CONVINcE
D4.3.1

Description of prototypes for energy saving mechanisms for security and privacy on terminals

Editor: Anders Plymoth (TelHoc)

Reviewer: Pawani Porambage (University of Oulu)

Authors: Pawani Porambage (University of Oulu)
Anders Plymoth (TelHoc)

EXECUTIVE SUMMARY

This document describes mechanisms and prototypes for saving energy in mobile terminals by focusing on the consumption of security and privacy algorithms used by video delivery networks. In addition to describing the prototypes themselves, some results from analysis and measurements are also presented.

As an evaluation scenario and test case, a video surveillance system has been developed that is based on a multi-tier wireless multimedia sensor network. The multi-tier operation enables keeping the camera sensor nodes in a sleep mode for most of the time and waking them up using low-power motion sensor nodes only when activity is detected. The different software prototypes described in this document were integrated and tested in this system. This includes a lightweight key establishment protocol which has been developed for resource constrained devices, called CHIP. It implements a collaborative Host Identity Protocol (HIP) with efficient proxy-based key establishment component and reduces the power consumption of the computationally complex operation of cryptographic key generation. The main observation was that using secure protocols do not inflict significant increase in the energy consumption. Using FTPS instead of FTP as transfer protocol increased the total energy consumption by around 2%, while using HTTPS instead of HTTP in streaming video increased the total energy consumption by around 1-5%.

A prototype has been developed that enables a device to dynamically apply different types of encryption for a video connection. In the convince project the focus is on MPEG-DASH video connections, that as such operates over HTTP, and by using this prototype it is possible to specify the type of encryption that is preferred when transporting the content through a parameter in the URL. Some encryption schemes are more power efficient than others, but at the expense of a less efficient key generation algorithm. Therefore, as the energy consumption can vary by as much 50% between encryption schemes, and the key generation operation, although used once per session, up to 4000%, this parameter allows the device and/or application to dynamically choose the most appropriate security protocols.
This prototype also support an option to link the video DRM protection mechanism to the TLS connection in HTTPs to that no additional over the type DRM encryption is needed, thus removing one level of encryption and lower power consumption. This can save up to 4.6% power consumption.
A power efficient AAA mechanism that be used in an access point or edge server has been developed that reduces the number of secure lookups needed by combining the different methods by applying CDN/ICN type functionality on the AAA process. This links with the TLS prototype described above. This can save up to 11% of the power consumption.

Table of Contents

Table of Figures

Table of Tables

No table of figures entries found.
In your document, select the words to include in the table of contents, and then in the Formatting Palette under Styles, click a heading style. Repeat for each heading that you want to include, and then insert the table of contents in your document. You can also create a table of contents by clicking the Create with Manual Formatting option and then type the entries manually.

1 DOCUMENT HISTORY AND ABBREVIATIONS

1.1 Document history

Version	Date	Description of the modifications
0.1	15/05/17	Initial version of table of contents.
0.2	01/06/17	Merge contributions by UO and TH
0.3	20/06/17	Added executive summary
0.4	27/06/17	Review by UO
1.0	27/07/17	Final Version

1.2 Abbreviations

SHA	Secure Hash Algorithm
MD4	Message Digest 4 algorithm
MD5	Message Digest 5 algorithm
IETF	Internet Engineering Task Force
DH	Diffie-Hellman public key encryption
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman public key encryption
ECDSA	Elliptic Curve Digital Signature Algorithm
SSH	Secure Shell
TLS	Transport Layer Security
SSL	Secure Socket Layer
MAC	Message Authentication Code
CA	Certificate Authority
PKI	Public Key Infrastructure
RC4	Ron's Code 4. RSA Variable-Key-Size Encryption Algorithm.
DES	Data Encryption Standard
AES	Advanced Encryption Standard
RSA	Standard Public-Key encryption protocol. Rivest, Shamir, and Adelman
HIP-BEX	HIP Base Exchange
HIP-DEX	HIP Diet Exchange
CHIP	Collaborative Hip Identity Protocol
KDC	Key Distribution Center
E2E	End-to-End
DH	Diffie-Helman
DoS	Denial of Service
DTLS	Datagram Transport Layer Security
HIP	Host Identity Protocol
IKE	Internet Key Exchange
IoT	Internet of Things
KMP	Key Management Protocols
MTSA	Media-Aware Traffic Security Architecture
PKC	Public Key Cryptography

QoS	Quality of Service
VSN	Visual Sensor Network
WMSN	Wireless Multimedia Sensor Networks
WSN	Wireless Sensor Networks

2 INTRODUCTION

2.1 Network Security and Privacy

Network security and privacy is an area of increasing importance that relates to the abilities of two devices to securely communicate without the ability of a third party to listen in on the communication. It also relates to ability of each device to ensure that it really is the device it claims to be (authentication), that the device are allowed to do the communication (authorization) and that the communication have not been compromised or altered, i.e. that of integrity.
The most common way of achieving these goals today is through the use of Cryptography and cryptographic keys.

2.1.1 Securing Device Communications

This most important tool today in order to secure the communication data between devices is through the use of Cryptographic encryption. Here, data is encrypted using a Cryptographic Cipher that employs a key, and without the key it is impossible or very hard to determine the data that was transmitted.
The cryptographic ciphers that are the commonly currently being used can be divided into two types, *symmetric* and *asymmetric*. *Symmetric* ciphers use a single encryption key that is used by both the sender and receiver. Here the plaintext data is first encrypted using the cryptographic key and the same key is then used by the receiver for decrypt the ciphertext data in order to retrieve the original data. In these schemes, it is important to properly protect the key or make sure it is kept secret unauthorized users. Popular symmetric cipher includes RC4, DES and AES.
Asymmetric encryption, also often called Public-key cryptography, is a class of cryptographic protocols based on algorithms that require two different and separate keys, one of which is private (secret) and one of which is public. The two keys are different by the key pair is mathematically linked. The public key is used, for example, to encrypt a small plaintext or to verify a digital signature; whereas the private key is used for the opposite operation; to decrypt the ciphertext or to create a digital signature. The term *asymmetric* stems from the use of different keys to perform these opposite functions, each the inverse of the other, compared to *symmetric* encryption described above. For network communication, this is immensely useful as the public key can be transmitted clear prior to communication, or posted in a directory or public database for anyone to see, as long the private portion of the key pair is kept secret.
Public-key algorithms are based on mathematical problems that currently admit no efficient solution and are inherent in certain integer factorization, discrete logarithm, and elliptic curve relationships. It is computationally easy for a user to generate their own public and private key-pair and to use them for encryption and decryption. The strength lies in it being "impossible" (computationally infeasible) for a properly generated private key to be determined from its corresponding public key. Thus the public key may be published without compromising security, whereas the private key must not be revealed to anyone not authorized to read messages or perform digital signatures. Public key algorithms, unlike symmetric key algorithms, do not require a secure initial exchange of one (or more) secret keys between the parties (Kessler 2003).

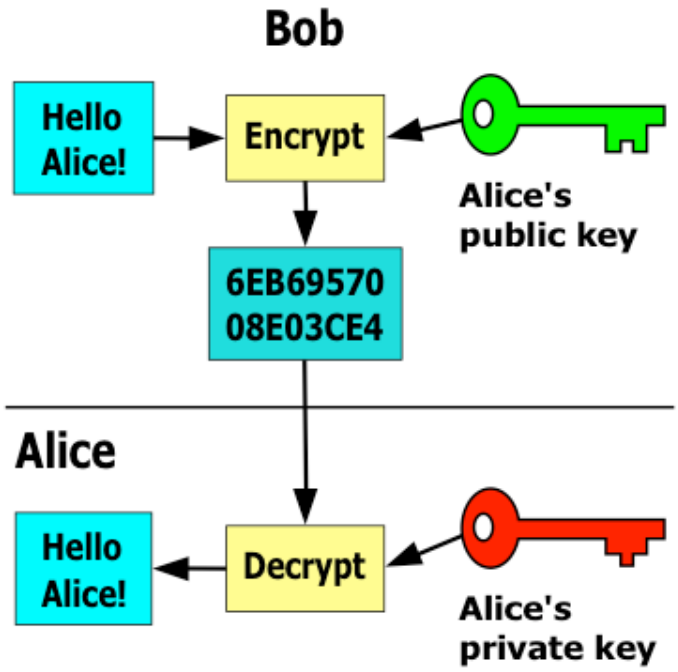

Figure 1. Assymetric (Public Key) Encryption and Decryption.

