The compromise between Security and Usability in the Internet of Things

Kary Främling, Jan Nyman
BIT Research Centre, Helsinki University of Technology, FI-02015, TKK, Finland
E-mail: Kary.Framling@tkk.fi, Jan.Nyman@tkk.fi

In this paper, a new information architecture for real-time data exchange between intelligent products and back-end systems is presented. As intelligent products that are capable of recording data about their surroundings become part of daily life, privacy protection must be addressed. Use-cases detailing different security requirements are presented, and issues related to the secure transfer of data are discussed on a technical level. Guidelines are presented for designing information systems involving intelligent products so that the consumer's right to privacy is taken into account. The applicability of state-of-the art security technologies for protecting the confidentiality of transactions in the information architecture is demonstrated through concrete demonstrator scenarios involving intelligent products and back-end systems.

Keywords: Security, Internet of Things, Intelligent product, Encryption, Authentication, Privacy.

1 INTRODUCTION

One interpretation of the Internet of Things consists in identifying physical things in a way that allows users to retrieve and update information related to the product, where the information is remotely accessible over the Internet (Främling et al., 2007). This interpretation includes applications such as consumers accessing supplementary information about products (user manuals or similar), where the main security issue is that the provided information is indeed the right one. If access to the information is restricted only to identified users, then the user has to authenticate her/himself. For information updates, authentication is almost always compulsory. Transactions between companies require both data encryption and authentication in almost all cases. Therefore, the security level to use for different services or applications tends to be a compromise between the requirements and interests of all the users of those services.

In this paper, a broader interpretation of the Internet of Things concept is used, where Things can be "intelligent" (Kärkkäinen et al., 2003), i.e. collect information about themselves and their environment and react based on that, possibly by informing or alerting users or other information systems about it. An information architecture for this kind of Internet of Things has been developed e.g. in the PROMISE project (PROMISE, 2008). The PROMISE information architecture and the handling of security in it are described in the next section.

2 PROMISE INFORMATION ARCHITECTURE AND RELATED SECURITY REQUIREMENTS

In PROMISE, the Internet is the main medium for communication between different information systems, no matter if they are Product Data and Knowledge Management (PDKM) systems, Decision Support Systems (DSS), Product Embedded Information Devices (PEID) or other. As illustrated by Figure 1, all these different information systems can be grouped together under the concept of a "**node**", whose internal implementation is irrelevant as long as it is capable of communicating over the Promise Messaging Interface (PMI).

The PROMISE connectivity model is similar to that of the Internet itself. Where the Internet uses the *HTTP* protocol for transmitting *HTML*-coded information mainly intended for *human users*, PROMISE uses *PMI* for transmitting *XML*-coded information mainly intended for automatic processing by *information systems*. This also signifies that the security architecture of PROMISE has to respond to the same security requirements as existing Internet-based services. One requirement is that a sufficiently high level of security should be possible to achieve when needed. Another requirement is that publicly available information should be accessible as easily as possible. An important requirement for the Internet is also that the privacy of its users should be respected so some level of anonymity can be required, which is typically in conflict with strict security requirements, where authentication of identities is crucial.

