Securely-Entrusted Multi-Topology Routing for Community Networks

Axel Neumann, Ester López, Llorenç Cerdà-Alabern, Leandro Navarro
Universitat Politècnica de Catalunya, Barcelona, Spain
{axel,esterl,llorenc,leandro}@ac.upc.edu

***Abstract*—**
Routing in open and decentralized networks relies on cooperation despite the participation of unknown nodes and node administrators pursuing heterogeneous trust and security goals. Living use cases for such environments are given by community-mesh networks due to their open structure and decentralized management and ownership. However, despite many active work in the field of routing security for mesh and MANET networks, practical solutions enabling a secured but decentralized trust management are still missing, leaving nowadays existing community networks vulnerable to various attacks and seriously challenged by the obligation to find consensus on the trustability of participants within an increasing user size and diversity.

This work presents the design, implementation and analysis of a routing protocol that enables cryptographically secured negotiation and establishment of concurrent and individually-trusted routing topologies for infrastructure-less networks without relying on any central management.

Benchmarking results, based on our initial implementation and tested on real and very cheap (10 Euro, Linux SoC) embedded routers, quantify the scalability of our approach supporting networks with hundreds of nodes and despite being based on supposedly CPU-expensive asymmetric cryptography.

***Index Terms*—routing, trust, decentralized security, multi-topology, cooperation, mesh networks, community networks.**

I. Introduction

Community mesh networks [1], [2], [3] are supposedly open and decentralized structures, growing and evolving organically as network infrastructure is contributed, deployed, and configured by their participants. Typically, a deployment of such networks, such as Guifi.net [4] with more than 27000 total active nodes, is structured in different clouds [5], each consisting of up to hundreds of nodes, constituting an autonomous system (AS), operating its cloud-specific internal routing protocol (e.g. OSPF, OLSR), and peering with neighboring clouds via an exterior gateway protocol (e.g. BGP).

The operation of such networks is based on the principle of cooperation among its members. These communities usually have participation rules as a membership license or peering agreement [6], that define their freedom, openness and neutrality. Nonetheless, current designs and implementations of mesh networks impose comprehensive technical definitions and restrictions to achieve functional data transit and end-to-end delivery among any pair of network nodes [2]. That includes the use of a specific routing protocol and routing metric so that nodes can consistently learn and inform about the state of the network and update their own routing tables. In practice, due to the lack of mature implementations, only a very few of the proposed routing protocols are used in real deployments or have been experimentally analyzed [3], [7], [8], [9]. Among them there are the AODV [10], Babel [11] BMX6 [12], the widely used OLSR [13] and batman-adv [14] protocol implementations.

One shortcoming of current solutions is given by the lack of routing-security support that comes without introducing centralized dependencies (e.g. certificate authorities) [15], which would contradict with the open and the decentralized objectives of such networks. Another problem lies in the protocol requirements for unified parametrization of metrics and policies to determine QoS, routing, trust and security decisions for all network nodes [16]. Such a strong level of unification prohibits the usage of individual defined policies and limits its openness. It also imposes a substantial effort, increasing with the number and diversity of community members, for finding consensus on related questions.

To dilute the limitations of a single and unified set of QoS parameters for routing, QoS Multi-Topology (MT) routing has been proposed [17], [18], allowing the concurrent support of multiple virtual topologies (on top of a single physical topology), each established based on a different definition of QoS parameters. This approach could also be adapted to concurrently support different security and trust sets. A network could for example maintain one topology (a) for nodes trusted by organization A and a second topology (b) usable only by nodes certified via organization B.

The security design of the protocol proposed in this work ensures that each node is the only authority able to define and publish its set of individually-trusted nodes via which forwarding rules (routes) for delivering its traffic should be selected, propagated, and maintained. This way, our protocol pursues the multi-topology approach as it establishes dedicated virtual topologies for each participating node. It further supports the cooperative, open, and decentralized philosophy that enables community networking, as deployed network infrastructures remain open for other nodes to join and be used while being independent from any central entity.

In summary, the main contributions of this paper are:

- Propose a novel secured and decentralized routing protocol called SEMTOR. In SEMTOR users can set up trust sets of nodes, which are the only ones allowed to route their traffic.
- Describe how SEMTOR is implemented in BMX6, a routing protocol currently used in production community

ISBN 978-3-901882-80-7

wireless mesh networks.
- Analyze the hardware requirements of SEMTOR by investigating its performance in terms of traffic, CPU, and memory overhead. Our results show that SEMTOR can be deployed using off-the-shelf inexpensive WiFi routers.

The remainder of this paper is organized as follows. After looking at related work in Section II, we identify our design objectives in Section III and describe the mechanisms to solve these objectives in Section IV. Section V presents the implementation and performance analysis to validate our approach with a very small device and real traffic in an emulated network, followed by a discussion about security implications in Section VI. We conclude our work in Section VII.

II. Related Work

Existing work on secure routing for ad hoc and mesh networks has been reviewed in [19], [20]. Authenticated routing for ad hoc networks (ARAN) [21] as proposed by Sanzgiri et al. as well as Admittance-control enabling extensions for OLSRv2 proposed by Herberg et al. [22] use digital signatures to verify the authenticity and integrity of control messages. Both rely on the existence of a central certificate server trusted by all participating nodes.