Data integrity and data authentication involves processing data with a private key to produce a digital signature. Thereafter anyone can verify this signature by processing the signature value with the signer's corresponding public key and comparing that result with the message. Success confirms the message is unmodified since it was signed, and that the signer, and no one else, intentionally performed the signature operation. In practice, typically only a hash or digest of the message, and not the message itself, is encrypted as the signature.

An important difference between *asymmetric* and *symmetric* cryptosystems, is that *asymmetric* systems are often much slower in terms of computation time than corresponding *symmetric* systems. This is also true for one of the most popular and widely used *asymmetric* cipher protocols, (Kaliski 1998). Due to the computationally complex nature of the RSA asymmetric encryption algorithm, the time taken to encrypt a large documents or files to be transmitted can take an increased amount of time to complete. To speed up the process of transmission, instead of applying the receivers public key to large documents or files, the sender can rather randomly generate a symmetric session key and then encrypt the generated session key with the public key. Symmetric encryption/decryption is often a much faster computation to complete as opposed to using the RSA encryption algorithm alone. Often, a hash of the plaintext data is also computed and encrypted with the public key for integrity protection, and included with the data. The transmission would then take place securely and with confidentiality and non-repudiation still intact. The receiver would then decrypt both the encrypted hash and session key and the encrypted documents or files with their private key. Finally, to ensure that the message being sent was in fact sent by the appropriate sender, the receiver would then use the sender's public key to decrypt the hash and then the receiver would hash the documents or files using the same hashing algorithm as the sender and see if the hash that is generated matches the hash key generated by the sender. If both hash keys match, the documents or files are the same as the sender's and non-repudiation has been preserved.

2.1.2 Public Key Infrastructure

In order to practically use public key cryptography in a communication network, a *Public Key infrastructure* (PKI) is needed that supports the distribution and identification of public encryption keys, and thus enabling users and computers to both securely exchange data over networks such as the Internet and to verify the identity of the other party. Without a PKI, data and keys can be still encrypted and exchanged as ciphertexts over the network, but there would be no way of assuring the identity, i.e. authenticating, the other party. Even if a public key can be published anywhere in the open, the PKI is needed is ensure that the party claiming to own the key is actually the owner and that the owner is authorized to use key. The PKI is also used for

authorization, i.e. enforcing user access policies through the use of public keys and digital certificates.

For public-key cryptography to work properly, users must be assured that the other communication party are identity is valid and trustworthy. To provide this assurance, all users of the PKI must have a registered identity. These identities are stored in a digital format known as a public key certificate. A very essential part of a PKI are those of Certification Authorities (CAs) that represent the processes and tools to create digital certificates that securely bind the names and identities of users to their public keys. By creating certificates, CAs act as agents of trust in a PKI. As long as users trust a CA and its business policies for issuing and managing certificates, they can trust certificates issued by the CA. This is known as third-party trust. Certificates created CAs include a set of digitally signed data that includes the users name (Distinguished Name), the public key, the lifetime and valid cryptographic operations (encryption, verification etc). The CA's signature on a certificate allows easy detection of any tampering of the contents of the certificate. As long as the CA's signature on a certificate can be verified as valid, the certificate has integrity. Since the integrity of a certificate can be determined by verifying the CA's signature, certificates are inherently secure and can be distributed in a completely public manner. Users can now retrieve a public key from a certificate and be assured that the public key is valid.

2.1.3 Energy Efficiency of Network Security Protocols

The by far most widely used communication security protocols in use today are the Secure Socket Layer (SSL) protocol and the Transport Layer Security (TLS) protocol, where TLS is an upgraded enhanced version of SSL and is the recommended protocol of choice. For historical reasons though, the term SSL is often used, when TLS is in fact used. TLS is standardized through the IETF via the RFC 6176 (Turner 2011). TLS (and SSL) offers the raw fundamental security services needed for encryption, source authentication, and integrity protection, enabling data exchanges over underlying unprotected networks. The SSL protocol is typically layered on top of TCP/IP layers of the protocol stack, and is either embedded in the protocol suite or is integrated with applications using libraries such as libssl and libcrypto.]
The first step when encrypting application data involves breaking the data into smaller fragments. Each fragment is then compressed, if compression options are enabled. The next step involves computing a message authentication code (MAC), which facilitates message integrity. The compressed message plus MAC is then encrypted using a symmetric cipher. If the symmetric cipher is a block cipher, then a few padding bytes may be added. Finally, an SSL header is attached to complete the assembly of an SSL record.
In the SSL protocol, the SSL handshake is the most complex and consists of a sequence of steps that allows a server and client to authenticate each other and negotiate the various cipher parameters needed to initiate a session. For example, the SSL handshake is responsible for negotiating a common suite of cryptographic algorithms (cipher-suite), which can then be used for session key exchange, authentication, bulk encryption and hashing. More than 250 such ciphers are defined for the SSL protocol, resulting from combinations of various cipher alternatives, such as cryptographic, digest, key exchange and key length algorithms for implementing the individual security services. In reality though, only a limited number of these used in practice as the older suites have been determined to be insecure and deprecated but are still included for historical and compatibility reasons.

3 ENERGY SAVING MECHANISMS FOR SECURITY AND PRIVACY

3.1 Architectures for security and privacy

3.1.1 Wireless Multimedia Sensor Networks (WMSN)

Wireless Multimedia Sensor Networks (WMSNs) interconnect devices that are able to ubiquitously retrieve multimedia content such as video and audio streams, still images, and scalar sensor data from the environment(Almalkawi;Zapata;Al-Karaki;& Morillo-Pozo, 2010), (Sharif;Vidyasagar;& Elizabeth, 2009). These are an evolutionary extension of Wireless Sensor Network (WSNs) that have highly resource constrained scalar sensor nodes. The resource constraints are in terms of computational power, memory footprints, battery life time, etc. However, the multimedia sensors

have to process multimedia data they possess comparably advanced capabilities and richer with more resources than the regular high constrained scalar sensors. WMSNs are deployed in a broad range of applications including tracking, home automation, environmental monitoring, surveillance, traffic congestion avoidance, advance healthcare delivery, and industrial process control. In order to provide high scalability to the application and handle large volumes of traffic generated by multimedia sensors (i.e. audio and video data), WMSN requires a hierarchical topology rather than a flat one. In the project on WMSN by Broadband Wireless Networking Lab, Georgia Institute of Technology, they converge several research areas for the development of an efficient and flexible architecture for WMSNs (Akyildiz;Dai;& Wang, 2007-2010),(Dai & Akyildiz, 2009). Figure 7 shows reference architecture for WMSNs that have single-tier flat, single-tier clustered and multi-tier sensor networks.

