DTLS for Lightweight Secure Data Streaming in the Internet of Things

Roy Fisher[1, 2], Gerhard Hancke[1]
[1]University of Pretoria, Pretoria, South Africa
[2]City University of Hong Kong, Hong Kong
u29237654@tuks.co.za, gp.hancke@cityu.edu.hk

JDIM
Journal of Digital Information Management

ABSTRACT: *The huge number of sensors envisioned to be deployed within Internet of Things applications will result in large amounts of likely confidential data that will be required to traverse the open Internet. This confidential data could range from the current security status of a smart house to the current health status of an elderly person connected to smart monitors. A secure and able connection will be required to connect either a cloud based or an in-house server to the embedded sensors. The ability to ensure that the data generated by the sensor networks is securely streamed to the central server will allow for real time access to the data. The use of the less favoured Datagram Transport Layer Security protocol is compared with the widely used Transport Layer Security protocol. Additionally, the use of Internet Protocol Security is investigated as an option. A comparison of the functioning and capabilities of Internet Protocol Security is also completed. The research shows that Datagram Transport Layer Security is a capable alternative to Transport Layer Security when it comes to streaming data across the Internet from connected Wireless Sensor Networks.*

Subject Categories and Descriptors:
C.2.6 [Internetworking]: Standards; **C.2.2 [Network Protocols]** C.2 [Computer-Communication Networks] Security and protection

General Terms: Internet of Things, Internet Protocols, Internet Security

Keywords: Internet of Things, Transport Layer Security, Datagram Transport Layer Security, Secure Streaming, IPSec

Received: 19 February 2015, Revised 29 March 2015, Accepted 3 April 2015

1. Introduction

The Internet of Things is the paradigm where everything and everyone will constantly be connected to the Internet [1]. This constant connection will be achieved by embedding Internet communication capable sensors within the environment in which people live and work. The ability to interact with the physical world through "smart" devices across the Internet could have a range of varied applications [2]. In this paper the term "smart device" describes a device with the following features, this is very similar to the definition as found in literature [2]:

- Has a set of physical features
- Minimal set of communication facilities
- Possess a unique identifier (some form of unique identification)
- Possess some basic computing capabilities
- Possess means to sense physical phenomena (change of temperature etc) or interact with the surrounding environment

Using these smart devices within the home environment could have many potential benefits. The main potential is for a smart home system to control energy use during "low-usage" times of the day [3, 4]. The times that can be defined as low usage are when the family members are either at work or school as well as when they are sleeping. The smart home wireless sensor network could also be used to manage the security of the household and monitor the current occupants of the house. Using motion

detection and similar technologies a video feed could be forwarded to the home owner when motion is detected and the house is supposed to be empty [5]. Using motion detection and facial recognition a smart home could also control the lighting within a house. Allowing for each occupant to have predefined light levels and enabling the home to switch the lights on and o as the occupants enter or leave the rooms. The data, this could include media and video as well as text based updates from embedded devices, that will be required for this type of smart home based product will require a large amount of information to be captured and forwarded across the Internet. This data will typically be in the form of video data.

Another possible home application of the Internet of Things is the connected healthcare network [6]. This application is specifically useful for management of elderly individuals health requirements. This technology could allow for fewer doctor visits, thereby reducing medical costs for elderly people. A dedicated center could be established that automatically monitors the health data received from the Internet capable Wireless sensor network. If an emergency situation is detected a response team can be dispatched automatically by the system. This system is not only for emergency situations but constantly receiving and keeping a record of blood pressure, blood sugar and heart rate over the course of a day can greatly assist doctors with diagnosing health issues.

To enable these applications to function across the Internet and thereby allow for an upgrade from an embedded Wireless Sensor Network to the Internet of Things a suitable solution for connecting these devices to the Internet needs to be utilized. A solution that has been proposed is to have an Internet capable border routing device that allows for the seamless conversion of the embedded sensor communication technologies (802.15.4) to the more common IP based communication technologies (ethernet/802.3) [7].

