Fragmentation Attacks in 6LoWPAN and Mitigation Mechanisms

Pranali Pawar
Guided By : Dr. Manas Khatua

January - April 2017

1 Abstract

6LoWPAN allows connectivity among nodes with limited power by importing IPv6 capabilities into the low-power nodes. 6LoWPAN adopts the physical (PHY) and Media Access Control (MAC) layer protocols defined in IEEE 802.15.4 standard as its lower layer protocols. The IPv6 protocol is used as the network layer protocol in 6LoWPAN. A packet fragmentation mechanism is required in 6lowpan when IPv6 packets exceeds the maximum frame size of the link layer.

In this work, we provide a comprehensive security analysis of the 6lowpan fragmentation mechanism. We consider the attack at the 6LoWPAN design-level that enable an attacker to prevent correct packet reassembly on a target node at considerably low cost. First, we will study the content chaining mechanism and then we will propose a modified lightweight defense mechanism of it, the hash tree based approach.

2 Introduction

6LoWPAN standards enable the efficient use of IPv6 over low power, low rate wireless networks on simple embedded nodes through an adaptation layer and optimization of related protocols. The Maximum frame size of LOWPAN packet is 128 octets as specified by IEEE 802.15.4 while the frame size of IPv6 is 1280 octets. Thus an incompatibility exists in accommodating the IPv6 frame in a LOWPAN frame. In order to alleviate this issue, 6LoWPAN working group has suggested an additional adaptation layer between MAC layer and the network layer.

As its main task, 6LoWPAN adjusts IPv6 packets to the unique characteristics and requirements of wireless multihop communication between low-power devices. The variety of applications thereby requires 6LoWPAN to support both small-sized transmissions, e.g., for sensor data or control commands, and

large transmissions, e.g., for firmware updates or security protocol handshakes [4, 6, 2].

The design of the 6LoWPAN fragmentation mechanism renders buffering, forwarding and processing of fragmented packets challenging on resource-constrained devices. Specifically, malicious or misconfigured nodes may send duplicate or overlapping fragments. Due to the lack of authentication at the 6LoWPAN layer, recipients are unable to distinguish these undesired fragments from legitimate ones for packet reassembly.

Our contribution in this paper gives a detailed security analysis of the 6LoWPAN fragmentation mechanism for networks that consist of resource-constrained devices. We identify fragment duplication attack that a malicious node can mount against the 6LoWPAN layer. An eavesdropping attacker can reactively prevent the successful processing of fragmented packets by duplicating an overheard fragment with the fragment duplication attack.

In our project we implement a lightweight mechanisms to protect resource-constrained devices against these attacks. The content-chaining scheme mitigates the fragment duplication attack by offering efficient per-fragment sender authentication while hash tree based approach can also authenticate the out of order fragments at receiver side.

3 6LoWPAN Packet Fragmentation

Fragmentation Mechanism The 6LoWPAN adaptation layer provides header compression and packet fragmentation functionality for IPv6 packets which is located between the network and the link layer. When an IPv6 packet at a sending node exceeds the available link layer payload size, the 6LoWPAN fragmentation mechanism treats the (compressed) IPv6 packet as a single data field and iteratively segments this field into fragments according to the maximum frame size at the data link layer. Each fragment includes a fixed-size fragment header and the remaining space of the link-layer frame is iteratively filled with the IPv6 packet content. Only the first fragment (FRAG1) contains end-to-end routing information. Hence, a receiving node needs to correlate the remaining fragments (FRAGN) to the FRAG1 in order to derive IP-based routing or processing decisions for these fragments.

Fragment Forwarding Mechanisms 6LoWPAN supports three routing mechanisms. Mesh-under routing is a link-layer routing schemes in which 6LoWPAN layer prepends each fragment with a mesh routing header that contains the end-to-end source and destination link layer addresses. The given information can be used to derive a routing decision on a per-fragment basis. Thus mesh-under routing is oblivious to packet fragmentation. As a result, individual fragments may take different paths towards the destination. In contrast, route-over routing does not require additional header information and derives forwarding decisions at the network layer. As forwarding nodes apply the routing decision on a per-packet basis, all fragments of a packet are sent along the

