Privacy-Preserving Publish/Subscribe: Efficient Protocols in a Distributed Model

Giovanni Di Crescenzo[1]

Brian Coan[1]

John Schultz[3]

Simon Tsang[1]

Rebecca N. Wright[2]

Presented to:

The 8th International Workshop on
Data Privacy Management
(DPM 2013, an ESORICS 2013 Workshop)
Egham, UK,
Sep 13, 2013

Affiliation of Authors:

[1] Applied Communication Sciences, NJ, USA
{gdicrescenzo,bcoan,stsang}@appcomsci.com

[2] Rutgers University, NJ, USA
rebecca.wright@rutgers.edu

[3] Spread Concepts, MD, USA
jschultz@spreadconcepts.com

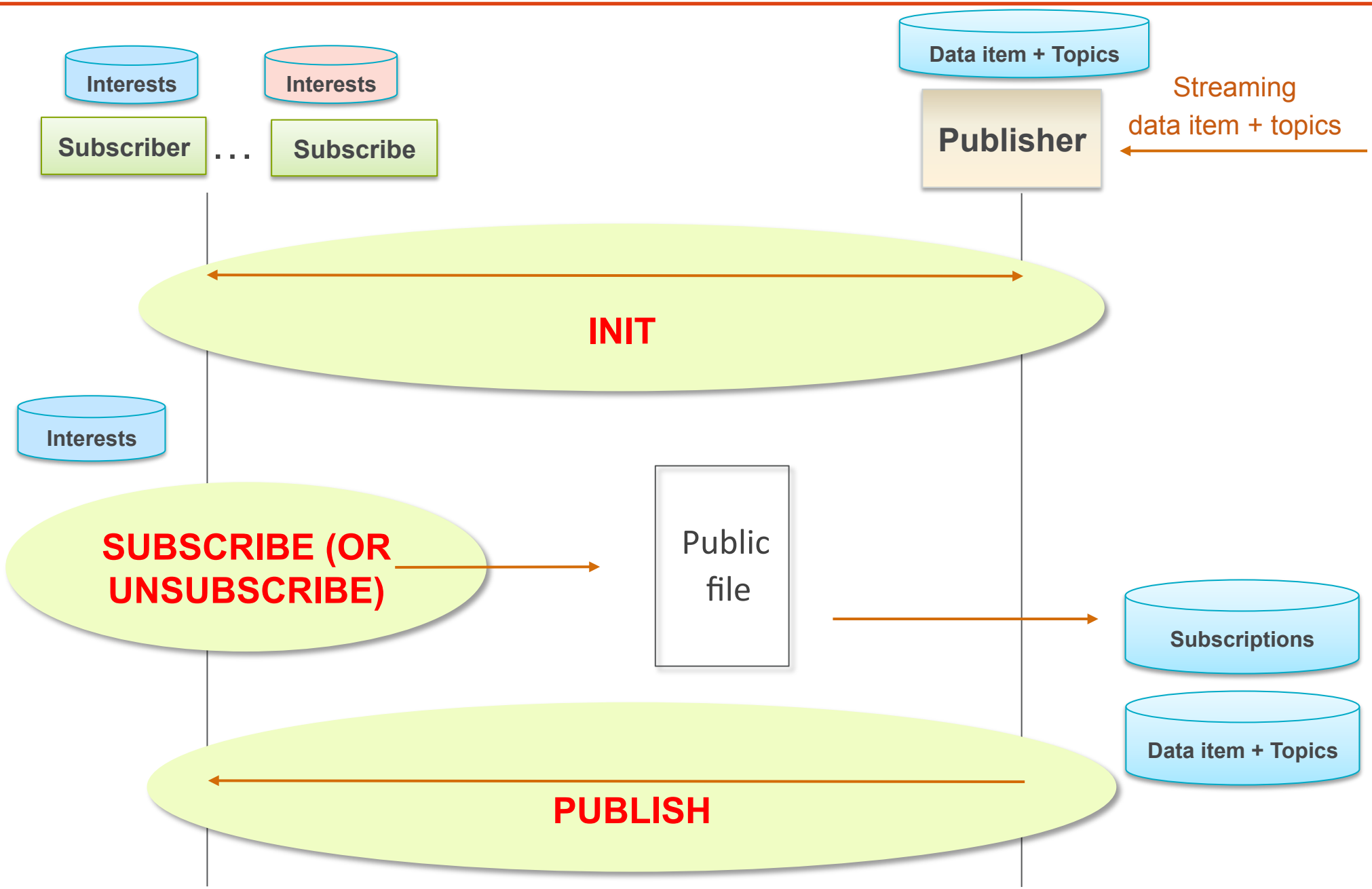
Objective: Efficient & Private Publish/Subscribe