1.1 Security Concerns

These applications discussed above have two similar traits: they require fast access to the data (which can be large) and the information is potentially highly sensitive. With the increase in public awareness on issues, such as data security, more and more ordinary people may not consider an application that offers no protection for the personalised data it generates as it moves across the Internet. There are many security issues that could possibly affect data that traverses the Internet. Some of the specific threats to data in Internet of Things applications are: man-in the-middle attacks, eavesdropping, denial of service, traffic analysis, spoong and compromised key attacks [8]. A man-in-the-middle attack is an attack that allows for the attacker to view the data that is sent across the Internet without the individuals involved in the communication being aware of the intrusion into their privacy, similar to the eavesdropping attack. This is an attack that can have an effect on the transmitted data other than allowing the attacker to view the information that is transmitted unlike an eavesdropping attack. By altering the data that is transmitted and retransmitting this false data the attacker can completely alter any of the communication. A denial of service attack results in the communication between the two intended participants being stopped due to a flooding of basic networking packets. This prevents any data being communicated between the two participants. These are only two of the available attacks that are available to an attacker of an Internet of Things application. These two classes of attacks however are capable of creating huge problems for a deployed Internet of Things application. Stealing, changing or denying the data that is an integral part of any application deployed within a Wireless Sensor network or Internet of Things application will have a huge negative impact on the entire deployed application.

Most of the attacks above focus on attacking and gaining access to the data that is transmitted. Two specific attacks focus more on disrupting and gaining information about the data that is transmitted. These are the denial of service (disrupting the communication) and traffic analysis (finding out information about the pattern of data transmission). Traffic analysis could give a potential criminal details as to when the occupants of the house are home and what their daily schedule involves. To prevent any traffic analysis the gateway device could randomly send dummy packets. Denial of service is a simple attack with powerful effects. In the home healthcare scenario a denial of service attack could potentially lead to a death. In the home security system a denial of service could allow for a criminal to steal at will. A denial of service attack is generally very difficult to prevent and a coordinated effort with an Internet Service Provider is required.

The rest of the attacks described above deal with either gaining access to, or changing the data that is communicated across the Internet. These are attacks that are not specific to the Internet of Things applications and can therefore be dealt with using standard technologies. The standard protocol used to secure data transmission across the Internet is Transport Layer Security (TLS). This protocol is used to secure everything from banking transactions to instant messaging communication [9]. The disadvantage of the TLS protocol is that it was designed to be used on standard computing devices that generally have high power and are not generally implemented on time critical applications. It was therefore designed to make use of the TCP (Transmission Control Protocol) which is designed to be reliable and ensure delivery. To enhance the protocols use in time critical applications the Datagram Trans- port Layer Security Protocol (DTLS) was created [10]. The DTLS protocol is designed to make use of the UDP (user datagram protocol) which is known as an unreliable protocol and is therefore designed instead to favour data throughput over ensuring the message is received. Very few major changes exist between the implemented operation of TLS and DTLS. This is due to

the fact that DTLS was designed to mimic the operation of TLS, in order to replace it within certain time critical applications. The DTLS protocol has numerous advantages and these will be discussed throughout the rest of the paper. It will be determined if the DTLS protocol is more suited to the home gateway scenario over the more traditional TLS protocol. The protocols are going to be compared for the purpose of securing the data transmissions for a limited, in terms of resources, Internet of Things application. Internet Protocol Security (IPSec) has been widely used in Virtual Private Network technology for many years and is considered as a possible end-to-end alternative to the widely implemented SSL. Due to the additional requirements, and its unversatile nature, IPSec has not been as widely implemented for use across the Internet. The term unversatile is used as IPSec is really designed to perform a slightly different task to SSL. IPSec operates across all network traffic it is implemented at the network layer of the OSI model. SSL is implemented at the session layer as is therefore far more applicable to a single communication session, encrypting only the information during that session. This makes SSL more versatile in its possible applications [11].

The rapid growth in the number of applications making use of Internet of Things technologies means that security concerns are of a high priority. These Internet of Things applications will be widely deployed and could potentially contain a huge amount of sensitive data that the developers of these applications will not want to fall into the wrong hands. The decision to implement DTLS, TLS or IPSec could have a huge effect on the final capabilities of the Internet of Things application. This decision could greatly impact the throughput capabilities of the Internet of Things application.

The rest of the paper will detail the process of testing the individual security approaches available to developers of Internet of Things streaming applications. The tests will be performed between both local and international communication partners. These tests will be completed to confirm the capabilities of the three security protocols within typical Internet of Things application zones. Each of the proceeding sections will cover one of these topics.

