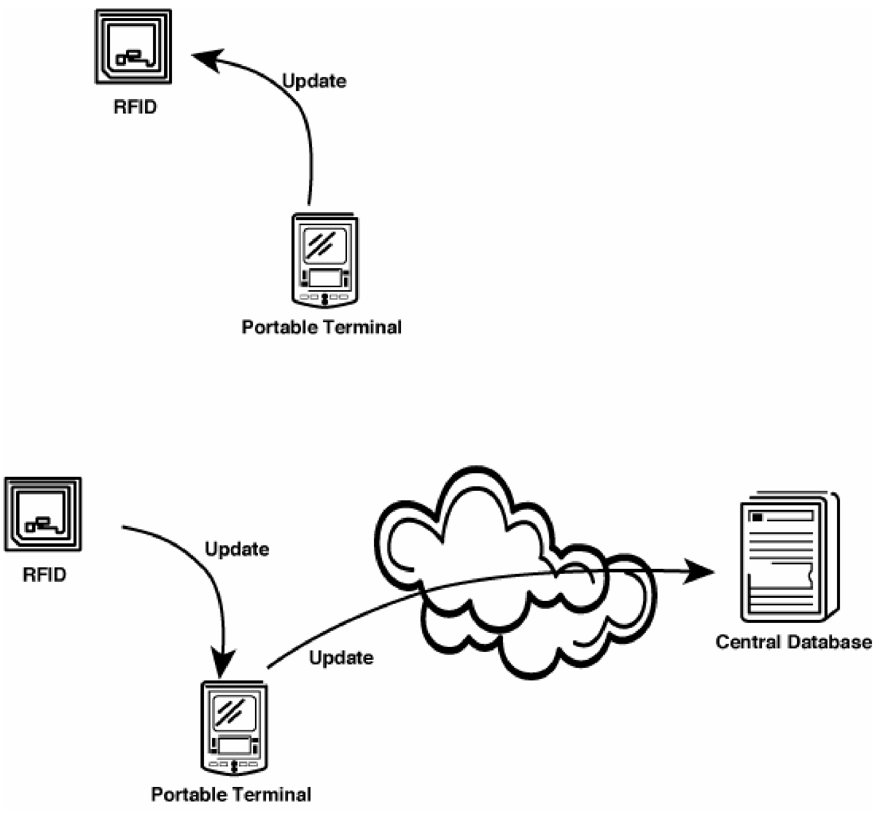

Figure 2.4 Allows disconnected operation, also allows tag write (but not database)

In contrast with the previous cases, since the online database is treated like a cache of the tag data and read-only, the data fields are not allowed to be updated in the online database while the tag is off-line.

When passive tags are in use in this case, the tag data cannot be treated as being "online" since the tag is only readable when a reader is interrogating it.

Some kinds of data, which require proof of authenticity of the data source, require digital signatures for updating information. By using digital signatures, the online master database can safely update the related records after checking the digital signature to authenticate the provider of the update (non-repudiation of data source) and checking the integrity of the data (i.e. that it corresponds to the signature and has not been altered).

Some of the data from sensor tags may be synchronized with this level of technique.

This way of updating requires significant changes to database operation.

2.2.5. Case 4: Allows disconnected operation, allowing updates at the tag and at the database, while disconnected

Scenario: In this case, either the database or tag can update information. The update of data on the tag has functionality similar to the previous case (Case 3).

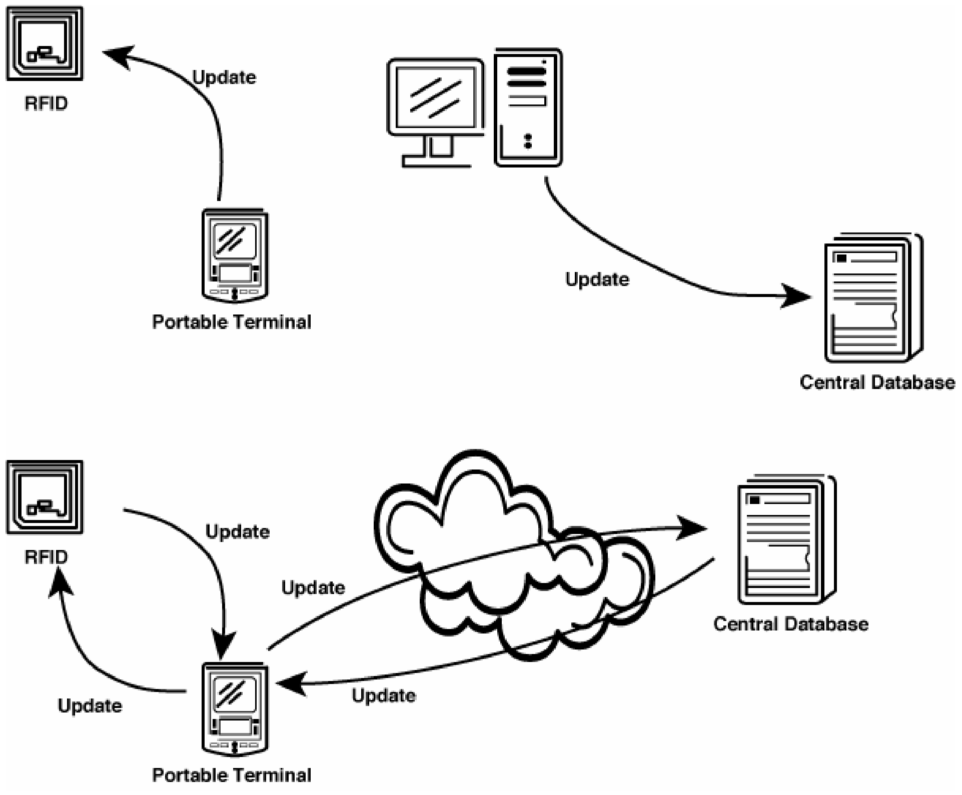

Figure 2.5 Allows disconnected operation and asynchronous update to both database and tag

