Technical Disclosure Commons

Defensive Publications Series

May 2020

AUTHENTICATED ROUTE DISCOVERY IN WIRELESS MESH NETWORKS

Niranjan M. M

Nagaraj Kenchaiah

AUTHENTICATED ROUTE DISCOVERY IN WIRELESS MESH NETWORKS

AUTHORS:
Niranjan M. M
Nagaraj Kenchaiah

ABSTRACT

Techniques are presented herein to provide an efficient and secure signature scheme to authenticate route discovery in Wireless Mesh Networks (WMNs). Specifically, the techniques presented herein provide a scheme where multi-signatures are generated with cryptographic keys provided by Trusted Platform Modules (TPMs) on each Mesh Router (MR) in the WMN. The keys can protect device identities, which may secure the network devices against attacks, and, in at least some instances, the cryptographic keys can also provide authentication and encryption at the software/application level. Overall, the techniques may eliminate the need for a Key Generation Center (KGC) in the WMN and do not require MRs to cooperate to construct a signature. Thus, among other advantages, the techniques described herein may be efficient and inexpensive to implement.

DETAILED DESCRIPTION

Generally, Wireless Mesh Networks (WMNs) are multi-hop radio networks whose nodes are Internet Protocol (IP) routers with one or more wireless interfaces, which often operate based on Institute of Electrical and Electronics Engineers (IEEE) 802.11 Wi-Fi protocols. A backbone of a WMN is typically made up of dedicated wireless nodes called Mesh Routers (MRs), which may be configured in an ad-hoc mode and use omnidirectional antennas. With the advent of 802.11ax, wireless bandwidth has become multi-folded and, thus, WMNs are becoming prominent in ad-hoc wireless network deployments. For example, WMNs are now often utilized for deployment of IP security cameras, Internet of Things (IoT) devices etc., in the places where wired connectivity is expensive and/or not feasible.

In a WMN, the MRs can be freely organized into any network topology and can communicate with each other using various protocols, such as Dynamic Source Routing (DSR) or Optimized Link State Routing (OLSR). However, collectively, the resource constraints of nodes, the capacity of the wireless medium, the mobility of the nodes, and the self-organized form of the network, may make it difficult to implement methods for securing traditional wired networks in ad-hoc wireless mesh networking environment. Moreover, DSR and OLSR have known vulnerabilities.

For example, DSR is a simple, robust reactive routing protocol that efficiently constructs routes, guarantees loop-free routing, and provides load balancing. However, DSR is vulnerable to several forms of attacks by malicious nodes. Meanwhile, OLSR is a simple, proactive link-state routing protocol that also guarantees loop-free routing, provides load balancing. However, OLSR is less reliable than DSR and is also vulnerable to many of the malicious attacks. In fact, both DSR and OLSR protocols, which are the two most popular protocols for on-demand source routing in multi-hop WMNs, are vulnerable to malicious attacks (e.g., injections of bogus routing information, formation of feedback loops by colluding adversarial nodes, identity spoofing, link spoofing, replay attacks, etc.). To address these attacks, OLSRv2 provides inherent resilience by having sequence numbers, ignoring unidirectional links, message interval bounds, etc., but still recommends incorporating a security mechanism to address vulnerabilities.

The classical approach to mitigate above attacks is to use cryptographic tools to authenticate information exchanged during the route discovery process. For example, current tools include: (1) symmetric key techniques; (2) asymmetric key techniques; (3) aggregate signature techniques; and (4) multi-signature techniques. Generally, these techniques are inefficient or ineffective, for example, because the techniques require extensive manual configuration, a trusted third party, and/or extensive computational power requirements. However, for clarity, these techniques are briefly discussed below.

First, symmetric key techniques utilize a single shared key across all MRs to generate integrity check values. Every MR verifies the integrity check value of a received message and creates a new integrity check value after adding the source route to the received message. Thus, in this method, the same secret key is configured on all MRs.

Unfortunately, this renders manual configuration difficult and creates a framework where overall security is compromised when one MR is compromised.

Second, asymmetric key techniques use identity-based signatures and avoid a shared secret key that results in a single point of failure vulnerability. Instead, a public and private/secret key pair (PK, SK) is generated for every MR. Then, the SK and identity of the MR are used to generate an identity-based signature, which can be used as an integrity check value. Thus, each MR verifies the identity-based signature of the previous MR using a public key and adds its identity-based signature after successful verification of the message from previous MR. However, with these techniques, every MR still needs to know the public keys (PKs) of all other MRs in the deployment.

