Integration of TESLA and FLUTE over Satellite Networks

L. Liang, M. Bhutta, H. Cruickshank, Z. Sun
CCSR, University of Surrey
Guildford, UK

C. Kulatunga, G. Fairhurst
University of Aberdeen
Aberdeen, UK

***Abstract*— Multicast research has explored the security challenges faced in group communications. Multicast transport and multicast security need to work in close collaboration to realise a multicast service. However, there has been comparatively little work to combine the two technologies. In this paper the authors is presenting an example of partially integrating Timed Efficient Stream Loss-Tolerant Authentication (TESLA) protocol and the File Delivery over Unidirectional Transport (FLUTE) protocol. The security concern raised by the proposed algorithm is analysed for satellite network. The proposed algorithm was implemented on a testbed with multicast tunnel between University of Surrey and University of Aberdeen and the results are presented in this paper.**

***Keywords*—Multicast security, satellite networks, TESLA, FLUTE, integration**

I. INTRODUCTION

IP multicast provides a way to simultaneously disseminate the same packet data to a group of clients. This enables a sender to transmit a single copy of the data, relying on the network to replicate the packets as and when required along the delivery tree towards the receivers. This allows multicast networks to serve large numbers of clients without wasting network capacity.

Many satellite systems natively support IP multicast [1], allowing connected systems to take advantage of the wide-coverage of most satellite down-link footprints. Furthermore, the cost of transmission is independent of the number of receivers; making multicast cost-effective, especially when low cost receive-only terminals are used. Such terminals support unidirectional transmission (i.e. there is no return path).

To reliably and securely transmit data over satellite using multicast requires close collaboration between the multicast transport technologies and the multicast security technologies. The possibility of (partially) integrating both technologies presents potential gains in the efficiency of transmission. This paper examines how Timed Efficient Stream Loss-Tolerant Authentication (TESLA) [2], a multicast security method, can be integrated with File Delivery over Unidirectional Transport (FLUTE) [3], a multicast transport framework from the Reliable Multicast Transport (RMT) family, providing unidirectional file transmission over IP networks.

FLUTE has been designed using Building Block (BB) architecture. The reliability and congestion control functionalities of FLUTE are provided by the Asynchronous Layered Coding (ALC) framework [4] with the Layered Coding Transport (LCT) BB [5] for session management. FLUTE itself provides a mechanism for signalling and mapping the properties of files to the concepts of ALC in a way that allows receivers to assign parameters for received objects. It is designed to work both with Any Source Multicast (ASM) [6] and Source Specific Multicast (SSM) [7] service models. FLUTE transmits session initiation information in-band using a special object called File Delivery Table (FDT). Receivers can distinguish the FDT from File objects using the Transmission Object Identification (TOI) filed in the LCT header.

FDT messages are periodically transmitted to all potential receivers in the FLUTE session. An FDT instance consists of the Header, which forms a part of the LCT header extension (EXT_FDT), and the Payload that consists of one or more file description entries composed and structured in accordance to an eXtensible Markup Language (XML) scheme. The TOI=0 in the EXT_FDT tells the receiver that FDT is in the payload.

The FLUTE sender encodes the File for reliability using Forward Error Correction (FEC) and starts transmission based on the Transmission Session ID (TSI) and the TOI values extracted from the FDT. The commonly used FEC schemes are Low Density Parity Check (LDPC) [8], Reed Solomon [9], and no-code FEC. After the receiver obtains sufficient packets to decode an object the receiver leaves the session. The FEC Payload ID header in each packet determines the FEC symbol within a block.

ALC, and hence FLUTE, can transmit data using multiple multicast groups each at a rate defined by the sender. This allows receivers with differing capabilities behind a network bottleneck to subscribe to a suitable number of groups. Receivers can join and leave groups according to the perceived congestion condition along the path.