When synchronizing data, a specific policy is required, per kind of data, about how information can be merged and synchronized. The resulting action may be to reject the update or to raise an alert if there is a conflict between the tag and the online database.

These policies might be complex. Thus, to increase flexibility, we need some language to describe these policies.

It is not clear whether this kind of update is needed or not, hence it would be subject to discussion with partners.

2.3. High level analysis of synchronization cases

In the previous section, we reviewed several degrees of synchronization. They are summarized as below.

Cases 0 and 1 do not require any special techniques.

Case 2 requires validation of data.

Case 3 allows disconnected operations and also allows updates at the tag. The online database will not update its data while the tag is disconnected. In this case, effectively, the master data is on the tag for particular kinds of data, such as local sensor data. Thus, the system allows updating of the data on a tag since it is the master. In contrast, since the online database is treated as a cache of the tag data and read-only, the corresponding records in the online database are not allowed to be updated while the tag is off-line. When passive tags are in use in this case, the tag data cannot be treated as "online" since the tag is only readable when the reader is interrogating it. Some kinds of data, which need verification of authenticity of the data source, require digital signature attached to the updated information. By using digital signatures, the online master database can update the related records safely. Some of the sensor tags may be synchronized using this technique.

Case 4 requires validation of data and updates the data to/from the tag, and also needs to implement a merge policy according to characteristics of the data. Possibly, some way to describe merging procedure to be executed on merge process may help. Since such merge policy descriptions require detailed study on data in this industry, we try to present a way to solve Case 3 in this report.

3. Proposal of Data Synchronization Protocol

In this section, the outline of the proposed data synchronization protocol is described. In this proposal, the tag data is divided into snapshot data, which is in-sync with the online database, and journal data, which contains the local modification history. Authenticity of these data is provided by use of public key cryptography. Validation of authenticity during disconnected operations is provided using a cached copy of a previously fetched public key.

3.1. Actors — Definition

In this document, we use the following actors to analyze the system.

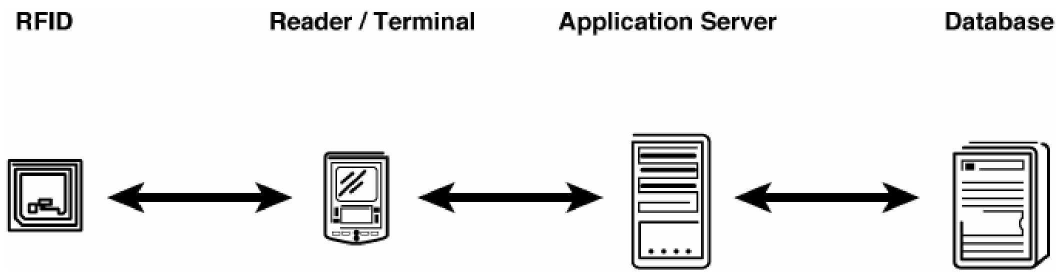

Figure 3.1 Actors

Inside these actors, there are tables of data fields, areas, etc. The following figure illustrates these. The following sections show the interactions between this information.