SEAD [23] and SAODV [24] encounter the dependency on a central trust authority with a self-securing control plane. Using anchored hash chains (instead of a hop counter) they ensure that a malicious node cannot claim better distances to any remote node than it really has. However, both remain vulnerable to data-plane attacks such as packet dropping or routing-table poisoning. Work in [25] and [26] address this problem by punishing malicious nodes based on their forwarding behavior as observed and assessed by neighboring nodes. Adnane et al. [25] build on top of SOLSR and extend it with detection and reaction mechanisms. Mogre et al. [26] present another holistic approach combining self-securing routing, detection, reputation, and counter-measure mechanisms.

SEMTOR follows a different approach. In fact, guaranteeing in all aspects the correct operation of nodes is indeed hard and, as pointed out by Adnane et al. [25], cannot be guaranteed (e.g. data-plane attacks cannot be prevented) by securing the topological information exchanged between nodes. Therefore, instead of aiming to ensure or enforce correct operation, SEMTOR enables each node admin to freely define his individual subset (and resulting sub-topology) from the whole set of participating nodes that he considers sufficiently trustworthy to meet his security and data-delivery objectives and concerns. In addition, none of the yet presented work relying on asymmetric cryptography for verification of control messages has yet been analyzed in terms of performance and benchmarked based on real embedded hardware and exposed to traffic and network characteristics that are typical for existing community mesh network clouds.

III. Design Objectives

For the design of the protocol, the following objectives have been identified and are shortly summarized as follows.

Regarding **Ownership** of data traffic, forwarding routes, and node identities, the objective is that any community-network related routing-table entry and packet can be attributed to exactly one node of the network. Each node can be unambiguously identified with a secure identity and an identity-proving address. In addition, each node is the exclusive owner of exactly those routes pointing towards this address and those packets carrying this address as destination.

Regarding **Security**, the general objective is that any contiguous group of nodes (node admins), trusting and willing to cooperate and support each other, can not be prevented by an external (e.g. an attacker) from doing so.

Regarding **Openness and Decentralization**, the objective is that each node admin can individually decide which of the other nodes he wants to trust and rely on, that multiple cooperative groups of nodes (admins) can coexist, group membership is not exclusive, and there is no need for a central registry or authority. Nonetheless, similar to social networking or so called networks of trust, public key-servers or other decentralized coordination platforms may be used in addition to facilitate the management of known and trustable nodes and corresponding node IDs.

Regarding **Scalability**, an implementation of the proposed mechanisms should be feasible and scalable for the characteristics (e.g. number of nodes and links per node) of typical community-network clouds and given the resource limitations (e.g. CPU, memory, and bandwidth) of low-budged but state-of-the art embedded routers as used in today's community-network deployments.

IV. Protocol Overview

A. Concepts

The basic idea to achieve our objectives may be best illustrated with a simple example: If person x declared that he fully trusts in persons a and b, then any person y could hand out value for x to a or b while fully respecting x's assumptions of trust and knowing x as being in full charge of this action and any subsequent consequences. Further, these trust declarations should not be weakened by composition (intransitive).

In order to apply this idea for routing in IP networks, SEMTOR extends the concepts of *receiver-driven routing* [16], which already allows nodes to express path-selection preferences via a descriptive profile. Both approaches employ principles of table-driven, proactive, destination-sequenced distant-vector (DSDV) routing protocols where sequenced routing updates, originated by each node of a network and updated and re-broadcasted by each hop, are used to propagate path cost and versioning information in the network. In contrast to the traditional DSDV protocol [27], the routing update of SEMTOR contains a reference value, instead of a destination IP address or network, which unambiguously identifies a particular node of the network and a specific version of this node's self-defined description. This reference is given by the descHash, the SHA1 hash of the node's current description (also called **description reference**). Another unambiguous reference to a particular version of a node's current description

is given by the the tuple of its public-key hash (nodePKHash) and description sequence number (descSqn) that are given in protocol packet headers and description messages. The relations between packet-header fields, descriptions, and other protocol messages and content as potentially contained in a protocol packet are illustrated in Figure 1 and is discussed in more detail in the following.

The node **description** itself aggregates information about the node's current configuration and defines at least the following information:

- A reasonable strong and permanent **node-public key** (nodePK) used for verifying the self-signed description (as signed by the node's corresponding private key). In addition, the SHA1 hash of this public key (nodePKHash) is used as a **global ID**, an unambiguous and permanent reference for identifying the node.
- A **description version** – sequence number (descSqn) that must be incremented with every **description update** or node reboot, allowing to always differentiate between the latest and out-of-date versions of any node's description. Sequence numbers of routing updates and descriptions do NOT wrap around. Instead, the exhaustion of the former requires the generation of a new description (update) after which routing-update SQNs are reset but also lead to a new descHash. The exhaustion of a descSqn indicates the end-of-life of a used nodePK.
- A secondary (typically less strong) temporary **transmission public key** (txPK) used by neighboring nodes for (authenticity and integrity) verification of received link-discovery and routing-update messages signed and broadcasted by this node.
- The **primary IPv6 address** of the node, being a 128 bit cryptographically generated address (CGA) that integrates into the address itself a proof of identity and ownership. It is computed by combining a small (between 8–48 bits) prefix, typically from IPv6 Unique Local Address (ULA) range [28], and the most-significant (at least 80 bits) of the node's permanent global ID. The concept of CGAs has been proposed by [29] which also describes mechanisms to further enhance the security of generated addresses.
- The **trust set** of this node, essentially being a list of other node's global IDs, that defines the **trusted nodes** of this node. This way, the trust set implicitly defines, together with the links possible between any pair of trusted nodes, the **trusted virtual topology** of this node. Only the good-listed nodes identified via this trust set are authorized for propagating routing updates originated by this node. However, instead of expecting untrusted nodes to respect this demand, all trusted nodes are expected to ignore routing updates originated by this node that were not received directly and securely authenticated via any of the trusted nodes. As a consequence, only neighboring nodes that are trusted nodes of a described node are considered as next hop towards this node and eventually also end-to-end established routing entries towards this node are only set along these trusted nodes.
- Optional content such as **IP4in6 and/or IP6in6 tunnel announcements** announcing the availability of other hosts and networks via this node.
- A **description signature** (descSignature) matching the permanent node-public key given in this description and used for verifying the authenticity and integrity (i.e. this description was indeed created by the node with the corresponding global ID and was not changed by anyone else) of the description.