- Publish/subscribe protocols: publishing data items to interested subscribers
- They come in many different formulations and variations; this paper:
  - a distributed protocol
  - participants can, at any given time, act as subscribers (with subscription keywords, called interests) or publishers (with data items and related publication keywords, called topics)
- Have applications in a large number of areas (e.g.: news, finance, web content) and are routinely used in practice (e.g.: RSS feeds)
- Motivations: efficiency and privacy are critical features, lack of privacy may even deter from the implementation or use of a publish/subscribe system
  - for instance, in finance, a publish/subscribe system assisting a market maker could allow subscribers to submit their interest in companies and publishers to issue data relative to companies; however, by revealing company names and data from either the subscribers or the publishers, it may not only impact participants' privacy but also significantly alter the market's pricing process and overall integrity

Modeling Publish/Subscribe Protocols

- Participant and execution model: a distributed model with many parties; after possibly a preliminary initialization phase, at any time
  - any party can subscribe by posting his subscriptions on a public file
  - any one party acts as a publisher and any subset of the remaining parties act as subscribers in one publication subprotocol
- Adversary model:
  - Honest but curious adversaries (i.e., adversary corrupts party, follows its protocol, and at the end tries to violate privacy)
  - Malicious adversaries, after authorized subscriptions (i.e., adversary performs only authorized subscriptions, then violates from the protocol in an attempt to violate privacy)
- Requirements (formal definitions are in the paper):
  - Publication correctness: at the end of each publication, matching subscribers (and only them) get the data item and topics
  - Privacy (formalized usual adaptations of standard crypto definitions): at the end of each subscription, noone else learns anything more than the number of subscriptions; at the end of each publication, matching subscribers learn the data item and topics, non-matching subscribers only learn the number of topics and the length of the data item, publisher learns nothing new
  - Efficiency: standard notions of low communication complexity, round complexity, time complexity + efficient publication latency (i.e.: up to a constant times the latency in a non-private protocol, for a practically large range of input parameters)

Publish/Subscribe Protocol Architecture



Private Publish/Subscribe: Limitations on Efficiency

- The stated privacy demands come with at least two main efficiency limitations:
  - Proposition 1: the publisher must send a message as long as the data item to each of the subscribers, regardless of whether the publication matches their interests or not
  - Observation 2: privately computing which subscribers are entitled to data items can be shown, using well-known fundamental results in cryptography, to require asymmetric cryptographic operations
- Note: general solutions from the area of secure function evaluation protocols would solve our problem but suffer from similar or worse inefficiency drawbacks
- We overcome the two efficiency limitations as follows:
  - we perform cryptographic processing of data items only once for all subscribers, by encrypting the data item once and distributing the key only to matching subscribers; in one variant, we use a repository server
  - we minimize the use of asymmetric cryptographic operations in distributing the encrypting key by using hybrid COT, a novel hybrid cryptographic primitive (i.e., starting with asymmetric cryptographic operations and then continuing with symmetric ones for the rest of the protocol lifetime)

Solution building block 1: cryptographic pseudonyms

- Computing on encrypted interests
  - Client interests and item topics are mapped to cryptographic pseudonyms (i.e., encryptions of interests, topics), based on a variant of El-Gamal encryption
  - Afterwards all computations (i.e., the actual publication) are performed on encrypted values
- Construction:
  - Each interest/topic pseudonym is a triple (h,u,v) where