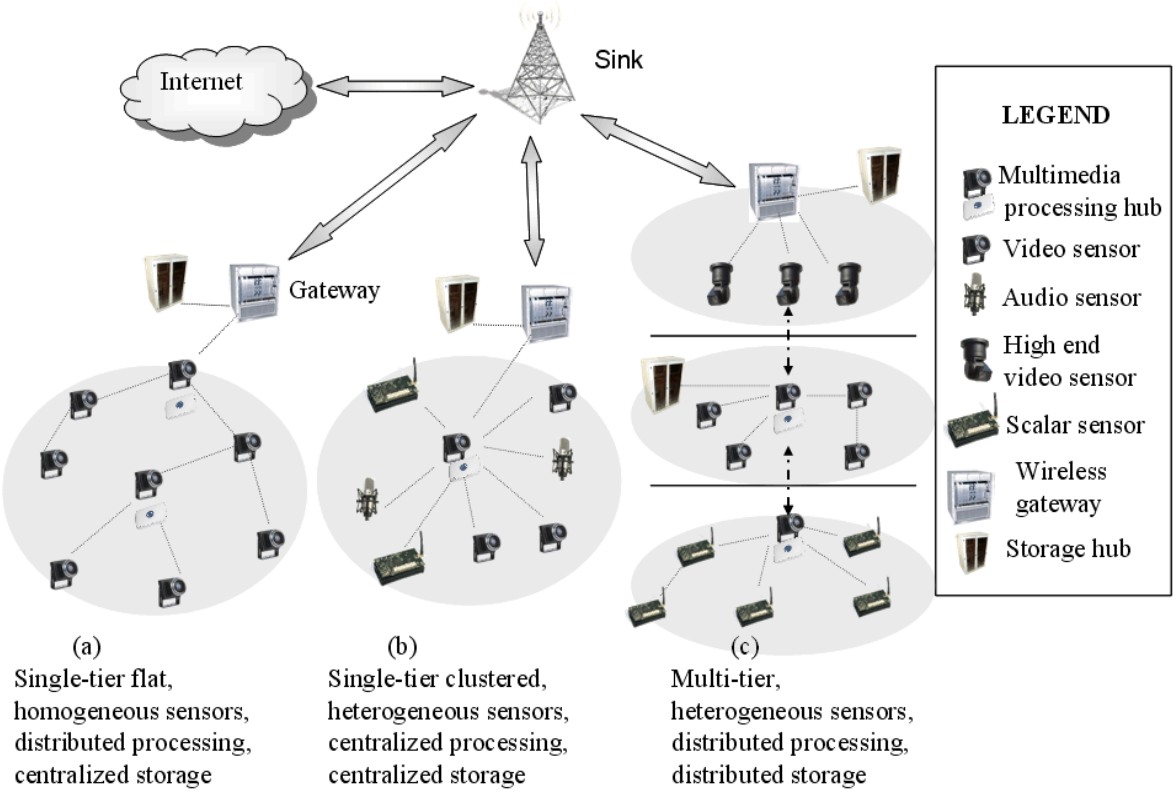

Figure 2: WMSN reference architectures with sub-classifications (Akyildiz;Dai;& Wang, 2007-2010)

Currently Internet of Things (IoT) has become the underlying fabric of the next generation networking technologies that provide seamless connectivity. This has extended the usability of WSNs and WMSNs to data communication between physical devices autonomously with least or no human intervention. Multimedia traffic in IoT can be divided as communication, computation, and service. Figure 8 illustrates a classic example of multimedia service architecture in the context of IoT.

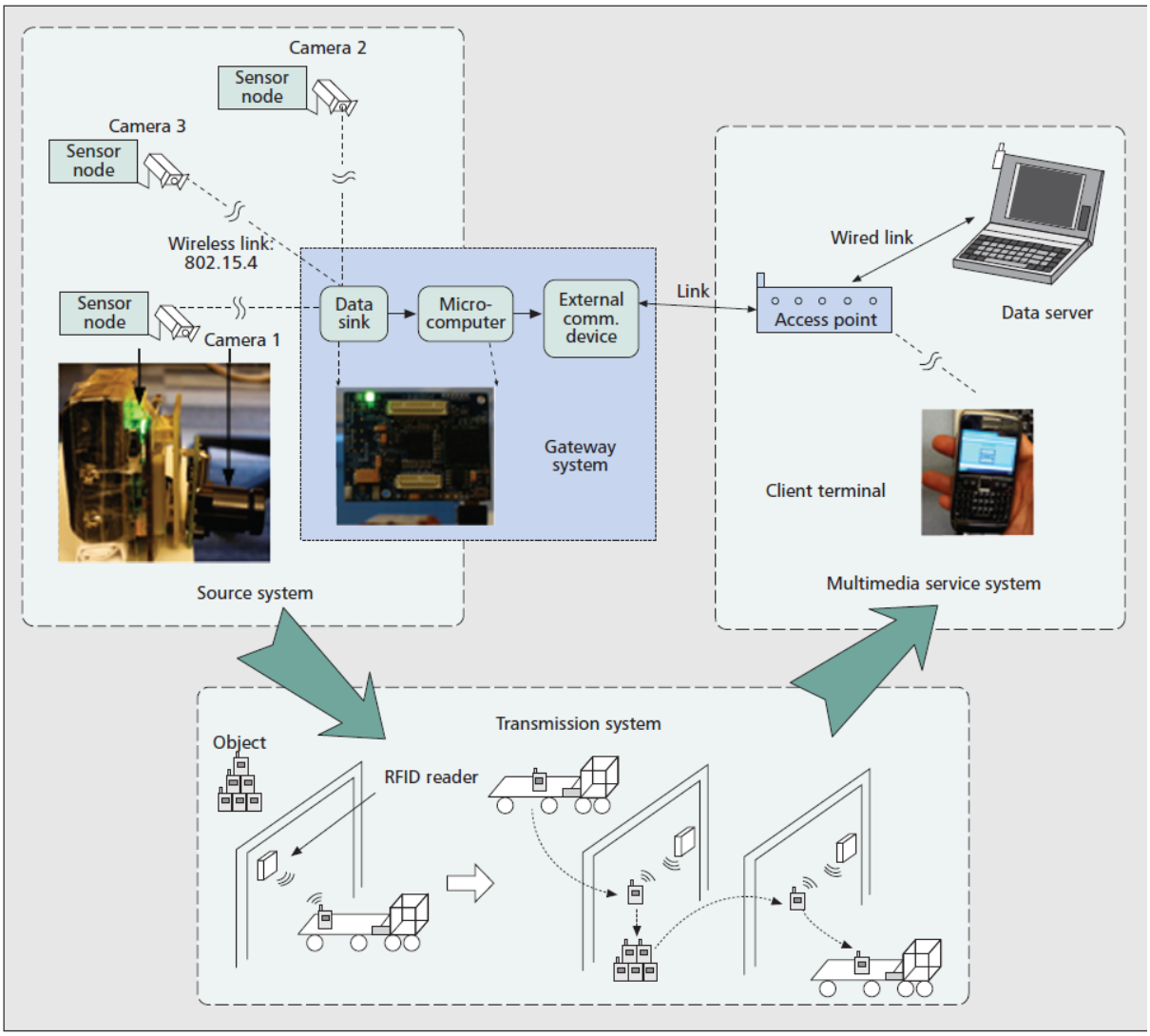

Figure 3: An example of multimedia service architecture in the context of IoT (Zhou & Chao, 2011)

3.1.2 Wireless Multimedia Sensor Networks prototype

University of Oulu has implemented a video surveillance system that is based on a multi-tier wireless multimedia sensor network. Figure 2 illustrates the setup. The multi-tier operation enables keeping the camera sensor nodes in a sleep mode for most of the time and waking them up using low-power motion sensor nodes only when activity is detected. The first version of the system was based on Libelium Waspmotes, but later the camera sensor nodes were replaced by more powerful Raspberry Pi embedded computers that allowed capturing and streaming Full HD video content.

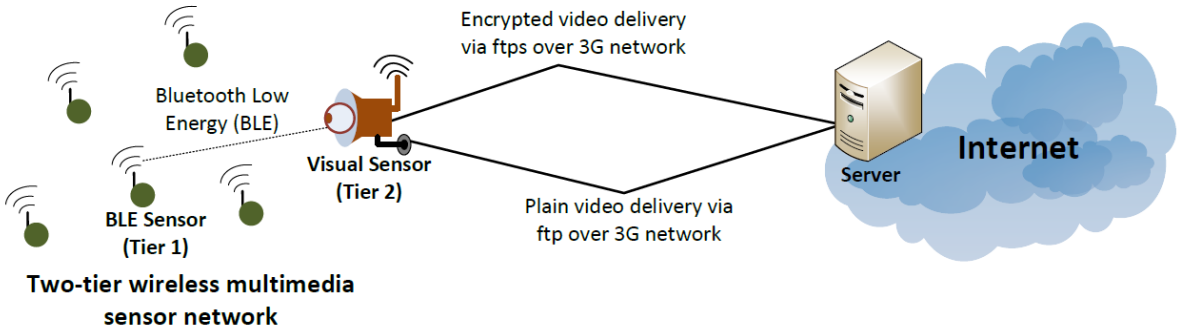

Figure 4. Two-tier WMSN scenario.