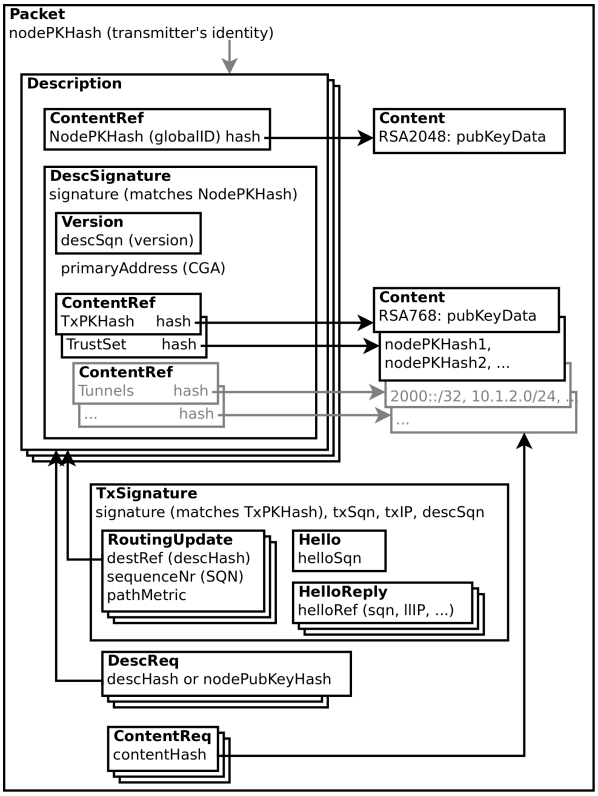


Fig. 1. Protocol packet, description-, content-, and reference structure

Instead of aggregating all the above data directly into a single and self-contained description message, parts of the description itself are given and defined only by the SHA1 hash of the actual original data (similar to the description reference used within routing updates to identify a node and its description). Such references-based definitions are called **content references** and are used for example to define the two public-keys or the trust sets of a node.

B. Bootstrapping

Our approach comes with the consequence that any received packet header or message, containing description or content references for which the corresponding original data is not known, cannot be instantly processed and must be requested first.

Figure 2 illustrates related bootstrapping procedures of the protocol in four phases: (1) the local node discovery, (2) the

link discovery, (3) the global node discovery and (4) the trusted path establishment.

The local node discovery (phase 1) shows how nodes $n2$ and $n3$ react on the reception of a new (for $n2$ and $n3$ yet unknown) nodePKHash or descSqn (here given by a received packet header) and resolve the description, identity (pubKey) and further referenced description content by sending corresponding **description or content request** messages (containing only the unresolvable hash) to node $n1$. Then $n1$ replies with the requested information. When all requested contents have been resolved, $n2$ and $n3$ can assemble the full description and verify its authenticity with the signature and public Key as defined by the initially received nodePKHash that triggered this process. The descHhash and signature is calculated over the raw description data and contained data hashes as they occur (not the referenced original data).

A transmission signature message (txSignature in Fig. 1 and 2) verifies the authenticity and integrity of all following **signature-demanding messages**. These include the locally exchanged link-discovery and routing-update messages but do not include the eventually globally exchanged descriptions (already signed explicitly via description signatures) or content advertisements (already signed implicitly via SHA1 reference hashes used in signed descriptions).

To protect against link-local replay attacks, the txSignature also covers the transmission sequence number (txSqn) that again can only be reused after renewing also the currently used txPK and a transmission IP (txIP) that must match the link-local src IPv6 address of the IP packet containing the protocol data.

The link discovery (phase 2) briefly indicates how the provided functionality can be used to authenticate the exchange of link-probing messages and subsequent deduced link qualities. Here, the hello message contains a sequence number that is unique during the lifetime of a node's description and the corresponding hello-reply message unambiguously references a previously received hello message and link via which this message has been transmitted (by specifying the hello-sequence number, the current description hash of the hello-sending node, and the txIP from which the hello has been send and via which own interface it has been received). By receiving correctly signed hello and hello-reply messages (referencing the receivers own hello messages via included txSqn and txIP), the exchange of link-probing messages becomes a continuous and bidirectional challenge-response handshake that only the holder of the corresponding private keys can maintain.

During the global node discovery (phase 3) nodes are discovered beyond the link neighborhood. The discovery is triggered by the detection of an unknown descHash contained in routing updates which causes the receiving nodes to resolve and verify the full node description (similar to local node discovery in phase 1).