A secure group communication system only allows authorized members to send objects to a group. Receivers check the identity of the source to authenticate each packet before accepting it. This procedure is called source authentication. Source authentication is trivial in a point-to-point communication system: the two communication parties can use one pair of keys to authenticate each other. In group communications, a group key is shared by all group members, which makes it challenging to identify the message source and determine its authorization. This authentication is even more challenging when authorized sources are changing. Several solutions have been proposed based on Message Authentication Code (MAC) and digital signature technologies for single source authentication in a multicast group [10] [9] [11] [12] [13].

TESLA [2], one of the source authentication approaches, is a standards-track method developed by the Multicast Security (Msec) working group of the Internet Engineering Task Force (IETF). It offers efficient source authentication using MAC, rather than a digital signature. It manages to trade time with source authentication in an untrusted group using a one way key chain.

The basic idea of the TESLA system is to achieve asymmetric cryptography by delaying the disclosure of the symmetric keys. The whole transmission time is divided into time intervals and packets sent in each interval have MACs using the same key. The key for interval i will be disclosed to all users in interval i+d. The receiver is responsible for buffering packets until the key for their authentication has been disclosed. After disclosure the receiver can authenticate the packet, provided that the packet was received before the key was disclosed. TESLA requires loose time-synchronization between receivers and the source as well as a bootstrap procedure to configure both receivers and sources before the data transmission.

Many research efforts have been put on FLUTE with reference to multicast security protocols such as TESLA. None of them considered how exactly these two protocols can be integrated to provide both reliability and security optimized. And none of them provided solutions for TESLA synchronization for unidirectional file transmissions. This paper is presenting solutions for these two issues by focusing on integrating TESLA with FLUTE with a view of applying in satellite networks.

The paper is organized into five sections. The first section provides a brief introduction to FLUTE and TESLA. The second section presents how to integrate the TESLA bootstrap within the FLUTE FDT object with a synchronization mechanism for TESLA using FLUTE. The third section present s the security consideration of the proposed synchronization algorithm in satellite networks. The fourth section is the implementation of the proposed algorithm on a testbed with positive results. Finally, a conclusion is drawn.

II. THE INTEGRATION OF TESLA AND FLUTE

The main idea of the integration of TESLA and FLUTE has been presented in [14] in details. We only give a brief here to recall the method proposed there.

For a file unidirectional transport application using multicast, authentications are needed for receivers to authenticate sources. It includes the need of receivers authenticating whether a data message is from the real sender and whether a FDT for FLUTE from the authorized sender. Unlike the encryption in link layer in a satellite network, the authentication mechanism is for user authentication which is an end-to-end approach and it can be applied in transport layer to protect only the user data content while saving data processing powers comparing applying to lower layer. Therefore, the source authentication mechanism proposed here can be integrated with the reliability transport protocol, FLUTE, at transport layer.

Integrating with FLUTE provides TESLA a lightweight synchronization method without restrict requirements on time accuracy. FLUTE periodically transmits the FDT message to enable receivers to join a session, which can be used as a “timing” loop for synchronization. It makes the synchronization dependent on the protocol, rather than a real-time clock. It overcomes the issue of delay from buffering and makes it less sensitive to actual transmission rate.

Time synchronization is needed in TESLA in the safe packet test step. If the receivers and the sources have a common start point (for example the synchronization point in FLUTE), the real-time clock time may be ignored. The formula used in IETF RFC 4082 for the safe packet test, which is the highest interval the sender could possibly be currently in: $x = \text{floor}((t_j - T_0) / T_{int})$, uses the receivers current local time minus the session start time T_0 with the assumption that receivers have gain synchronization with sources. Therefore, if T'_0 is the time when the receivers received the FIRST (the first packet sent to the group, marked by a sequence number or special header field) data packet from the source in the beginning of a session, we can use the non-clock-synchronized receiver local time t_j, marked by the lower layer to the packet, minus the time T'_0 to replace the $(t_j - T_0)$ in the above formula to calculate highest interval index x the sender could possibly be in.