Often, a trusted third party is used to distribute certified public keys. In distributed deployments, a key generation center (KGC) is often used as the trusted third party. However, any identity-based cryptographic system using a KGC has an inherent key escrow problem since the KGC always knows user keys. Thus, if the KGC is malicious, it can impersonate a user. Moreover, in a WMN, all MRs may not have direct access to the KGC. Without direct access, an MR must make a connection to another node, but this connection will not be trusted. Still further, typically, a KGC signs keys for a particular MR to allow other MRs to trust the particular MR, but this signed key does not allow the particular MR to trust other MRs (i.e., the signed key does not necessarily create two-way trust). Thus, KGCs may create a 'chicken and egg' type of problem where MRs have to find a common KGC and make sure their keys have not been compromised. Some techniques have been introduced to try to address these vulnerabilities. For example, some techniques propose using strongly unforgeable certificateless signatures that resist attacks from a malicious, but passive KGC. However, these techniques may be complicated and expensive to implement, especially in a fully distributed self-organized WMN, where all MRs need to have access to the certified public keys of all other MRs in the WMN.

Third, aggregated signature techniques authenticate route discovery in DSR by having each MR sign route request (RREQ) packets they forwarded toward the destination, as is shown in Figure 1 below. Then, the destination authenticates the accumulated source route before generating a route reply (RREP) packet. However, as can be seen, with these techniques, DSR floods the network with RREQ packets during route discovery, so most

MRs waste computation and communication resources by signing, verifying, and forwarding RREQ packets that are not included in the eventual route. Moreover, the destination MR needs to verify each and every signature of a RREQ packet (signed by all MRs in the path) by using the public key of each MR in a path. Unfortunately, verifying sequential aggregate signatures is computationally intensive/expensive, so performing this verification for a potentially unused path may be undesirable. Moreover, a KGC would need to be used to distribute certified public keys and private keys, which is undesirable for the reasons laid out above.

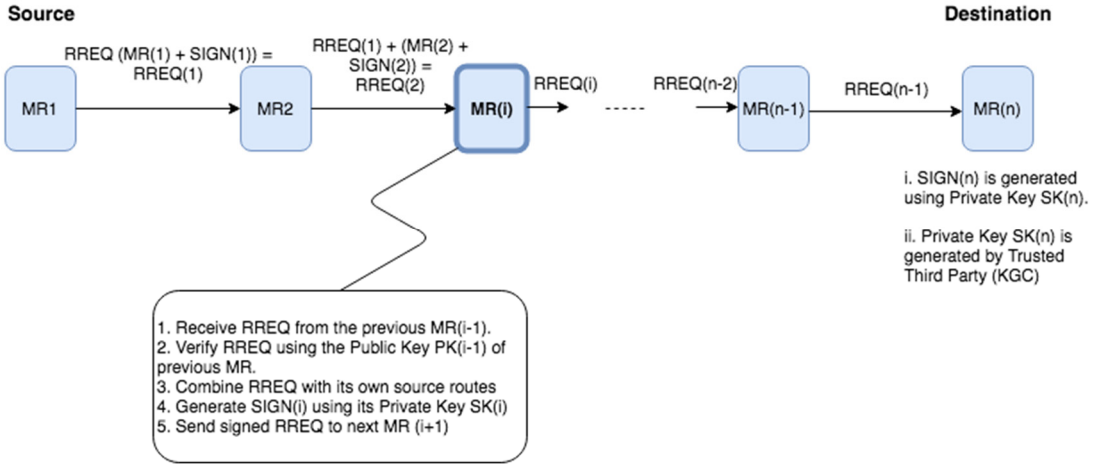

Figure 1: Conventional Aggregated Signature Techniques

Fourth, conventional multi-signature techniques use a private key to generate a signature while the public keys of individual MRs are used to generate an aggregate public key. The aggregate public key is then used to verify a multi-signature of the previous MR and to generate a new multi-signature before sending to next MR. However, with conventional multi-signature techniques, each MR must access a KGC to obtain a private key and the certified public keys of all other MRs in the WMN, which is undesirable for the reasons laid out above.

In view of the foregoing, the techniques presented herein use trusted platform modules (TPMs) on MRs to provide certified private and public keys. In particular, the techniques presented herein leverage cryptographic keys generated by TPMs to provide a multi-signature scheme that enhances efficiency of authenticated route discovery in WMNs. Typically, TPMs protects network device identities, but TPMs can also be used to generate cryptographic keys. Utilizing the TPMs in this manner eliminates a KGC from route

discovery and allows MRs to trust each other (instead of KGC) without the costs and vulnerabilities associated with KGC. In fact, since cryptographic keys are provided by TPMs of MRs, the techniques implicitly provide trustworthiness of the routes learned between source and destination MRs.