The security (T4.3) related study using the WMSN system included two sets of measurements: one comparing the encrypted FTPS protocol versus unencrypted FTP protocol when sending still images from WaspMote-based camera sensor nodes to the server, and another comparing the encrypted HTTPS protocol versus unencrypted HTTP protocol when streaming security camera video from Raspberry Pi -based camera sensor nodes to the server. The main observation was that using secure protocols do not inflict significant increase in the energy consumption. Using FTPS instead of FTP protocol in transferring still images increased the total energy consumption by around 2%, and using HTTPS instead of HTTP in streaming video increased the total energy consumption by around 1-5%.

3.2 Use and Test cases for security and privacy

The reason for wanting enhanced security and privacy protections for video streams depends on the different use cases. It may also depend on how you may want to test the security of your connection or video. Within this project we are mainly working with three use cases:

- On demand video streaming
- Live video streaming
- Camera Based Sensor Networks

These are all good use cases as the first one focuses more on protecting the stream itself via encryption and performing authorization and authentication for protecting who and when a video streaming can be accessed. This could for example be an online video streaming service where data needs to be protected for royalty and copyright protection reasons. The second and third uses cases can in this task closely related, as a camera based sensor network may produce the live stream being consumed, although in the third use case they may also be consumed directly in the local network closer to the source. The third (and as mentioned, sometimes the third) use cases therefore focus a bit more on privacy as the video content may be sensitive and may depict clients and users whose privacy rights needs to be protected. In this sense, these two use and test cases differs in terms of the first one focusing more on security protections for commercial reasons while the third one focusing more on user rights and privacy protections.

Within the Convince project, the prototypes presented in this document have all been tested and verified for all three of the above used cases within the work of WP5. The last two are also part of the final demonstration, where a live video stream from the based sensor network is sent using a encrypted MPEG-DASH stream.

3.3 Securing video and algorithms for energy efficient video streams

With choosing to secure a video stream by means of encryption, the system designer or policies implemented within application basically has three different options. Encrypt the video stream before encoding, during encoding as part of the codec, or after encoding by encrypting the bit level data stream. In practice, only the last option is actually used, as the first two is either expanding the amount of data needed to transmit to much, or is too complex to achieve any gain in terms of power savings.

TelHoc has developed a prototype that enabled a device to dynamically apply different types of encryption in a video connection. In the convince project the focus is on MPEG-DASH video connections, that as such operates over HTTP. Very often the video content is encrypted for DRM purposes, and very often the connections is secured with TLS, i.e. it is transmitted with HTTPs. This means that two forms of encryption is applied on top of each other, as the TLS connection itself is also encryption, in addition to that applied by the DRM process. TelHoc's prototype allows a content provider to provide the DRM protection at the point of the video server when the TLS connection is established. When the device connects to the video server, the TLS server can perform client authentication by looking at the client certificate, and use database lookup to determine what content the client is allowed to fetch, and how it should be protected. This functionality is closely linked to functionality described in section 3.3. Measurements indicate that the savings of this mechanism is about 3.3%, which corresponds to about 10 minutes extra battery time of continuous video streaming on a Nexus 5 mobile phone.

TelHoc has also developed a proxy module that dynamically allows for encrypting MPEG-DASH streams by employing a proxy service to mobile devices. A corresponding server module runs at either the content server, or at any location between the client and the content server. Between the server and client a number of different types encryption can applied, or none at all. This is the same module that TelHoc developed in T4.2 and T3.2 of the Convince project that supports fountain coding, and data is transported over UDP rather than TCP (or TLS), which in itself can often save energy. The proxy module runs as a local HTTP server on the device, and access to it is achieved by using the local address of the device in the video url, e.g. 127.0.0.1, rather than the address of the content server. The application can either embed this URL or transform to the local address, or the user can manually change it in the video client, or the server can send a redirect to the local address, For example, if the video content resides at:

http://video.convince.com/tears.mpd

the URL will be changed to:

http://127.0.0.1/tears.mpd

It is also possible to specify the encryption protocol as part of the URL:

http://127.0.0.1/AES256/tears.mpd
http://127.0.0.1/Blowfish/tears.mpd
http://127.0.0.1/None/tears.mpd

3.4 Edge CDN content discovery and authentication, authorization and charging

As part of this project, Convince, we are looking into the usage of an edge cloud architecture, that will also provide services. An important part of any CDN is how the actual content is discovered, and where replicas can be found as close to the request client as possible. Part of the idea with an edge cloud, is to enable these types of services and more at the edge, rather than as close as possible. Another important aspect of any CDN service is also how authentication and authorization and eventually charging is performed, as only user client with the correct verified credentials should be allowed access to the content.

3.4.1 Standard Authentication, Authorization and Charging

Normally the Authentication, Authorization and Charging process is performed towards a centralized AAA server that performs the verification process. This process typically involves:

1. User provides username and password to server.
2. Server verifies username and password through lookup of an authentication database.
3. After user is authenticated, the server checks with an authorization database if the user is allowed to utilize the requested service.
4. If the user is authenticated and authorized, the server replies with an acceptance message to the client.
5. Communication commences, and the server logs charged activity in a charging database based on the type of service used and charging model. It also possible that the charging database is continuously updated towards an account level based on activity, i.e. a pre paid charging model is used.

Normally this phase also include a few more steps is involved if cryptographic keys are involved. Within this part of the project this is true as we rely heavily on TLS connections for communication. The process then also involves the verification of public keys and client and server certificates, and a handshake process that setup session keys and verifies all parameters.

3.4.2 Edge vs Cloud Authentication, Authorization and Charging

The major difference between Authentication, Authorization and Charging in an edge cloud versus regular cloud environment are the localization of these elements, meaning shorter distances and faster connections. In terms of content discovery this also means that content might be discovered and served directly at the edge without the need to contact several different server elements before the actual transfer of content takes place. This can lead to significant reductions in setup delay, faster throughput of content delivery and reduced energy consumption.

As part of this task in the Convince project, we are looking into the usage of an edge cloud architecture that will also provide services. An important part of any CDN is how the actual content is discovered, and where replicas can be found as close to the request client as possible. Part of the idea with an edge cloud, is to enable these types of services and more at the edge, rather than as close as possible. Another important aspect of any CDN service is also how authentication and authorization and eventually charging is performed, as only user client with the correct verified credentials should be allowed access to the content.

The prototype developed and described section 3.1, can be run a WiFi access point and includes an embedded database. When requests for content arrives at the HTTP server running in the AP, the server can request the content from a CDN, or at the original source, and then cache it. Subsequent requests for the same content can then be served directly from the AP. This software can also perform Authentication, Authorization and Accounting (AAA) functions, and is actually part of another deliverable in Convince, M3.2.4. Rather than running AAA functions far out in the cloud, this software allows them to performed much closer to the user. It also allows a more sophisticated and energy efficient TLS based authentication method when HTTPs is used, as authentication and authorization is already supported as part of the TLS protocol and its usage of X.509 PKI infrastructures, although this is almost never used or even well known.