2. Secure Data Transmission

Secure data communication requires the fullfilment of three primitive security objectives [12]. These objectives are commonly referred to as the CIA triad and consists of:

- Condentiality
- Integrity
- Authentication

For an Internet based application to be considered secure at least the primitive security objectives need to be met [8]. The basis for most secure communication across the Internet is the TCP based Transport Layer Security (TLS) protocol [9]. This protocol provides for the primitive security objectives by using built-in features of the protocol. Confidentiality is provided for with the use of cryptographic techniques, integrity is provided using secure hash functions and authentication is provided using digital certificates. The nature of the underlying TCP connection ensures that any data that is sent is reliably received. This reliability introduces additional overhead onto the TLS communication as acknowledgements need to be sent to each received message as well as additional messages that ensure that the communication is reliable.

TLS operates through the use of four sub protocols [12], namely:

- **Record Protocol**
- **Handshake Protocol** - Performs initial handshake and sets up connection
- **Alert Protocol** - Any change in connection parameters or errors received
- **Change Cipher Spec** - Used to change the cipher type of the client and server

The Record protocol is the basic messaging protocol passed by TLS and contains either user generated data or the messages passed by the other three protocols.

Recently the development of a new protocol to oset the increased cost acquired when using TLS has been developed. This protocol is based on the UDP protocol instead of the more popular TCP protocol. The protocol is known as Datagram Transport Layer Security (DTLS) [10]. The initial developers of the DTLS protocol designed it to imitate as closely as possible the operation of TLS [13]. A few amendments to the TLS protocol are required, for example: when completing the secure key transfer for confidential encryption, instead of being able to depend on the reliable transport of the key between the communicating devices; the protocol needs mechanisms to ensure that the key is received. The UDP protocol does not cater for acknowledgement messages from the recipient and therefore in the case of DTLS, timers are implemented to continually repeat the transmission of the key until the next stage of the protocol is received, allowing the communication to proceed. For general data transmission there is no ability to ensure that the data arrives. The sending device sends the data across the medium and then moves on.

The DTLS protocol was developed with standard Internet devices in mind and not with a direct application into the Internet of Things. The aim of the protocol was to replace TLS based communication in circumstances where time constraints were critical, such as multi-player gaming [10]. Although not specifically designed for application within the Internet of Things the protocol does hold the possibility of being applied within the Internet of Things due to its lighter weight when compared to the more standard TLS protocol. This may allow for a higher data throughput when

applied within a low resource environment.

Another form of end-to-end communication protocol is known as IPSec [11]. This is a method of securing communication that is truely capable of completely securing the communication. IPSec is capable of operating in two modes: transport mode and tunnel mode. In transport mode the payload of the IP packet is encrypted. This is similar in operation to that of DTLS and TLS where the application data is encrypted only. It provides for confidentiality, integrity and authentication. In tunnel mode the entire IP packet is encrypted and then placed inside a new ly made packet with a destination header added. This provides for CIA as well as anonymity of the communication. In order to correctly operate IPSec relies on three primary security architypes to achieve its goals. These are: encapsulated security payloads (ESP) which are responsible for confidentiality and authentication; authentication headers (AH) that are responsible for providing connectionless integrity and data origin authentication and finally the set of security association tools, these provide for the extra implementations that allow the AH and ESP to properly function. The most commonly used SA is IKE (Internet Key Exchange), which facilitates secure key management across a unsecure network. IPSec is commonly used in applications such as virtual private networks (VPNs). These provide a tunnel of secure communication in which the sent data can be s afely and securely transmitted without external interference or recording.

3. Streaming in the Internet of Things

Many applications within the Internet of Things could require the streaming of data from a device embedded within the environment to a centralized server. One such application could require the embedded sensor to send temperature updates to a centralized server so that the temperature can be carefully controlled within a laboratory space. With in a city environment the embedded sensors will generate a huge amount of data [2], from sources that range from traffic monitoring to water and electricity management systems [14]. The ability to quickly and securely send this data will be key in ensuring that the Internet of Things gains even more traction within the public sphere and within industry applications.

The current design of the Internet of Things using border routing devices will require that the throughput capabilities of the border router are sufficient to successfully transmit all the data that is being received. If a single border router is used to transmit the sensor data across the network a powerful device will be required. A more sensible solution may be to use a number of inexpensive devices that can manage and transmit a smaller subnetwork of data to the central servers. These are generally considered to be high power cloud based solutions with the capability of handling multiple communication streams simultaneously. This model would require more border routers but the devices would require less throughput capability and would thus be cheaper.