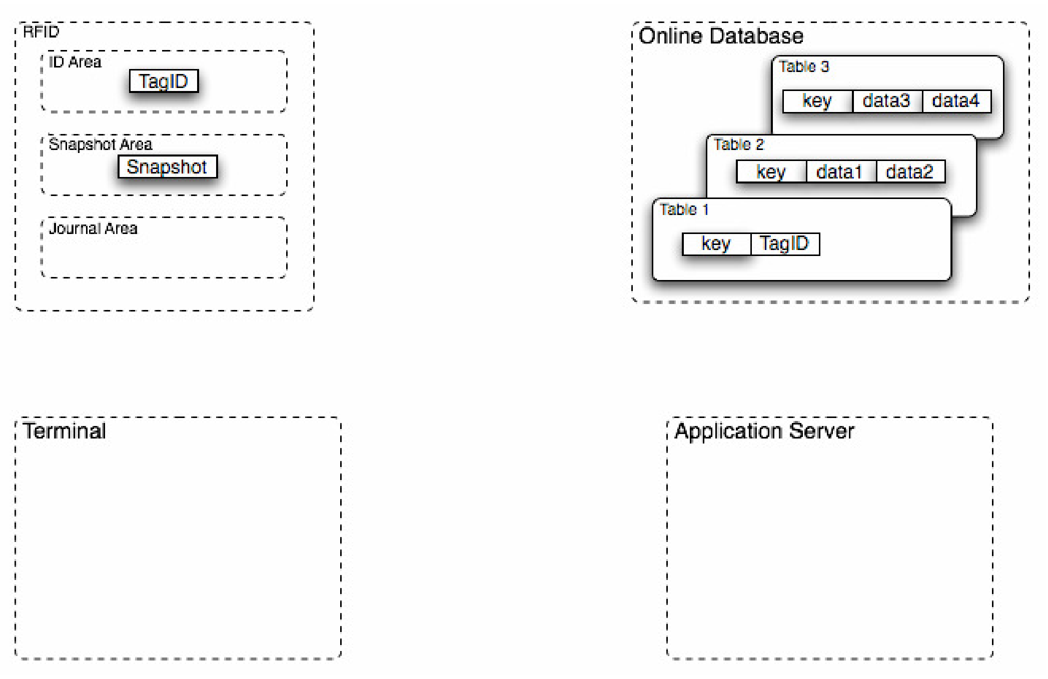

Figure 3.2 Data in each Actor

3.1.1. RFID

RFID is an intelligent ID, which has some amount of memory to store information.

As an assumption, we use the term “ID” to refer to the number, which identifies a tag. Implementation details (such as whether this is a permanent tag ID written by the tag manufacturer or an EPC or other unique item identifier written by the part manufacturer) are outside the scope of this document.

In this report, we assume the use of passive RFID tags. Active RFID will have a different event model and relationship to other systems and would therefore need to be covered by a separate report.

3.1.2. Reader/Terminal

The Reader/Terminal is a general-purpose terminal to access “synchronized” data. It will retrieve information from either RFID tags or the online database (via the Application Server). It has built-in functionality to verify the authenticity or source of a data.

3.1.3. Application Server

The Application Server acts as a backend processor for the Reader/Terminal. It accepts requests from the Reader/Terminal and provides data update logic such as merge and transaction handling. It also provides the online database's source authenticity and signs data with its own private key.

3.1.4. Online Database

We assume, “database” is a typical database program/service such as a relational database that can be queried via SQL, and that it provides at least a transaction safety mechanism, for instance, to allow for the database to either be updated to a consistent state if the transaction succeeds or to roll back to its previous state if the update cannot be completed.

3.2. Procedures required

Currently, there are three possible procedures, which must be treated as a transaction. These are:

- Snapshot update
- Journal update
- Journal merge

Within each of these procedures, it is important to perform Data Validation procedures. We examine each procedure in the following section.

Note: The number in parenthesis denotes the step number in the corresponding figures.

3.2.1. Snapshot update

Snapshot update is a procedure to replicate information from an online database to a tag. The key technique of the snapshot update is provision of authenticity using the data source signature. The data source signature is kept at the data source side. (In this scenario, it resides in the Application Server.) Each participating party can verify the authenticity by checking the signature against the data source public key.

Snapshot update will take place in the following steps:

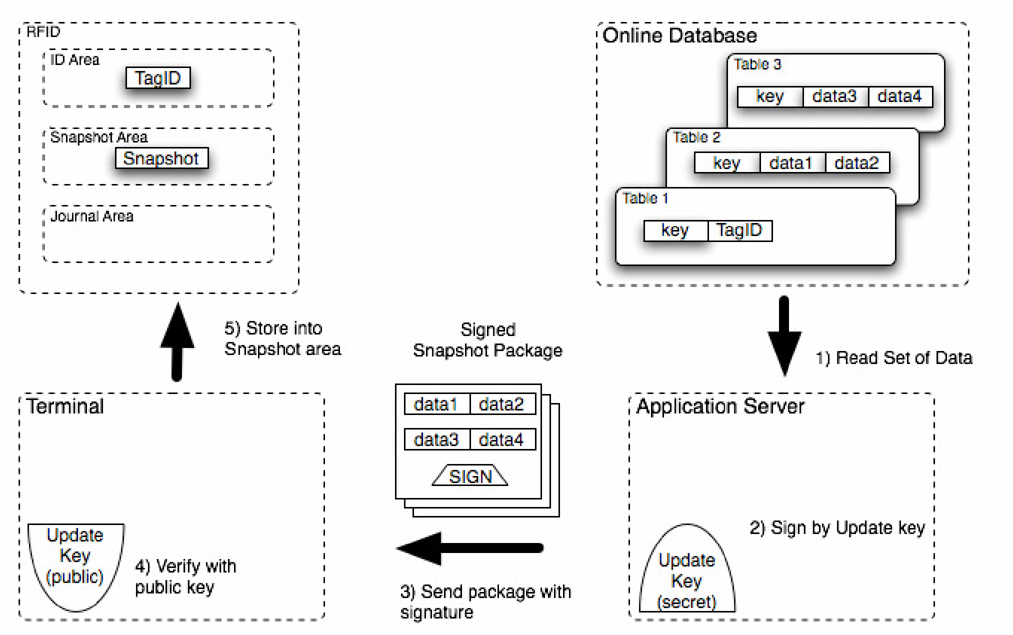

Figure 3.3 Snapshot update sequence

1. Initiation of update by the Terminal (with target tag ID)

Data update is initiated by the Terminal, either by proximity or manually. Firstly, the Terminal will know the ID of the tag, which needs to be updated. Then the Terminal accesses the Application Server to request a snapshot update.