3.4.3 AAA Client Software

As part of the client software is designed to be running on terminals, including mobile terminals, the client may choose to use different authentication, authorization and accounting methods in terms of the improved TLS method described or the basic method from HTTP that uses username and password. In addition, as there is a whole range of different cryptographic functions that can be used within TLS to provide the AAA functionality as well as the data connection itself, they all have different energy consumption characteristics in addition to the offered security level. Based on current network level conditions, mobile terminal capabilities and battery energy levels, the terminal or user may choose a different method and thereby reduce energy consumption, possibly at the price of reduced security. This software allows these considerations to be possible through the TLS configuration, and offers the required tools to investigate the differences between various AAA primitives.

3.4.4 Power savings of implemented AAA software

We evaluated how much savings you might gain but using the TLS based authentication and authorization vs standard HTTP based authentication and authorization. The standard HTTP based method usually involves imputing and username and password followed by a database lookup to perform the authentication, and if successful, do another database lookup retrieve authorization and access policies. In many cases authentication and authorization databases are located in different parts of the network due to the heterogeneous nature of modern access networks.

AAA type	Consumption AAA	Savings	
TLS - TLS	4.75 W	0.53 W	11%
TLS - BASIC	5.16 W	0.12 W	2.5%
BASIC - BASIC	5.28 W	-	-

Figure 5. Power savings of different types Authentication and Authorization

The TLS based method is mainly more efficient because it removes the need for the database lookup, which in itself is often over an encrypted authenticated channel. TLS invokes authentication using a client certificate signed by the system, and nonce provide by the server and signed by client public key within the certificate. As this can be configured to happen as part of the TLS setup, there is significant gain as all external lookups are eliminated. TLS Authorization happens in a similar manner, but instead used authorization policies specified with the certificate.

From our measurements on our prototype implementation, we can see that the gain is up 11% for a pure TLS-TLS based authentication and scheme, vs HTTP-HTTP. The biggest gain actually comes from authorization part, as it happens after the cryptographic challenge phase of the authentication process, which is more computationally heavy.

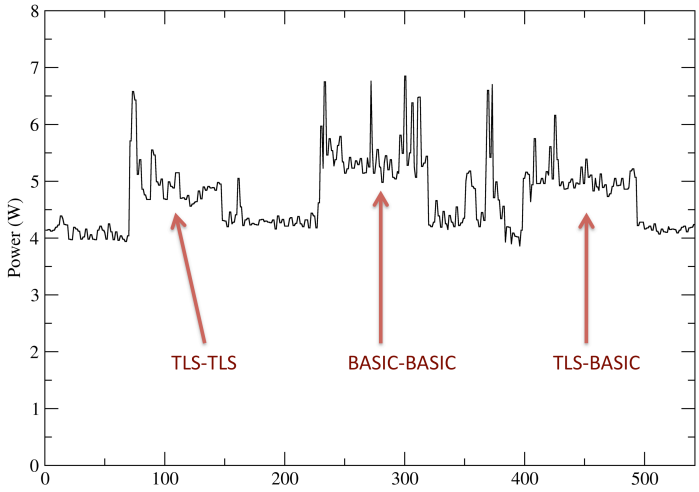

Figure 6. Power Profile of different types Authentication and Authorization

3.5 Lightweight key management and secure group communications

Processing of audio and video data requires at the multimedia sensors require the highest energy consumption. Therefore it is significant to exploit the most energy efficient security solutions with at the terminal side. Lightweight key management and secure group communication are two noteworthy issues in establishing secure connections for internode communication in WMSNs. Figure 10 illustrates several application scenarios where the scalar and multimedia sensors work collaboratively in order to retrieve user data. As shown in Figure 6, the scalar sensors are used to monitor the elderly persons' or patients' health conditions, kids' behavior, athletes' fitness, or traffic jam. The scalar sensors can be integrated into mobile personal devices such as smart phones, smart watches or even fixed to the human body or the vehicles as wearable devices. Whenever the scalar sensors receive data about critical health conditions or susceptible situations, then they are responsible for invoking the multimedia sensors in their closest proximity. At the critical situations, the visual sensors can take the still images, audio streams, or video streams of the target and send to the responsible authorities (e.g., care taker, doctor, nurse, parents, traffic control office). Since the targeted objects travel to different places (e.g., elderly home, hospital, library, park, roads), the scalar sensors may need to communicate with the multimedia sensors deployed in completely unknown environments. Besides, the mulltimedia sensors can be located in less accessible areas, where changing the battery is hard.

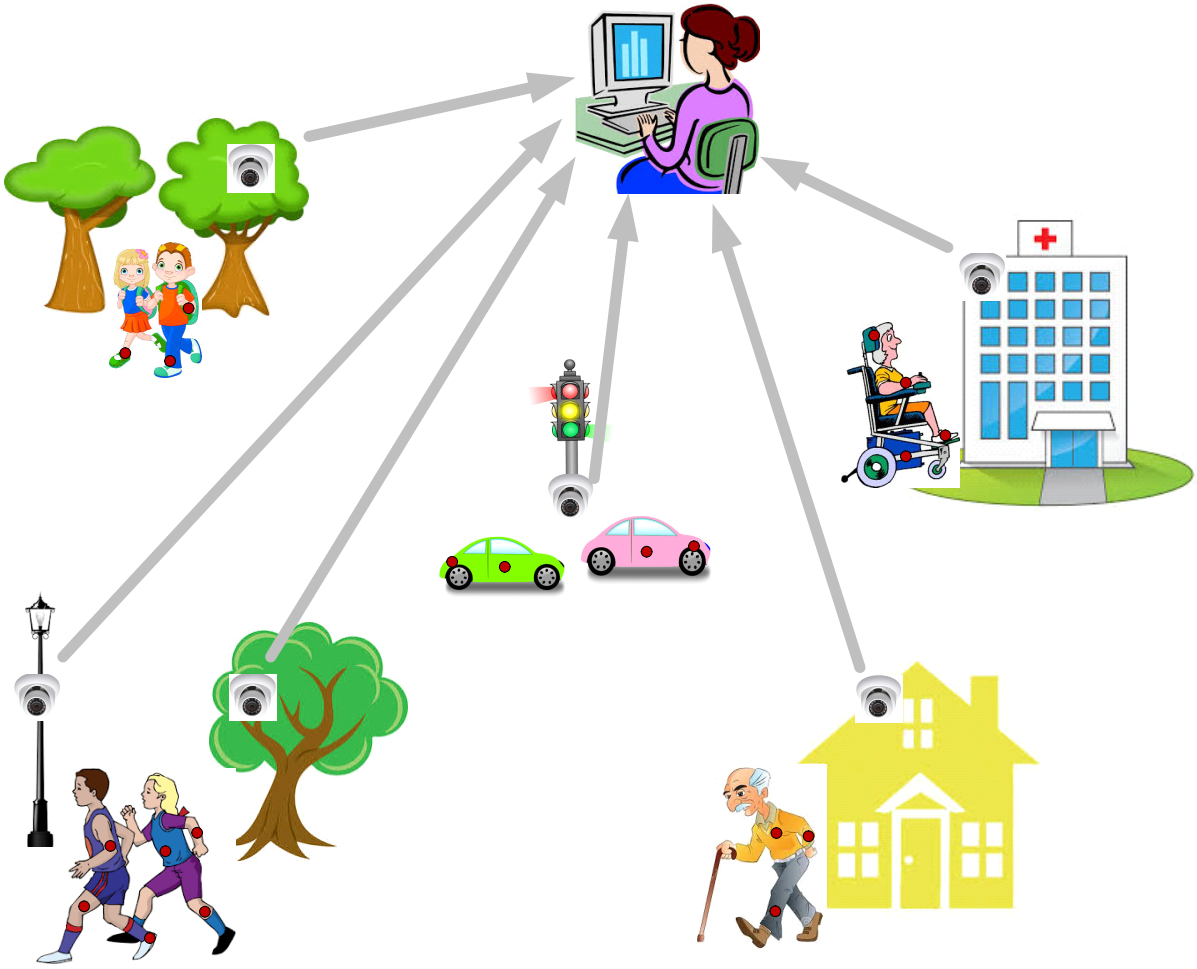

Figure 7: Application scenarios of IoT enabled multimedia sensor networks

3.5.1 Key management

The key management is a major prerequisite to construct secure communication channels among network devices for both unicast and multicast scenarios. However, the resource consuming cryptographic operations on the calibre of the key establishment, key revocation, and key distribution could be unaffordable or remarkably expensive to perform by a very wide range of the resource constrained devices in WMSNs. This would bring an extra overhead to these devices and their normal operations since they exhibit constraints in both computational power and battery capacity. A key establishment occurs only at the initialization phase of a secure communication channel. Later on, the key can be reused until there is a necessity for rekeying. Therefore, a lengthy key establishment process, such as few seconds, is still acceptable as long as it occurs once in a while during the entire operational mode. In certain IoT applications, the multimedia sensor nodes deployed in hazardous environments or battle fields that are difficult or impossible to access frequently, face the challenge of replacing batteries. Under such circumstances, it would be very critical to use high energy efficient security schemes in order to conserve the battery life.