3.1 Secure Streaming from a Border Routing Device

The proposed approach is to use the Raspberry Pi (which is an inexpensive ARM based device) as the embedded networks border router. The Raspberry Pi is a lightweight device with suitable expansion capabilities for bridging the wireless sensor network technologies (802.15.4 networks) with the more common ethernet Internet Protocol (IP) based networks [15]. This would require the Raspberry Pi to implement the required security protocol, either TLS, DTLS or IPSec, and forward the secure data to the central server. Each of the proposed security mechanisms has a number of advantages and weaknesses when applied to the operation of a border routing device within the Internet of Things.

A number of important concerns need to be considered when applying any of these security approaches to the communication from the Raspberry Pi. The primary concern is the low resource nature of any devices that participate in traditional Internet of Things or Wireless Sensor Network applications [16]. The limited hardware capabilities will have a direct effect on the intensive calculations completed when security is applied to any form of communication. A limited processor speed and relatively low amount of RAM will greatly reduce the speed and capability of the devices involved in the communication. Additionally the limitations extend further than the simple hardware directly connected to the devices. In Internet of Things applications the power provided to many of the devices is also considered to be a limited resource. Applications can be deployed using solar power, battery power or even mains power. The wide range of application types means that power requirements are often closely scrutinised and maintained within an Internet of Things application. This lead us to perform a study of the power effects that each of these security mechanisms have when they are applied to the devices.

The proposed lightweight solution involves the use of the Raspberry Pi and a suitable lightweight security protocol that will allow for the protection of the data across the network.

A comparison of the performance of TLS and DTLS was required. The embedded sensor networks data stream would be simulated using random data created by the Raspberry Pi. Two sets of tests were completed. One set of tests was to test the throughput capability of the Raspberry Pi when streaming data to a local network centralized server. The data was “received” (random data) at the Raspberry Pi and through a secure connection was streamed to the server. This set of tests was to confirm the capabilities if the Raspberry Pi when streaming data across a stable and simple network. These tests also give the best case operation of the security protocols. This can be described as best case as minimal opportunities are presented for packet loss and the protocols have minimal delay due to server latency. The

client and server are geographically locateA comparison of the performance of TLS and DTLS was required. The embedded sensor networks data stream would be simulated using random data created by the Raspberry Pi. Two sets of tests were completed. One set of tests was to test the throughput capability of the Raspberry Pi when streaming data to a local network centralized server. The data was "received" (random data) at the Raspberry Pi and through a secure connection was streamed to the server. This set of tests was to confirm the capabilities of the Raspberry Pi when streaming data across a stable and simple network. These tests also give the best case operation of the security protocols. This can be described as best case as minimal opportunities are presented for packet loss and the protocols have minimal delay due to server latency. The client and server are geographically located within the same room. The server is located on a high speed WIFI network and the Raspberry Pi (client) is located on a wired network. The two networking technologies connect via a router. d within the same room. The server is located on a high speed WIFI network and the Raspberry Pi (client) is located on a wired network. The two networking technologies connect via a router.

The second set of tests were designed to test the reliability and speed of both DTLS and TLS from a client connected in South Africa to a server based on the Amazon Elastic Cloud Compute (EC2) in the Eastern North America region. These tests tested a two way communication stream. Data was sent by both the client (Raspberry Pi) and the server. EC2 was chosen as the host for the server as it is free for a certain length of time and it can be guaranteed where the server is located.

The tests involved the use of a C++ client and server that are able to independently adjust the packet size of the transmitted data as well as the length of the test. One set of client/server was written for DTLS and one was written for TLS. The testing then involved the manipulation of these values to determine the effect this will have on the communication stream in terms of throughput. In order to test the IPSec protocol a virtual private network (VPN) service was hosted on a server and the Raspberry Pi made use of the built in Linux VPN capabilities to test the throughput of the system. All of these tests were then repeated for a connection which made use of a local network and then a international global network was tested. More information is available below regarding each of the individual tests completed.

3.2 Secure Local Streaming Test

The secure local streaming test was constructed using a TLS or DTLS client and corresponding server; written in the C programming language using the OpenSSL security toolkit [17]. The server was hosted on a high power desktop machine. Two sets of tests were run to determine the throughput capability of the protocols and the Raspberry Pi's on a local network. The initial test altered the packet size of the data portion of the TLS and DTLS Record protocol.

The test was conducted for ten seconds in which the total number of packets sent and received was recorded. The results of this test can be seen in figure 1.