Phase 4 illustrates the establishment of forwarding paths along trusted nodes. First $n2$ and $n3$ receive the routing update from $n1$, which description has already been fully resolved in phase 1. Because the update was received via an authenticated

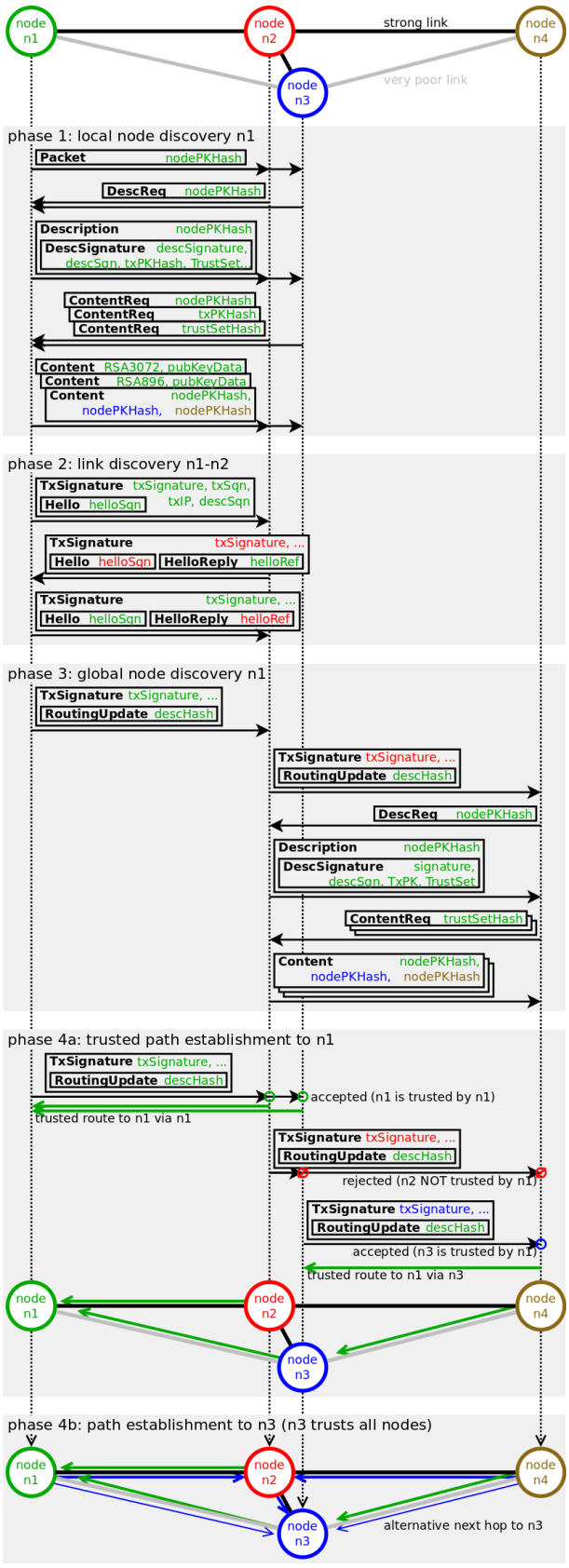


Fig. 2. Network bootstrapping (node, link, and path discovery)

link (due to its txSignature) with known link-quality (see phase 2) and because $n1$ itself is listed as a trusted node in the description, $n2$ and $n3$ know that $n1$ trusts the transmitter of the received message with regards to correctly process, update, and re-broadcast its routing update messages. In the following, the end-to-end path cost to the originating node and via the trusted link peer can be calculated based on the pathMetric of the update and the current link quality. In case the calculated cost identifies this link as the best next hop towards the originating node, the routing update is re-broadcasted with a modified pathMetric value that reflects the cost of the additional link (see [30] for more details).

In the next illustrated step, node $n3$ and $n4$ receive the routing update originated by $n1$ and re-broadcasted by $n2$. This time, the update-conveying txSignature verifies $n2$ as the transmitter and last modifier of the routing update. However, the identity (nodePKHash) of $n2$ is not listed in the trustSet described by the originating node $n1$ and therefore the update is discarded (by $n3$ and $n4$) for path establishment towards $n1$. Node $n4$ does not have a trusted route to $n1$ unless it receives the routing update via $n3$ which is listed within the trustSet of $n1$ and can therefore be considered as next hop towards $n1$. The topology depicted at the bottom of phase 4a illustrates with arrows the resulting virtual topology for routing traffic towards $n1$. In this example, $n2$ is the only node which $n1$ does not trust. Consequently, its traffic is directed around $n2$. Given the assumption that $n3$ trusts all nodes, phase 4b illustrates with blue arrows also the concurrent established topology for routing traffic towards node $n3$. Traffic from $n4$ to $n3$ will be routed via $n2$ (due to potentially strong involved links $n4-n2-n1$). But traffic towards $n1$ remain routed via $n3$ (being the only trusted path option towards $n1$).