The proposed source authentication mechanism improves the transmission efficiency by partially integrating TESLA and FLUTE. It uses this new synchronization mechanism in a unidirectional satellite network so that the TESLA bootstrapping message and the FLUTE FDT message can be combined into one signaling procedure. It saves the transmission overhead and simplified the security requirements, i.e. only one signaling protocol need authentication instead of two.

III. THE SECURITY CONSIDERATION

The proposed loose synchronization raises security concern .If the FDT messages is delayed by attackers between the satellite and the end user, the synchronization will fail. The recommended synchronization methods in the TESLA RFCs, both direct and indirect, face the same threat. This section is to analyze the possibility and impact of such attacks. The attack is analysed here in two scenarios where requirements for launching such attacks are stated and different factors are analyzed that can decide if it can be detected.

A. *Scenario 1*

Attack starts before the receiver terminal joins the satellite network.

- Physical layer: attacker signal is high enough to suppress the original satellite signal received power, but low enough not to saturate the receiver terminal's amplifier. Our experiments showed that the attacker should generate the signal 19dB higher to override the satellite signal.
- Link layer: attacker will repeat all carrier signals sending by the satellite. That means he will act as a satellite hub, which is very difficult to do. He will delay only the multicast signals the receiver interested, which means he knows the receiver will join the multicast channel later.
- Attacker need to make the receiver believe the signal he forwarded is from the satellite. He can achieve this only if all relative signals are delivered without modification. This is a sort of hijack of the satellite HUB. However, he will not be able to make the receive join a faked satellite network generated by him because the real satellite HUB should sign all the messages using a secret key.
- If the attacker achieved all the above goals, the integrity protection will be down.

B. *Scenario2*

Attack starts after the receiver terminal joins the satellite network.

- Physical layer: same as scenario 1. But it will be detected immediately by the terminal because the S/N ratio suddenly changed.
- Link layer: attacker will repeat all carrier signals sending by the satellite. That means he will act as a satellite hub.. He have to delay only the multicast signals the receiver interested, otherwise the terminal will loss the satellite connection due to lack of reception of keep-live messages in the right time. However, his step-in can be immediately detected by the terminal because same messages received multiple times.
- Attacker will have difficulty to cheat the receiver terminal that his signal is the original satellite signal in both physical layer and link layer.
- If the attacker achieved all the above goals, the integrity protection will be down.

The conclusion is that such replay attack is very difficult to launch in terms of both equipment and receiver activity prediction (It is much easier to destroy the receiver's terminal physically.). Such attack can be immediately detected by the receiver terminal in the scenario 2. It can break the TESLA authorization procedure in scenario 1 if the attacker can really launch such attack. He can cause more damage than a DoS attack with the right authentication key he received before the end users. The attacker can simply cause more serious DoS attack by purely suppressing the original satellite signal received power on the receiver's terminal or even burn down the amplifier of the receiver terminal using extreme high power transmission.

Hence, it can not be a real serious threat to the proposed TESLA synchronization algorithm, which is the only way of doing so in a unidirectional satellite network without using any other third party timing services.

In the commercial broadcasting services, such attack is not a serious threat due to the very limited receiver will be affected and an attacker will not benefit much from this kind of attack. In our multicasting services, such attack will not result in serious damage as well. Purely from research point of view, it can be easily solved as well by adding timestamp to each satellite data frame on the satellite HUB. Any delay can be immediately detected on the receiver terminal due to the synchronization between it and the HUB. The point is that this extra timestamp will cost money and it is not worth for such a weak threat.

However, for a Digital Video Broadcasting - Satellite services to Handhelds (DVB-SH) service, this attack could be much more serious where the attacker can jam a terrestrial relay station which will affect hundreds of users in its coverage. That can be a serious Deny of Service (DoS) attack. The attacker can also use high transmission power to suppress the relay station's signal and pretend to be a relay. Such attack can be more difficult to detect because terrestrial relay signal is expected besides the original satellite signal.