More specifically, with the techniques presented herein, each MR can be configured once (for its key lifetime) by a trusted authority, independently of all other MRs. For example, a MR can connect to an authority (typically in a secure environment) to receive a private key or can have a private key delivered securely (out of band) from the authority. Thus, every MR will have a public and private/secret ley pair (PK, SK), provided by its TPM (e.g., stored in microchip). The TPM itself could be built into hardware (e.g., an ACT2 chip) and, thus, tampering would make hardware/device unusable. Moreover, since TPMs do not expose security keys to the applications – and, instead provide options to migrate the keys so that applications can use them for the authentication and encryption methods – the TPM will provide an added layer of security. Thus, the present techniques avoid the issues associated with KGCs discussed above. Once the private and public keys are generated by TPM, the private keys can be used to generate signatures while the public keys of individual MRs are used to generate an aggregate public key (e.g., in a multi-signature scheme). The aggregate public key can be used to verify a multi-signature of a previous MR and to generate a new multi-signature before sending to next MR (e.g., in a multi-signature scheme).

Figure 2 below provides an example of how the techniques presented herein may be utilized for authenticated route discovery in a WMN. Generally, the multi-signature authentication scheme presented herein consists of five phases: (1) initial setup; (2) key generation; (3) signature generation; (4) multi-signature verification; and (5) multi-signature generation.

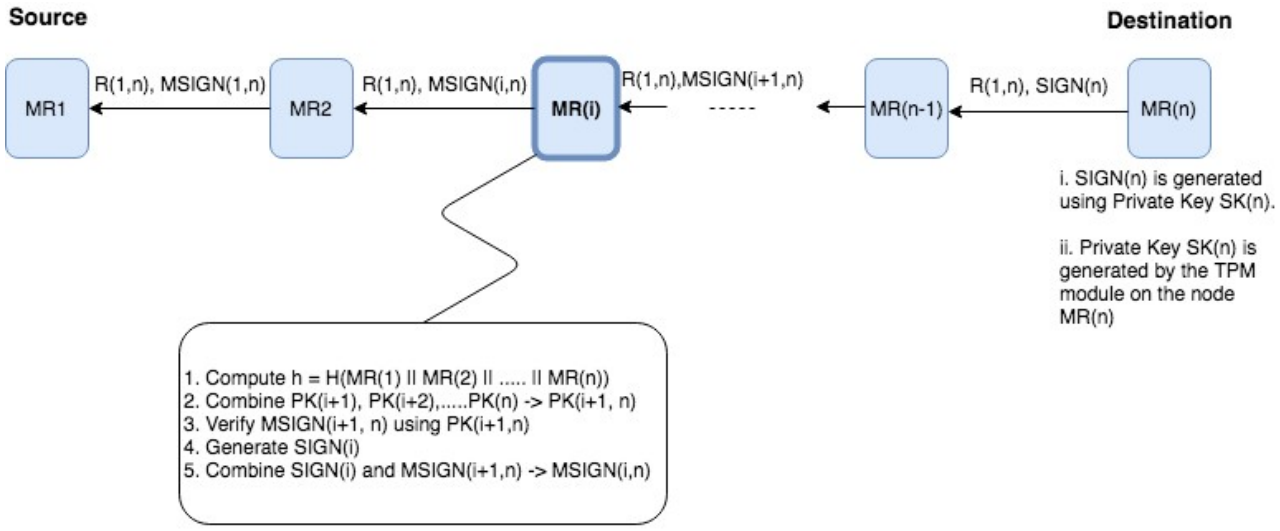

Figure 2: Authenticated Route Discovery with the Techniques Presented Herein

In Figure 2, the following notations are used:

- R(1, n) is a source route, e.g., {MR(1), MR(2) . . . MR(n)};
- SIGN(i) is an individual signature on “h” using a private/secure key (e.g., “SK(i)”);
- PK(i,j) is the aggregate public key of nodes MR(i), MR(i+1), . . . MR(j);
- MSIGN(i,j) is a multi-signature on “h” by aggregating SIGN(i), SIGN(i+1), . . . SIGN(j); and
- R(,), MSIGN(,) is a RREP packet containing source route R(,) and multi-signature MSIGN(,).

Each of the steps of Figure 2 is also described in more detail below.

First, during initial setup, all MRs get a public parameter and all nodes choose and agree upon the system public parameters generated using a probabilistic polynomial-time (PPT) algorithm. The nature of the public parameter depends on the underlying crypto system, but generally, the input includes a number of MRs in the deployment, security parameters, etc., and the output is the public parameter. Additionally, MRs can be onboarded in any order.