2. Application Server creates a snapshot record

The Application Server creates a snapshot record for the tag in the following steps:

- Fetching data from the online database (1)
- Converting data into a normalized format

- Signing the data with the Application Server's data source private key (2)
- Attaching the signature
- Sending the snapshot record back to the Terminal (3)

3. Terminal writes data onto the tag

When Terminal receives the snapshot record from Application Server, it first verifies the authenticity of the data source (4). If it is valid data, the Terminal will write the information onto the tag (5).

In addition to these steps, to verify the write to the tag from the database point of view, it is possible to introduce several more handshakes between participating entities, e.g. to confirm when the data has been updated to the tag.

3.2.2. Journal update

Journal update is a procedure to record information of incremental updates between snapshot updates. Initially, the tag will store a snapshot, synchronized with the online database. Afterwards, until journal merge takes place, journal entries will be appended to the journal area in the tag. Each journal update will be signed by the Terminal, which provides journal update functionality. Each terminal owns its Journal Update Key, so that 1) other terminals can verify the journal's validity using the signature, and 2) the Application Server can check the journal's validity during merging.

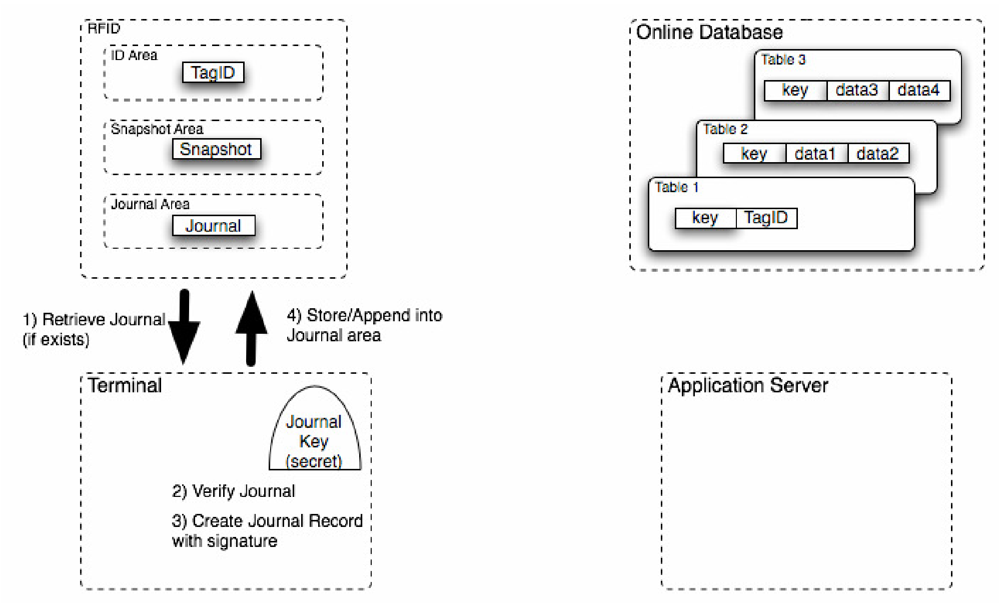

Figure 3.4 Journal update sequence

The journal update takes place in the following steps:

1. Terminal decides to update the data

2. Terminal prepares the journal entry

To prepare the journal entry, the following steps are taken:

- Preparing data for entry
- Reading current journal from the tag (1)
- Verifying the journal according to its signature (2)

Note: Other entities (not in proximity of the current Terminal) may add journals later, before synchronization with the online database. To verify the current journal's authenticity, it needs all the keys that created journal entries. If the Terminal can verify the previous journal entries, it can update (add) the journal. If it cannot verify previous journal entries, two actions can be taken: 1) Adding an extra journal without modifying previous journal, or 2) Not modifying the journal (aborting the current transaction).

3. Addition of latest data to journal

These steps are the same as part of snapshot update:

- Converting journal into normalized format
- Signing the normalized journal with the Journal Update Key (3)
- Attaching signature to the journal

4. Replacing the tag's journal data with newly created journal data

See step (4) in Figure 3.4

Note that, journal replacement must be an atomic operation and should be completely rolled back if it cannot complete the whole process.

3.2.3. Journal merge

Journal merge is a procedure to merge currently available journal data on the RFID to online database, and then replace the RFID's snapshot data with the up-to-date one.