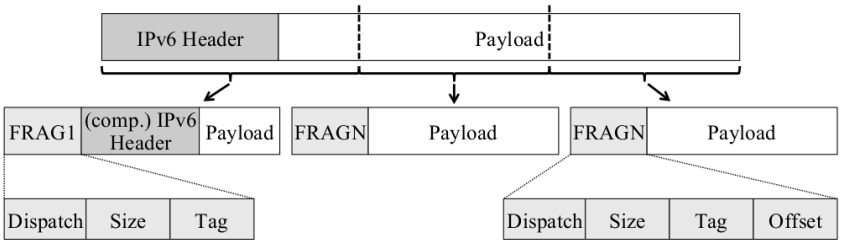

Figure 1: 6LoWPAN packet structure of a first fragment FRAG1 and subsequent fragments FRAGN.

same path. To afford mesh-under-like forwarding efficiency, enhanced route-over proposes an optimization of the route-over approach. Enhanced route-over derives forwarding decisions directly based on the IP header information in the FRAG1. It then stores the forwarding decision, forwards the FRAG1, and applies the same forwarding decision on reception of a FRAGN that belongs to the same IPv6 packet. Hence, while FRAGNs can be forwarded individually, they are transmitted along the same path, similar to route over.

4 Fragmentation Attacks

At the physical layer, DoS attacks can be launched by jamming or tampering the radio signal. Moreover, the cryptographic secrets stored inside the node can be extracted allowing replay attacks, packet injection, making a clone, or node reprogramming. At the link layer, an attack on network availability consists in flooding the network with large packets to occupy the entire bandwidth. Packet injection can also lead to battery exhaustion or to packet collision followed by packet loss.

Hummen et al.[7] presents two fragmentation attacks on "mesh-under" routing protocol handled by the 6LoWPAN adaptation layer. As the destination address is mentioned only in the first fragment, an attacker can easily flood the network with next fragments duplicated at the time of reception. Another attack consists in maliciously reserving space in the reassembly buffer with incomplete packets until saturation.

5 Our Goals and Existing Solution

The goal of this paper is to efficiently protect the authenticity and integrity of packets. Ideally, we hope that

1.**Node-compromise resilience.**Every sensor node can authenticate and verify the integrity of the program code disseminated from a base station. An

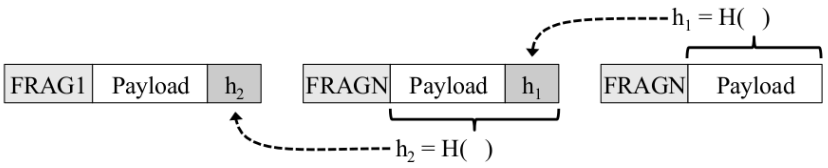

Figure 2: Example of a content chain for a packet consisting of three fragments.

adversary cannot spoof the base station or change the contents of a code image without being detected by other nodes.

2.**DoS-attack resilience.**Receiver can verify the fragment as soon as it receives it. Otherwise an adversary can potentially launch denial of services attacks against the sensor network due to delayed authentication, For example, if an adversary injects fake packets to a node and that node cannot verify those packets immediately, then either those packets will consume memory or the computing time of this node and eventually exhaust its resources, or this node has to drop packets without verification, potentially dropping valid data packets.

3.**Low cost.**The resource consumption of the proposed security mechanism must be light weight in terms of communication, computing and memory usage.

Our proposed method consists of two stages:

1) *Content chaining* based scheme.

2) *Hash Tree* based scheme which is an advanced version of the content chaining.

In the following two sections we describe these two stages in detail.

A. Chain-based Scheme

Resource-constrained nodes could defend against the fragment duplication attack if they were able to identify the sender on a per-fragment basis. Pairwise keys at the link layer would prevent spoofing of link layer addresses. To avoid the overhead of a pairwise-key management, we propose a content-chaining scheme that binds the content of a fragmented packet to its FRAG1 instead of binding fragments to cryptographic sender identities. To this end, the legitimate sender adds an authentication token to each fragment during the 6LoWPAN fragmentation procedure. This allows the recipient to cryptographically verify the link between fragments at the time of reception and to discard maliciously duplicated fragments early on the forwarding path. Content chains are based on the concept of hash chains [5, 3], a lightweight, efficient mechanism to authenticate the sender of a data stream of finite length. Thus, they naturally fit the properties of fragmented packets when treating each packet as a fragment stream.