C. Summary of Properties and Advantages

In the following, fundamental properties and advantages of this approach are summarized:

- Descriptions can be rather short while being able to securely specify large amounts of data (e.g. a single 160-bit SHA1 hash compared to a 2048-bit RSA signature or a trust set listing the global IDs of hundreds of nodes).
- Referenced data that does not change over a description update or that is used by several nodes (e.g. defining equal trust sets) does not need to be re-propagated since their reference (hash) would not change and can be reused.
- A significant processing advantage is achieved by using a secondary and rather weak temporary key for the continuous packet signing and validation operations (typically occurring several times per second), leaving a potential attacker with insufficient time for cracking this key before it gets replaced by a new one (typically every few hours). On the other hand, a reasonable strong key can be used for signing the rather seldom propagated description updates (typically once per hour) and thereby allowing a long-term protection of node IDs and descriptions.
- The mechanisms allow (without requiring any pre-deployed infrastructure or registry) the network-wide, dynamic, and secure (in terms of authenticity and integrity) deployment of a public-key infrastructure that can be used for further integrity verification of link-discovery and routing-update messages between neighboring nodes and ensuring the propagation of routing updates along nodes trusted by the originator of these updates.

V. Implementation and Performance Analysis

To test our approach, the message structure and processing of BMX6 [12], a mesh-routing protocol that already supports node-descriptive profiles, uses SHA1 hashes as referencing identifier, and provides a "hash-based profile-propagation mechanism" to disseminate node descriptions, has been extended to implement the proposed behavior. For cryptographic operations such as hashing, asymmetric-key generation, signing, and verification the PolarSSL [31] library has been used. The preliminary source code is publicly available via the SEMTOR git-repository branch at [12].

Functional support for objectives as outlined in Section III, such as data-, route-, and address- ownership or for allowing any individual selection of trusted nodes without disrupting the route-establishment of other nodes, has been tested by emulating selected networking scenarios with Linux nodes running our SEMTOR-enabled BMX6 implementation and using MLC [32], a suite of LXC[1]-based scripts, that allow to virtualize hundreds of Debian systems inside a single host and to configure virtual topologies, essentially by linking virtual or real nodes' interfaces via a virtual bridge and using ebtables[2] and tc[3] to control packet loss and delay between nodes on layer two. The host system was given by an Intel i5-3230M CPU @2.60GHz with 8GB RAM, allowing to run more than 200 virtual systems and protocol instances in parallel.

To evaluate our design and implementation with respect to its performance implications and our scalability objectives outlined in Section III, the effect of network and protocol parameters on protocol overhead and performance has been measured by benchmarking our implementation on a cheap embedded device with hardware characteristics as summarized in Table I and that can be considered the bottom end of devices typically used within community networks [3], [5]. This device, running the OpenWRT OS [33] and a cross-compiled version of our implementation, has been connected as a core node via Ethernet to the MLC controlled bridge that instantiated the virtual topology and stressed with real traffic generated by the MLC emulated SEMTOR instances. Relevant configuration parameters for our measurements, essentially consisting of a 10x10 nodes grid topology, are given in Table II. Selected defaults and probing ranges comprise network sizes and densities that are typical for deployments in clouds and core-graph zones of real community networks [3], [4], [5].

The overhead has been measured in terms of relative system CPU and virtual memory usage (using the Linux top command) and protocol-data (using tcpdump). Graphs in Figure

[1]Linux Containers http://lxc.sourceforge.net/

[2]Linux ebtables, http://ebtables.sourceforge.net

[3]Linux traffic control: tc, http://tldp.org/HOWTO/Traffic-Control-HOWTO/

TABLE I
HW AND OS CHARACTERISTICS OF USED TARGET DEVICE

Characteristic	Details
Type / CPU	TP-Link TL-WR703N, Atheros AR7240@400 MHz
Wireless	AR9331, 802.11bgn 150 Mbps @100 mW
Flash / Memory	4 MB / 32 MB
Ports	100 MBit Ethernet, USB 2.0
Power supply	5 V, 100 mA, 0.5 W
Cost	approx 10 Euro (2013)
OS and distro	Linux OpenWrt (Chaos Calmer, r46943)
Further reading	http://wiki.openwrt.org/toh/tp-link/tl-wr703n
Routing	BMX6 semtor branch, git revision 2fb169f
Libraries	PolarSSL version 1.3.4

TABLE II
DEFAULT PARAMETRIZATION OF EMULATION AND PROTOCOL

Parameter	Default [range]	Fig.
Network size (number of nodes)	100 [10..150]	3(b)
Density (number of links per node)	4 [4..20]	3(c)
Node interfaces	1	
Grid network structure (topology)	10x10 [10x1..10x15]	
Link dynamics and loss	static @ zero loss	
Cryptographic strength of primary key	RSA3072	
Cryptographic strength of txKey	RSA896 [512..1536]	3(d)
Description-update interval	36000 s [100..4 s]	3(a)
Routing updates interval	6 s	
Link-probing interval	0.8 s	
Max message aggregation (TX) interval	0.8 s	

3 represent the results of these measurements averaged over 30 seconds. To save space and ease comparability, CPU, memory, and protocol-traffic overhead are represented in a single Figure for each analyzed input parameter. Protocol-data measurements are given per node and distinguish between received (RX) and sent (TX) data and in packets and bytes per second. Due to a default of four neighbors (links) per node, RX traffic is divided by four, resulting in (except for Fig. 3(c)) essentially overlapping RX and TX lines.

The impact of a node updating its description at higher (non-default) rates is illustrated in Figure 3(a). The measurements show that protocol-data overhead and CPU usage are indeed significantly affected by such bootstrapping events and that update intervals occurring at rates higher than twice per second have the potential to seriously harm the stability of the protocol; a finding that we'll get back to later. To avoid capturing massively parallel bootstrapping phases, an unlikely scenario for real nodes under distributed administration, further measurements have been delayed for a stabilization period of 120 seconds after each protocol-configuration change.