Next, during key generation, each mesh router (MR(i), where "i" - represents a mesh node ID in subsequent usage, and "n" represent number of MRs in the path) generates keys. Specifically, public keys of all MRs and TPM migration are enabled, which results in the generation of public and private key pairs (PK(i), SK(i)), as well as an aggregate

public key, PK(i+1,n). For example, the TPM on each mesh router (MR(i)) can generate pairs of a long-term public key, PK(i), and a private key, SK(i), while also generating the aggregate public key, PK(i+1,n), using public keys of all other MRs in the path (i.e., PK(i+1), PK(i+2) . . . PK(n)).

Third, during signature generation, each mesh router (MR(i)) uses the private/secret key (SK(i)) of the mesh router (MR(i)) that was created using TPM and a message M containing the source route information to output an individual signature (SIGN(i)). Specifically, each mesh router (MR(i)) participating in the RREP propagation generates an individual signature, SIGN(i), using a private/secret key (SK(i)) of the MR on the hashed concatenation of the address in the source route of the message M by the PPT algorithm.

Fourth, during multi-signature verification, each mesh router (MR(i)) uses the aggregated public key (PK(i+1,n)), aggregated multi-signature (MSIGN(i+1, n)), and the source route information from previous MR included in the message M to render a validity determination (e.g., to output valid or invalid). Specifically, each intermediate mesh router MR(i) (other than the destination), receives a signed RREP packet containing the multi-signature MSIGN(i+1,n). Then, each mesh router (MR(i)) verifies the multi-signature (MSIGN(i+1,n)) using the aggregate public key (PK(i+1,n)) generated in the second step. Notably, for the last hop before the destination MR(n) (e.g., MR(n-1)), the multi signature (MSIGN(i+1,n)) denotes MSIGN(n).

Fifth, and finally, during multi-signature generation, each mesh router (MR(i)) uses the multi-signature (MSIGN(i+1, n)) sent by previous MR and its signature (SIGN(i)) to generate a new multi-signature (MSIGN(i,n)) to be sent to the next MR. If the multi-signature (MSIGN(i+1,n)) verification is successful, a mesh router (MR(i)) generates the multi-signature (MSIGN(i,n)) by using its signature (SIGN(i)) (e.g., by using its secret key, SK, like in the third step). Then, each mesh router (MR(i)) replaces the multi-signature (MSIGN(i+1,n)) present in the RREP with the newly generated multi-signature (MSIGN(i,n)) before forwarding the RREP to the next hop MR (MR(i-1)).

Overall, the sequence of signature generation, multi-signature verification, and multi-signature aggregation continues until the RREP packet containing the multi-signature (MSIGN(1,n)) is delivered to the source. If the multi-signature (MSIGN(1,n)) is verified using aggregated public key (PK(1,n)), the individual signatures (e.g.,

SIGN(1),…..SIGN(n)) of corresponding MRs (e.g., MR(1),…MR(n)) in the discovered source route to the destination MR (e.g., MR(n)) are verified collectively. Notably, unlike aggregate signature techniques, the techniques presented herein verified signatures for a selected path.

Among other advantages, these techniques provide an efficient, authenticated route discovery protocol for WMNs. These techniques are particularly advantageous because they eliminate a KGC, rendering the issues associated with using a KGC in a WMN deployment moot. In fact, since the techniques provide a high level of security, the techniques can be used in enterprise WMNs, including mesh networks with access points servicing wireless clients (e.g., root access points (RAPs) and/or mesh access points (MAPs) that typically require high-level of security. Moreover, the techniques presented herein can be used to implement authenticated route discovery in any source-based routing protocol and. The techniques can also be incorporated into WMN deployments utilizing IEEE 802.11ax, which supports higher bandwidth and is becoming increasingly utilized. Moreover, since the Multi-signature scheme presented herein is applied on the RREP from the destination to the source, hop-by-hop, ordering is unnecessary and, thus, the techniques are easily scalable.

In summary, the techniques described herein provide an efficient and secure signature scheme to authenticate route discovery in WMNs. Specifically, the techniques provide a scheme where signatures are generated with cryptographic keys provided by a TPM on each MR in the WMN. The keys can protect device identities, which may secure the network devices against attacks, and, in at least some instances, the cryptographic keys can also provide authentication and encryption at the software/application level. Overall, the techniques may eliminate the need for a KGC in the WMN and do not require MRs to cooperate to construct a signature. Thus, among other advantages, the techniques described herein may be efficient and inexpensive to implement.