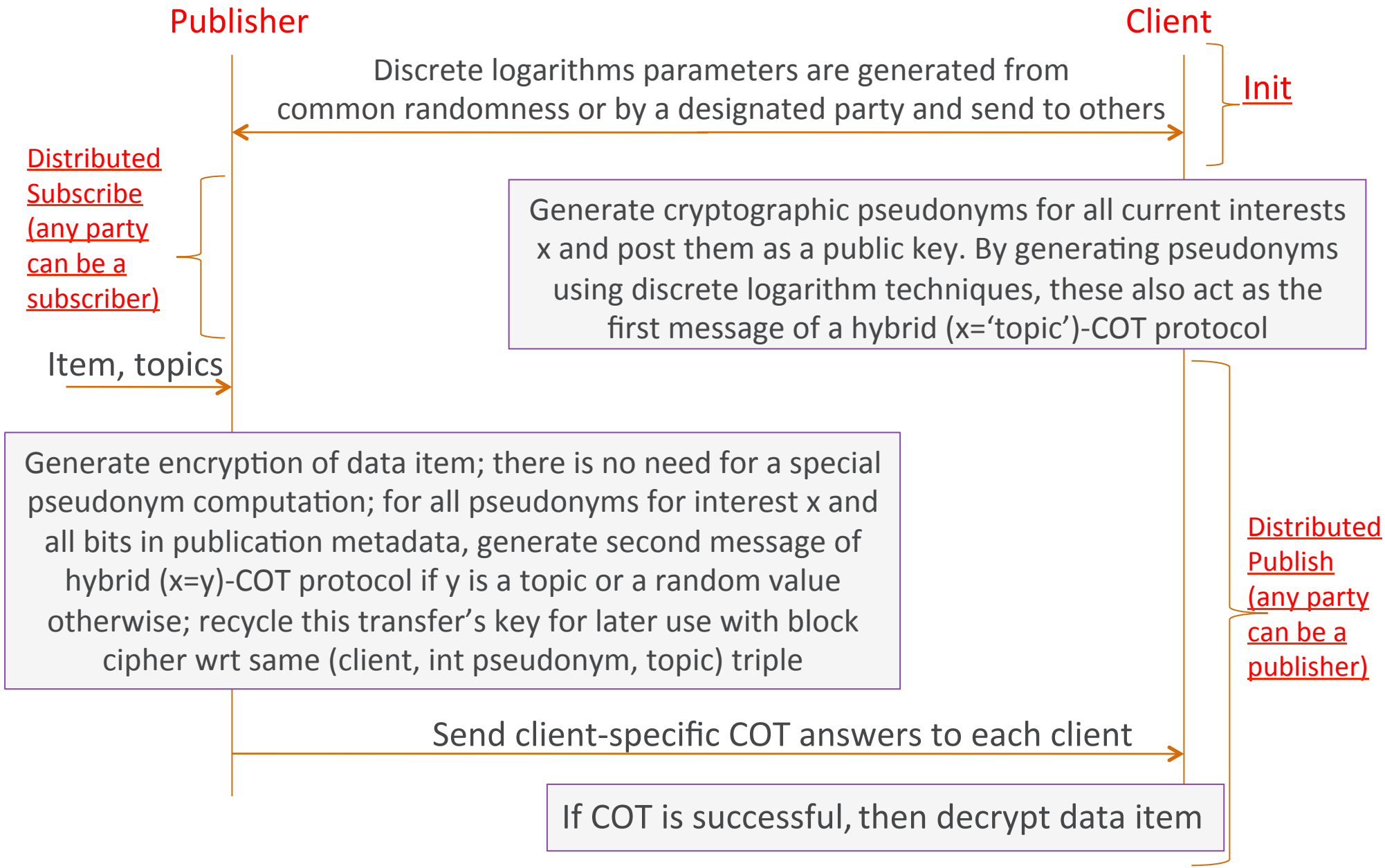
$$h=g^r \bmod p,\ u=g^s \bmod p,\ v=h^s \cdot x \bmod p$$

- Properties:
  - The value v (hiding x) is "division-malleable" into a value v' (hiding x/y for any given y)
  - The value v' (hiding x/y) is "randomization-malleable" into value v'' (hiding x^z for any given random z)
  - $x \neq y$ → applying to v division + randomization malleability returns $v''=h^s (x/y)^z \bmod p$
  - $x=y$ → applying to v division + randomization malleability returns $v''=h^s \bmod p$