IV. THE IMPLEMENTATION

The proposed integrated authentication mechanism has been implemented in C# and validated on a testbed. The implemented contains two pieces of software: the sender and the receiver. The sender generates FDT messages and adds new tags to it as proposed in section II and periodically multicast them to a pre-defined multicast channel. It also generates a one-way key chain using SHA1 function provided in C# with key size of 20 bytes. The key chain is used to calculate the MAC for each data packet during the FLUTE session. The following is a FDT example with TESLA parameters:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<FDT-Instance>
  <File maxOccurs="Unbound">
    <Content-Location>file7.txt</Content-Location>
    <TOI>0</TOI>
    <Expires>633727989608437500</Expires>
    <Interval-Duration>3</Interval-Duration>
    <Key-Chain-Length>20</Key-Chain-Length>
    <Interval-Start-Time>633727989518437500</Interval-Start-
Time>
    <Interval-Index>7</Interval-Index>
    <Key-Disclosure-Delay>3</Key-Disclosure-Delay>

<Key>221:198:145:53:161:69:38:24:165:77:37:86:48:48:238:18
0:177:242:98:150:</Key>
  </File>
</FDT-Instance>
```

The interval length in above code is 3 seconds. The key chain length is 20. The session start time is 633727989518437500s. The current interval index is 7, the key disclosure delay is 3 and the last key of the key chain is "221:198:145:53:161:69:38:24:165:77:37:86:48:48:238:180:177:242:98:150". Such a FDT message was sent to the multicast group in the beginning of each time interval. The sender also repacks the packet as shown in table 1 in [14] and sends it to another predefined multicast group.

On the receiver side, the software listen to the FDT in the corresponding multicast group when it joins the session and gets synchronized with the sender using the proposed method and carries out the packet safe test upon each coming packet using the formula described in section III in [14]. The whole process of the sender and the receiver can be shown in Figure 1.

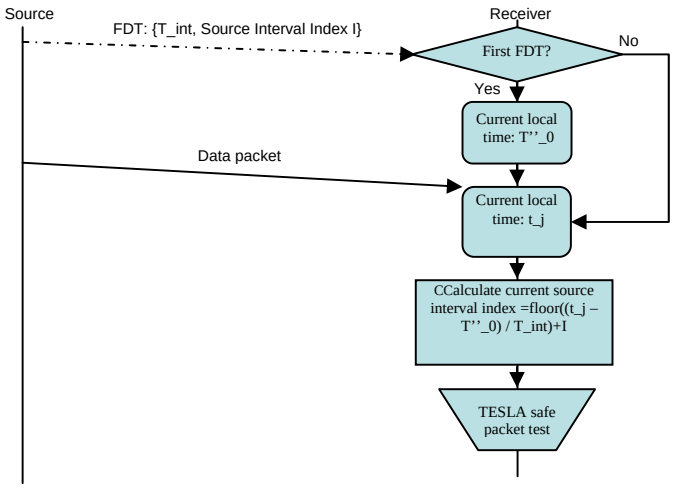

Figure 1. The implemented integration of TESLA and FLUTE

The implementation has been validated on a testbed which connected two networks in University of Surrey (UniS) and University of Aberdeen (UoA) respectively. In the UniS, a Cisco 2811-sec/k9 router is connected to the internet via our university to UoA where locates a Rendezvous Point (RP). The Cisco router was used as a designated router (DR) with a computer running sender in UniS and the other computer running receiver in UoA. A Generic Routing Encapsulation (GRE) tunnel is established between the DR and the RP in order to carry the multicast sessions.