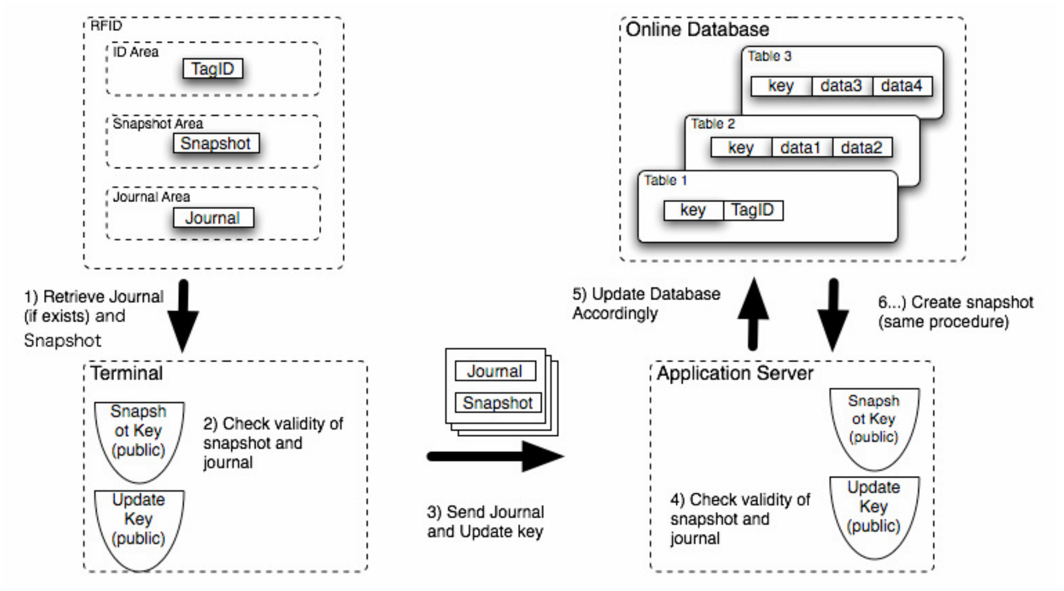

Figure 3.5 Journal merge sequence

The journal merge takes place in the following steps:

1. Initiation of update by the Terminal (with target tag ID)

2. Terminal reads the journal from the RFID tag and verifies it

Steps (1) and (2). Since Terminal has access to the online network, it is possible to retrieve public keys in order to verify these. Thus, it is possible to check the authenticity of the journal data before sending it to the Application Server.

Note: Need to define the way to process verification failure.

3. Terminal sends journal data to the Application Server

See Step (3) in Figure 3.5

4. Application Server verifies the journal

Server also verifies the journal (4).

5. Application Server updates the database according to the journal

See Step (5) in Figure 3.5

6. Application Server creates the snapshot record

Procedure same as snapshot update (6).

7. Terminal writes data onto the RFID tag

Procedure is the same as for a snapshot update.

Note: All operations should be transactionally safe.

3.2.4. Validation

If there are snapshot data or journals, it is possible to verify with public keys corresponding to the signature attached. When checking a journal, public key for the journal may or may not be available. If it is available at the Terminal, Terminal can provide validation of the journal too. The Data Source Key (Application Server Key) is designed to be cached in the Terminal always. Thus, it is possible to validate without external network access.

3.3. Functionalities needed

3.3.1. Validation

It is necessary to verify the data's authenticity either by using the Data Source Keys or Journal Keys. It is possible to test the integrity of the signed data content by using public key cryptography to check that the digital signature for the data corresponds to the message digest when decrypted with the appropriate public key, either the Data Source Key or the Journal Key.

3.3.2. Key management

Both the Application Server and the Terminal need access to public keys. A key ID will identify a public key. All digital signatures to a data will be accompanied by a key ID to specify which public key needs to be used to verify. All keys are created and installed on the Terminal or Application Server prior to the participation of these entities in the system. The key installed on a Terminal is a Terminal Update Key and the key installed on an Application Server is a Data Source Key.

Also, the keys need to be revoked as soon as they are known to be compromised. Thus, the entire key management process, including generation, distribution, rollover and revocation is essential and critical to the system. All the public keys such as Data Source Keys will be kept by participating parties in a cache. Other terminal specific keys might not be in the Terminal.

4. Standard Technologies to Use

In this section, key standard technologies used by this proposal are described briefly. Please refer to reference materials for further reading.

4.1. Public key cryptography

Public key cryptography is a type of cryptography, which allows the user to communicate using encryption or to provide a digital signature without prior exchange of secret keys.

The signing party (or its maintenance staff) creates a key pair — a public key and a private key — which are mathematically related. It is safe to provide the public key to anybody, but the private key must be kept secret by the signing party. It is mathematically difficult to find the private key from the public key.

To provide digital authenticity, the signer re-signs the data using the private key. The signer provides the public key, which corresponds to the private key used, to be publicly available. Parties wishing to check the authenticity of the signature will use the public key to verify.

To provide encryption, the originator can encrypt a data using the receiver's public key. Only the owner of the corresponding secret private key can decrypt the data.

In data synchronization, we use public key cryptography to provide a digital signature service.

4.2. Public Key Infrastructure

By using public key cryptography, we can verify whether the data is signed by a specific private key or not. But the parties who try to verify the key also need to verify the authenticity of the public key. For example, even if a signature is valid against a public key, if the public key is not certified by a party that the Terminal trusts, the Terminal cannot treat it as valid data. In case of compromise of the private key of some party, any compromised key needs to be changed.

Public Key Infrastructure (PKI) provides a way to certify a public key and its relationship with its owner. The public key is stored and distributed as a certificate. By providing the digital signature using a trusted certificate authority, the recipient of the key can be sure of the authenticity of the key.

Note: PKI is implemented in every Internet browser to provide end-point verification in secure communication, called the Secure Socket Layer (SSL), which provides a secure access in web browser. URLs that include “https://” access the webserver using SSL.

4.3. Domain Name System

Domain Name System (DNS) provides a way to find Internet-related basic information such as server addresses, from symbolic names. Virtually all clients using Internet Protocol (IP) use DNS as the basis of mapping server names to Internet addresses.