Solution building block 2: hybrid equality-COT

- Hybrid encryption is the typical way to efficiently encrypt long messages: first encrypt a short key k using asymmetric encryption and then use k to symmetrically encrypt the long message
- Conditional Oblivious Transfer (COT) is a variant of oblivious transfer (OT) [Rabin81] proposed by Di Crescenzo, Ostrovsky and Rajagopalan [DOR99]. In a COT protocol for a predicate *P*, a sender holds message *m* and value *x* and a receiver holds a value *y*. The sender wants to send *m* to the receiver so that at the end:
  - Conditional: receiver obtains *m* if and only if P(x,y) is true
  - Obliviousness: sender does not gain any info about *P(x,y)* (thus, whether client receives *m* or not);
  - Privacy: the inputs *x* and *y* remain private.
- Equality-based Conditional Oblivious Transfer (eqCOT): COT where P(x,y) is "x=y"
- Hybrid Equality-based Conditional Oblivious Transfer (h-eqCOT)
  - Consider multiple executions with the same (x,y) inputs, but a different message m on each execution
  - Instead of repeating the eqCOT scheme for each execution, a hybrid eqCOT scheme uses asymmetric crypto techniques only on the first execution and (much more efficient) symmetric crypto thereafter
  - Basic idea: send a shared key k using the eqCOT in a first execution and then use k to symmetrically encrypt the messages to be sent in all executions

Our solution: informal description



Hybrid equality-COT: some more details

- Consider, for simplicity, a single interest x and a single topic y
- Publisher wants to conditionally transfer a key k to the susbcriber

- The asymmetric phase goes as follows (symmetric case is simple and omitted here):
- Subscriber sends triple (h,u,v) where $h=g^r \bmod p$, $u=g^s \bmod p$, $v=h^s \cdot x \bmod p$, where x is the interest
- Publisher sends pair (d,e) where $d = g^a \cdot u^b \bmod p$, $e = h^a \cdot (v/y)^b \cdot k \bmod p$
- Subscriber computes $e \cdot d^{-r} \bmod p =$

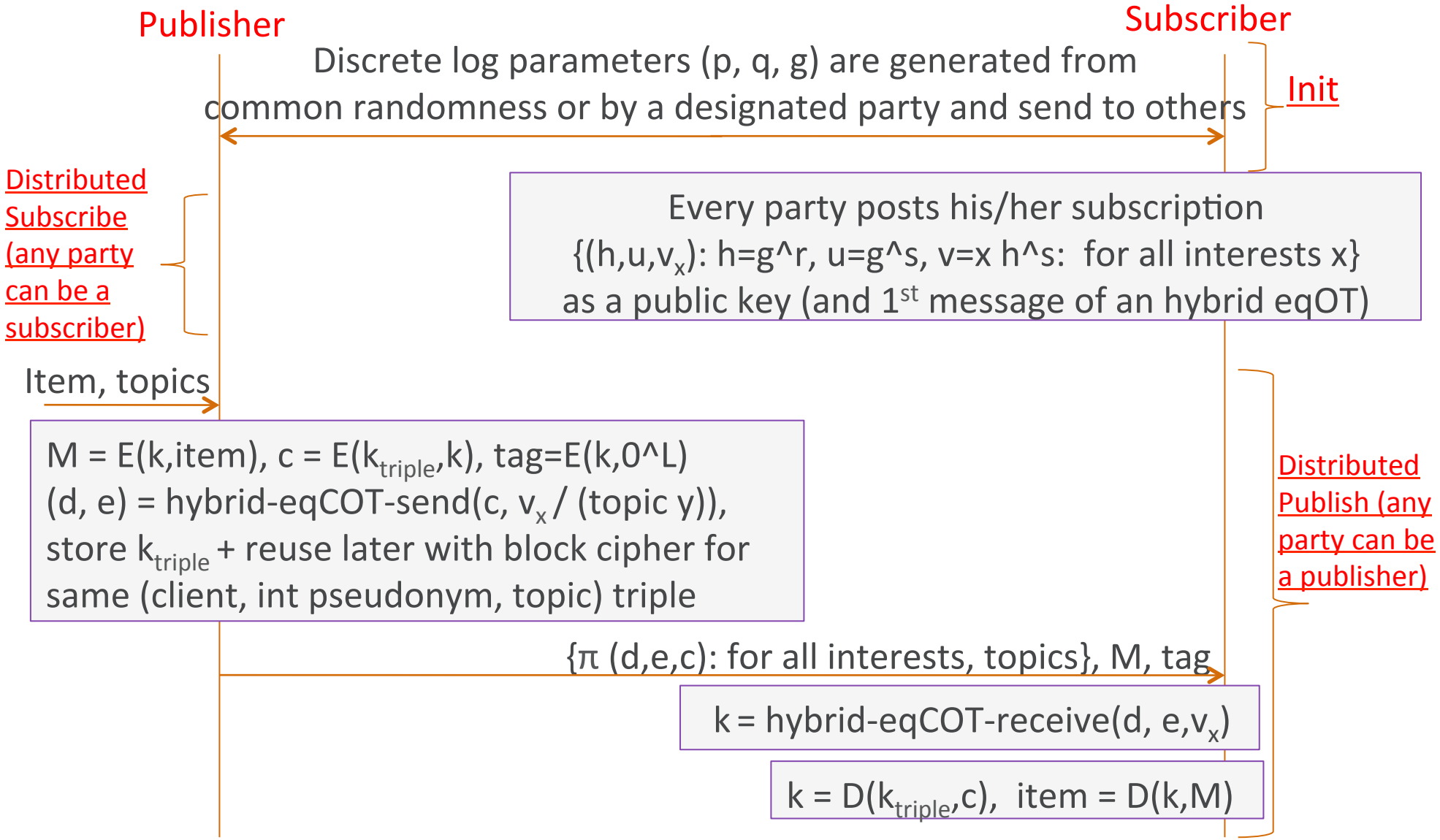
$$= h^a \cdot (v/y)^b \cdot k \cdot (g^a \cdot u^b)^{-r} \bmod p$$