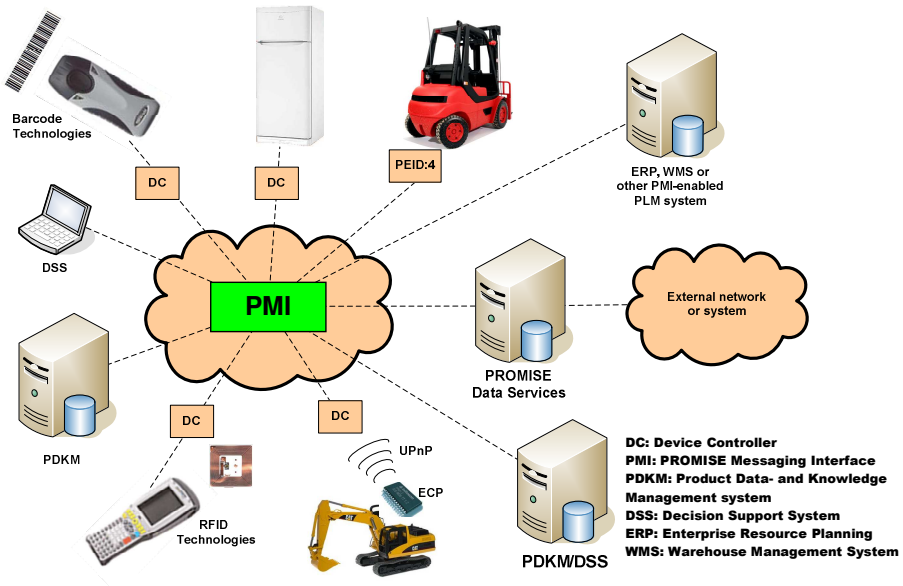

Figure 1. Illustration of PROMISE connectivity (PROMISE, 2008).

2.1 PROMISE security requirements

PROMISE security must be applied in a consistent way in all levels of the PROMISE architecture. If a PEID does not have sufficient computation capacity for implementing the PMI, then it can join the network through a proxy device, the **Device Controller** (**DC**). The most universal DC implementation uses the CorePAC (Core Product Embedded Information Device Access Container) interface defined using the Universal Plug-and-Play (UPnP) protocol. In that case, the security mechanisms provided by

UPnP can be applied to the CorePAC interface in the same way as the security mechanisms of the Internet can be applied to the PMI. In applications where the CorePAC interface is replaced by some proprietary solution, i.e. a *proprietary DC*, the implementation of security is up to the proprietary DC and therefore outside the scope of PROMISE. A myriad of such PEIDs exist already, e.g. engine control units in cars give access to diagnostics and measurement information using the OBD-II or CAN protocols. The PROMISE security model must take also take into consideration and support such existing systems. In practice this means that the PMI must support the passing of necessary proprietary security credentials when needed. Because any kind of information can be sent using the PMI, the sending of such security credentials (encrypted or not) is already supported.

2.2 Overview of relevant security terminology and technologies

Security requirements can be roughly grouped together into the following categories:

1. *Encryption of stored data and network traffic*: This is necessary in order to avoid that sensitive data is accessed by unauthorized parties.
2. *Integrity check of data (stored and transmitted over network)*: Especially when data is transmitted over a network, it may be necessary to verify that none of it has been modified during the transmission.
3. *Authentication*: In order to restrict access only to trusted parties, it is necessary to authenticate the computer and/or the user that attempts to connect over a network before providing access to information.
4. *Access control*: When a requesting computer and/or user has been authenticated, access rules can be defined for what information is accessible and in what ways, e.g. read, write, execute, delete etc.

Examples of corresponding solutions are:

1. *Asymmetric and symmetric encryption*. Asymmetric encryption uses one key to encrypt information and another to decrypt the information, usually called a public key and a private key. The private key is kept secret, while the public key may be widely distributed. The keys are related mathematically, but the private key cannot be practically derived from the public key. A message encrypted with the public key can be decrypted only with the corresponding private key. Examples of widely used asymmetric encryption methods are RSA [Kaliski and Staddon, 1998] and the Digital Signature Algorithm (DSA) [NIST, 2000]. Symmetric encryption uses only one key that is kept secret. Because symmetric encryption is less computation-intensive, it is usually used for network transmission. In that case, a joint symmetric key is generated by one of the communicating parties and transmitted to the other parties using asymmetric encryption.
2. *MD5 (Message-Digest algorithm 5)* [Rivest, 1992] is often used for integrity check of e.g. un-encrypted software packages distributed as ISO CD-images.
3. *Public Key Infrastructure (PKI)* is the most used authentication method for Internet services. It is based on the use of certificates according to the X.509 standard [Adams et al., 2005]. A certificate specifies e.g. who is the owner of the certificate, what organization certifies the authenticity of the certificate, the asymmetric encryption method used for encrypting data with the certificate, a public key and validity dates. The PKI is a hierarchical structure where the uppermost level consists of trusted certificate authorities such as VeriSign, Comodo etc., who sell

