Query Privacy in Sensing-as-a-Service Platforms

Ruben Rios David Nuñez Javier Lopez

Network, Information and Computer Security Lab
Department of Computer Science
University of Malaga
{ruben,dnunez,jlm}@lcc.uma.es

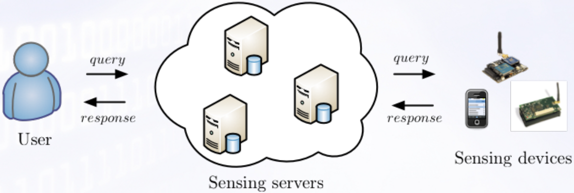
IFIP SEC 2017
May 29, 2017. Rome (Italy)

Network, Information and Computer Security Lab

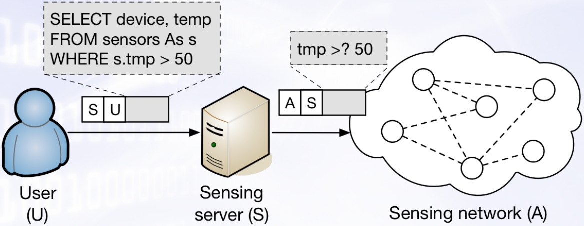
Sensing-as-a-Service Platforms

S^{2}aaS platforms allow querying for data from sensing devices via a sensing server

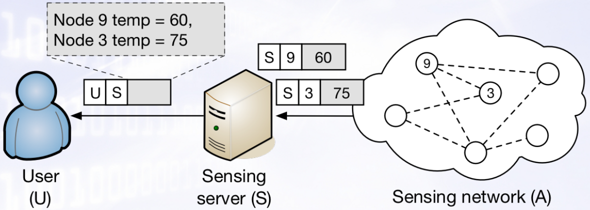
- Sensing devices may belong to companies, administrations or citizens
- Sensing servers act as communication gateways
- The user issues queries and waits for the response



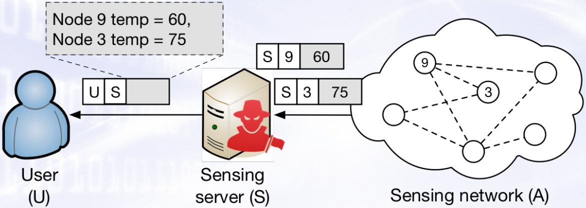
How does it work?



How does it work?



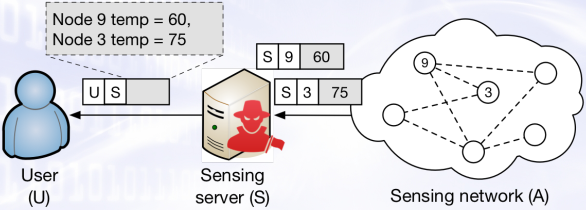
Honest-but-curious Sensing Server



Sensing servers may access to the contents of the queries as well as contextual information to route the queries

- User privacy is at stake!

Why Not Encrypt Traffic?



Traditional end-to-end encryption has several drawbacks:

1. The user needs to know the key of every single sensing device
2. The user has to check the status of the keys
3. Multi-/Broadcast queries demands multiple transmissions

Our Solution

We propose the QPSP (**Q**uery **P**rivacy for **S**ensing **P**latforms) protocol

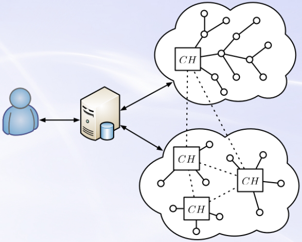
QPSP is built on techniques inspired by proxy re-encryption and k-anonymity to provide

- Query confidentiality: hide the query and response contents
- Query privacy: hide the communication end points

System Model

We assume a number of sensing devices organized into clusters

There are several cluster heads and they must be able to communicate with one another and with the sensing server



The readings of the sensing devices are publicly available to anyone requesting them

– Example: Smart City scenario

The sensing server and the sensing devices are assummed not to collude against the user

Adversarial Model

The sensing server is semi-honest (a.k.a. honest-but-curious)

- Wants to learn the interests of a particular user based on his/her queries

We assume it has the following capabilities:

- Content analysis: it can observe packet payloads and headers
- Statistical analysis: it can analyze features of the communication flow