DNS is a naming system, but since it is capable of distributing relatively small records, it is also possible to use DNS as a public key distribution platform. Due to its distributed nature, DNS is one of the well-deployed distributed databases. Its scalability is sufficient to use in a conceivably large distributed system.

4.4. Domain Name System with Security Extension (DNSSec)

There are some noteworthy security problems existing in DNS. One such problem is called “cache poisoning”, which allows third party to inject wrong information into DNS. This forces the application implementer to use some out-of-band mechanism to authenticate at the peer level, such as client/server certificates for SSL (secure connectivity for web browsers).

DNSSec is a technology that provides a way to check the validity of the set of records involved to resolve a domain name into a resource (values). By using DNSSec, a program can be certain about the authenticity of the resource records bound to the name.

It is recommended to use DNSSec inside a company or service-specific network.

4.5. Tag standards

The report on ID Technology Application Matching (AEROID-CAM-006) discusses the factors which influence the choice of the ID technology (e.g. 1D/2D barcodes, RFID, contact memory buttons etc.) depending on desired features (e.g. read/write memory, non-line-of-sight reading) and ability to withstand particular environmental conditions (extreme temperatures, shock, pressure, electrostatic and electromagnetic fields, corrosive solvents etc.). Readers are advised to refer to that report for further information.

4.6. Aerospace industry e-Business standards

Several efforts have been made to define standards of e-Business such as data exchange formats, protocols and policies. Needless to say, new systems will be created based on these standards.

4.7. US government guidelines

The US government has issued several documents setting standards or guidelines on security concerns. These guidelines are useful to describe the degree of security policies and other pertinent issues. Needless to say, other countries also have similar standards.

For example, US Federal Information Processing Standards Publications (FIPS PUBS) describe several important standards, and one of its documents, FIPS 140-2 *Security Requirements for Cryptographic Modules -- 01 May 25* describes the requirements of cryptographic modules. Many security hardware vendors describe their hardware conformance level according to these standards. Thus, it is reasonable to say, “This part of the system is required to comply with FIPS 140-2.” In further developments of this effort, it is important to refer to these guidelines.

5. Developments Based on Top of Standards

In this section, some major development topics to achieve this proposal are described.

5.1. RFID-related developments

5.1.1. RFID tag data area and access control

As described in previous sections, disconnected data updates and their synchronization require ways to store 1) snapshot data, and 2) journals for updates after the snapshot is taken. These should be treated separately in terms of access control/protection.

Snapshot data must be protected from tampering by terminals without proper permission. In the current design, we want to have some functionality to allow updates inside the snapshot data section in the memory bank, if and only if the proper signature with data source is attached. Secondly, the journal part can be updated by anybody who has the proper key certified by the data source (or other party).

Some kinds of information (e.g. handling of FAA 8130-3 certificates) can be implemented by 1) installation of the certificate on snapshot, and 2) revocation of the certificate at the Terminal, with the signature of the maintenance staff (via Terminal).

Note: On-chip security mechanisms such as two-way authentication and update are already available in HF tags such as the SONY Felica chip. It should also be noted that password-based access control is too weak.

5.1.2. Fine-grained access control to memory of tag

At the very least, write access to the data area and journal area must be controlled separately. As tags increase in functionality, it may be that a new “area” of tag will be reserved for control/policy information. To use the tag memory efficiently and increase access speeds, fine-grained access control is necessary, initially for the resolution of blocks of memory and eventually perhaps to control access to individual data fields.

5.2. Key

5.2.1. Key management

Systems based on public keys or Public Key Infrastructure require a carefully-designed key management system. The key management system also needs to have a set of management policies, with tools used to comply with these policies.

5.2.1.1. Initial key distribution and key rollover

When a new data updating entity is introduced to the system, the entity needs to have its own set of keys as initial keys. If a system is designed to include a mechanism to update keys (rollover), further key maintenance will be minimal, unless problematic events happen. In such a case, initialization should re-occur.

On initial installation of keys, out-of-band distribution mechanisms are necessary. Such key distribution should be secure. Reasonable initialization techniques should be developed.

5.2.2. Policy and supporting mechanism to change a key

Keys need to be changed periodically, either by scheduled replacements or following an emergency. In an emergency, the system needs to replace the key, if some of the important keys are compromised. If the problem itself is not an emergency, the key rollover mechanism can solve most of the cases. However, for an emergency case, a key revocation mechanism must be deployed.

We expect to have lots of terminals deployed all over the system. Thus, replacing or revoking some key in this system can be very challenging.

5.2.2.1. Terminal Key (Journal Key) Update and caching

To allow disconnected operations, certain levels of caching of public keys are essential. All the public keys related to updating of the online database must reside in the Terminal, and there needs to be a mechanism to manage keeping these keys up-to-date when the Terminal is connected to the network.

5.2.2.2. Certificate authority

The certificate authority, which provides the chain of assurance of a key, must be maintained and managed by the airplane manufacturer or central organization of MRO or such.

Since there are discussions regarding how to operate the certificate authority in the airline industry, we can refer to specifications of these efforts.

5.2.2.3. Key identification

Public keys are identified by a Fully Qualified Domain Name (FQDN) of the update node in presentation (human readable) format, and also use some compressed binary format of the FQDN (DNS wire format is one of a candidate). It may also be necessary to distinguish between several keys that share the same FQDN but whose periods of validity are different. In addition to FQDN, to identify a specific key, a DNSSec style key tag is also used.