The elements h_i of a hash chain are generated by iteratively applying a cryp-

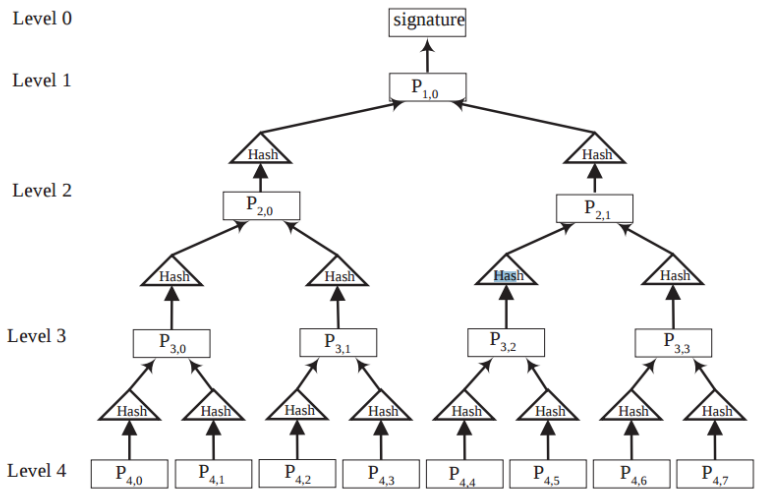

Figure 3: Signed Hash Tree Scheme..

tographic hash function H(·) to the output of the previous iteration:$h_i = H(h_{i1})$. For the first iteration, a (random) seed value h_0 is used as input. The last token

$$h_n = H^n(h_0) = H(H(...H(h_0)))$$

is referred to as the anchor element of the hash chain. Hash chains are used in reverse order of their generation, i.e. starting with h_n, as the one-way property of the hash function then prevents others from computing undisclosed tokens that are closer to the seed value. Thus, a token represents a cryptographic commitment to all previous tokens of the hash chain.

When processing out-of-order FRAGNs, a verifying node follows a simple policy. First, it only forwards fragments that have been verified successfully. Second, it stores out-of-order FRAGNs without prior verification until all previous FRAGNs have been received and verified.

Referring back to Section 2.2, we see that this chain scheme satisfies security goals 1 and 3, but not adequately goal 2. In fact, the chain scheme meets the second goal only in a limited manner, as follows:

2a. Limited DoS-attack resilience. Suppose the data packets are disseminated as sequence from 1 to n. Only when a node X has received all packets from 1 to $k1$ can it verify packet k immediately when X receives it.

B. Hash Tree Based Mechanism

In [1], Jing Deng et al. proposed a hash tree method for wireless sensor network. To achieve DoS-attack resilience and allow immediate verification of out-of-order packets, we implemented this hash tree method for secure propagation based on a signed hash tree. We assume an underlying code distribution

mechanism like Deluge: a node sends a group of packets to its neighbor nodes, and after a certain time, each neighbor node sends back a NACK message to tell the sender which packets it missed. Then the sender retransmits missed packets. This process saves traffic since the receiver doesn't have to acknowledge every packet. To simplify the algorithm description, we just assume that the sender transmits all packets from 1 to n to its neighbor nodes.

To enable nodes to authenticate and verify the integrity of code image packets quickly, even when the packets may arrive out of order, it is important to propagate the hash values of those packets a priori. This can of course be done by sending index packets containing only the hash values of code image packets in advance.

Figure 3 illustrates our basic hash tree scheme. The code image is divided into packets at the base station, and a secure hash is computed on each packet. These hash values are themselves input to create a new level of hashes, and so on up the tree. A packet at level i contains hash values of w packets in level $i+1$.

$$P_{i,j} = Hash(P_{i+1,jw})||...||Hash(P_{i+1,jw+w1})||otherinfo$$

The hash value of every data packet is included in one of the packets in level $m-1$, and the hash value of every packet in level $m-1$ is included in one of the packets in level $m-2$, and so on. This tree is built in such a way that there is exactly one packet at level 1.

