It's About Time: Securing Broadcast Time Synchronization with Data Origin Authentication

Robert Annessi, Joachim Fabini and Tanja Zseby
Institute of Telecommunications, TU Wien, Austria
{robert.annessi, joachim.fabini, tanja.zseby}@tuwien.ac.at

***Abstract*—Due to the increasing dependency of critical infrastructure on synchronized clocks, network time synchronization protocols have become an attractive target for attackers. We identify data origin authentication as the key security objective and therefore conduct a comprehensive, theoretical evaluation of data origin authentication schemes from different application fields with regard to their applicability to secure broadcast time synchronization. Some evaluated schemes were found to be susceptible to message delay attacks in the context of time synchronization - including TESLA, the approach currently favored by the IETF NTP working group and also on the shortlist of the P1588 Security Subcommittee for PTP. Two of the evaluated schemes, however, come somewhat close to meeting the evaluation criteria derived from our time synchronization specific threat analysis, and therefore qualify as promising candidates to secure broadcast time synchronization.**

I. Introduction

Time synchronization protocols have become essential to critical infrastructures such as telecommunication, industrial automation, avionics, or energy distribution. Attacks on time synchronization can originate faulty sensor reports, endanger control decisions, and adversely affect the overall functionality of applications depending on it. A successful attack on time synchronization can even undermine the security of essential cryptographic protocols [1] such as TLS. For these reasons, time synchronization protocols have to be secured whenever they are used outside of fully protected network environments [2].

In accordance with RFC 5905 [3] we use the term broadcast communication in this paper for any one-to-many communication, i.e., multicast and broadcast. In broadcast communication, a server[1] periodically sends time synchronization messages that can be received by multiple clients. In contrast to unicast communication, time synchronization messages are not sent to clients individually in broadcast communication. Instead, the server sends each time synchronization message only once, and the message is replicated by the communication network along distinct paths whenever needed. In this way, broadcast communication is more efficient when handling multiple clients, because the load on the server is reduced and the number of copies that traverse a network is minimized. In the Network Time Protocol (NTP) [3], broadcast communication is primarily used for more efficient communication because the increasing use of network time synchronization results in additional load on time servers and networks. In the Precision Time Protocol (PTP) [4], on the other hand, broadcast communication is inherently integrated into the protocol.

While unicast communication can be appropriately secured, the two most widely used time synchronization protocols, NTP and PTP, do not protect broadcast time synchronization messages sufficiently, leaving applications vulnerable to attacks. Although various broadcast authentication schemes have been proposed, no general solution as yet exists that can satisfy all constraints and requirements of the different applications. Authenticating broadcast time synchronization, therefore, remains an interesting open problem that requires significant attention [5].

Contributions

In this paper, our aim is to maintain both data origin authentication, which we identify as the foundation of any security solution to broadcast time synchronization, as well as the lowest possible degradation of time synchronization precision. By establishing an explicit threat model, we identify the properties that need to be provided by data origin authentication schemes to secure broadcast time synchronization. This threat model entails a network model of broadcast time synchronization, an adversary model, and a comprehensive set of attacks that adversaries may conduct. From the threat model, we infer a set of (time synchronization specific) requirements that need to be provided by data origin authentication schemes.

With regard to the requirements identified in the threat analysis, we conduct a thorough, theoretical evaluation of numerous data origin authentication schemes. To the best of our knowledge, this paper is the first evaluation of data origin authentication schemes with specific focus on securing broadcast time synchronization messages. Among others this systematic evaluation (a) yields new insights such that the currently favored candidate, TESLA, is unsuitable to secure broadcast time synchronization and (b) reveals two promising alternatives.

[1] In accordance with NTS (see Section III), we use the term server for both an NTP server and a PTP master, and client for NTP clients and PTP slaves.

II. Background

Due to the lack of access control in many communication networks, cryptographic schemes are required to ensure that clients can verify that messages have been, indeed, sent by the claimed server. This security property is called data origin authentication. Data origin authentication directly affects integrity, authenticity, and non-repudiation, and is therefore essential to any security solution. In the context of time synchronization, confidentiality is generally of less concern.

The term group refers to a set of communicating entities. In the context of time synchronization, group refers to one server and multiple clients. Group Authentication assures that messages originate from a valid but not uniquely identifiable group member and are not been modified by entities outside the group. Message Authentication Codes (MACs) with a key shared by the server and all clients are a well understood and efficient solution for achieving this level of authenticity. However, clients cannot distinguish between the individual parties sharing the key and, therefore, cannot know the exact identity of the origin, since not only the server but any client can generate authenticated messages. This indistinguishability is of particular importance in broadcast communication, as there are numerous clients involved, and a single malicious or compromised client is capable of imitating the server.

Since it cannot be assumed that every single client is generally trustworthy, a level of authentication is required that allows clients to identify a specific server: data origin authentication. To this end, a server needs to use an asymmetric mechanism that allows clients to verify authenticity without providing means to generate valid authentication information themselves on behalf of the server. Table I summarizes the security properties provided by group authentication and by data origin authentication.

Table I
Group Authentication vs. Data Origin Authentication

Security Property	Group Authentication	Data Origin Authentication
Integrity	✓	✓
Non-repudiation	✗	✓
Authenticity	Group	Server

Digital signatures provide data origin authentication. The main downside of today's digital signature schemes such as RSA, DSA, and ECDSA is that they involve high computational cost and substantial penalty in terms of delay, both in the server and in the client. Consequently, it is widely believed that digital signatures are roughly 2 to 3 magnitudes slower than MACs [6], such that signing each packet is not a practical solution. In particular, the time that is needed for signing packets and verifying digital signatures is detrimental to the precision of time synchronization protocols (as will be detailed in Section V).

