Towards Understanding Pure Publish/Subscribe Cryptographic Protocols

DRAFT

Pekka Nikander[1,2], Giannis F. Marias[3]
[1] Ericsson Research Nomadic Lab, Hirsalantie 11, 02420 JORVAS, Finland
[2] Helsinki Institute for Information Technology, Metsänneidonkuja 4, 02015 TKK, Finland
[3] Dept. of Informatics, Athens University of Economics and Business, 10434 Athens, Greece

Pekka.Nikander@nomadiclab.com, marias@aueb.gr

Abstract

In this paper, we pursue towards understanding how to design and analyse cryptographic protocols in a (large) network setting where all communication is solely based on the publish/subscribe paradigm. That is, we expect a stack and network architecture where all message passing is based on publish/subscribe rather than send/receive, all the way down to the link layer. Under those assumptions, it looks like that the majority of present work on cryptographic protocol analysis applies to an extend, with only minor modifications mostly on the notation side, while the protocol design aspects will need larger modifications. Furthermore, the paradigm shift opens a number of interesting problems, requiring modifications to many of the traditional intuitions guiding protocol design and analysis.

1. Introduction

With the advent of serious proposals for moving from the present inter-networking of computers to inter-networking of information as the primary communication paradigm [Jacobson][Scott et al], the publish/subscribe model [Eugster et al] has been raised again as a potential primary communication method, replacing the present send/receive model.

In this paper, we take the first baby steps towards understanding how to design and analyse cryptographic protocols when the underlying network is solely based on the public subscribe paradigm. Interestingly, while many of the intuitions and traditional ways of designing protocols appear to need changes, the actual formalisms underlying the various analyse methods appear to be better equipped to encounter the new reality.

2. Towards a Pure Publish/Subscribe Paradigm

Consider a network built solely up using the publish/subscribe paradigm instead of the commonly used send/receive paradigm. In such a network, there are no receiver or sender names. Only messages have names. We expect the message name space to be very large and essentially random, making it virtually impossible to guess a message's name. Whenever receivers or senders need to be named, e.g., in order to be able to argue about authentication of principals, it must be understood that such names are given outside of the networked system. This also means that there is no primitive that would allow a message to be sent to a given principal. The only thing a sender can do is to publish the message and hope that the intended principal has subscribed or will subscribe to it.

As an example, in a pure publish/subscribe system the routing and forwarding tables could be based on the reachability of the subscribers. That is, at each intermediate node there would be a forwarding table that indicates the physical port or radio characteristics of the next hop(s) for each message. A publication tag in the message would be used to find the right entry in the table. For example, when a message arrives over a physical link to a forwarding network node, the forwarding node looks up the next physical link(s) where the message needs to be delivered and re-sends the message over that (those) link(s).

Each individual message can be considered as a primitive publication, having an unique name (an identifier). For a message to be meaningful, some node must publish it, and there there must be or more nodes that have indicated or will indicated their interest to receive it. Hence, when Alice and Bob agree that Alice will send a message to Bob, they must have an a priori agreement of the name of the message that Alice will send.

In reality, such a simple model does not scale; it is unrealistic to assume that all subscribers will explicitly subscribe for each message they want to receive. On the other hand, as a conceptual model, it has some very appealing properties. First, it requires that each principal explicitly expresses what messages they are interested about, with the assumption that the network will deliver them only messages matching with their interests. Second, it requires one to think more thoroughly some of the underlying assumptions, as one can no longer assume messages to be delivered by simply sending them. Together, these seem to map nicely to a number of existing protocol analyses tools.

2.1. Recursive composition

A key structure to enable pure publish/subscribe are recursively composed publications; without them the system would not scale. Basically, a composed publication is one that consists of a collection of more primitive publications, such as messages. For example, at the implementation level the forwarding table entries might contain such publication identifiers instead of identifiers of single messages, thereby allowing the routing and forwarding system to scale to more realistic dimensions [Särelä et al].