5.2.2.4. Public key distribution

The participating party who wants to retrieve a public key will retrieve the public key via the Domain Name System's CERT Resource Record. CERT resource records are defined in RFC 2538.

Whether we use the public DNS tree or an internal DNS tree is also a topic of discussion but at present we will assume that we have access to a DNS available throughout the system.

5.3. Data merge (synchronization) service

5.3.1. Automatic merge

A journal with a valid signature can be merged into the online database, and then a new snapshot can be created. The whole process will be run as a transaction. If part of the transaction fails, the whole transaction will be rolled back. This requires tight transactional coordination of the whole system.

5.3.2. Exception handling

If an exception arises, some decision must be made. Some of these policies may be provided by a set of rules, but in some cases, humans need to review and decide what to do. For example, if the Application Server receives a journal without a signature or with an invalid signature, it may — abort the whole transaction; accept the transaction if the transaction fits certain criteria; or forward the transaction to an operator for review.

5.4. Tag-related technology

5.4.1. EPCglobal UHF Class 1 Generation 2

Currently, EPCglobal UHF Class 1 Generation 2 protocol does not provide for fine-grained locking of bit ranges or blocks within the user memory bank as a standard feature; the standard only provides for locking, unlocking or permanently locking (permalocking) of an entire memory bank, rather than individual bit ranges. Recent conversations with Sue Hutchinson of EPCglobal US have confirmed that this is the status quo regarding Gen 2, although she did mention that Chris Diorio (co-chair of the Hardware Action Group at EPCglobal) is drafting an amendment for a future revision to the Gen 2 standard, which would provide a standard mechanism for such fine-grained locking within the user memory. It should be noted that the Gen 2 specification does provide for vendor extensions, so the block-level locking provided by Intelliflex is presumably currently using proprietary vendor extensions, rather than a standard method built into the air protocol. It is better to amend the air protocol soon to provide fine-grained locking mechanism in order to avoid the risk of vendor lock-in and the cost of some future system re-integration when a standard method for fine-grained locking is provided.

5.4.2. ISO 15961/15962

The ISO standards 15961 and 15962 are concerned with the way in which identifiers and user data is recorded in RFID tags.

ISO 15961 describes standard commands that can be sent to a tag's Data Protocol Processor. Commands are provided for locking particular data fields, i.e. at a higher, logical level than merely locking blocks of memory, since the Data Protocol Processor is responsible for ensuring that the data objects and the object ID are aligned within block boundaries and that those blocks are then permanently locked. No “soft locking” of blocks is provided via the ISO 15961 standard.

ISO 15961 provides for tags with no directory structure, tags with a directory structure (i.e. supporting a form of random-access memory) and also self-mapping tags, which are capable of dynamically allocating (and re-allocating) where data objects are stored within the user memory.

6. Further Considerations

This proposal is still immature, but there already are several important topics that need to be discussed while realizing this proposal. In this section, such topics are covered.

6.1. Key management

This is partly a hardware system configuration problem, but since the updating entities (Application Server and Terminal) sign the data online, they also need to keep secret keys online. As the secret keys need to be kept in a safe place, a proper way to protect these keys at a hardware level is very important. For example, the server update key must not be on the hard drive; it must reside in a special kind of hardware device like one that complies with FIPS 140-2 level-2.

6.2. Journal management and sensor data

One good candidate for the data source of data synchronization is sensor data. However, sensor data has several special properties, which are very different from other kinds of data. For example, a typical sensor will record data at particular time intervals, specified by requirements, for instance at regular time intervals or if the value exceeds a particular threshold value. Since all collected data will be stored as a journal, the journal will keep growing. Thus, it is necessary to define some sort of policy on the journal memory overflow. Handling a corrupted journal may require similar policy. Therefore, a detailed study of characteristics of the data and the study of workflow are very important.

6.3. Key management policy

Defining the key management policy is very important. For example, key rollover is a topic of significance. However, since studies on the use of PKI in aerospace industry already exist (such as efforts at the Digital Security Working Group at ATA), we can build on the results of such studies. Since every industry using public key techniques faces the same technical problems, other studies from different industries should also be examined for useful findings and approaches.

6.4. Authentication of operator and secure token

Operators such as maintenance staff will access the tag or online database via a Terminal. There is design choice to use the operator’s security key (such as that contained on a smart ID card or badge) as the Journal Update Key. There is already a discussion on the use of badges with PKI, thus it is possible and reasonable to also store keys to update a journal into these badges.

6.5. XML normalization and signature, ASN.1

A good candidate of a compact binary data representation format is ASN.1 (Abstract Syntax Notation One). In addition to ASN.1, we need to have a way to represent data in XML format, as the use of XML enables the use of a wide range of software to develop the system, which will increase the chances of cost reduction. Also note that ATA SPEC 2000 standard also specifies the process of encoding a data into an XML format, thus use of XML is a must.

In contrast, it is seems prudent to avoid directly storing XML data onto tag, since tag memory is limited and at a significant premium compared with the cost of the corresponding amount of memory capacity in an online database. Thus, it is natural to store data in ASN.1 format, but we may still require to convert to XML when necessary.