In the experiments, the sender and the receiver is no synchronized using any method recommended by [2]. Their time is 1hour and 4 minutes different. The result shows that the proposed algorithm works well when the source and receiver is not timely synchronized and bandwidth has been saved with the integrated approach. Figure2 and Figure 3 show the integrated authentication implementation works well on two machines, acting as source and receiver respectively, that have different local times. In Figure 2, the sender output the data, the disclosed key, and the MAC for the last packet it sent in the time interval 4. After that, it displayed the 5th FDT it sent in the 5th interval with the correct interval index. The clock at the right side of Figure 2 shows the current local time is 09:39:56.

Figure 3 shows the output result on the receiver side that displayed the data has been successfully received for the last packet sent in the interval 4 after it passed the 4 steps of safe packet test where its local time, shown at the right side of the figure, is 10:43:26, which is different from the sender's time.

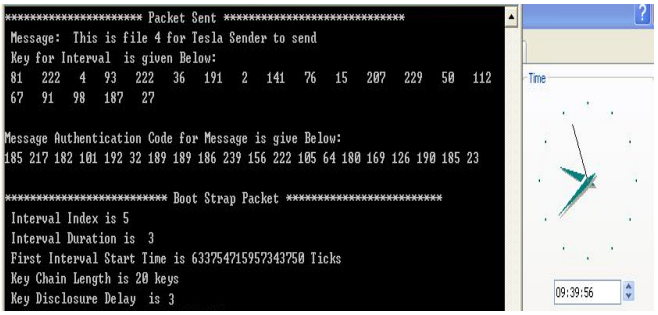

Figure 2. Screen shot on sender computer

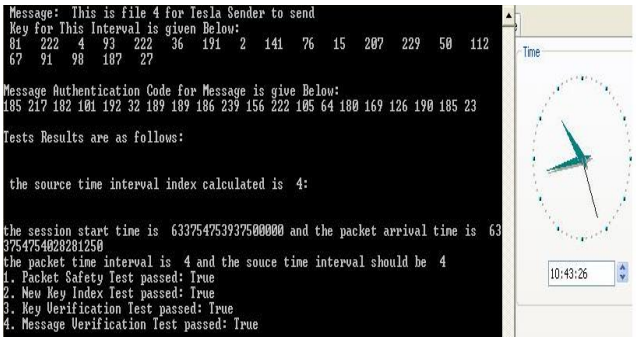

Figure 3. Screen shot on receiver computer

In additional to the synchronization success, the proposed algorithm is expected to save bandwidth after combining the TESLA bootstrapping signalling and the FLUTE FDT signaling and the test results proved it.

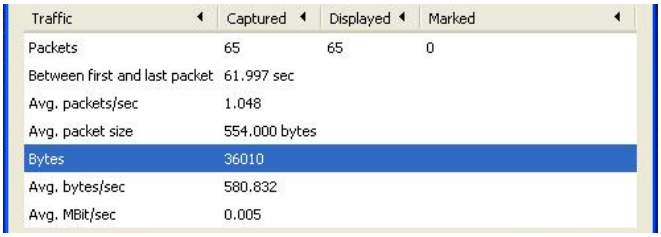

Figure 4. Screen shot on receiver computer

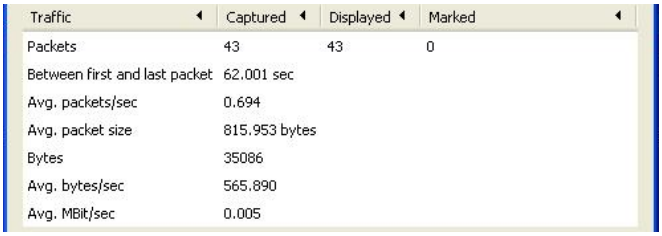

Figure 5. Screen shot on receiver computer

Figure 4 shows a Wireshark capture result that 36010 bytes have been transported in a short time slot around 1 minute using a separated FLUTE and TESLA approach. Figure 5 shows the same session but using integrated approach as proposed that only transmitted 35086 bytes. The integrated approach saved about 2% bandwidth to transmit the same mount of data. It is because the separated approach introduced more overhead for more signaling packets. In our test, there are not much data were generated which definitely contribute the figure of bandwidth saving. However, in a satellite network, extra signaling implies more overhead due to the encapsulation and MPEG TS header. Moreover, if we use integrity protection to the separated signaling procedures, the overhead will get even bigger. So our algorithm does help save the expensive bandwidth in a satellite network.

V. CONCLUSION

This paper has examined the use of TESLA with FLUTE, considering the building block architecture, use of congestion control and FEC building blocks. In the proposed method, the FLUTE File Delivery Table, FDT, object is used to convey the TESLA bootstrap information. A new synchronization mechanism is proposed that eliminates the requirement for using a time server. This method is particularly attractive in a unidirectional transmission environment, (e.g. a satellite network without a return channel).