But... we consider it may also

- Collude with external entities located in the vicinity of the sensing devices
- Try to cheat by slightly modifying its behaviour as long as it does not deviate from the protocol specification

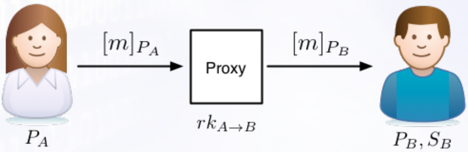
Outline

Cryptographic Notions

Proxy Re-encryption

Proxy re-encryption is a type of PK encryption that enables a proxy to transform ciphertexts under Alice's public key (P_A) into ciphertexts decryptable by Bob's secret key (S_B).

To that end, the proxy is given a re-encryption key ($rk_{A\to B}$), generated by Alice.



Most of these schemes are based on pairing-based cryptography

Overview

The QPSP protocol consists of three phases:

1. **Initialization**: a global public key (pk_P) is generated by the cluster heads[1]. Re-encryption keys are also generated in this phase.
2. **Query**: The user encrypts the query using pk_P, which is transformed by the sensing server using the re-encryption key ($rk_{P \to i}$) of an arbitrary cluster head. The cluster head decrypts the query and forwards it to the appropriate sensing device.
3. **Response**: the confidentiality of the response is secured from the user end by incorporating a fresh key into the query.

Some traffic obfuscation mechanisms are introduced to prevent leaking information.

[1] No single entity controls the corresponding decryption key

Phase1: Initialization

Each cluster head CH_i generates a key pair $(pk_i, sk_i) = (h^{x_i}, x_i)$ and shares the pk_i with the other cluster heads

Next, each CH_i generates a temporal secret value p_i and computes

$$u_i = Z^{p_i}$$

$$v_{ij} = (pk_j)^{p_i} = h^{x_j p_i}$$

The sensing server receives $(u_i, \{v_{ij}\})$ from all cluster heads and computes the global public key and the re-encryption keys:

$$pk_P = \prod_{i=1}^{N} u_i = \prod_{i=1}^{N} Z^{p_i} = Z^{p_1+\ldots+p_N} = Z^p$$

$$rk_{P\to i} = \prod_{j=1}^{N} v_{ji} = \prod_{j=1}^{N} h^{x_i p_j} = h^{x_i(p_1+\ldots+p_N)} = h^{x_i p}$$

Phase2: Query

User

Message 1: Encryption

The user encrypts $m = Q \parallel K$, using the global public key pk_P

$$\mathsf{Enc}_P(m) = (g^r, m \cdot (pk_P)^r) = (g^r, m \cdot Z^{p \cdot r}) = M_1$$

Sensing Server

Message 2: Re-encryption

The sensing server sends M_2 to an arbitrary CH_i

$$\mathsf{ReEnc}_i(M_1) = (e(g^r, rk_{P \to i}), m \cdot Z^{p \cdot r}) = (Z^{p \cdot r \cdot x_i}, m \cdot Z^{p \cdot r}) = M_2$$

Cluster Head

Decryption

The cluster head CH_i uses its secret key sk_i to decrypt M_2

$$\mathsf{Dec}_i(M_2) = CT_2 \cdot (CT_1)^{-1/sk_i} = m \cdot Z^{p \cdot r} \cdot (Z^{p \cdot r \cdot x_i})^{-1/x_i} = m$$

Phase3: Response

The query Q is delivered to the actual destination using a k-anonymous transmission protocol

- For any given identifier, k destinations are chosen using a deterministic function
- Destinations may receive the actual or bogus queries

All k destinations must behave in the same way to cover the actual query recipient. They all respond to the query and the cluster head filters out cover messages.

The true response R is encrypted by the CH using key K and finally sent to the sensing server, which forwards it to the user

Outline

Experimental Evaluation

Proof of concept in C using the Apache Milagro Crypto Library

We needed an elliptic curve that supports a Type-3 pairing

- 256-bit Barreto-Naehrig (BN) curve

The following table shows the average value after 100 experiments

Entity	**Platform**	**Operation**	**Cost (ms)**
User	Laptop†	Encryption	7.58
Sensing server	Laptop†	Re-Encryption	11.55
Cluster head	RPi 1 B§	Decryption	46.20
Cluster head	Intel Galileo 1*	Decryption	122.20

† Core2Duo@2.66GHz, 8GB   § SoC@700MHz, 512MB   * SoC@400MHz, 256MB

Outline

Conclusion and Future Work

We have presented the QPSP protocol as a mechanism to prevent user profiling in semi-trusted S^2aaS platforms

The solution is built on proxy re-encryption primitives and traffic obfuscation at the sensing network

As future work we are considering

- Scenarios where users need to be authorized to query for data
- Issues related to node revokation and the addition of new cluster heads
- Dealing with a portion of compromised sensing devices

Thank you for your Attention!