authenticated certificates to other organizations. However, authentication may also be performed without using a PKI. In trusted networks, it may be cheaper and simpler to store public keys of trusted parties and use them directly for performing an authentication check as is done by the Secure Shell (SSH) system [Ylönen, 2006], for instance. The most traditional authentication method are usernames and passwords, which are usually as valid as the ones mentioned above as long as they are stored and transmitted in encrypted format. Internet banking applications typically use this technique, where the PKI first certifies the bank's identity to the user and allows creating an encrypted communication channel using Secure Sockets Layer (SSL) [Dierks and Rescorla, 2006] encoding with the HTTPS protocol [Rescorla, 2000]. Fingerprints, iris scanning etc. are other variants of this authentication category.

4. *Access control* is based on authentication; once a computer and/or user has been authenticated using one of the methods mentioned above, then access rules can be applied. Access rules are stored in different ways depending on the information system. The diversity of proprietary existing and future access control systems demonstrates that there is no "one-fits-all" solution. Therefore PROMISE provides means to integrate with these diverse solutions when needed, rather than attempting to propose some new system.

This background information on the PROMISE security context should be sufficient for analyzing and understanding different security requirements and potential solutions presented in the next section.

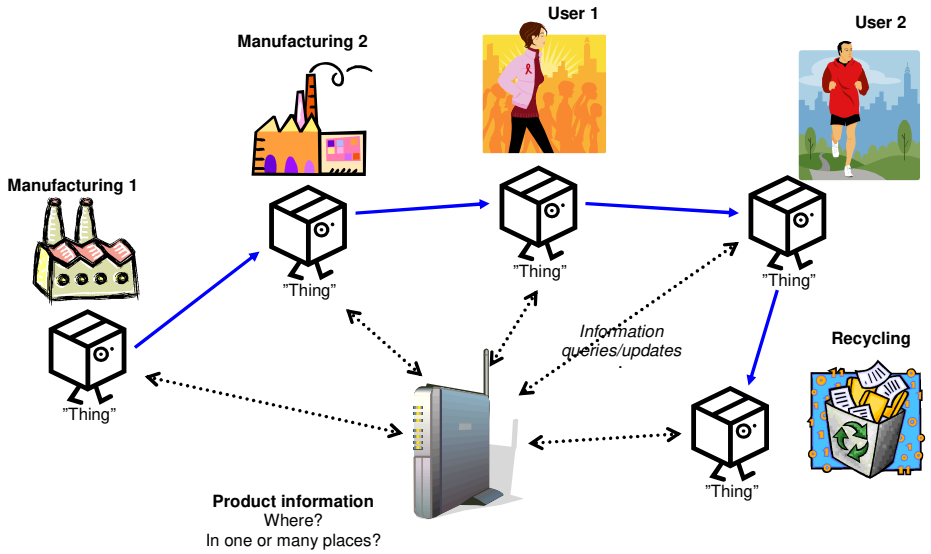

Figure 2: The product lifecycle seen from an Internet of Things point of view. Information about the "thing" is used and produced during all phases of its lifecycle [Främling and Holmström, 2006].

3 ANALYSIS OF REPRESENTATIVE USE CASES WITH DIFFERENT LEVELS OF SECURITY REQUIREMENTS

The implementation of security mechanisms even using standard technologies requires managing certificates, keys, access rights, firewalls etc., which implies a lot of manual work by different people. Many organizations do not even have any personnel who would know how to set up secure servers, manage certificates etc. In order to also allow

such organizations to provide and use PLM-related services, it is important to adjust the level of security according to the requirements of the service, rather than always imposing the highest possible level of security. Furthermore, not only organizations may need to query and update product information during its lifetime as illustrated in Figure 2. At least when the users are individuals, extensive security requirements could discourage the use of many services. At the same time, privacy issues become relevant (Langheinrich, 2001). As shown by the still ongoing debate about the use of RFID technology, universal identification of individuals is a controversial issue. In the following sub-sections we will analyze cases where different levels of security are appropriate, beginning with no security requirements and finishing with the greatest security requirements.