$$= g^{ra} \cdot (h^s x /y)^b \cdot k \cdot (g^a \cdot g^{sb})^{-r} \bmod p$$

$$= \text{(when x=y)}\ g^{ra} \cdot (g^{rs})^b \cdot k \cdot (g^a \cdot g^{sb})^{-r} \bmod p$$

$$= k$$

P3: detailed description



Sketch of Properties: Correctness, Privacy

- Formal proofs are in the paper
- Publication Correctness:
  - if at least one interest = one topic, then at least one h-eqCOT is successful and the subscriber obtains the data item; otherwise, the subscriber obtains random values and cannot decrypt the data item
- Privacy against publisher:
  - the interest pseudonyms are encryptions of the interests
- Privacy against subscriber:
  - in the case of no match, the h-eq-COT does not reveal any information about the data-encrypting key
- Privacy against eavesdroppers (or traffic analysis):
  - as all clients receive an encrypted data item + h-eqCOT messages, traffic analysis does not help detecting which clients received the data item or not

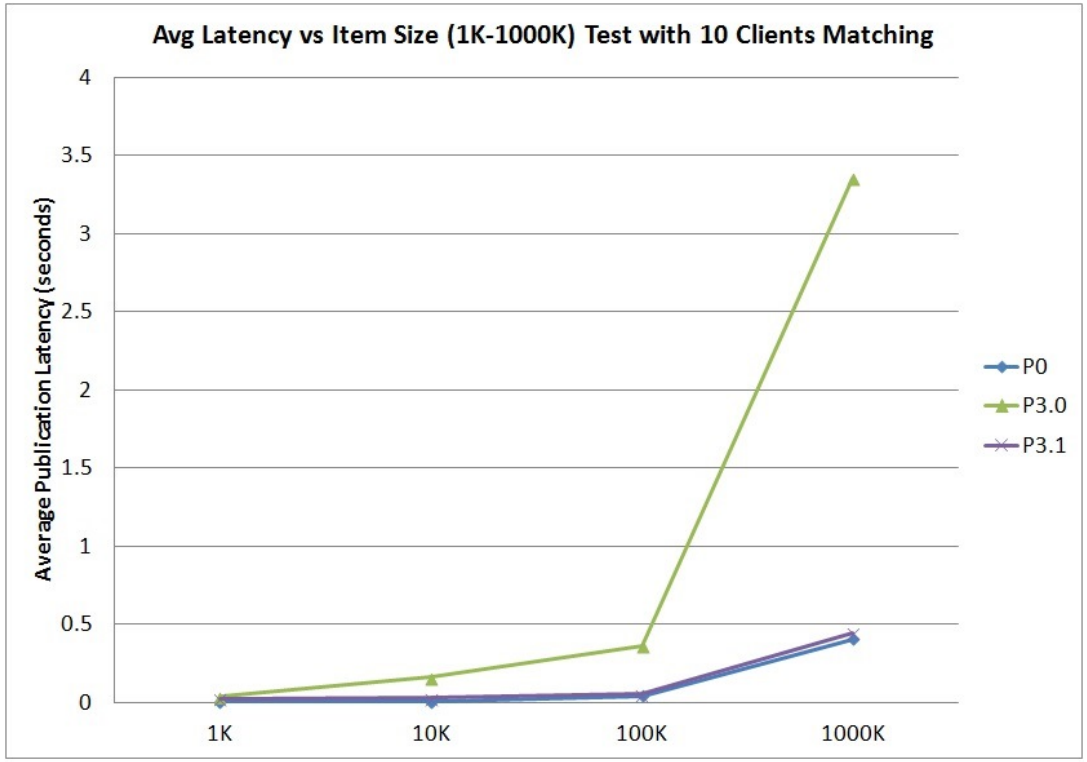
Summary of Properties: P0, P3.0 and P3.1

P0: protocol with no privacy mechanism
P3.0: our main protocol;
P3.1: a P3.0 variant, where the data item is only recovered through a repository server

Properties → Protocol	Publication Correctness	Privacy (hbc or malicious with authorized subscriptions model)		Privacy against Eavesdropper / Traffic analysis	Efficiency / Trust
		Publisher	Subscriber		
P0	yes	No privacy	No privacy	No privacy	Very efficient
P3.0	yes	Only leaks number of interests	Only leaks number of topics and length of data item	Only leaks number of interests, topics, and length of data item	Up to 8 times less efficient than P0 for large parameter ranges
P3.1	yes	As above	As above	Additionally leaks which subscribers received the data item	Essentially as efficient as P0 but trusting a new server

Implementation and Performance results