III. State of the Art

Security measures can either be implemented as an integrated part of time synchronization protocols or through external security mechanisms, such as (D)TLS or IPsec, which are (somewhat) independent of time synchronization. There are three main drawbacks associated with external security mechanisms, in particular with (D)TLS and IPsec. First, they significantly lower the precision of time synchronization [7] because of the delays and the delay variation in the security stack. Second, (D)TLS and IPsec can provide only group authentication for broadcast communication but not data origin authentication. Third, they prevent intermediate devices in PTP (transparent clocks) from adding information, which would improve the precision of time synchronization by communicating the delay introduced by that devices to a particular message. Another external security measure is MACsec, which is limited to layer 2 and can, therefore, only provide security to local networks. Furthermore, each involved node needs to be trusted in MACsec as it follows a hop-by-hop encryption approach.

In this paper, our aim is to maintain both data origin authentication and the lowest possible degradation of time synchronization precision. To this end, we conclude that an authentication scheme should be integrated into the time synchronization protocol, and data origin authentication should be provided using an asymmetrical cryptographic mechanism.

PTP includes an experimental security extension, Annex K [4], which provides message integrity and replay protection. Annex K, however, is based on symmetric-key cryptography and, therefore, cannot provide data origin authentication. Furthermore, several flaws were discovered and it was never properly formalized [8, 9]. NTP, on the other hand, incorporates two integrated security mechanisms to provide authenticity and integrity: symmetric [3] and Autokey [10]. Both integrated security mechanisms can only achieve a group authentication level of security for broadcast communication. Furthermore, the use of Autokey is strongly discouraged as severe security weaknesses of the algorithm have been discovered [11, 12]. As potential successors of Autokey two distinct proposals are being discussed: ANTP [13] and Network Time Security (NTS) [2, 14, 15]. ANTP is intended to provide security for unicast NTP only, and cannot be extended to provide efficient data origin authentication for broadcast time synchronization as it is based on symmetric-key cryptography. NTS consist of a set of Internet Engineering Task Force (IETF) drafts that aim to provide authenticity and integrity for unicast and broadcast time synchronization protocols. A formal analysis was conducted for the unicast mode [16] but no evaluation is provided to motivate the use of Timed Efficient Stream Loss-tolerant Authentication (TESLA) [17] as favorite candidate for securing broadcast time synchronization. TESLA is highly rated for securing also the next version of PTP [9]. Besides other existing authentication schemes, TESLA will be evaluated in Section VI for its suitability to secure broadcast time synchronization.

Although evaluations of data origin authentication schemes for broadcast communication have been conducted before, they either include only few, specific schemes [18, 19], are dated [20], or are tied to other applications [21] so that their results are only partially applicable to broadcast time synchronization. In contrast to existing work, we conduct a comprehensive evaluation of data origin authentication schemes with regard to their suitability to secure broadcast time synchronization.

IV. Threat Model

Throughout the paper, we assume that clients already know the public key of the particular server and can trust that this public key is both valid and correctly bound to the server; i.e., we assume that the certified public key has been transmitted initially in a reliable and authenticated manner. A secure key publication mechanisms can be conducted in the clients' initial setup phase or provided during runtime by a certificate authority, for example.

A. Network Model

The overall purpose of time synchronization protocols is to convey time information in order to compensate for clock drifts. We consider communications involving one server and a potentially large number of clients. Messages are delivered from the server to each client through an unreliable, potentially lossy communication network, such as the Internet. Furthermore, we assume that neither network devices nor clients can be trusted, since the larger the number of clients the higher the probability that at least one is compromised; besides that clients might not even be controlled by the same entity who controls the server. The network only forwards packets, it does not provide any security guarantee such as integrity, authenticity, or availability. For this reason, the communication channel is considered insecure [22], which means in particular that messages may be read, modified, dropped, or injected by entities other than the intended clients. This corresponds to the Dolev-Yao threat model [23].

B. Adversary Model

We assume that the adversary does not permanently prevent the communication between server and clients; i.e., clients will receive some messages unaltered. But we do not make any assumptions about the actual ratio of received to sent messages. The adversary has full control over the network and can selectively capture, drop, resend, reorder, inject, delay, and alter messages arbitrarily with negligible delay. The computational power of the adversary is limited but not necessarily bounded to that of the server nor the clients; i.e., the adversary can use more powerful devices and larger storage. Furthermore, the adversary can compromise an arbitrary number of clients and learn all secrets they know. An important aspect of assuming such powerful adversary is that if security properties hold up, they also hold up against less capable adversaries without having to redesign the security solution from scratch.

C. Attacks

The goal of the adversary is to make clients adhere to false time values, degrade the precision of time synchronization, or deny access to the time synchronization service. For this purpose, the adversary can conduct various attacks [24]:

1) Message Substitution Attack: In a substitution attack [25], the adversary intercepts valid time synchronization messages during transmission and modifies them in such a way that clients accept the forged messages as if they had been sent by the original server.

2) Message Replay Attack: An adversary can record time synchronization messages and replay them without modification at a later time, since successful verification of a message does not certify the correctness of the message's send time [6]. In this way, inaccurate information can be intentionally provided to clients. It is, therefore, important to protect against replay attacks by ensuring that every message is distinct and can be tied to a specific session or time.

3) Server Impersonation Attack: For conducting an impersonation attack [25], the adversary utilizes knowledge of the authentication scheme used by the server. By constructing authentication information for forged messages, the adversary attempts to fool the clients into thinking that these messages were sent by the server. In this way, the adversary impersonates the server in order to distribute false time information to clients.

4) Message Removal Attack: A message removal attack results when time synchronization messages are intentionally dropped by the adversary from a privileged middleman position. As previously stated, we assume that the adversary does not drop all messages because this would then be an issue of availability rather than authenticity and integrity. Dropping messages, however, can degrade the precision of time synchronization. Obviously, authentication schemes cannot prevent such message removal attacks; some schemes, however, are adversely affected by packet loss.

5) Message Delay Attack: To conduct a message delay attack, an adversary intercepts time synchronization messages and delays these messages artificially for some time before forwarding them. If the effective delay is calculated maliciously, the clock of the intended client(s) can be manipulated [24]. Since the time synchronization protocol has no detailed information of the underlying communication infrastructure, symmetric delay between server and client is assumed, i.e., the one-way delay from server to client is the same as the delay from client to server. Message delay attacks exploit this assumption of symmetric delay by maliciously introducing asymmetric delay in such a way that the client synchronizes to an inaccurate time. Authentication schemes cannot prevent adversaries from conducting message delay attacks, as successful verification of messages does not imply that send and propagation times along the network path are correct [6]. In previous work, the proposed mitigation strategies coupled time synchronization protocols with delay measurement [26], either by using multiple servers [3] or multiple paths between the server and the client [26] (or by comparing the round-trip-time

to a predetermined threshold [27]). In this way, the probability of an adversary controlling the majority of servers or paths, respectively, is reduced.