Let us now consider some recursive publication structures, starting from file transfer. (We are ignoring a number of very interesting but thorny issues related to reliability for the moment, assuming that message delivery is error free.) Typically, today's networking technology places an upper limit to the size of messages, requiring one to use multiple messages to send a file. Using the publish/subscribe paradigm, we can accomplish that by first publishing a separate "meta" message containing the names of those messages that form the segments of the file. Hence, if a subscriber wants to receive a file, it can first subscribe to the meta message, learn the identifiers of the file segments, subscribe to each of the segments, and reconstruct the file from the segments.

In a similar manner, we can imagine a media stream (e.g. a voice call) to consist of a number of tiny messages (segments), each of which the receiver conceptually subscribes to individually. However, opposed to the previous case, the number and content of the messages is not known a priori. Hence, it would make more sense to create the segment identifiers algorithmically instead of listing them in the meta message. Instead, the publisher of the media stream could publish a meta message containing an algorithm that allows the receivers to compute the subscription identifiers for the media segments that will appear in the stream at well defined points in the future.

As a third example, let us consider scalability at the network level a little bit deeper. Given that in our model each message is (conceptually) a separate publication, a global publish/subscribe network would have a huge number of publications. Hence, it would be unrealistic to assume that each intermediate node separately has a distinct forwarding table at the message granularity. To solve this problem, consider wrapping the message level publications into separate coarser granularity publications that each corresponds to a subscriber group. These group-granularity publications can further be clustered into even coarser-level publications, creating a partially redundant multicast forwarding table[1]. Using this structure, each forwarding node has a fairly simple forwarding table that contains only the identifiers and next-hop information for a relatively small number of these coarser-grain publications.

From a security protocols point of view, it is important to understand that such composed publications consist of multiple messages, separated by time. Hence, from a protocol analysis point of view it is an open question when exactly they should be considered as simple composed (concatenated) messages, when as separate steps in the protocol specification. Apparently, both cases will appear, e.g., depending on how the message names are used or how they are bound together.

2.2. Unifying forwarding and storage

Returning to the publish/subscribe paradigm, we can further enrich our total system by adding a (persistent) caching function to the publish/subscribe network and providing an API that unifies storage and forwarding.

---

[1] In practise, such recursive wrappings could be implemented with recursive, MPLS-like lables.

Let us return to our previous file transfer example. With caching, the network can now cache the file in addition to forwarding it to the present subscribers. Such caching allows the network to forward the file to any future subscribers more efficiently[2] by simply sending the packets from the closest cache instead of re-requesting them from the original publisher. If the cache is persistent (enough), even the original publisher can take advantage of this, as it can now discard the original file and request it on demand by subscribing to it.

2.3. Multicast and concast

While traditional two-party protocols can be easily implemented even in a publish/subscribe network, publish/subscribe as a paradigm makes multicast and concast (reverse multicast) natural modes of operation. In the multicast case, there is one principal who publishes new messages belonging to a known publication, which in turn may be subscribed by several receivers. Conversely, in the concast case, there is just one subscriber but there may be multiple principals that add messages to the publication. In that case, the network is expected to merge multiple messages before they reach the subscriber [Calvert et al].

From the cryptographic protocols point of view, in both cases we enter the domain of group communication. Furthermore, in the latter case, where the network acts actively on the messages, it may be necessary to include parts of the network to the analysis, moving even further from the traditional Dolev-Yao intruder model.

2.4. Main differences from the present model

The main differences of our publish/subscribe model from the present send/receive model can be summarised as follows:

- The messages are not directed to any explicitly named receiver; the expected receiver or receivers must be understood from the context.
- The primary mode of communication is one-way rather than two-way. Two-way communication requires at least two explicit channels.
- The senders and receivers have no network-provided names.
- All data is conceptually identifiable; each message has an (unique) identifier.
- The basic message-passing primitive is based on multicast rather than unicast.
- Network nodes are able to and are expected to cache messages.

[2] For brevity, we ignore some the very interesting problems related to cache consistency.

3. An initial formal model

Based on the ideas above, let us outline an initial formal model to present publish/subscribe communication. Herein we focus on the exchange of single messages, leaving issues related to message names and composition for later.