- Implementation: used a collection of 6 Dell PowerEdge 1950 processors and one Dell PowerEdge 2950 processor. Subscribers were divided in 4 groups of size 25 each, and each group was run on a PowerEdge 1950 processor. The publisher was run on a dedicated 1950 processor, the third party was run on dedicated 1950 processor, and the testing control was run on the 2950 processor. All initialization, subscription, and publication traffic was run over a dedicated gigabit Ethernet LAN. Testing control and collection of timing measurement traffic was isolated on a separate dedicated gigabit Ethernet LAN.
- Parameter settings: We compared P3.0, P3.1 and P0 using a publication rate of 1 item/sec, with values 1K, 10K, 100K, and 1000K bytes for data items, 10 matching subscribers and 10 topics per item.
- Results: performance of P3.0 is always within a small constant (e.g., 8) from the performance of P0 (remarkable, as P3.0 sends information to all subscribers to safeguard privacy against publisher), while P0 is only sending to interested participants. P3.1, using a repository server, has performance very close to that of P0 (losing some privacy against traffic analysis, and requiring 1 server).



Conclusions

- We formally defined a distributed model for publish/subscribe protocols where participants can act as publishers or as subscribers in any given publication transaction
- In this challenging model, we showed that solutions with provable privacy and efficiency are possible. In particular, two inherent efficiency limitations (the use of asymmetric cryptography operations and the fact that data items need to be sent to all subscribers) can be mitigated to have only a very small impact on performance, allowing private solutions with efficiency comparable to non-private solutions, and without the need of a broker (as in our recent solution [NSS 2013])
- Our approach, based on a novel cryptographic primitive (i.e., hybrid conditional oblivious transfer protocols), can also be generalized to more elaborate publish/subscribe conditions