6) Message Flooding: In Denial of Service (DoS) attacks [28], the computational or storage capacities of clients are exhausted in order to prevent or delay the reception of messages. Such DoS attacks can be conducted, for example, by an adversary sending an excessive number of time synchronization messages to a client. Authentication schemes cannot prevent message flooding attacks but they can reduce their impact as they can provide means for clients to distinguish valid from invalid messages (up to some number of packets per second).

7) Summary: Table II summarizes the threat analysis. The impact column provides a measure of the attack's severity, which can have two possible values: (1) false time and (2) degraded precision. False time (1) means that clients synchronize to a false time due to the attack, which may cause a disturbance of applications that rely on precise time. Degraded precision (2) means that the attack causes time synchronization of clients to be degraded.

Table II
THREAT ANALYSIS SUMMARY

Attack	Impact	Prevented by or with data origin authentication
Message substitution	False time	✓ Yes
Message replay	False time	✓ Yes
Server impersonation	False time	✓ Yes
Message removal	Degraded precision	✗ No
Message delay	False time	✗ No
Message flooding	Degraded precision	✗ Partially

As pointed out in this section, attacks that just degrade precision, message removal and message flooding, can not or only partially be prevented by data origin authentication. Also, additional means are required in order to mitigate delay attacks. Nevertheless, two severe attacks, message substitution and server impersonation, can be prevented entirely by data origin authentication, which is also a prerequisite to prevent replay attacks.

V. REQUIREMENTS

In this section, we discuss how the threats that have been identified in the previous section can be countered. Complementing the main security goals, authenticity and integrity, a set of requirements for data origin authentication schemes is defined that enable clients to detect injected or manipulated messages. Furthermore, criteria are introduced in order to evaluate the suitability of authentication schemes for securing broadcast time synchronization.

Authentication of Servers

Clients must securely authenticate the server [3, 24], which can be achieved by a data origin authentication scheme that provides end-to-end authenticity.

Message Integrity

Authentication schemes must provide integrity of time synchronization messages so that time synchronization protocols are not exposed to substitution attacks. This property is also provided by data origin authentication schemes. As pointed out in Section III, it is difficult for external security measures to provide message integrity for PTP because intermediate devices need to be able to modify messages in transit in order to improve precision.

Availability

The availability of time synchronization services can be compromised by DoS attacks. Authentication schemes should at least limit DoS attacks, but above all not introduce itself as a new DoS attack vector.

Message Replay Protection

There are various ways to perform replay attacks, and no single measure can prevent them generally. In broadcast time synchronization, however, replay protection can be ensured by making every message distinct. As an example, one way could be adding a cryptographic nonce such as a monotonically increasing number [6] and clients (and server) keeping track of the nonce used last.

Computational Efficiency

The generation as well as verification of authentication information should require as low computational effort as possible to minimize the impairment onto the precision of time synchronization [7, 5, 13]. Conformance to this requirement also supports implementation of authentication schemes within resource constrained devices.

Low Communication Overhead

For efficiency reasons, the authentication information should be as small as possible because bandwidth is a valuable resource.

Immediate Authentication

Some authentication schemes require the server or the client to buffer packets before the authentication information can be generated or verified, respectively. In this way, schemes can reduce the computational resources needed overall for signing and verification. In case of server-side buffering, additional delay is introduced, which lowers the precision of time synchronization. When clients need to buffer messages, an adversary can flood the clients' buffers with bogus messages. In this way, a DoS attack can be conducted, because clients are unable to store and authenticate incoming messages when their buffers are full. Depending on the authentication scheme, server-side and client-side buffering may apply to the same packet.

Collusion Resistance

As mentioned in Section IV, clients are not necessarily trustworthy. Data origin authentication schemes must, therefore, provide protection against clients that collude in order to impersonate the server. Some authentication schemes, however, are parameterized on the number of colluders they can resist.

Robustness Against Packet Loss

Time synchronization protocols are often built upon communication networks that provide only best-effort delivery of messages and cannot guarantee reliability. Many data origin authentication schemes, however, do not take the lossy nature of these networks into account as they require the underlying communication networks to provide reliability. Examples for such network reliability include anything from requiring in-sequence ordering of messages to requiring no packets be lost at all. It is important that data origin authentication schemes still provide authenticity of received messages in case some other messages were lost. If an authentication scheme fails to verify the authenticity of a valid message, because some messages were lost, the authentication scheme is said to be degrading [29].

Independence of Time Synchronization

Some schemes require clients to be time synchronized with the server. This is a very critical requirement, since an authentication scheme depending on time synchronization leads to a circular dependency in the context of time synchronization protocols. Such circular dependency introduces an additional attack vector because the security of the authentication scheme breaks when the underlying assumption of time synchronized server and clients is violated. However, the dependency between authentication scheme and time cannot be avoided completely since the lifetime of cryptographic keys must be enforced, for example. Nevertheless, this dependency introduces a security risk, and, therefore, should be as loose as possible (as in the case of NTP 34 or 68 years [3]).

Summary

Table III summarizes the requirements and evaluation criteria. Some properties can supposedly be fulfilled by every authentication scheme - others can not. In our evaluation of data origin authentication schemes, we focus on the requirements and criteria that cannot be fulfilled by every authentication scheme.