These results indicate that the Raspberry Pi is capable of achieving a maximum throughput rate of 6 324.48 KBps for DTLS communication and 3 146.24 KBps for TLS communication. Table 1 shows the percent of packets that are lost during the course of the communication. This shows that for the smaller packets the communication is less reliable and for the larger packets the converse is true. This is to be expected due to the congestion that occurs on the communication medium. This congestion is also the reason that the TCP based TLS performs better when the packet sizes are smaller. The smaller packet size means that there is a larger number of individual packets. TCP has built-in features to manage flow control (congestion control) this allows for the better handling of the smaller packet numbers [18].

The 16 000 byte packet size was chosen due to the maximum packet size for DTLS and TLS being $2^{14} = 16384$ bytes [19]. It can be seen that both protocols have a higher throughput capability when dealing with a larger packet size. This is also due to the bottleneck forming onto the connection medium. A large number of very small packets arriving very quickly puts the lower level hardware under pressure whereas for the larger packets moving the data through the lower levels of the Open Systems Interconnect model seems to perform an automatic flow control.

Once the packet sizes reach into the kilobytes the advantage gained through control mechanisms by TLS is lost and the nature of the UDP based DTLS leads it to be far superior in terms of throughput. From a packet size of over 5 000 bytes the throughput is around 100% better for UDP based DTLS than the TCP based TLS. This is attributed to the reliable nature of the TLS protocol and the requirement for a response to be received for each transmitted packet.

The Raspberry Pi has a 100 Mbps Ethernet port onboard. This device is theoretically capable of achieving a data throughput of 12 500 KBps. However in practice the overhead of all the networking protocols often results in a lower actual throughput. For the used switch the speed rating can not be found. The impact from DTLS and the overhead of underlying protocols forces the device to only operate at around 60% efficiency. As a comparison result high definition video streaming requires around 2 400 kbps [20].

The next set of tests that where completed where done to ensure that the small testing period of 10 seconds did not negatively affect the results achieved. The best performing packet size was run over three different time periods, namely: 30 seconds, 60 seconds and 120 seconds. The results of these tests can be found in figure 2. Running over these lengths allows us to ensure that the data stream and communication medium are stable. As can be seen

these results confirm the results achieved by the previous tests. It is also seen that the DTLS communication seems to become more efficient with time as the longer the connection is kept available the higher the throughput is achieved.

After successfully testing the capabilities of both DTLS and TLS the final test that was completed was to examine the capabilities of IPSec as a security protocol. The setup involved a Raspberry Pi connected in point-to-point mode directly to a laptop and the Iperf tools where used to perform

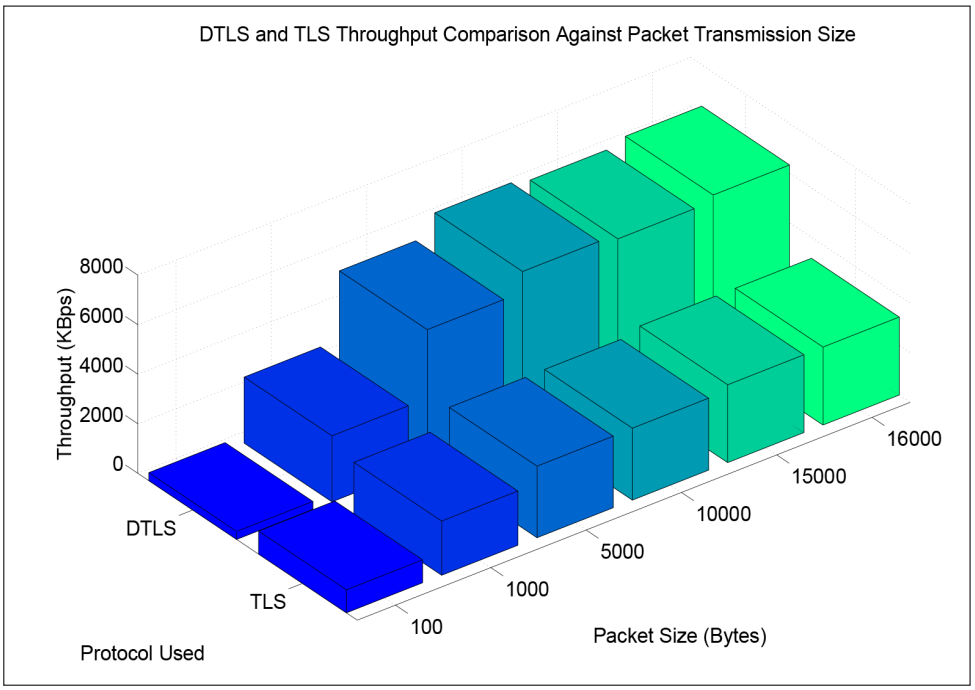

Figure 1. DTLS versus TLS throughput for different run lengths

Packet Size (Bytes)	Percent Lost
100	5%
1000	6%
5000	1.5%
10000	1%
15000	<0.5%
16000	<0.5%