Adopting notation from [Alice&Bob], we will use upper-case letters $A, B, C, ...$ to denote principals, N and I to denote numbers, and M to denote an arbitrary message. All of these symbols may be annotated with subscripts or superscripts.

Messages are built inductively from atomic messages (identifiers and number symbols) by pairing, encryption, inversion, and hashing. For M, M_1, and M_2 messages, we write the pairing (concatenation) of M_1 and M_2 as $M_1; M_2$, the asymmetric encryption of M_1 by M_2 as $\{|M_1|\}^a_{M_2}$, the symmetric encryption of M_1 by M_2 as $\{|M_1|\}^s_{M_2}$, the asymmetric inversion of M by M^{-1}, and the application of a hash function H to M as $H(M)$. As conventional notation for keys, we will write K_A to denote a public key of the principal A, with the corresponding private key K_A^{-1}, and we will write K_{AB} to denote a symmetric key that is shared by the principals A and B.

Given the above, a protocol specification consists of a finite sequence of message-publication steps, each of the form $(p_q) A_p \rightarrow: (N_1, ..., N_t).M$, where (p_q) is a name of the protocol step, A_p is a distinct symbol referring to the principal that publishes the message M, and $N_1, ..., N_t$ are distinct number symbols (nonces). A step indicates that the principal A_p generates fresh random numbers (nonces) $N_1, ..., N_t$ and then publishes the message M. Note that this notation does not indicate who is or are the intended recipient(s). This is to reflect the different nature of publish/subscribe as opposed to send/receive.

Moving forward from a specification to the actions of the principals, we can define three basic actions, roughly corresponding to those in [Alice&Bob]:

- $p(M)$ — publication of the message M,
- $r(M)$ — receiving the message M, and
- $f(N)$ — generating the fresh number N.

These actions reflect the fact that the underlying network is both publish/subscribe in nature and may also be hostile. A publication action does not name the intended recipients; in some cases the set of recipients may not even be known by the publisher. The receiving actions do not name the message's publisher either, it may not be known or the message may originate from some other than the expected principal. For example, one can assume that the network is controlled by a Dolev-Yao intruder [DolevYao] who can compose, send, and intercept messages at will, but cannot break cryptography.

Using these notations, we can now express the principals' action sequences using a direct interpretation (cf. [Alice&Bob]) by simply composing the permissible runs (or traces) for the principals.

Example. Consider the Otway-Rees Authentication/Key-Exchange Protocol (OR) [Otway-Rees], adopting from Example 9 in [Alice&Bob]:

$$\begin{array}{rll}
(or_1)A \rightarrow: & (N_1). & I; A; B; \{|N_1; I; A; B|\}^s_{K_{AS}} \\
(or_2)B \rightarrow: & (N_2). & I; A; B; \{|N_1; I; A; B|\}^s_{K_{AS}}; \{|N_2; I; A; B|\}^s_{K_{BS}} \\
(or_3)S \rightarrow: & (K). & I; \{|N_1; K|\}^s_{K_{AS}}; \{|N_2; K|\}^s_{K_{BS}} \\
(or_4)B \rightarrow: & & I; \{|N_1; K|\}^s_{K_{AS}}
\end{array}$$

With this, realising that B passes some messages verbatim, we can describe the permissible actions by B as follows:

$$\begin{array}{l}
r(I; A; B; \gamma_1). \\
f(N_2). \\
p(I; A; B; \gamma_1; \{|N_2; I; A; B|\}^s_{K_{BS}}). \\
r(I; \gamma_2; \{|N_2; K|\}^s_{K_{BS}}). \\
p(I; \gamma_2)
\end{array}$$

where $\gamma_1 = \{|N_1; I; A; B|\}^s_{K_{AS}}$ and $\gamma_2 = \{|N_1; K|\}^s_{K_{AS}}$ represent submessages that B cannot interpret due to not possessing the key K_{AS}.

Of course, this is essentially the same as in the formalisms based on send/receive, as the protocol nor the semantics have changed. However, some of the intuitions may have changed. Furthermore, as we will note further down, the way protocols are designed may need more fundamental changes.