3.1 No security requirements

Figure 2 illustrates the Internet of Things but if the "Thing" would be replaced with a computer screen and a web browser, it might as well illustrate the current Internet. In the current Internet, "query" and "update" operations are mainly performed using the HTTP protocol's GET and POST actions. In the PROMISE view of the Internet of Things, the PMI is used instead. Because PMI also has HTTP as the underlying protocol (augmented with SOAP), it could indeed be considered as an extension to HTTP. We will attempt to illustrate the case when no security is (or should be) required, both for the case of information queries and information updates.

Querying for information about product items could signify asking for user manuals, asking for online diagnosis of some trouble based on sensor values of the "Thing" (sent as parameters of the query) or a microwave oven that asks how long to heat the pizza that was inserted into it. These examples may not be relevant business cases as such but they do illustrate useful services where managing certificates or passwords would mainly be a burden.

Updating information could consist in reading a cinema publicity with a unique identifier that allows buying a ticket by reading the identifier with a mobile phone, registering for product update messages for the car that was just bought, tracking of items in a closed environment, i.e. updating an item's location (e.g. for laboratory equipment). However, updating information usually requires a higher level of security than just querying for it.

The conclusion is that a great number of business scenarios exist where no security is needed and where introducing security would make systems more complicated, difficult to use and subject to privacy issues. It is even probable that there are services that people would not use at all if they are required to identify themselves (i.e. authenticate themselves) unambiguously straight away.

3.2 Data encryption and integrity check is necessary

Encryption and data integrity checks become necessary in all applications where an external spy could gain access to confidential stored data or data sent as messages over a network. As pointed out in Section 2, data encryption usually also signifies data integrity check so that prevents an external party from tampering with the data. Data

encryption can be achieved even when only one of the communicating parties has security credentials (typically a PKI certificate), as in Internet banking applications.

Some examples of situations when it is sufficient to authenticate only one party through certificates or similar are:

- When authentication is done by username/password or some other credentials as in Internet bank applications
- A serial number of an RFID tag, a vehicle identification number or similar existing identification methods can also be used; since encryption is activated in the beginning, none of this information is visible to external observers anyway

Data encryption and integrity check are technically quite simple to set up because existing protocols such as HTTPS (supported by all web servers) can be used directly. Even small companies might have the necessary know-how for this or can cheaply buy the setting up of it. Since each organization only needs to manage its own certificate (i.e. just have it stored in the right place), there is no further maintenance effort needed. Because HTTPS is supported in all "client" applications such as web browsers, introducing this level of security doesn't introduce hardly any overhead. One advantage is also that the identity of the information provider is guaranteed, as well as the integrity of the information provided.

3.3 Authentication of all communicating parties is necessary

In the previous section, only one communicating party was authenticated by X.509 certificates or similar electronic authentication. In Internet banking applications, the authentication of the user is performed by usernames, passwords and/or some other means. In machine-to-machine communication it is often required that all parties would be electronically authenticated, i.e. that all parties want to verify the security credentials (X.509 certificates or similar) of the others through a "handshake" operation before proceeding with the communication. HTTPS supports this by a technique called "client authentication". When using client authentication, the web server takes care of that only authenticated parties can connect.

Despite the existence of client authentication, the technical challenges may increase considerably when using this level of security. Some reasons for this are:

- SSL client authentication is usually configured on the web server level, which implies some modification of configuration files by the web master. In small companies with skilled web masters, this will not be a problem but in bigger companies and/or with less skilled web masters it could be a major problem.
- The result of the authentication may not be accessible to the actual service (e.g. a PDKM) so it may be necessary to do the authentication again if needed e.g. for access control.
- If someone deletes or copies a certificate, then it could enable unauthorized access to information of many parties. Certificates can be revoked but it may be difficult to notice that someone else is using the certificate so significant damage may occur before the certificate is revoked.
- For individual users, creating a certificate is still technically challenging and the user may not even want such a certificate to be transferred.

For Web Services using the Simple Object Access Protocol (SOAP) [W3C, 2007], an alternative to "client authentication" is to use the Web Services Security standard [Oasis, 2004]. The WS-Security standard specifies how to include security credentials in the header of SOAP messages, thereby allowing the services themselves to do the authentication. Various software products exist (including open-source products) that implement WS-Security. With WS-Security it is also relatively easy to encrypt parts of a message. Still, setting up WS-Security is technically even more challenging than setting up HTTPS client authentication.