Security concern has been analyzed in terms of man-in-the-middle attack that can override the satellite signal and gain users trust to fail both the proposed synchronization and the recommended synchronization method in TESAL RFCs. The result showed that such kind of attack is extremely difficult and will not have serious impact in the commercial satellite broadcast/multicast networks.

The proposed algorithm has been implemented using C# and validated on an inter-university testbed. The test result shows that the proposed algorithm works successfully and it saves bandwidth comparing with the separated TESLA and FLUTE approach.

ACKNOWLEDGMENT

The work presented in this paper is sponsored by the Engineering and Physical Sciences Research Council (EPSRC), UK, within the Satellite-Based Secure Multicast Employing Hybrid Reliability project, and the European Satellite Communications Network of Excellence (SatNEx) project.

REFERENCES

[1] Z. Sun, M. P. Howarth, H. Cruickshank, S. Iyengar, and L. Claverotte, "Networking issues in IP Multicast over Satellite", International Journal of Satellite Communications and Networking, vol. 21, pp. 489-507, July 2003

[2] A. Perrig, D. Song, R. Canetti, J. D. Tygar, and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction", IETF RFC 4082, June 2005

[3] T. Paila, R. Walsh, M. Luby, R. Lehtonen, and V. Roca, "FLUTE - File Delivery over Unidirectional Transport," IETF Work in progress, October 2007.

[4] M. Luby, Watson, L. Vicisano, "Asynchronous Layered Coding Protocol Instantiation", IETF Work in progress, November 2007.

[5] M. Luby, Watson, L. Vicisano, "Layered Coding Transport (LCT) Building Block", IETF Work in progress, November 2007

[6] S. Deering, "Host Extensions for IP Multicasting," IETF RFC 1112, August 1989.

[7] H. Holbrook, "A Channel Model for Multicast", in Department of Computer Science, Stanford University, California 2001.

[8] R. Vincent, "INRIA," http://planete-bcast.inrialpes.fr

[9] Rosario Gennaro and Pankaj Rohatgi, "How to sign Digital Streams", Advances in Cryptology – CRYPTO '97, pp. 180-197, August 1997

[10] Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas, "Multicast Security: A Taxonomy and some efficient Constructions", INFOCOM'99, March 1999

[11] C. K. Wong and S. Lam, "Digital Signatures for Flows and Multicasts", IEEE/ACM Transactions on Networking, vol. 7, No. 4, August 1999

[12] P. Rohatgi, "A Compact and Fast Hybrid Signature Scheme for Multicast Packets", 6th ACM Conference on Computer and Communications Security, Singapore, November 1999

[13] S. Even, O. Goldreich, and S. Micali, "On-Line/Off-Line Digital Signatures", Journal of Cryptology, 9(1), pp. 35-67, 1996

[14] L. Liang, H. Cruickshank, Z. Sun, C. Kulatunga and G. Fairhurst, "TESLA with FLUTE over Satellite Networks," 2008 IEEE International Conference on Communications (ICC2008), Beijing, China, 19-22 May 2008.