In the light of the early formalism and the example above, the basic elements of traditional cryptographic protocol analysis appear to be essentially the same in publish/subscribe and the more conventional send/receive worlds. The only difference is that the sender may not know, at some level, the identity of the indented recipient. However, as most "standard" cryptographic protocols do expect that the sender simply must know some (cryptographic) identifier for the recipient (cf. e.g. [Syverson & Carversato], such an "insight" does not lead as far.

Hence, we have to look at other intended purposes (beyond simple authentication) that a cryptographic protocol may have. For example, instead of knowing the identity of the communication peer, it may be enough to know that there is only one peer (e.g. a group of fully synchronised nodes) that remains the same throughout some session. More generally, it may be necessary to look at the intention more from the application point of view, and try to understand the economic mechanism, contract, or other purpose which the protocol has been build for.

Some of the properties from more traditional protocols still apply, such as gaining ex post assurance of the identity of those that have access to a key (ex post identity management), making sure that the holder of a particular key is currently reachable (freshness), etc. (cf. also e.g. [Syverson & Carvesato]).

Another aspect that we haven't yet tapped to are message identifiers. As we explained above, in a pure publish/subscribe network all messages are supposed to have a distinct name. If these names are cryptographically meaningful, they per se create a set of implicit protocols, needing analysis.

4. Towards a Problem Statement

Given the pub/sub communication model and its constraints, we tentatively can make the following observations.

- While the traditional Alice & Bob like protocols with the Dolev-Yao intruder model still pertain, they form only a small subset of the interesting problems. Furthermore, the existing models may need to be extended and enriched by the facts that all communication in the pub/sub network is naturally multicast and that two-way communication requires explicit establishment of a return channel.
- Moving focus from authenticating principals to various security properties related to the data itself may require completely new methods.
- A number of interesting problems are apparently related to the group communication aspects of publish/subscribe.
- Another set of open problems can be found from within the infrastructure. Apparently, a number of new publish/subscribe based protocols are needed. Open problems in designing such protocols include resource control, such as issues related to fairness, compensation, and authorisation.

Given this all, it becomes necessary to reconsider what we mean with authentication goals and assumptions. As the network provides no names for the active entities (nodes), the next generation applications are likely to be more interested in the ability to receive correct and properly protected information rather than communicating with predetermined nodes.

The threats and security goals can be divided, in a perhaps more standard fashion, as follows:

- Secrecy of security-related entity identities and identity protection.
- Secrecy of keys and other related information, typically needed for confidentiality and data integrity of the transmitted information.
- Denial of service.
- Threats to fairness, including mechanisms such as compensation and authorisation.
- Authenticity of the information, including its integrity and trustworthiness, reputation of the origin, and evidence of past behaviour, if available.
- Privacy and integrity of subscriptions to information.
- Privacy and integrity of the forwarding state (as a result of subscriptions).

At the mechanism level, there must be in place mechanisms to enable communication through potentially malicious networks and nodes, as well as to establish mutual trust between different administrative domains. This may require new kinds of cryptographic protocols that draw insight from micro-economics, e.g. algorithmic mechanism design, and have explicit structures for handling compensation, authorisation, and reputation instead of relying solely on more traditional and lower-level identity authentication and key distribution.

5. Design and modelling of cryptographic protocols

The majority of work in the area of cryptographic protocol design and modelling has been based on the two-party communication model, with a Dolev-Yao [Dolev Yao] intruder. As discussed above, such a model appears insufficient for pure publish/subscribe networks, where the network provides no identity (other than the implicit identity provided by the location-related forwarding information) for the active parties. Furthermore, the set of interesting security problems goes beyond the standard end-to-end examples, such as authentication, key distribution, and secure file transfer; in addition to those, we need to consider group communication, denial of service, security goals related directly to data or database transactions, and the overall security of the network infrastructure itself.

In this section, we briefly look at existing work, trying to figure out possible ways to enhance them to cover some of the new challenges.