In (Zhang;Li;Yang;Liu;Xiong;& Vasilakos, 2013), the authors present a Real-time Dynamic Key Management (RDKM), a splay tree-based rekey management, which can efficiently enhance the network security and survivability in the LEACH-like protocol. Compared to other LEACH-like security solutions, their work offers some salient advantages. First, it establishes a real-time rekey mechanism based on the access-triggered splay tree architecture. In the mechanism, the keys will be changed during messages exchanging phase. RDKM is the first real-time key management for security in WMSNs. Second, it designs and realizes the rekey mechanism based on the splay tree, which can provide dynamic architecture to generate new keys and make the dynamic key management feasible without any overhead. The novel mechanism can efficiently protect the network against attacks from eavesdropping or captured nodes compromise and address challenging security issues of runtime in WMSNs.

The most consistent candidates that have been currently proposed to establish E2E secured communications among the IoT devices, are Datagram Transport Layer Security (DTLS) handshake, Internet Key Exchange (IKEv2) scheme, and Host Identity Protocol Diet Exchange (HIPDEX) protocol. All these security protocols require the key agreements with the asymmetric cryptographic primitives that have the variants of Diffie-Helman (DH) protocol.

According to the lightweight collaborative key establishment scheme in(Saied;Olivereau;Zeghlache;& Laurent, 2014), a constrained device may delegate its heavy cryptographic load to less constrained nodes in the neighbourhood exploiting the spatial heterogeneity of IoT environment. While initiating a secure E2E connection between two unknown nodes in distinctive networks, they exploit one set of intermediary nodes as proxies in order to support the key establishment process. However, this would not be realistic in the actual scenarios, since both the end nodes, which are completely unknown, may not have the securely pre-established communication links with those common proxies.

3.5.1.1 Proxy-based KMP

Therefore, as shown in

Figure 8: Network system architecture, we consider the E2E key establishment between two highly resource constrained devices (e.g., a scalar sensor and visual sensor) located in distinctive local networks. Each node is capable of formulating logical networks with a set of resource rich devices (e.g., smart phones, PDAs, laptops) in their neighbourhood performing as proxies. This creates two logical local networks (i.e., Network 1 and 2) and the proxies collaboratively support the two nodes for computing the shared secret key. The proxies in Network 1 and 2 can securely communicate with each other since they have enough resources and well-known techniques can be used.

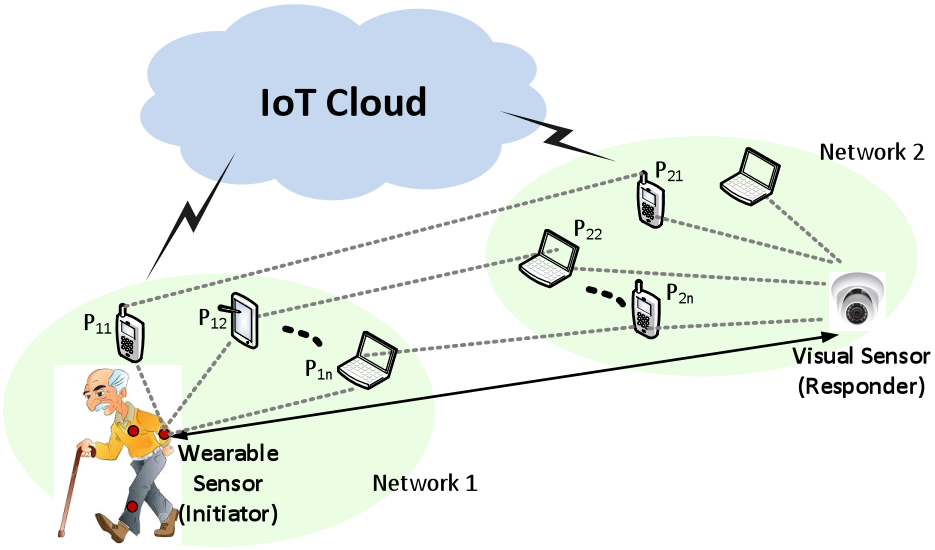

Figure 8: Network system architecture

Host Identity Protocol (HIP) is an IETF standard that establishes secure signalling channels which inherently support node mobility and multihoming. The lighter version HIP Diet Exchange (HIP DEX) is introduced to WMSNs, due to the resource consuming cryptographic operations in the original HIP Base Exchange (HIP BEX). However, the exploitation of static Diffie-Hellman (DH) keying materials in HIP DEX may not be very pragmatic for the mobility and scalability requirements of IoT. We propose a Collaborative HIP (CHIP) solution with an efficient key establishment phase to provide E2E secure connectivity among the resource constrained devices in IoT. Accordingly, the highly constrained device delegates the computational resource demanding cryptographic operations of HIP protocol to resource rich devices in the neighbourhood which are acting as proxies.

In order to evaluate the performance and quantify the energy efficiency of the key establishment component of the proposed CHIP protocol at the responder's side, we implemented the corresponding cryptographic operations on Libelium Waspmote platform using Waspmote cryptographic libraries. Waspmote has an Atmega1281 microcontroller running at 8 MHz with 8 KB SRAM, 4 KB EEPROM, and 128 KB Flash memory. We measure the execution time (t) for individual cryptographic operations and there by calculate the computation energy cost using formula U_I_t based on the execution time (t), the nominal voltage (U), and the current draw in active mode (I) on Waspmote sensors. As given in the data sheet, specifically we select I = 9 mA and U = 3 V. As shown in Figure 8 while increasing the number of proxies involved, the total energy cost for the key establishment component of CHIP grows exponentially.