As a conclusion, this level of security will probably be reserved for inter-organizational communication between organizations that have the means to set up the needed authentication mechanisms. However, as technology evolves, this level of security could become more feasible for other contexts also.

3.4 User-based access control is necessary

By the word "user" we will in this section normally assume that it is an individual user, an organization or a computer that has been authenticated by one of the means described in the two previous sections. User-based access control is needed if the accessible information or services depends on who is trying to access them. As pointed out in Section 2, many systems already have in-built access control. Therefore the implementation of a "sufficient" degree of access control is rather a question of specification and technology choice of the application owner.

3.5 Parts of messages need to be encrypted

Transmitted messages could contain pieces of data that are intended for different recipients, e.g. if a product consists of several sub-systems from different sub-contractors. Therefore different parts of the message may be encrypted using different keys so that only the final authorized party may access it. PMI messages can contain any kind of data, and thus this is again an application-specific issue that does not affect the PROMISE architecture or interfaces as such.

4 SECURITY REQUIREMENTS AND IMPLEMENTATION IN REAL-LIFE SCENARIOS

One of the PROMISE demonstrators involves an ordinary household refrigerator, whose intelligence has been "augmented" with the incorporation of self-diagnostics, statistical data collecting, and communications capabilities. The statistical and diagnostic data obtained from the appliance installed at the customer's premises can be used to detect service needs in advance, before a failure occurs (condition-based maintenance), thus offering improved service for the customer. It could also be possible to determine in advance which spare parts are needed for the job and improve the scheduling of service personnel. Another demonstrator created within the project shows how an existing intelligent product, namely a state-of-the-art automobile, can be a part of the Internet of Things. In that case, the product's capabilities are "augmented" by creating a piece of software running on a Java-enabled handset, that can read diagnostic and sensor data from a OBD-II-enabled car using Bluetooth and suitable OBD-II reader hardware, and send the real-time data over PMI to a remote system (Figure 3). Security considerations

in the two demonstrators are similar to each other. The demonstrators are implemented upon the Dialog software platform (DIALOG, 2008), augmented by PROMISE interfaces.