5.1. Adversary model

The standard attacker model in cryptographic protocol design and analysis is that of Dolev and Yao [Dolev-Yao], often enriched with the correspondence assertions by Woo and Lam [Woo-Lam]. The Dolev-Yao model assumes two honest parties that are able to exchange messages through a powerful adversary that is able to intercept, eavesdrop, and inject arbitrary messages. Given that in our model primary communication is expected to be one way data transfer rather than two way transactions, requires two distinct channels for two way communication, and that in a more realistic model the attackers are typically able to compromise only part of the infrastructure (a byzantine model) instead of having complete control over it, a richer attacker model is needed.

Given the primarily multicast nature of the publish/subscribe paradigm, some insights may be attainable from the work on group protocols [References needed]. It may even turn out that discrete attacker models are not sufficient, but that instead one has to turn attention to probabilistic or micro-economic models, such as Meadows' model for analysing resource-exhausting denial of service [Meadows] or Buttyán and Hubaux micro-economics flavoured models [Buttyan Hubaux].

5.2. Modelling logic and beliefs

To our knowledge, the vast majority if not all the work on logic-based modelling and verification of cryptographic protocols is inspired by the Alice & Bob two-party setting (see e.g. [Caleiro et al, Syverson et al]), sometimes enriched with a Server. Considering the publish/subscribe paradigm, this does not appear very useful. In the case of a single publication (channel), the publisher basically knows nothing, or, rather, does not gain any new knowledge when publishing. The subscribers, on the other hand, may learn new knowledge from the message contents. However, some properties, like freshness, appear impossible to implement without either two-way communication or additional, external data (such as roughly-synchronised clocks).

Digging slightly deeper, it becomes evident that also in the publish/subscribe world there will necessarily be two-party or multi-party protocols. Using our basic model, the initial messages will contain information that allows the receivers to subscribe to some messages expected to be published in the future, or publish messages in a way where they can expect there to be a subscriber. Hence, already here we have some basic beliefs:

> Alice believes that there is a party ("Bob") that is subscribed to a message named M and will do some well-specified action X once it receives a valid M.

As this belief expresses expectations about the allowed future states of the system, an open question is whether adding temporal modalities some of the existing modal-logic based approaches would be sufficient.

5.3. Spi calculus

Process algebras, such as Spi calculus [Abadi & Gordon], and especially Pattern-matching Spi-calculus [Haack & Jeffrey] seem to be readily capable of modelling our basic model, including multicast communication and explicitly named messages. However, in order to derive useful and interesting results, one may want to consider various richer description for the net. That is, instead of assuming a Dolev-Yao type all-capable intruder, one may want to model an intruder that is capable to subscribe to (eavesdrop) any messages and message sequences (publications) that it knows about, but has limited capabilities of eavesdropping messages whose names they do not know or publishing messages on message sequences that they do not know about.

5.4. Strand spaces

Like Spi calculus, strand spaces [Thayer et al] appear capable for basic modelling. For example, multicast is naturally modelled, requiring no extensions. However, as in the case of Spi calculus, an open question is how to model the network and the penetrator in order to derive interesting results. One approach might be to continue using the basic penetrator model, but add new strands that model the publish/subscribe nature of the network in between.

5.5. Information-theoretic models

At the time of this writing, it is an completely open problem how the more information theoretic models, such as the one underlying Huima's tools [Huima] or developments thereof (e.g. [Millen & Shamatikov]), could be applied to publish/subscribe.

6. Discussion

The publish/subscribe paradigm being completely different from the prevailing send/ receive paradigm, with the starting point of naming information instead of principals of actors, we expect that much of our intuition on how to efficiently build security protocols will fail. As a reference point, consider the intuitions behind circuit-oriented and packet-oriented communication and the affect of those intuitions to the definition of authentication, as discussed by Dieter Gollman [Gollman]. Basically, there are subtleties in how, exactly, to define "authentication" depending on the intuitive model of the network between Alice and Bob, i.e., whether the network is expected to be a circuit with Mallory sitting there in between or the network delivers packets that may (sometimes) be intercepted by Mallory. That exemplifies the problem we have with our intuitions; we expect the difficulties to be much larger when moving from send/receive to publish/subscribe. That is, the problem appears to go deeper when the entities (principals) are no longer the primary objects of communication. Most intuitions based on thinking about the principals in terms of humans fail. In addition to the necessity of focusing more on other security goals but authentication, even the very concept of principal identity may need to be reconsidered, perhaps leading to a new definitions for authentication.