From Figure 3(b) it can be seen that network size[4] roughly linearly affects CPU, memory, and traffic usage per node; but CPU raises slower than the others. This could be explained by the fact that the most CPU-intensive operation, being the continuous and asymmetric signing and verification of hello and routing update messages, is performed on a per-packet basis (and not on a message basis) and that almost no additional transmissions (packets) are needed to convey the increasing amount of protocol-data.

As can be seen from Figure 3(c), CPU and traffic overhead heavily depends on the number of links to which the test node is exposed while no additional memory requirements were observed for maintaining an increasing amount of links. This could be explained with the relatively small related memory requirements for maintaining these links compared to the allocations needed for maintaining node descriptions and thereby referenced data which must be tracked anyway, also for non-neighboring nodes.

The impact of RSA key strength of txSignatures is illustrated in Figure 3(d). The results confirm the expectation that stronger keys lead to linearly increasing protocol-data overhead and exponentially increasing CPU load. However, it is interesting to note that using RSA key strength such as RSA1024, which can be considered secure against factorization attacks by today and thus sufficient for the supposedly short lifetime of only few hours in our approach, for continuous signing and verification is well feasible and requires only about 8% of the target device CPU resources.

Throughput (TP) was measured (using iperf) with the objective to detect degradations that could be attributed to the concurrent load caused by the routing protocol. Interestingly, the maximum TCP throughput, ranging around 80 Mbps for purely routed traffic (forwarded in and out via a single symmetric 100 Mbps Ethernet link), was not notably affected at any point. This can be explained with the fact that (even with the highest observed, in- and out-going protocol traffic of less than 0.14 Mbps) the consumption of link-capacity, given a 100 Mbps link is much less than 1 percent.

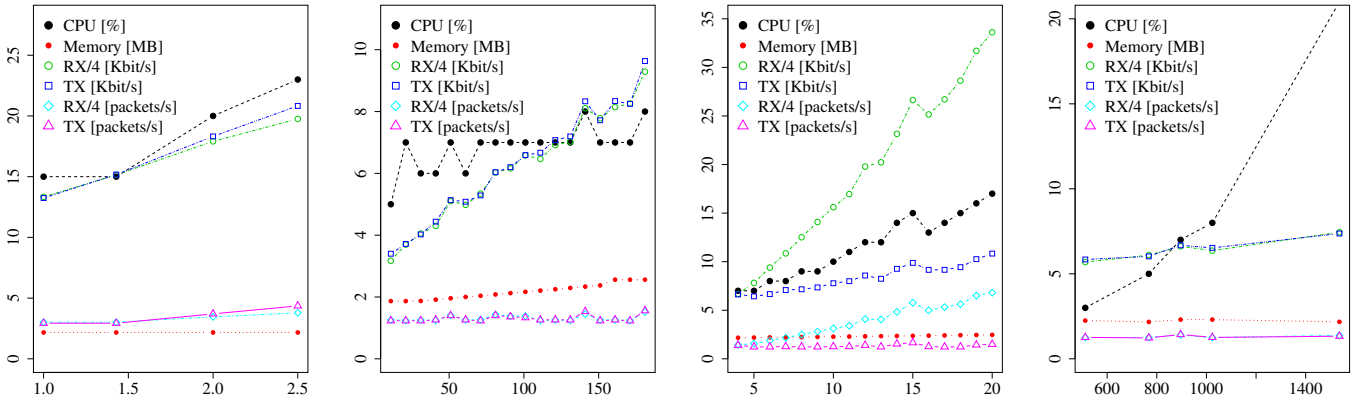
VI. SECURITY DISCUSSION

In the following, routing-security aspects are discussed in the context of the objectives defined for this work and grouped into (i) those out of the scope or not considered an attack (although worth discussing), (ii) those addressed and explicitly handled, and (iii) those remaining a potential threat.

Confidentiality attacks via eavesdropping of disseminated protocols data fall into the first category. Such data includes topology, trust, identity, and if published, personal information. Particularly, the disclosure of trust sets has raised concerns[5] as its transparent dissemination allows others to deduce trust relations between the users of a community network and exploit this knowledge for attacking individuals or groups of them in higher layers (e.g. social layer). However, it should be noted that the provisioning of personal information via node descriptions or other parallel infrastructure services is left to the preferences of each node admin and could be omitted (thus revealing only trust relations between anonymous cryptographic identities) at the cost of complicating the task for identifying the nodes considered as trustworthy.

[4]Trust sets described per node were kept constant despite the increasing topology size, adding only a constant memory overhead per additional node.

[5]E.g. during discussions and presentations of our approach at community events such as the International Summit for Community Wireless Networks 2013 and the Wireless Battle Mesh in Germany 2014 and Slovenia 2015.