Table III
LIST OF REQUIREMENTS AND EVALUATION CRITERIA

Requirements and evaluation criteria	Part of the evaluation
Authentication of server	No
Message integrity	No
Protection from DoS attacks (against the time protocol)	No
Message replay protection	No
Computational efficiency	Yes
Low communication overhead	Yes
Immediate Authentication	Yes
Collusion resistance	Yes
Robustness against packet loss	Yes
Independence of time synchronization	Yes

VI. EVALUATION

In this section, we evaluate data origin authentication schemes for their suitability to secure broadcast time synchronization messages and assess to which degree existing authentication schemes fulfill the requirements identified in Section V. We assume that every scheme can provide similar end-to-end authenticity, message integrity, protection from DoS attacks against the time synchronization protocol, and message replay protection. For this reason, we evaluate the schemes according to their (1) computational efficiency, (2) communication overhead, (3) capability to sign and authenticate messages immediately, (4) resistance against collusion of clients, (5) robustness against packet loss, and (6) independency of time synchronization.

All data origin authentication schemes for broadcast communication comprise a trade-off between the performance and the security they provide. Some schemes, for example, aim to tolerate packet loss by creating redundancy in the authentication information, which generates additional communication overhead. Other schemes provide information-theoretic security against adversaries with potentially unlimited computational resources (and time) by trading-off resistance against collusion of clients. It is generally accepted that no ideal scheme exists that provides high-performance and high-security at the same time. Instead, each scheme aims to make the best trade-off from a specific point of view [20].

Due to the sheer number of authentication schemes that have been proposed for broadcast communication in the last twenty-five years, we use the classification of broadcast authentication schemes by Challal, Bettahar, and Bouabdallah [20] as our basis. They identified six distinct classes, which we reuse for our evaluation: deferred signing[2], signature propagation, signature dispersal, secret-information asymmetry, time-based asymmetry, and hybrid asymmetry. For each class, we give a brief description and evaluate one or two schemes we find representative or most suitable to secure broadcast time synchronization messages.

A. Deferred Signing

With deferred signing, the signing process is split into two parts: (1) a slow offline and (2) a fast online part. The offline part is independent of the actual message to be signed. It is used to pre-compute one-time keys, which are used in the online part for signing the actual messages. Since one-time signature schemes are computationally very efficient, messages can be signed and verified very fast.

[2]Challal, Bettahar, and Bouabdallah originally used the term "differed signing" but we think that they actually meant "deferred signing" as it makes more sense in this context.

1) Offline/Online Signing: Even, Goldreich, and Micali's offline/online signing scheme [30] is resistant to collusion of clients, tolerates packet loss, is independent of time synchronization, and authenticates messages immediately. Each one-time public-key is signed with a conventional signature to certify that the one-time public keys originate from the claimed server, Since one-time public keys can be signed offline, it does not negatively impact online message signing performance. During the online phase, however, clients have to certify the signature of the one-time public key from a conventional signature scheme. For this reason, a client has to verify a signature from a conventional signature scheme in addition to the one-time signature on the message. The performance of one-time signatures is, therefore, impaired by the use of conventional signatures, which is a serious performance drawback. Furthermore, the size of one-time signatures is very large, which implies high communication overhead.

2) Offline/Online k-time Signing: The offline/online signing has been improved by various schemes. The most notable is offline/online k-time signing [31]. The difference to the offline/online signing scheme is that k-time keys are created during the offline phase instead of one-time keys. By signing and verifying k-time keys (instead of one-time keys), the use of a conventional signature scheme can be amortized over multiple messages. Since verification is the most expensive (online) operation, computational efficiency is improved. Packet loss resistance suffers from this approach, however, because if the message containing the k-time keys' signature is dropped, none of the k messages can be verified. If that signature message is sent more often, communication overhead increases, and the problem is mitigated (but not solved).

Since one disadvantage of the offline/online k-time signature scheme is its communication overhead (around 1 kB per message), the author provided a method to reduce the communication overhead. This improvement in communication overhead has a negative impact on the computational efficiency, however. Furthermore, the offline/online k-time signing scheme requires frequent key generation and distribution throughout the communication, since the keys can be used to sign only a fixed number (k) of messages. For this purpose, a secure channel from the server to the clients is required, in order to transmit the new keys periodically.

B. Signature Propagation

One approach to reduce the cost of conventional signatures is to amortize it across multiple packets. With signature propagation schemes, a digital signature from a conventional signature scheme is appended to just one packet, the signature packet. Instead of signing each packet individually, hashes of non-signature packets are included in the preceding packet before they are sent. In this way, a chain of packets is built where each packet carries a hash of the subsequent packet. Only the first packet in the chain (containing the hash of the second packet) is signed. In this way, the digital signature propagates through the packets and its computational cost is amortized, since hash operations are computationally inexpensive. However, signature propagation schemes require some packets to be buffered at the sender or at the client before they can be authenticated and their signature be verified. Furthermore, they rely on the successful reception of signature packets.

1) Simple Online Chaining: In the simple online chaining scheme [32, 33], only one signature from a conventional signature scheme is transmitted in the first message. Each message contains an association (a hash or a one-time key) with the subsequent message, which can be used to verify the authenticity of the subsequent message. The simple online chaining scheme is computationally very efficient, since signing and verification consist of very fast hash operations only (apart from the first message that is signed using a conventional signature scheme). The scheme has low communication overhead, is resistant to collusion of clients, and independent of time synchronization. The server has to buffer one message, however, which introduces additional delay. Furthermore, if a single message is lost, the authentication chain is broken and cannot be recovered.

2) RLH: Many other schemes have been proposed in the signature propagation class. In this class, we find one scheme to perform better with regard to securing broadcast time synchronization messages: Receiver driven Layered Hash-chaining (RLH) [34]. RLH adds several packets to the communication. These packets contain hashes of other packets to increase resistance to packet loss while minimizing communication overhead and maximizing authentication probability. RLH uses two constructions for determining the position of the added packets containing the hashes: mix deterministic and random hash distribution. While packet loss resistance is improved compared to the simple online chaining approach, RLH is not resistant against attacks by an adversary that intentionally drops packets containing the hashes. Furthermore, computational efficiency and communication overhead are slightly worse, and clients have to buffer messages before authentication.

C. Signature Dispersal

With signature dispersal schemes, messages are divided into blocks of several packets. Each of the blocks is then signed independently with a digital signature scheme. The signature of a block is split into small parts and each part is appended to one packet within the block. Additional information is appended to each packet that helps clients to reconstruct the signature even if some packets were lost. In this way, signature dispersal schemes overcome the limitation of signature propagation schemes that rely on the reception of signature packets. The aim of signature dispersal schemes is to increase the tolerance to packet loss by providing means to recover authentication information in case packets were lost.