The root value at the top of the tree, level 0, is a signature that is obtained by encrypting the hash value of the packet at level 1 using the private key (K_s) of the base station, i.e

$$P_{0,0} = E_{K_s}(Hash(P_{1,0}))||Hash(P_{1,0})||otherinfo$$

To authenticate the source of $P_{1,0}$ and to verify the integrity of $P_{1,0}$, signature $P_{0,0}$ can be used. We can see that if an adversary tampers with the contents of a packet, it can be detected using the hash of the original (untampered) packet received in a packet in the previous level.

Referring back to requirement 2a, we see the hash tree scheme satisfies a weaker statement, i.e. applies more broadly:

2b. DoS-attack resilience. When a node received all packets from 1 to $k-1$, it can verify any packets from k to m with any order, where $m-k > k$.

6 Performance Evaluation

The goal of this experiment is to understand the costs of our security mechanism, and see if it is feasible for 6LoWPAN applications. For our evaluation, we implemented our proposed defense mechanisms on ns3. As a proof of concept for the fragment duplication attack, we implemented a simple sender that transmits a constant stream of fragmented UDP packets. To evaluate the behavior and overheads of our proposed mechanisms for IPv6 packets of up to 640 bytes. The average computation time for a single content token, i.e., generation or

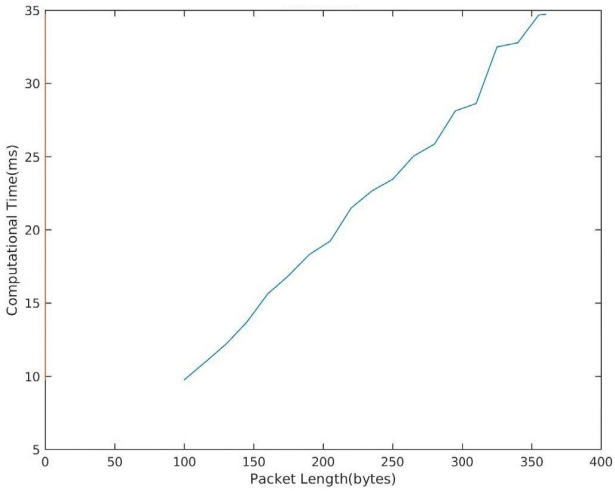

Figure 4: Processing time at the 6LoWPAN layer at sending node

verification, increases from 0.87 ms to 5.22 ms for growing payload sizes. For our evaluation, we did not consider explicit overheads due to link layer security operations such as encryption and decryption. Our content chaining scheme require 10.90% additional memory.

7 Conclusion and Future Work

This paper develops a novel secure code propagation protocol for 6lowpan that employs private key signing of the root of a joint structure comprised of hash chains and hash trees for authentication. This solution has the following advantages: node-compromise resilience, since the public key and linked chain-tree hash structures protect all packets; DoS attack resilience since the tree structure allows immediate and rapid verification of packets; and low cost realization in terms of 1) low delay due to public key authentication being performed only once initially 2) low delay due to fast hash based verifications for all subsequent packets 3) low overhead since the hash tree tolerates disorder and reduces unnecessary retransmissions. In the future, we plan to improve these scheme to provide a more secure channel and for better performance.

References

[1] J. deng, r. han, and s. mishra. "secure code distribution in dynamically programmable wireless sensor networks. in proc. of ipsn, 2006.".

[2] K. hartke and o. bergmann. "datagram transport layer security in constrained environments." draft-hartke-core-codtls-02 (wip), 2012.

[3] P. g. bradford and o. v. gavrylyako. hash chains with diminishing ranges for sensors. international journal of high performance computing and networking, 2006.

[4] P. thubert and j. hui. "lowpan fragment forwarding and recovery. draft-thubert-6lowpan-simple-fragment-recovery-07 (wip), 2010.".

[5] R. gennaro and p. rohatgi. how to sign digital streams. 1997.

[6] R. hummen, j. h. ziegeldorf, h. shafagh, s. raza, and k. wehrle. "towards viable certificate-based authentication for the internet of things." in proc. of acm hotwisec, 2013.

[7] René hummen, jens hiller, hanno wirtz, martin henze, hossein shafagh, klaus wehrle. "6lowpan fragmentation attacks and mitigation mechanisms. in wisec'13, april.".