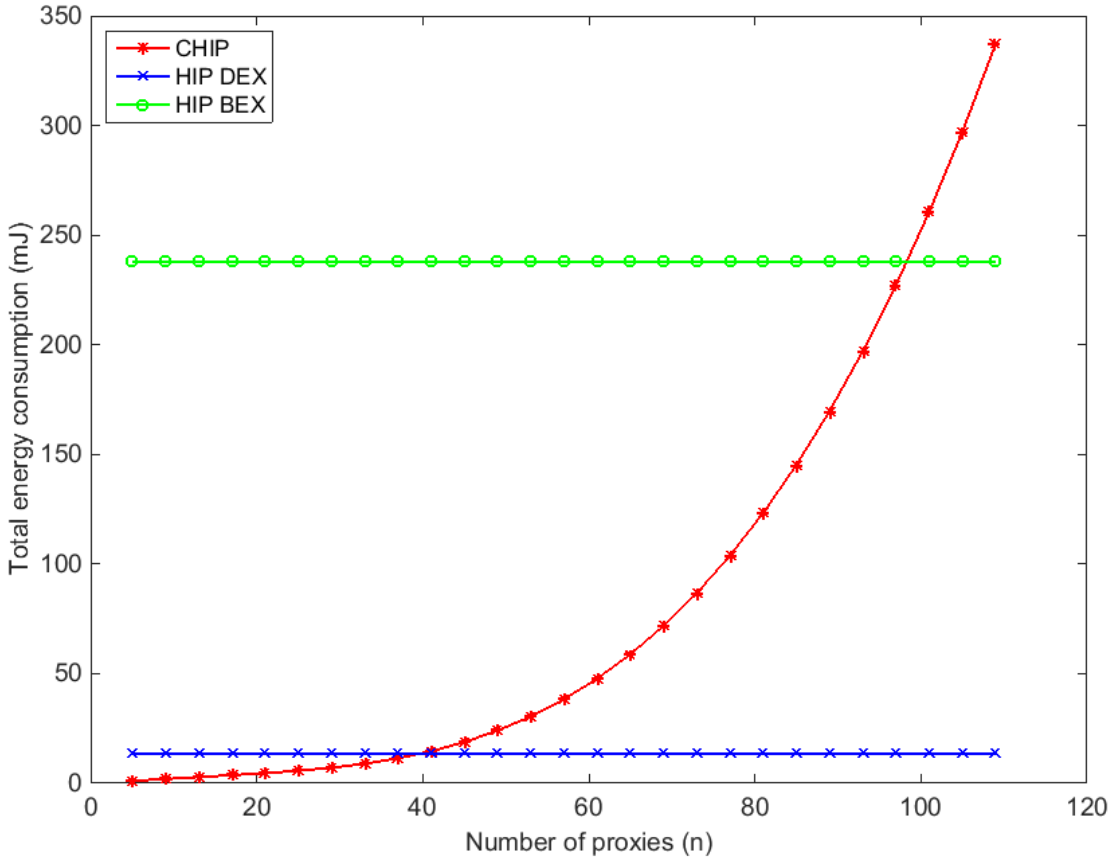

Figure 9: Variation of total energy cost for key establishment with collaborating proxies (n).

The proposed key establishment phase outperforms than that of HIP DEX and HIP BEX when the cooperative proxies in one side are kept below 40. However, involvement of a very low number of proxies may also create a higher probability for them to communicate among the group of proxies and reconstruct the secret key. Therefore, it is important to maintain the balance between the number of involved proxies and the risk they tend to cooperate.

3.5.2 Secure group communication

Collaboration in visual sensor networks (VSNs) typically involves not only 1:1 (unicast) communication, but 1:n (multicast or broadcast) communication where, for example, tracking information has to be distributed to all devices within the immediate neighbourhood(Thomas & Bernhard, 2014). A primary challenge in this context is the management of cryptographic keys required for message authentication and integrity protection. Asymmetric cryptography is an appropriate tool for authentication and integrity protection in broadcast and multicast scenarios. If hardware-accelerated implementations of asymmetric cryptography are available, they can be used to strengthen and simplify the implementation of VSN communication mechanisms. Although such hardware is usually included in state-of-the-art designs, legacy devices might not be powerful enough for asymmetric cryptography. In these situations, symmetric encryption offers far better performance. Keyed hash functions such as HMAC are based on symmetric encryption and shared keys. The fundamental problem in multicast or broadcast scenarios is that the shared key has to be distributed to all members of the group to enable them to authenticate received messages. Being in possession of the shared key enables all members of the group to generate valid messages, which contradicts the idea of individual authentication of messages. Popular approaches that eliminate this shortcoming are the TESLA and uTESLA protocols, which are based on hash chaining and delayed disclosure of the symmetric keys.

3.5.2 Design for secure video delivery in WMSN

Figure 9 illustrates the evaluated network architecture of a visual sensor network that can be considered for a video surveillance applications. The visual sensor network (VSN) exhibits a two tier architecture with both scalar and visual sensors. According to this setting, it is assumed that clustering can be performed according to the field of view of the visual sensors as one VS per cluster. Each visual sensor is responsible for reacting to the trigger-signals coming from the scalar sensors from the given cluster and then capturing the videos during the time intervals accordingly. The visual sensors encrypt those interval-videos with the ephemeral interval-keys and transmit to the mobile agent (MA) which is performing as a server as shown in Figure 9 (e.g., mobile phone, laptop or tablet with the internet connectivity). The server (MA) first needs to derive the interval-keys with the consent of sufficient number of corresponding users (scalar sensors) and then decrypts the video.

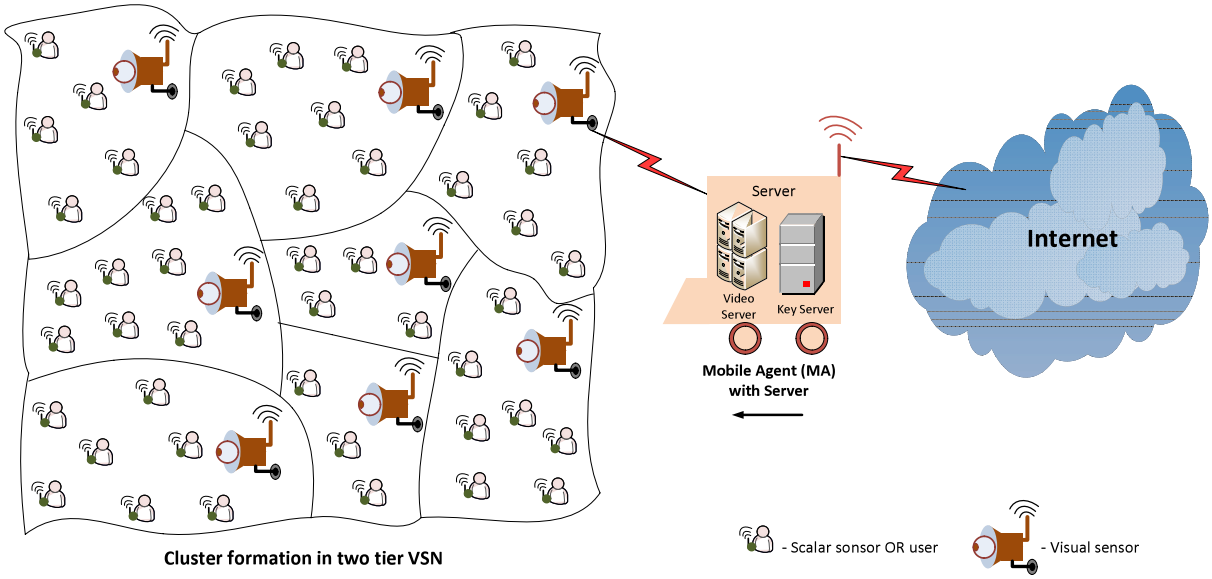

Figure 10: Evaluated network architecture

During the key-distribution phase, all the nodes in the WMSNs should acquire the corresponding cryptographic keys from key distribution centre (KDC), Figure 10. For a given cluster in the WMSN with one visual sensor and m scalar sensors, The KDC generates a key-pair (*d, e*) for the ELGamal cryptosystem: *d* and *e* are private and public keys. The shares of the private key (d_1, d_2, …, d_m) and the public key $e = gd$ are respectively delivered to the scalar sensors and the visual sensor in a secure manner. The secret key *d* is generated following a t-degree polynomial *f()* (i.e., $d = f(0)$) and d_i shares are computed.