1) SAIDA: SAIDA (Signature Amortization using Information Dispersal Algorithm) [35, 36] balances packet loss resistance and communication overhead. Due to the use of the Information Dispersal Algorithm (IDA) [37] it is computationally rather inefficient [18]. It is resistant to collusion attacks

of clients and independent of time synchronization. However, it requires both server and clients to buffer messages before signing and verification, respectively. Furthermore, the scheme is only resistant to packet loss for a specific, pre-defined packet loss ratio.

2) Tartary et. al: Tartary, Wang, and Ling [38] propose to use list recoverable codes, which allow injected packets to be efficiently detected. Furthermore, the authors show that various other schemes from the signature dispersal class can be treated as particular cases of their general approach.

The scheme expects the fraction of received packets to be above a pre-defined threshold, and the fraction of injected packets below some other pre-defined threshold. As long as these assumptions hold, the scheme can recover lost authentication information entirely. The scheme is independent of time synchronization and resistant to collusion of clients. The communication overhead and computational efficiency are acceptable, but packets need to be buffered before they can be signed and before they can be authenticated.

D. Secret-Asymmetry

With secret-asymmetry schemes, the server shares a set of keys with the clients instead of a single key. The server knows the entire set of keys and can generate valid authentication information, but the partial view of each client allows only to verify but not to generate authentication information. Clients, therefore, cannot forge valid messages on behalf of the server.

The class of secret-asymmetry schemes is the only class that includes information-theoretically secure schemes that protect against adversaries with potentially unlimited computational power and time. They provide strong authentication guarantees but require substantial computational resources for signing and for verification. Requiring that many resources as well as the need to distribute new keys often make information-theoretically secure schemes impractical for securing broadcast time synchronization messages. Secret-asymmetry schemes, furthermore, are prone to collusion of clients, where fraudulent clients collaborate in order to reconstruct the entire set of the server's keys.

1) k-MAC: The basic idea of the k-MAC approach [39] is that the server knows several secret keys, and a subset of these keys are shared with each client to maintain some properties of the subsets held by the clients. One property is that no subgroup of n clients should know the set of keys known by any other client.

The server can authenticate a message by computing MACs using its full set of keys and appending all MACs to the message. Each client can verify the parts of the MACs for which it knows the keys. If all these parts are valid, the client accepts the message as genuine. Note that one client cannot forge a MAC, because it does not know all the keys of the server. But if more than n clients collude, the security of the scheme could break down. Furthermore, if there are enough colluders to know all the servers' keys, the security of the scheme breaks completely.

The k-MAC scheme is independent of time synchronization and provides immediate authentication. Furthermore, it is resistant to packet loss, since every packet is individually verifiable. However, k-MAC is prone to collusion attacks of clients, and its communication overhead and computational requirements increase with the number of clients, which effectively negates the benefits of broadcast communication.

E. Time-based Asymmetry

In time-based asymmetry schemes, server and clients are assumed to be time synchronized. As outlined in Section V, data origin authentication schemes cannot provide protection against every kind of attack: message delay attacks can be conducted even if an authentication scheme is in place. Assuming time synchronized server and clients is very dangerous for securing time synchronization messages because it includes circular reasoning and creates a dependency on time [5]; in case the assumption does not hold, the security of the authentication scheme breaks entirely.

1) BiBa: In BiBa [40], the server uses a set of keys for authentication; when signing a message, the server releases a subset of these keys as signature. After a specific number of signatures, clients would know the whole set of keys and, therefore, be able to forge signatures on behalf of the server. To prevent that, the server switches to a different set of keys periodically. The different sets of keys are associated using a one-way chain so that clients can recover when some messages were lost. However, at some point the last subset of keys is used, and new keys need to be generated and distributed (over a secure channel). Furthermore, both the time to generate keys and BiBa's communication overhead are quite high. Nevertheless, the scheme is resistant against collusion of clients and provides immediate authentication of messages.

2) TESLA: The NTS drafts [2, 14, 15] propose to employ the TESLA scheme [41, 42, 17] to address the broadcast authentication problem. Since TESLA is the currently favored approach by the IETF NTP group and also highly rated by the P1588 Security Subcommittee for securing the next version of PTP, we evaluate it in greater detail.

In TESLA, the private and public keys are identical, only separated by time. The server keeps a key secret for some time; i.e., the disclosure of the key is delayed. While the key is secret, the server uses it to sign messages. Clients buffer messages and can verify their authenticity after the key has been disclosed. Once the key is disclosed, the server switches to a new secret key in order to sign new messages. A common notion of time guarantees that a key is known by clients only after it is not used anymore by the server to sign messages. For this reason, time synchronization between server and clients becomes a security requirement so that clients know when to consider a key as valid. The time synchronization between server and clients is required not only initially but for the whole duration of the communication. Like in BiBa, keys are associated using a one-way chain in TESLA. However, at some point the last set of keys is used, and new keys need to be generated and distributed (over a secure channel).

TESLA meets the performance requirements perfectly, since it has minimal communication overhead of just one MAC per packet. Furthermore, it requires very few computational resources, tolerates packet loss, and resists collusion of clients. Still, we find TESLA unsuitable to secure broadcast time synchronization messages because it is vulnerable to message delay attacks in the context of time synchronization. Since key disclosure is delayed in TESLA, time needs to be partitioned into pre-defined intervals. During such interval, the server sends signed messages that the clients buffer. At the end of an interval, the server discloses the key so that the clients can verify the authenticity of the buffered messages. Verifying the authenticity of a message consists not only in verifying the MAC but also in checking whether the key belongs to the correct time interval. Otherwise, it would be trivially possible for an adversary to drop the original message and wait for the server to disclose the key in order to forge messages on behalf of the server.