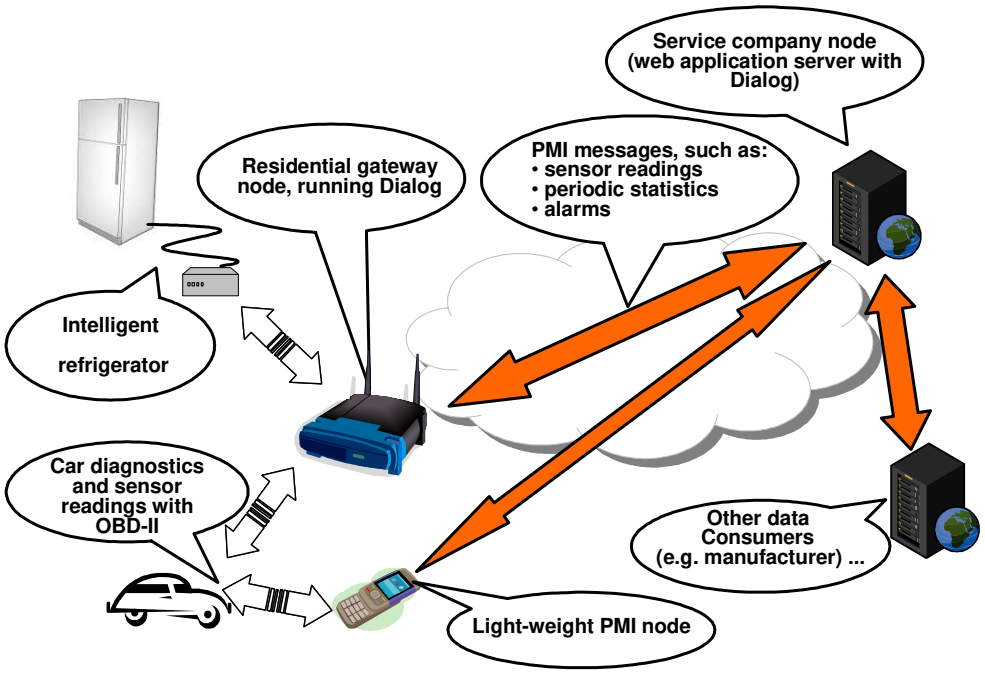

Figure 3. A residential gateway acts as a message interface that enables simple product-embedded information devices to participate in PMI communications over the Internet. Mobile intelligent products, such as cars, can send PMI messages through a mobile phone with suitable client software.

The ability to provide this improved service requires that the appliance is connected to a network (The Internet), so that the run-time data can be transferred to a product data system for analysis. As devices that are part of our daily lives start to share data with entities that are not under the customer's control, we need to address questions related to what data is collected, how and to whom it is sent, and what the data is being used for. Privacy-related issues need to be taken into account in the design phase of the product (and related service), because most of the issues that might compromise the customer's privacy can be handled by taking them into account in the design phase of the product/service architecture. Marc Langheinrich has identified the notions of Notice, Consent, Anonymity, Locality and Adequate security (Langheinrich 2001), that should be taken into account when designing the service involving an intelligent product. The user should be notified and give his or her explicit consent before any data is shared with a remote system. The data sent should be anonymized, if possible. The system handling the customer's private data should be near the user in the sense that the individual should have the legal means of controlling how his or her data is being used. It is only with the last notion, that the data should be protected with adequate security during storage and transfer, that security-related technologies become relevant.

A hypothetical business scenario involving an intelligent refrigerator and data analysis systems could consist of an intelligent fridge installed at the customer's house. The fridge has sufficient network connectivity so that it can periodically send statistics and diagnostics data to the remote analysis system run by the local refrigerator service

company. The service company could have a contract with the manufacturer of the refrigerator so that the manufacturer obtains usage-time statistics data from the installed base of devices to their own analysis system for the needs of product development.

As the intelligent refrigerator is installed at a customer's premises, it is relevant to assume that its connectivity to the Internet goes through a residential gateway/firewall along with other networked devices of the household. The "subscription" of sending statistical data from the appliance to the service company's analysis system could be initialized to the fridge already at installation time. The intelligent appliance could also be initialized, e.g. at the moment of sale, with information indicating to which address the alarms should be sent in case of an anomalous operating condition. Upon receiving an alarm from the intelligent product, the monitoring system could respond with a request indicating that the intelligent appliance should start sending specified sensor values at regular intervals.

As the data sent from the fridge to the monitoring system at the service company contains data that could be used to deduce the customer's daily habits, the data should be encrypted while in transit. This can be achieved using the HTTPS protocol. The system used to establish the secure link in HTTPS is a form of the asymmetric encryption scheme. The server sends to the client its public key, which the client will use to encrypt the data needed to establish a secure link. The data encrypted using the public key is de-cryptable only using the private key of the server, which is kept secret. If the authenticity of the server is important, the server's public certificate could be signed by a certificate authority, which the client can contact to verify the fingerprint of the public key, or the server's public key (or just the fingerprint) could be stored permanently on the client hardware (pre-shared key). On the other hand, pre-shared keys might be problematic, if the server's private key is compromised in the future.

The service company and the fridge should be able to authenticate each other, if the authenticity of the data is critical: The customer does not want anybody else to issue alarms to the monitoring system in his or her name, and the service company does not want to have its database contaminated by bogus data. If HTTPS (or some other SSL-based protocol) is used to encrypt the data, the server is already authenticated (either by a CA or by the fingerprint stored on the client). HTTPS also allows for client authentication using asymmetric encryption. Usually though, the client can be authenticated by simply including the appliance's serial number to the data, which is then compared to the list of known appliances. If the data is judged to be so trivial, that is not necessary to encrypt the data, but the authenticity of the remote server is important, DNSSEC could be used to sign the DNS entry for the server, to guard against DNS poisoning attacks. Finally, the monitoring system's database should allow the intelligent product only to append more data to the database, to prevent malicious users from deleting existing data from the database.