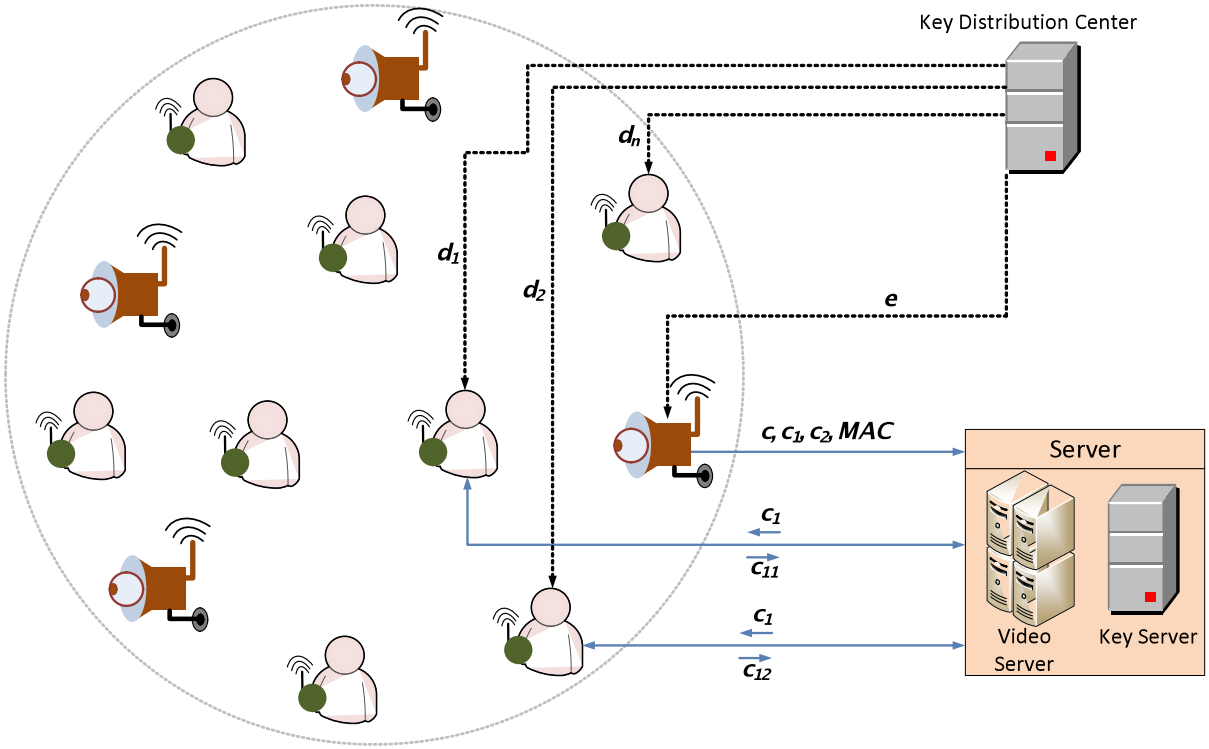

Figure 11: Security architecture and message flow of evaluated solution.

3.5.2 Implemented Approach

University of Oulu has implemented a video surveillance system that is based on a multi-tier wireless multimedia sensor network. Figure 11 illustrates the setup. The multi-tier operation enables keeping the camera sensor nodes in a sleep mode for most of the time and waking them up using low-power motion sensor nodes only when activity is detected. The first version of the system was based on Libelium Waspmotes, but later the camera sensor nodes were replaced by more powerful Raspberry Pi embedded computers that allowed capturing and streaming Full HD video content.

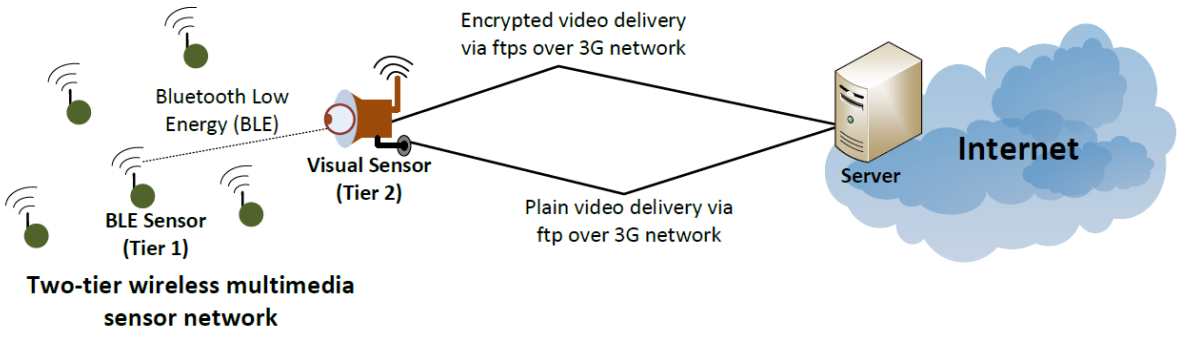

Figure 12. Two-tier WMSN scenario.

The security (T4.3) related study using the WMSN system included two sets of measurements: one comparing the encrypted FTPS protocol versus unencrypted FTP protocol when sending still images from WaspMote-based camera sensor nodes to the server, and another comparing the encrypted HTTPS protocol versus unencrypted HTTP protocol when streaming security camera video from Raspberry Pi -based camera sensor nodes to the server. The main observation was that using secure protocols do not inflict significant increase in the energy consumption. Using FTPS instead of FTP protocol in transferring still images increased the total energy consumption by around 2%, and

using HTTPS instead of HTTP in streaming video increased the total energy consumption by around 1-5%.

4 CONCLUSIONS

We have showed our initial findings about how security and privacy algorithms affect the energy consumption of video streaming. Based on this we can see that depending on which algorithms and protocols that are used, different amounts of energy will be consumed. By considering these differences we have outlined methods and designs for how these can be considered in order to make more energy aware decisions while designing the security aspects of a video delivery network. Future will investigate these designs further along with more specific energy consumption measurements.

5 BIBLIOGRAPHY

Akyildiz, I.;Dai, R.;& Wang, P. (2007-2010). *Broadband Wireless Networking Lab*. Noudettu osoitteesta Wireless Multimedia Sensor Networks: http://www.ece.gatech.edu/research/labs/bwn/WMSN/index.html

Almalkawi, I.;Zapata, M. G.;Al-Karaki, J. N.;& Morillo-Pozo, J. (2010). Wireless Multimedia Sensor Networks: Current Trends and Future Directions. *Sensors* , 6662-6717.

Dai, R.;& Akyildiz, F. (2009). Joint Effect of Multiple Correlated Cameras in Wireless Multimedia Sensor Networks. *Proc. IEEE ICC.*

Ghadi, M.;Laouamer, L.;& Moulahi, T. (2015). Securing data exchange in wireless multimedia sensor networks: perspectives and challenges. *Multimedia Tools and Applications* , 1-27.

Guerrero-Zapata, M.;Zilan, R.;Barceló-Ordinas, J. M.;Bicakci, K.;& Tavli, B. (2010). The future of security in Wireless Multimedia Sensor Networks. *Telecommunications Systems* , 77-91.

Saied, Y. B.;Olivereau, A.;Zeghlache, D.;& Laurent, M. (2014). Lightweight collaborative key establishment scheme for the Internet of Things. *Computer Networks , 64* (0), 273-295.

Sharif, A.;Vidyasagar, P.;& Elizabeth, C. (2009). Wireless multimedia sensor network technology: A survey. *7th IEEE international conference on Industrial Informatics*, (ss. 606-613).

Thomas, W.;& Bernhard, R. (2014). Security and Privacy Protection in Visual Sensor Networks: A Survey. *ACM Computer Survey , 47* (1), 2:1-2:42.

Yiying, Z.;Xiangzhen, L.;Jucheng, Y.;Yuanan, L.;X., N.;& Athanasios, V. (ei pvm). A real-time dynamic key management for hierarchical wireless multimedia sensor network.

Zhang, Y.;Li, X.;Yang, J.;Liu, Y.;Xiong, N.;& Vasilakos, A. V. (2013). A real-time dynamic key management for hierarchical wireless multimedia sensor network. *Multimedia Tools and Applications , 67* (1), 97-117.

Zhou, L.;& Chao, H.-C. (2011). Multimedia traffic security architecture for the internet of things. *IEEE Network* , 35-40.

Kessler, Gary C. "An overview of cryptography." (2003).

Kaliski, Burt. "PKCS# 1: RSA encryption version 1.5." (1998).

Turner, S., and Polk, T. "RFC 6176." *USA: IETF* (2011).

Potlapally, Nachiketh R., et al. "Analyzing the energy consumption of security protocols." *Proceedings of the 2003 international symposium on Low power electronics and design*. ACM, 2003.