A large open security problem in the proposed architecture appears to be the composition of more complex publications from simple ones. This composition process needs to be both secure and efficient, and may require multiple different methods depending on the nature of the more complex publication. For example, for link-local administrative streams or sensor-type even streams a protocol resembling TESLA might be a suitable one. For wider-range communication and higher data rates public key cryptography combined with more traditional session keys may be more efficient. Consequently, the structure of the message and other data identifiers clearly plays a crucial security role. An ability to use self-certifying identifiers, such as ones based on hashing, hash chains, or hashes of public keys, may enable secure bootstrapping of the system without too many assumptions about external infrastructure. One potential approach here might be to tag message is explicitly with a public key [Järvinen et al].

Finally, the separation between the applications and networks needs to be clarified, along with the technology challenges that are new, such as privacy of subscriptions. The resulting network properties can then lead to a deeper the technology review.

References

1. Abadi M and Gordon AD. A Calculus for Cryptographic Protocols — The Spi Calculus. Research report (1998) SRC 149 p. 110,
2. Buttyán L and Hubaux JP. A Formal Analysis of Syverson's Rational Exchange Protocol. IEEE Computer Security Foundations Workshop (2002)
3. Caleiro C, Viganò L, and Basin D. On the Semantics of Alice&Bob Specifications of Security Protocols. Theoretical Computer Science (2006) vol. 367 (1–2) pp. 88-122

4. Dolev D and Yao AC. On the security of public-key protocols. IEEE Transactions on Information Theory, 2(29):198–208, March 1983.
5. Eugster GTh, Felber PA, Guerraoui R, and Kermarrec A-M. The Many Faces of Publish/Subscribe. ACM Computing Surveys (2003) vol. 35 (2) pp. 114–131
6. Gollmann D. On the Verification of Cryptographic Protocols – A Tale of Two Committees. Electronic Notes in Theoretical Computer Science (2000) vol. 32 (1) pp. 1-17. Revised version, 10 May 2005.
7. Haack C, Jeffrey A. Pattern-matching Spi-calculus. FAST'04 (173) pp. 193–205, 2004.
8. Huima A. Efficient Infinite-State Analysis of Security Protocols. Workshop on Formal Methods and Security Protocols (1999).
9. Jacobson V. If a Clean Slate is the solution what was the problem? Stanford 'Clean Slate' Seminar (2006).
10. Järvinen, K. et al. FPGA Design of Self-certified Signature Verification on Koblitz Curves. In Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems, CHES 2007, pp. 256-271. Vienna, Austria, September 2007.
11. Meadows C. A formal framework and evaluation method for network denial of service. In Proceedings of the 12th IEEE Computer Security Foundations Workshop. IEEE Computer Society Press, June 1999.9(1):47–74, 2001.
12. Millen J and Shmatikov V. Constraint solving for bounded-process cryptographic protocol analysis. In 8th ACM Conference on Computer and Communication Security, pages 166–175. November 2001.
13. Särelä M, Rinta-aho T, and Tarkoma T. RTFM: Publish/Subscribe Internetworking Architecture. ICT Mobile Summit, June 10-12, 2008. Stockholm 2008.
14. Scott J, Crowcroft J, Hui P, and Diot C. Haggle: a Networking Architecture Designed Around Mobile Users. (2006)
15. Syverson P et al. The Logic of Authentication Protocols. Lecture Notes in Computer Science (2001) vol. 2171 pp. 63–
16. Thayer Fábrega FJ, Herzog JC, and Guttman JD. Strand Spaces: Proving Security Protocols Correct. Journal of Computer Security (1999) vol. 7 pp. 191–230
17. Woo TYC and Lam SS. A Semantic Model for Authentication Protocols. IEEE Security and Privacy. 1993.