In the context of time synchronization, however, TESLA is vulnerable to message delay attacks since the content of the message influences the client's notion of time. In this way, one of TESLA's underlying assumptions - time synchronized server and clients - may be violated. If the adversary delays the server's message only by a short time, the client still accepts it as timely. The client will then calculate the difference from its local time to the time value in the message and set its time backward to match the difference. This, on the other hand, slightly changes the client's perception of the time intervals during which a signing key is valid. If the adversary continues to delay messages in this way, then small-scale time changes accumulate into a large-scale time inaccuracy in the long term [13]. Eventually, the desynchronization between server and client will be large enough so that the adversary knows the disclosed key while the client accepts messages signed with that key as timely – effectively breaking the whole authentication scheme. This message delay attack on TESLA in the context of time synchronization is shown in Figure 1. In the first interval, the adversary delays the time-synchronization message by a short time; the adversary delays also the key disclosure message so that the client sets its clock backwards. For this reason, the client's perception of the interval is slightly shifted. This difference in the client's perception increases as the adversary continues to delay messages maliciously (Interval 2). The small-scale delays accumulate into a large-scale time shift eventually (Interval n) so that the adversary gets access to the key before the end of the interval. For this reason, the adversary can then sign arbitrary messages that the client will accept as valid.

To mitigate this delay attack on TESLA, NTS proposes two distinct strategies: (1) an additional check [2] extending the TESLA protocol, and, (2) leaving the client vulnerable [14]. We argue that both are failure criteria: (1) the goal of the additional check ("keycheck") is to guarantee *"to the client that the key belonging to the respective TESLA interval communicated in the exchange had not been disclosed before"* [2]. To verify that the broadcast message has not been delayed by some malicious entity, the client is expected to establish a unicast connection to the server after each broadcast message. But establishing a unicast connection after the reception of each broadcast message defeats the purpose of using broadcast communication in the first place. (2) Leaving the client vulnerable to delay attacks by proposing the additional check to be optional, as suggested in [14], effectively negates the whole authentication procedure.

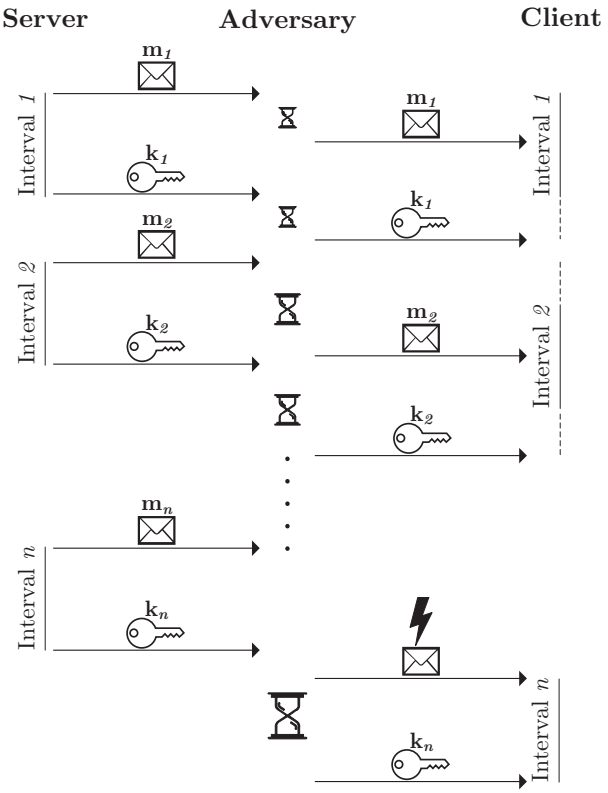

Figure 1. Message delay attack on TESLA as part of time synchronization.

F. Hybrid Asymmetry

Secret-information asymmetry schemes provide immediate authentication but are prone to collusion attacks of clients. Time-based asymmetry schemes, on the other hand, are immune to collusion attacks but cannot provide immediate authentication. In hybrid asymmetry, the aim is to combine the strengths of secret-information asymmetry and time-based asymmetry approaches, while mitigating their limitations [20]. Like in time-based asymmetry schemes, the keys used in hybrid asymmetry schemes can only sign a fixed number of messages. Once this limit is reached, a new key must be generated and distributed in order to sign more messages securely.

1) TV-HORS: Among the many schemes proposed in the hybrid asymmetry class, we find one to be more suitable than any other for securing broadcast time synchronization messages: Time Valid Hash to Obtain Random Subsets (TV-HORS) [43]. TV-HORS is a broadcast authentication scheme based on the HORS [44] one-time signature scheme.

Table IV
SUMMARY: EVALUATION OF DATA ORIGIN AUTHENTICATION SCHEMES

Class	Scheme	Computational efficiency	Low communication overhead	Immediate authentication	Collusion resistance	Resistance against packet loss	Independence of time synchronization
Deferred Signing	Offline/online signing	--	--	+	+	+	+
	Offline/online k-time signing	~	~	+	+	-	+
Signature Propagation	Simple online chaining	++	+	--	+	-	+
	RLH	~	~	~	+	~	+
Signature Dispersal	SAIDA	-	~	--	+	~	+
	Tartary et. al	~	~	--	+	~	+
Secret-Asymmetry	k-MAC	-	-	+	--	+	+
Time-Asymmetry	BiBa	~	~	+	+	+	--
	TESLA	++	+	~	+	+	--
Hybrid Asymmetry	TV-HORS	++	~	+	+	+	~

Evaluation criteria is either: strongly satisfied (++), satisfied (+), somewhat satisfied (~), unsatisfied (-), or strongly unsatisfied (--).

First, the HORS one-time signature scheme is extended to a k-time signature scheme. Then, multiple k-time key pairs are associated using a one-way hash chain. In this way, numerous messages can be signed very efficiently. However, signature size and, therefore, communication overhead is quite large because of the use of the one-time signature scheme. To reduce the signature size, only a part of the message's hash is signed. Of course, this reduces security in general since an adversary needs to find only a partial hash collision. Since TV-HORS assumes that server and clients are loosely time synchronized, the time a signature is exposed to a potential adversary can be controlled. By estimating the adversary's computational power, it can be assured that no partial collision can be found (during the time span a key is valid). Since TV-HORS involves a k-time signature scheme, the frequent key updates require a secure out-of-band channel between server and each client throughout the whole communication.