However, problems are encountered when some of these data are converted into XML format (or vice versa). If ASN.1 data is signed, and if it is converted to XML, the data with the attached signature cannot be verified, since the signature corresponds to the original format, in this case ASN.1. Thus, if we use ASN.1 and want to use XML as the presentation format, we have to establish a nondestructive, normalized conversion policy between ASN.1 and XML. Since both ASN.1 and XML are very expressive, we need to limit the way to express the data in both formats to make the normalization process easier.

6.6. Data encryption

If necessary, both data on tags and transactions between Terminal and Application Server can be encrypted using automatically-generated session keys. Since we already have ready-to-use key pairs in hand, it is easy to exchange these session keys between entities that require encryption and decryption.

6.6.1. Data overlay and revocation

When revoking a certificate like FAA 8130-30, it is not necessary to MODIFY the certificate itself; simply adding a revocation record and overlaying it on top of the data is sufficient to achieve the functionality. Indeed, the Sarbanes-Oxley rules within the US may forbid the modification of the certificate, so appending with a record to indicate the changes or revocation may be the only viable option.

In this system, an operator may attach his endorsement of revocation to these certificates. Other parties in the system can verify and according to the endorser's reliability, they can decide whether this certificate is good enough or not.

6.7. Clock synchronization

To provide a good security-based system and a merge service, clock synchronization is essential. Even disconnected environments should have precise clocks, accurate to the second.

6.8. Memory consumption

We need to discuss memory use at a high level and at a low level. Some of the high-level layout needs to be aligned to other tag memory standards. We also need to carefully design memory efficiently and enable a flexible memory layout, especially for the journal area.

Note that, public keys are typically large (order of some K bits) but signatures are relatively small (of order of a few hundred bits).

6.9. Computational power

Computational power required to use cryptography is relatively expensive, creating a new set of public key is especially so. Signing is also expensive, but cheaper than key generation.

The client will only sign their data, so it is possible to use software-based solution. However, for an Application Server, which will add signature to every possible transaction, special hardware may be required to achieve high performance.

Note that generation of the key happens from time to time, so there will be an initial cost as well as a periodic cost (computation requirement) for the key rollover scenario, which should also be taken into account.

6.10. Key parameter selection

Several parameters affect the strength of a digital key. One of them is the selection of the length of the key. Since computation power, which directly relates to the time to break a specific key, is steadily increasing, it is wise to use longer keys in order to be able to continue using a key for a long time period.

There are several studies in this area and there are some recommendations available. In short, at least 3,248 bits length of key is required if one uses asymmetric cryptography. Since length of the key affects the size of the signature and the time to sign/verify a signature, we need to balance timing of key rollover and key length according to operational requirements.

7. Conclusion

In this study, we have presented current and early results, and possible solutions using techniques to separate snapshot data and a journal of updates with the data source's endorsement by digital signature. Although this is very early result, this study will present some views to the readers about how synchronization between an RFID tag and an online database can be implemented, and which areas require further study. With future involvement of partners, we can increase the accuracy of the study to further refine the description of the data synchronization algorithm.

8. References

Air Transport Association e-Business Program Working Groups
http://www.ataebiz.org/apps/group_public/

Arends, R., Austein, R., Larson, M., Massey, D., & Rose, S. March 2005, "DNS Security Introduction and Requirements", RFC 4033.

Arends, R., Austein, R., Larson, M., Massey, D., & Rose, S. March 2005 "Protocol Modifications for the DNS Security Extensions", RFC 4035.

Arends, R., Austein, R., Larson, M., Massey, D., & Rose, S. March 2005 "Resource Records for the DNS Security Extensions", RFC 4034.

Federal Bridge Certification Authority
http://www.cio.gov/fbca/

Housley, R., Polk, W., Ford, W., & Solo, D. April 2002, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC3280.

ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002), Information Technology - Abstract Syntax Notation One, 2002.

ITU-T Recommendation X.690 Information Technology - ASN.1encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER), 1997.

Josefsson, S. March 2006, "Storing Certificates in the Domain Name System (DNS)", RFC4398.

KeyLength.com – Cryptographic Key Length Recommendation
http://www.keylength.com/

Mockapetris, P. November 1987, "Domain names - concepts and facilities", STD 13, RFC 1034.

Mockapetris, P. November 1987, "Domain names - implementation and specification", STD 13, RFC 1035.

National Institute of Standards and Technology (NIST – US)
Cryptographic Toolkit
http://csrc.nist.gov/CryptoToolkit/

National Institute of Standards and Technology (NIST – US)
Federal Information Processing Standards Publications
http://www.itl.nist.gov/fipspubs/

Santesson, S. & Housley, R. December 2005, "Internet X.509 Public Key Infrastructure Authority Information Access Certificate Revocation List (CRL) Extension.", RFC4325.

Schneier, B. 1996, "Applied Cryptography", John Wiley & Sons, ISBN 0-471-11709-9

SONY Felica technology
http://www.sony.net/Products/felica/index.html

Suzuki, S. & Nakamura, M. 2005, "Domain Name System—Past, Present and Future", IEICE Transactions on Communications, Vol. E88-B(3), pp. 857–864.

US National Institute of Standards and Technology
FIPS PUB 140–2: Security Requirements for Cryptographic Modules
http://csrc.nist.gov/cryptval/140-2.htm