(a) Ovhd. vers. reconfigurations. (b) Overhead vers. network size (c) Overhead vers. network density (d) Ovhd. vers. TX-signature

Fig. 3. Overhead versus different network and protocol characteristics

Malign attacks also fall into this category, being the case where a malicious node aims to influence the reputation of another node by incorrectly reporting on its negative behavior. On the protocol layer such attacks are not possible simply because no functionality exist for detecting nor reporting behavior (good or bad) of other nodes. This also means that **selfish, non-cooperative, and unfair behavior** is not considered by the protocol and (as yet) left to be solved independently (e.g. via reputation systems such as [25], [26]) or by future work. However, it should be noted that with the support for verifiable and dynamically-updatable node descriptions, self bootstrapping public-key infrastructure, and individually-definable trust topology, powerful tools are provided, that can be used for arguing on the trustability of nodes and enforcing individual decisions without requiring consensus among network participants.

Data-plane attacks, compromising confidentiality, authenticity, integrity, non-replication, and non-repudiation of user data are also considered as out of scope. Various existing and well established end-to-end security solution should be used instead such as TLS-based protocols like https, VPN solutions like openvpn or tinc, and anonymization and privacy protocols like TOR. In fact, the security aspects covered by our protocol, operating on layer 3, complement these higher-layer solutions as these build on the availability of a functioning end-to-end data-delivery service in the first place, partially for creating encrypted overlay networks on top of it.

Such service should be provided by the **control plane** which can be attacked by disturbing the propagation of node and topology information or the forwarding of data packets. Attacks on this plane from non-trusted nodes are falling into the second category and preventing related attacks [20] (such as blackhole, partition, detour, routing-table poisoning, impersonation via replication, dropping, modification, or fabrication of protocol messages) is what the main contribution of this work is about. This is essentially achieved by excluding all non-trusted nodes from related attack-susceptible tasks such as route-discovery, establishment, and forwarding. The key enabler for this approach are the strictly non-overlapping receiver-driven responsibility for these "trusted tasks", the usage of a permanent strong asymmetric key pair (used for authenticating node identity, cryptographically generated addresses ensuring conflict-free IPv6 addresses and route ownership, trusted nodes, and secondary-key replacements) and a secondary lightweight key pair of which security relies on its short lifetime and frequent replacement (used for frequent tasks of verifying communication between neighboring nodes and to identify and discard data from non-trusted nodes).

Wormhole and Denial of Service (Dos) attacks are falling into the third category that remain unsolved with yet defined mechanisms. While several approaches, such as TIK [34] using packet leashes, or NPA [35] observing standard deviation of round trip times exist to defend against wormhole attacks, the threat of DoS attacks on the security of mesh routing protocols has received much less attention.

The susceptibility to DoS attacks is actually a pillar stemming from the open and decentralization objectives pursued with our approach that provides no means for excluding malicious or selfish nodes from exploiting or exhausting other nodes resources. For example, an attacker can fake and propagate large numbers of physically non-existing virtual node IDs with frequently updated node descriptions that can hardly be distinguished from real nodes but whose processing may easily exceed existing memory and CPU resources in real nodes. The consequences of such attack could already be seen from the measurements performed in Section V showing overhead depending on the number of nodes or description update frequency. To defend against such kind of attacks, without sacrificing the decentralization and open preambles defined for this work, nodes may prioritize the processing of certain known nodes and throttle the support of unknown, an approach we plan to research further in future work.

VII. Conclusions and Future Work

In this work we pointed to new trust and security requirements arising for open and decentralized networks such as community networks and described SEMTOR, a novel routing protocol that can be used for satisfying these demands.

SEMTOR allows the verifiable and undeniable definition and distributed application of user-individual trust topologies for routing traffic towards each node. One particular advantage of SEMTOR is that it does not require a global consensus on the trustiness of any node. This gives each node admin the freedom to individually define the subset (and resulting sub topology) from the whole set of participating nodes that he considers sufficient trustworthy to meet his security and data-delivery objectives and concerns.

Addressed security aspects have been discussed and put in context with related and orthogonal security solutions and how these can benefit by building on it.

The proposed mechanisms have been implemented, tested and evaluated for their scalability regarding network size, density, reconfiguration dynamics, and strength of used crypto parameters. Measurement results, obtained via benchmarking on low-end embedded hardware, show scalability limits and parameters with significant impact on performance and overhead. The results also show that the usage of strong asymmetric cryptography for building trusted routing-topologies is possible; even given the scalability requirements imposed by environment and characteristics of nowadays community-network clouds. In the future we plan to research further the impact and challenges due to the presence of large numbers of anonymous nodes and denial of service attacks.

ACKNOWLEDGMENTS

This work is partially supported by the European Community Framework Programme 7 (FP7) within the FIRE Initiative, Community Networks Testbed for the Future Internet (CONFINE), contract FP7-288535, and by the Spanish government, contract TIN2013-47245-C2-1-R.

REFERENCES

[1] C. Szabó, K. Farkas, and Z. Horváth, "Motivations, design and business models of wireless community networks," *Mob. Netw. Appl.*, vol. 13, no. 1-2, pp. 147–159, Apr. 2008.

[2] P. Pathak and R. Dutta, "A survey of network design problems and joint design approaches in wireless mesh networks," *Communications Surveys Tutorials, IEEE*, vol. 13, no. 3, pp. 396–428, Third 2011.

[3] J. Avonts, B. Braem, and C. Blondia, "A questionnaire based examination of community networks," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*, Oct 2013, pp. 8–15.

[4] D. Vega, R. Baig, L. Cerdà-Alabern, E. Medina, R. Meseguer, and L. Navarro, "A technological overview of the guifi.net community network," *Computer Networks*, vol. 93, Part 2, 2015, community Networks.