TV-HORS provides fast message signing and verification (roughly 8k messages per second [43]). It supports immediate authentication of messages, is resistant to collusion of clients, and tolerates packet loss. However, TV-HORS does assume the client and the server to be time synchronized. The accuracy of time synchronization between clients and server can be configured; if looser time synchronization is assumed, communication overhead increases. For this reason, TV-HORS comprises a trade-off between communication overhead and the accuracy of time synchronization it assumes. Nevertheless, in contrast to TESLA, the dependency on time synchronization between server and client can be configured in TV-HORS so that it is less sensitive to delay attacks.

G. Summary

Table IV summarizes the theoretical evaluation of existing classes and schemes with regard to the evaluation criteria proposed in Section V. Our evaluation shows that most schemes are unsuitable for securing broadcast time synchronization messages. Nevertheless, two schemes from two distinct classes come somewhat close to meeting the evaluation criteria: RLH and TV-HORS. For RLH, computational efficiency should be improved so that the precision of time synchronization is less degraded. TV-HORS, on the one hand, just needs to be configured appropriately so that it tolerates very loose time synchronization but, on the other hand, constantly requires a secure channel between server and clients for key updates.

VII. CONCLUSION

In this paper, we looked at the challenge of data origin authentication for broadcast time synchronization. A security measure integrated into time synchronization protocols is required in order to achieve high precision time synchronization. First, we identified the security and performance requirements of time synchronization protocols in the context of broadcast communication. We then conducted a thorough evaluation of existing data origin authentication schemes against these requirements and criteria. The main outcome of our evaluation is that most data origin authentication schemes are unsuitable to secure broadcast time synchronization. Even worse, the TESLA scheme, currently proposed in the IETF to secure broadcast messages in NTP and highly rated by the P1588 Security Subcommittee for securing PTP, is susceptible to message delay attacks in the context of time synchronization. This precludes TESLA from providing the required security

features. Of all evaluated schemes, two classes entail data origin authentication schemes that are suitable to secure broadcast time synchronization: the signature propagation class with the RLH scheme and the hybrid asymmetry class with TV-HORS. Both schemes may well be promising candidates to secure broadcast time synchronization, especially when their weak spots are mitigated in the future.

We are confident that our evaluation of data origin authentication schemes in the context of broadcast time synchronization provides valuable insights in order to motivate future work on making broadcast time synchronization more secure.

REFERENCES

[1] David L Mills. *Computer Network Time Synchronization: the Network Time Protocol on Earth and in Space*. 2nd ed. CRC Press, 2011. ISBN: 978-1-4398-1463-5.

[2] Dieter Sibold, Stephen Roettger, and Kristof Teichel. *Network Time Security*. Internet-Draft draft-ietf-ntp-network-time-security-15. https://tools.ietf.org/html/draft-ietf-ntp-network-time-security-15. IETF Secretariat, Sept. 2016. (Visited on 03/01/2017).

[3] D. Mills et al. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905 (Proposed Standard). Internet Engineering Task Force, June 2010. URL: http://www.ietf.org/rfc/rfc5905.txt.

[4] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". In: *IEEE Std 1588-2008* (July 2008), pp. 1–269. DOI: 10.1109/IEEESTD.2008.4579760.

[5] Aanchal Malhotra and Sharon Goldberg. "Attacking NTP's Authenticated Broadcast Mode". In: *ACM SIGCOMM Computer Communication Review* 46.1 (2016), pp. 12–17.

[6] Jonathan Katz. *Digital Signatures*. Boston, MA: Springer US, 2010. ISBN: 978-0-387-27711-0 978-0-387-27712-7.

[7] Albert Treytl, Bernd Hirschler, and Thilo Sauter. "Secure tunneling of high-precision clock synchronization protocols and other time-stamped data". In: *8th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, 2010, pp. 303–312.

[8] A. Treytl and B. Hirschler. "Security flaws and workarounds for IEEE 1588 (transparent) clocks". In: International IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS). IEEE, Oct. 2009, pp. 1–6. DOI: 10.1109/ISPCS.2009.5340204.

[9] N. Moreira et al. "Security mechanisms to protect IEEE 1588 synchronization: State of the art and trends". In: International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS). IEEE, Oct. 2015, pp. 115–120. DOI: 10.1109/ISPCS.2015.7324694.

[10] B. Haberman and D. Mills. *Network Time Protocol Version 4: Autokey Specification*. RFC 5906 (Informational). Internet Engineering Task Force, June 2010. URL: http://www.ietf.org/rfc/rfc5906.txt.

[11] Stephen Röttger. "Analysis of the NTP Autokey Procedures". Master's thesis, Technische Universitt Braunschweig, 2012.

[12] Harlan Stenn. *[ntpwg] Antw: Re: Proposed REFID changes*. http://lists.ntp.org/pipermail/ntpwg/2015-July/002291.html. [Online; accessed 01-March-2017]. 2015.

[13] Benjamin Dowling, Douglas Stebila, and Greg Zaverucha. "Authenticated Network Time Synchronization". In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 823–840. ISBN: 978-1-931971-32-4.

[14] Dieter Sibold, Stephen Roettger, and Kristof Teichel. *Using the Network Time Security Specification to Secure the Network Time Protocol*. Internet-Draft draft-ietf-ntp-using-nts-for-ntp-07. https://tools.ietf.org/html/draft-ietf-ntp-using-nts-for-ntp-07. IETF Secretariat, Oct. 2016. (Visited on 03/01/2017).

[15] Dieter Sibold et al. *Protecting Network Time Security Messages with the Cryptographic Message Syntax (CMS)*. Internet-Draft draft-ietf-ntp-cms-for-nts-message-06. https://tools.ietf.org/html/draft-ietf-ntp-cms-for-nts-message-06. IETF Secretariat, Feb. 2016. (Visited on 03/01/2017).

[16] Kristof Teichel, Dieter Sibold, and Stefan Milius. "First Results of a Formal Analysis of the Network Time Security Specification". In: *Security Standardisation Research*. Lecture Notes in Computer Science 9497. Springer International Publishing, Dec. 15, 2015, pp. 218–245. ISBN: 978-3-319-27151-4 978-3-319-27152-1.