Prior to transmission of product data from installed appliances to the manufacturer the data should be anonymized, to prevent the unnecessary transfer of personal information. The confidentiality of data transferred between the service company and the manufacturer is very important, as data related to performance is of great value to the manufacturer, and also a potential target for industrial espionage. Authentication of participants, encryption and integrity of the transferred data can be achieved using HTTPS, or even with an asymmetric encryption system with a pre-shared key.

5 CONCLUSIONS

The PROMISE information architecture might be the first one that would allow an implementation of an Internet of Things where the Things themselves can be "intelligent". Most of the ongoing work on sensor networks, ubiquitous computing etc. tends to focus only on parts of such an Internet of Things and therefore also have a partial view on security. This paper proposes both an information architecture for the Internet of Things together with guidelines and solutions for handling security in it. Issues related to the privacy of consumers and confidentiality of transferred data should be taken into account during the design of the product/service system. Security-related technologies already deployed on the Internet are sufficient for the needs of future applications involving intelligent products.

ACKNOWLEDGEMENTS

The PROMISE project is funded through the EU 6th Framework program.

REFERENCES

Adams, C., Farrell, S., Kause, T., Mononen, T. (2005). *Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP) – RFC4210*. Available online (accessed 10 October, 2007): http://tools.ietf.org/html/rfc4210

DIALOG 2008, [Online] Available at: http://dialog.hut.fi/

Dierks, T., Rescorla, E. (2006). *The Transport Layer Security (TLS) Protocol - RFC 4346*. Available online (accessed 10 October, 2007): http://www.ietf.org/rfc/rfc4346.txt

Främling, Kary, Holmström, Jan (2006). How to create evolving information models by a layered information architecture. In: *Proceedings of The Modern Information Technology in the Innovation Processes of the Industrial Enterprises (MITIP'2006)*, 11-12 September 2006, Budapest, Hungary. pp.173-178.

Främling, Kary, Harrison, Mark, Brusey, James, Petrow, Jouni 2007, 'Requirements on unique identifiers for managing product lifecycle information - comparison of alternative approaches', *International Journal of Computer Integrated Manufacturing*, vol. 20, no. 7, pp. 715-726.

Kaliski, B., Staddon, J. (1998). *PKCS #1: RSA Cryptography Specifications, Version 2.0 – RFC 2437*. Available online (accessed 10 October, 2007): http://www.ietf.org/rfc/rfc2437.txt

Kärkkäinen, Mikko, Holmström, Jan, Främling, Kary & Artto, Karlos 2003, 'Intelligent products - a step towards a more effective project delivery chain', *Computers in Industry*, vol. 50, no. 2, pp. 141-151.

Langheinrich, Marc 2001, 'Privacy by Design – Principles of Privacy-Aware Ubiquitous Systems', Ubicomp 2001, LNCS 2201, pp. 273-291.

NIST (2000). *Digital Signature Standard (DSS) – FIPS 186-2*. Available online (accessed 10 October, 2007): http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf

OASIS (2004*). Web Services Security: SOAP Message Security 1.0*. Available online (accessed 10 October, 2007): http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

PROMISE (2004), *Product Lifecycle Management and Information Tracking using Smart Embedded Systems*, Available from http://www.promise-plm.com/ and http://www.promise.no/, accessed March 20, 2008.

PROMISE (2008), *PROMISE Architecture Series Volume 1: Architecture Overview*, Forthcoming.

Rescorla, Eric (2000). *HTTP Over TLS – RFC 2818*. Available online (accessed 10 October, 2007): http://www.ietf.org/rfc/rfc2818.txt

Rivest, R (1992). *The MD5 Message-Digest Algorithm – RFC1321*. Available online (accessed 10 October, 2007): http://tools.ietf.org/html/rfc1321

Ylönen, Tatu (2006). *The Secure Shell (SSH) Protocol Architecture – RFC4251*. Available online (accessed 10 October, 2007): http://www.ietf.org/rfc/rfc4251.txt

W3C (2007). *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*. Available online (accessed 10 October, 2007): http://www.w3.org/TR/soap12-part1/