[5] L. Cerdà-Alabern, A. Neumann, and P. Escrich, "Experimental evaluation of a wireless community mesh network," in *The 16th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM'13*, Barcelona, Spain, Nov. 2013.

[6] "Pico peering agreement," http://picopeer.net.

[7] D. Murray, M. Dixon, and T. Koziniec, "An experimental comparison of routing protocols in multi hop ad hoc networks," in *Telecommunication Networks and Applications Conference (ATNAC), 2010 Australasian*, Oct 2010, pp. 159–164.

[8] D. Seither, A. Konig, and M. Hollick, "Routing performance of wireless mesh networks: A practical evaluation of batman advanced," in *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, Oct 2011.

[9] A. Neumann, E. López, and L. Navarro, "Evaluation of mesh routing protocols for wireless community networks," *Computer Networks*, vol. 93, Part 2, 2015, community Networks.

[10] I. D. Chakeres, "Aodv routing protocol implementation design," in *In ICDCSW '04: Proceedings of the 24th International Conference on Distributed Computing Systems Workshops - W7: EC (ICDCSW'04*. IEEE Computer Society, 2004, pp. 698–703.

[11] "Babel – a loop-avoiding distance-vector routing protocol," http://www.pps.univ-paris-diderot.fr/~jch/software/babel/.

[12] "BMX6 mesh networking protocol," Jul. 2014, http://bmx6.net.

[13] "olsrd - an adhoc wireless mesh routing daemon," http://olsrd.org.

[14] "batman advanced - a layer 2 implementation of the B.A.T.M.A.N. routing protocol," http://www.open-mesh.org/projects/batman-adv/wiki.

[15] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Comput. Netw. ISDN Syst.*, vol. 47, no. 4, Mar. 2005.

[16] A. Neumann, L. Navarro, R. Baig, and P. Escrich, "Receiver-driven routing for community mesh networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, June 2013, pp. 1–7.

[17] P. Psenak, S. Mirtorabi, A. Roy, L. Nguyen, and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF," RFC 4915, Internet Engineering Task Force, June 2007.

[18] M. Hauge, M. Brose, J. Sander, and J. Andersson, "Multi-topology routing for qos support in the consis convoy manet," in *Communications and Information Systems Conference (MCC), 2012 Military*, Oct 2012.

[19] S. Gupte and M. Singhal, "Secure routing in mobile wireless ad hoc networks." *Ad Hoc Networks*, vol. 1, no. 1, pp. 151–174, 2003.

[20] L. Abusalah, A. Khokhar, and M. Guizani, "A survey of secure mobile ad hoc routing protocols," *Communications Surveys Tutorials, IEEE*, vol. 10, no. 4, pp. 78–93, Fourth 2008.

[21] K. Sanzgiri, D. LaFlamme, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer, "Authenticated routing for ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 3, March 2005.

[22] U. Herberg, T. Clausen, and J. Milan, "Digital signatures for admittance control in the optimized link state routing protocol version 2," in *Internet Technology and Applications, 2010 Int. Conf. on*, Aug 2010.

[23] Y.-C. Hu, D. B. Johnson, and A. Perrig, "Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks," in *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, ser. WMCSA '02, Washington, DC, USA, 2002.

[24] M. Guerrero-Zapata, "Secure ad hoc on-demand distance vector (saodv) routing," Sep. 2006, draft-guerrero-manet-saodv-06.txt.

[25] A. Adnane, C. Bidan, and R. T. de Sousa Júnior, "Trust-based security for the {OLSR} routing protocol," *Computer Communications*, vol. 36, no. 10–11, pp. 1159 – 1171, 2013.

[26] P. S. Mogre, K. Graffi, M. Hollick, and R. Steinmetz, "A security framework for wireless mesh networks," *Wirel. Commun. Mob. Comput.*, vol. 11, no. 3, pp. 371–391, Mar. 2011.

[27] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, ser. SIGCOMM '94. New York, NY, USA: ACM, 1994.

[28] R. Hinden and B. Haberman, "Unique Local IPv6 Unicast Addresses," RFC 4193 (Proposed Standard), Internet Engineering Task Force, October 2005.

[29] T. Aura, "Cryptographically Generated Addresses (CGA)," RFC 3972 (Proposed Standard), Internet Engineering Task Force, March 2005.

[30] L. Cerd-Alabern, A. Neumann, and L. Maccari, "Experimental evaluation of bmx6 routing metrics in a 802.11an wireless-community mesh network," in *4th International Workshop on Community Networks and Bottom-up-Broadband (CNBuB'2015)*, Rome, Italy, Aug. 24–26, 2015. [Online]. Available: http://research.ac.upc.edu/CNBuB2015

[31] "POLARSSL – Straightforward, Secure Communication," http://polarssl.org.

[32] Axel Neumann, "Investigating Routing-Protocol Characteristics with Mesh Linux Containers (MLC)," Workshop, UPC, Barcelona, Spain, Nov. 2011, https://raw.github.com/axn/mlc.

[33] "OpenWrt Linux distribution for embedded devices," http://openwrt.org.

[34] Y.-C. Hu, A. Perrig, and D. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, March 2003, pp. 1976–1986.

[35] J. Zhou, J. Cao, J. Zhang, C. Zhang, and Y. Yu, "Analysis and countermeasure for wormhole attacks in wireless mesh networks on a real testbed," in *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, March 2012.