[17] A. Perrig et al. *Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction*. RFC 4082 (Informational). Internet Engineering Task Force, June 2005. URL: http://www.ietf.org/rfc/rfc4082.txt.

[18] Tommaso Cucinotta, Gabriele Cecchetti, and Gianluca Ferraro. "Adopting redundancy techniques for multicast stream authentication". In: *Proceedings of the Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, FTDCS 2003*. IEEE, 2003, pp. 183–189.

[19] Stefaan Seys and Bart Preneel. "Power consumption evaluation of efficient digital signature schemes for low power devices". In: *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob 2005)*. Vol. 1. IEEE, 2005, pp. 79–86.

[20] Y. Challal, H. Bettahar, and A. Bouabdallah. "A taxonomy of multicast data origin authentication: Issues and solutions". In: *IEEE Communications Surveys Tutorials* 6.3 (2004), pp. 34–57. ISSN: 1553-877X. DOI: 10.1109/COMST.2004.5342292.

[21] Yee Wei Law et al. "Comparative Study of Multicast Authentication Schemes with Application to Wide-area Measurement System". In: *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*. ASIA CCS '13. New York, NY, USA: ACM, 2013, pp. 287–298. ISBN: 978-1-4503-1767-2. DOI: 10.1145/2484313.2484349.

[22] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[23] Danny Dolev and Andrew C Yao. "On the security of public key protocols". In: *IEEE Transactions on Information Theory* 29.2 (1983), pp. 198–208.

[24] T. Mizrahi. *Security Requirements of Time Protocols in Packet Switched Networks*. RFC 7384 (Informational). Internet Engineering Task Force, Oct. 2014. URL: http://www.ietf.org/rfc/rfc7384.txt.

[25] Josef Pieprzyk, Jennifer Seberry, and Thomas Hardjono. *Fundamentals of Computer Security*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2002. ISBN: 3540431012.

[26] Tal Mizrahi. "A game theoretic analysis of delay attacks against time synchronization protocols". In: *International IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*. IEEE, 2012, pp. 1–6.

[27] Saurabh Ganeriwal et al. "Secure time synchronization in sensor networks". In: *ACM Transactions on Information and System Security (TISSEC)* 11.4 (2008), p. 23.

[28] R. Shirey. *Internet Security Glossary, Version 2*. RFC 4949 (Informational). Internet Engineering Task Force, Aug. 2007. URL: http://www.ietf.org/rfc/rfc4949.txt.

[29] Christophe Maurice Andre Tartary. *Authentication for Multicast Communication*. Macquarie University, 2007.

[30] Shimon Even, Oded Goldreich, and Silvio Micali. "On-line/off-line digital signatures". In: *Journal of Cryptology* 9.1 (1996), pp. 35–67.

[31] Pankaj Rohatgi. "A Compact and Fast Hybrid Signature Scheme for Multicast Packet Authentication". In: *Proceedings of the 6th ACM Conference on Computer and Communications Security*. CCS '99. New York, USA: ACM, 1999, pp. 93–100. ISBN: 1-58113-148-8. DOI: 10.1145/319709.319722.

[32] Rosario Gennaro and Pankaj Rohatgi. "How to sign digital streams". In: vol. Advances in Cryptology. Springer, 1997, pp. 180–197.

[33] Rosario Gennaro and Pankaj Rohatgi. "How to Sign Digital Streams". In: *Information and Computation* 165.1 (Feb. 25, 2001), pp. 100–116. ISSN: 0890-5401. DOI: 10.1006/inco.2000.2916.

[34] Yacine Challal, Abdelmadjid Bouabdallah, and Yoann Hinard. "RLH: receiver driven layered hash-chaining for multicast data origin authentication". In: *Computer Communications* 28.7 (2005), pp. 726–740.

[35] Jung Min Park, Edwin KP Chong, and Howard Jay Siegel. "Efficient multicast packet authentication using signature amortization". In: *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 2002, pp. 227–240.

[36] Jung Min Park, Edwin KP Chong, and Howard Jay Siegel. "Efficient multicast stream authentication using erasure codes". In: *ACM Trans-*

actions on Information and System Security (TISSEC) 6.2 (2003), pp. 258–285.

[37] Michael O. Rabin. "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance". In: *J. ACM* 36.2 (Apr. 1989), pp. 335–348. ISSN: 0004-5411. DOI: 10.1145/62044.62050.

[38] C. Tartary, Huaxiong Wang, and San Ling. "Authentication of Digital Streams". In: *IEEE Transactions on Information Theory* 57.9 (Sept. 2011), pp. 6285–6303. ISSN: 0018-9448. DOI: 10.1109/TIT.2011.2161960.

[39] R. Canetti et al. "Multicast Security: A Taxonomy and Some Efficient Constructions". In: Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM '99. Vol. 2. Mar. 1999, pp. 708–716. DOI: 10.1109/INFCOM.1999.751457.

[40] Adrian Perrig. "The BiBa One-time Signature and Broadcast Authentication Protocol". In: *Proceedings of the 8th ACM Conference on Computer and Communications Security*. CCS '01. New York, USA: ACM, 2001, pp. 28–37. ISBN: 1-58113-385-5. DOI: 10.1145/501983.501988.

[41] Adrian Perrig et al. "Efficient authentication and signing of multicast streams over lossy channels". In: *Proceedings of the IEEE Symposium on Security and Privacy, S&P 2000*. IEEE, 2000, pp. 56–73.

[42] Adrian Perrig et al. "The TESLA broadcast authentication protocol". In: *RSA CryptoBytes* 5 (2002).

[43] Qiyan Wang et al. "Time Valid One-Time Signature for Time-Critical Multicast Data Authentication". In: *IEEE INFOCOM 2009*. Apr. 2009, pp. 1233–1241. DOI: 10.1109/INFCOM.2009.5062037.

[44] Leonid Reyzin and Natan Reyzin. "Better than BiBa: Short one-time signatures with fast signing and verifying". In: *Information Security and Privacy*. Springer, 2002, pp. 144–153.