Table 1. Table showing DTLS packet sizes against percentage of dropped packets

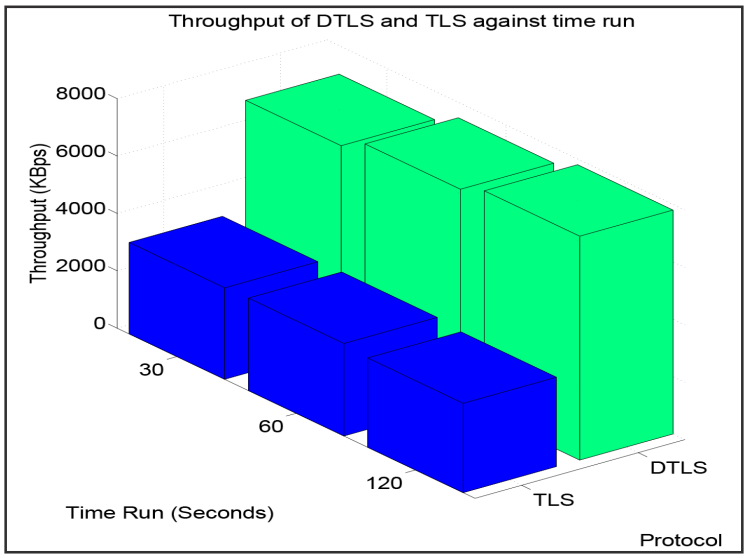

Figure 2. DTLS versus TLS throughput for different run lengths

Protocol	Throughput (KBps)
TCP	2450
UDP	131.26

Table 2. Table showing the IPSec throughput results

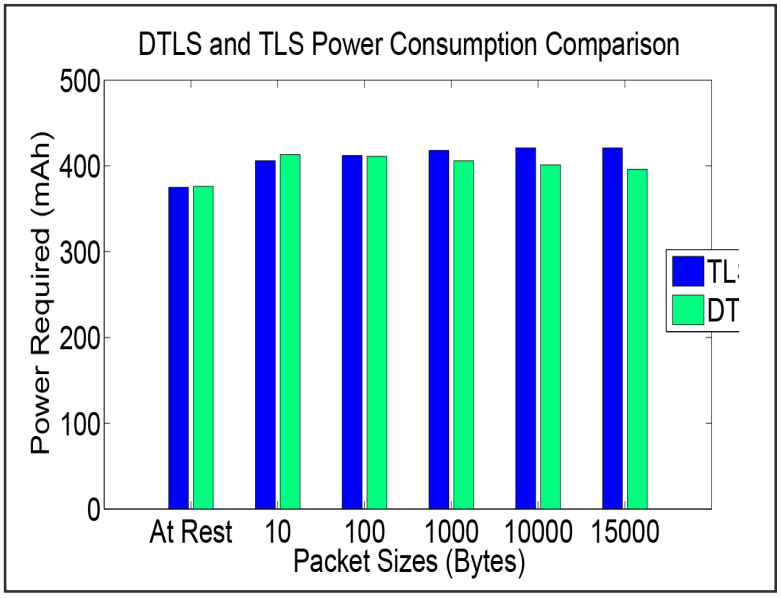

Figure 3. DTLS and TLS power required

Protocol Power	Consumed (mAh)
TCP	415
UDP	391

Table 3. IPSec Power Consumed

the tests on the connected network. In order to maintain parity between the tests the exact same network setup as the other tests was used [21]. These final tests resulted in the throughput capability for the IPSec protocol as shown below in table 2. The tests were performed in two ways. The first way was to use the Raspberry Pi as the server device and then to use the Rasoberry Pi as the client device.

It is expected that the protocol will perform worse than either DTLS or TLS. This is due to the previous research that has been completed that shows that this protocol performs significantly worse [11]. The results show that the capability of the Raspberry Pi to communicate using IPSec are greatly hampered when implemented in the Internet of Things. The greatest surprise is the difference between the TCP and UDP protocols. This can only be attributed to the flow control that exists built in to the TCP protocol. The use of IPSec is possible and the performance is similar to that achieved for TLS based communication

3.3 Secure Global Streaming Test

The next set of tests completed involved the testing of a communication stream across a continent. The setup for this program was altered slightly. Instead of performing a simple stream of output from the Raspberry Pi. The Raspberry Pi is also required to receive some data from the server allowing for two way communication. Due to the nature of TLS being a reliable communication protocol the synchronisation of the send and receive of the client and server are carefully controlled and blocking read and write were implemented.

The TLS security protocol achieved an average of 112 packets of size one hundred bytes sent in 30 seconds. In 10 seconds 33 one hundred byte packets were sent. The DTLS protocol achieved 597 one hundred byte packets in 10 seconds with a 30% packet loss percentage. In 30 seconds 1 272 one hundred byte packets were sent with a 29% packet loss.

The first major concern was the extraordinary low amount of data transmitted by the TLS protocol, however this is deemed to be as a result of the high latencies experienced between communicating devices in South Africa and North America where latencies in excess of 300ms are common [22]. Another concern highlighted by the results above are the high amounts of packet loss during communication using the DTLS protocol. These percentages could be as a result of the poor broadband connections and network

distances when communicating with North American servers from South Africa. Additionally this could be as a result of the low quality server supplied when using the Amazon EC2 free tier.

3.4 Power Requirements

As discussed previously the power requirements of the implementation will be of the utmost importance. Being able to ensure that the power requirements are met and the power is well controlled will allow for a more robust application.

In order to understand what effect each of the protocols have on the application power requirements a study into this was completed.

Figure 3 shows the impact that each of DTLS or TLS protocol have on the power consumption of the Raspberry Pi. These results show some surprising results. One of the important things to note is the differing power charts. DTLS has a lower power consumption as the packets increase whereas TLS power consumption increases as the packet sizes increase.

Table 3 shows the power consumption of the Raspberry Pi when applying IPSec across either of the two protocols. Due to the performance tool used, the comparison for IPSec only occurs between the two protocols. Unlike the other comparisons completed the measurements do not occur across ranging packet sizes. We can see that, similar to the other tests completed, the UDP protocol outperforms the TCP protocol for implementations. The two protocols perform similarly but the UDP protocol again shows its capability to be used in place of the TCP protocol and its capability to be applied within the Internet of Things.

We can also see that, the differences between the security options available, in terms of the power used, do not have that much of an impact when applying different security tools. The major concern is the amount of throughput that we are able to achieve and the power analysis shows us that the capabilities of the Raspberry Pi as a gateway device are possible. It is important to note that this comparison has not had the opportunity of comparing other major gateway devices against the Raspberry Pi..

4. Conclusion

The results provided by the streaming tests performed show that using larger packet sizes allows for a higher throughput on border routing devices for a secure Internet of Things application. It also shows that if the application requires for small packets of data to be sent, the Transmission Control Protocol based Transport Layer Security allows for a higher throughput than the User Datagram Protocol based Datagram Transport Layer Security. IPSec does not have similar performance capability to the other protocols and we can see why the many applications currently run on Transport Layer Security. If the application requires larger packets and a higher data throughput it is better to secure the communication using Datagram Transport Layer Security.

This investigation also shows that the Raspberry Pi deployed as a gateway device is capable of streaming secure data to a centralized server. The power requirements additionally show that the Raspberry Pi performs better when using Datagram Transport Layer Security.

A more thorough investigation into both the large number of packet loss for Datagram Transport Layer Security and the low number of packets sent and received for Transport Layer Security when communicating with internationally hosted servers needs to be completed. This investigation should include a comparison between a server hosted locally (in South Africa) versus a server hosted internationally.

The research shows that securely streaming data collected by an Internet of Things application is achievable when done using a local or in-house server. The streaming rates achieved allow for the streaming of a gigabyte of data in just over two minutes. In home applications this is a capable enough speed.

Acknowledgment

This work is based on the research supported in part by the National Research Foundation of South Africa (Grant reference TP1207183332) and our industry partners Telkom, Bytes Universal Systems, Tellumat and EMC. The grant holder acknowledges that opinions, findings and conclusions or recommendations expressed in any publication generated by the NRF supported research are that of the author(s), and that the NRF and our industry partners accept no liability whatsoever in this regard.

References

[1] Atzori, L., Iera, A., Morabito, G. (2010). The Internet of Things: A Survey, *Computer Networks*, 54 (15), 2787- 2805, October.

[2] Miorandi, D., Sicari, S., De Pellegrini, F., Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges, *Ad Hoc Networks*, 10 (7), 1497-1516, September. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1570870512000674

[3] Wang, M., Zhang, G., Zhang, C., Zhang, J., Li, C. (2013). An IoT-based appliance control system for smart homes, in 2013 Fourth International Conference on *Intelligent Control and Information Processing (ICICIP).* IEEE, June. 744 -747.

[4] Saad al sumaiti, A., Ahmed, M. H., a. Salama, M. M. Smart Home Activities: A Literature Review, *Electric Power Components and Systems*, 42, 3 - 4, 294-305, March 2014.

[5] De Silva, L. C., Morikawa, C., Petra, I. M. (2012).

State of the art of smart homes, *Engineering Applications of Artificial Intelligence*, 25 (7), 1313-1321, October.

[6] Liu, R., Wang, Y., Shu, M. (2013). Internet of Things Healthcare Cloud System Based on IEEE 802.15.4, *Journal of Applied Sciences*, 13, (9) 1582-1586.

[7] Zhu, Q., Wang, R., Chen, Q., Liu, Y., Qin, W. (2010). IOT Gateway: Bridging Wireless Sensor Networks into Internet of Things, *In*: 2010 IEEE/IFIP *International Conference on Embedded and Ubiquitous Computing*. IEEE, December 2010, 347-352.

[8] Shafi, Q. (2012). Cyber Physical Systems Security: A Brief Survey, in 2012 12th International Conference on *Computational Science and Its Applications*. IEEE, June, 146-150.

[9] Suo, H., Wan, J., Zou, C., Liu, J. (2012). Security in the Internet of Things: A Review, *In*: 2012 International Conference on *Computer Science and Electronics Engineering*. IEEE, March, 648-651.

[10] Modadugu, N., Rescorla, E., The Design and Implementation of Datagram TLS, *In Network and Distributed System Security Symposium (NDSS)*.

[11] De Rubertis, A., Mainetti, L., Mighali, V., Patrono, L., Sergi, I., Stefanizzi, M., Pascali, S. (2013). Performance valuation of end-to-end security protocols in an internet of things, *In*: *Software, Telecommunications and Computer Networks (SoftCOM)*, 21st International Conference on. IEEE, 1-6.

[12] Stallings, W. (2013). Network Security Essentials, 5th ed. Prentice Hall.

[13] De Rubertis, A., Mainetti, L., Mighali, V., Patrono, L., Sergi, I., Stefanizzi, M., Pascali, S. (2013). Performance evaluation of end-to-end security protocols in an Internet of Things, In: *Software, Telecommunications and Computer Networks (SoftCOM)*, 21st International Conference on. 1-6.

[14] Schaers, H., Komninos, N., Pallot, M. (2011). Smart cities and the future internet: towards cooperation frameworks for open innovation, In: The future internet. *Springer*, 431-446. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-20898-0 31

[15] Edwards, C., Not So Humble Raspberry Pi Gets Big Ideas, *Engineering and Technology*, 8 (3), 30 - 33.

[16] Catapang, S. A., Wang, K. I. K., Salcic, Z., Roberts, Z. J. (2014). Design of a hybrid router for bridging heterogeneous embedded ip networks in ambient intelligence applications, *In:* Intelligent Environments (IE), 2014 International Conference on. IEEE, 56- 62.

[17] OpenSSL, OpenSSL About, 1, 2013. [Online]. Available: http://www.openssl.org/about/

[40] Postel, J. (1981). Transmission Control Protocol.

[18] Stallings, (1998). SSL: Foundation for Web Security, *The Internet Protocol Journal*, 1 (1), 1. *Anonymous, Bit Rates for Live Streaming*, 1, 2014.

[19] Tirumala, A., Qin, F., Dugan, J., Ferguson, J., Gibbs, K. (2005). Iperf: The tcp/udp bandwidth measurement tool, http://dast. nlanr. net/Projects.

[20] Chetty, M., Calandro, E., Feamster, N. (2013). *Latency Eects on Broadband Performance in South Africa*,in 31st ISOC, 3 - 4.