Any questions?

Ruben Rios
ruben@lcc.uma.es

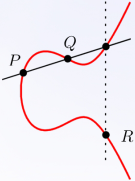
Cryptographic Notions

Bilinear Pairing

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order q. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ satisfying the properties of bilinearity, non-degeneracy, and computability

1. Bilinearity: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab} = e(g_1^b, g_2^a)$
2. Non-degeneracy: $e(g_1, g_2) \neq 1$
3. Computability: There is an efficient algorithm that computes e

Bilinear pairings for cryptography are usually constructed over elliptic curves



Query Confidentiality

The encryption scheme is IND-CPA under the External DH assumption.

(Informal) Proof

Challenger		Adversary
DDH tuple (g^a, g^b, g^x) $\longrightarrow$		
Sample $h \in \mathbb{G}_2$ $pk_P^* = e(g^a, h)$		
	pk_P^* $\longrightarrow$	
	m_0, m_1 $\longleftarrow$	
$\delta \xleftarrow{R} \{0, 1\}$ $c^* = (g^b, m_\delta \cdot e(g^x, h))$		
	c^* $\longrightarrow$	
	δ' $\longleftarrow$	
If $\delta = \delta'$ output "$x = a \cdot b$" $\longleftarrow$		

Query Privacy

The sensing server can only learn (with the help of external colluders) the k destinations but not the actual query recipient

- This is true for a single and multiple runs of the protocol

What if the sensing server chooses the cluster head at will?

- He learns nothing since all cluster heads use the same mapping function

What if the sensing server crafts its own queries?

- The only thing it learns is the mapping function for a particular node
- But this is not sensitive

Related Work

Most research in query privacy has been done in WSN

The trivial solution is not scalable nor energy efficient

- Consists of making all nodes reply to every query

Solutions that aim to reduce the overhead while preserving privacy

- Data-aggregation [DPV11]
- Bogus queries [CYS+10]
- Actual destination is hidden with the query path [DCDT09]
- Sensed data is unlinked from sensing device [DS11, CP13]
- Query transformations [LL12, CL12, ZDP+14]

References

- F. Chen and A. X. Liu, *Privacy- and integrity-preserving range queries in sensor networks*, IEEE/ACM Transactions on Networking **20** (2012), no. 6, 1774–1787.
- E. De Cristofaro and R. Di Pietro, *Adversaries and countermeasures in privacy-enhanced urban sensing systems*, IEEE Systems Journal **7** (2013), no. 2, 311–322.
- Bogdan Carbunar, Yang Yu, Weidong Shi, Michael Pearce, and Venu Vasudevan, *Query privacy in wireless sensor networks*, ACM Trans. Sen. Netw. **6** (2010), no. 2, 14:1–14:34.
- Emiliano De Cristofaro, Xuhua Ding, and Gene Tsudik, *Privacy-Preserving Querying in Sensor Networks*, 18th International Conference on Computer Communications and Networks (San Francisco, CA), ICCCN '09, IEEE Computer Society, Washington, DC, USA, 3-6 Aug. 2009, pp. 1–6.

References (contd.)

- Roberto Di Pietro and Alexandre Viejo, *Location privacy and resilience in wireless sensor networks querying*, Comput. Commun. **34** (2011), no. 3, 515–523.
- T. Dimitriou and A. Sabouri, *Privacy preservation schemes for querying wireless sensor networks*, IEEE International Conference on Pervasive Computing and Communications Workshops,, 2011, pp. 178–183.
- Xiaojing Liao and Jianzhong Li, *Privacy-preserving and secure top-k query in two-tier wireless sensor network*, 2012 IEEE Global Communications Conference (GLOBECOM), Dec 2012, pp. 335–341.
- Xiaoying Zhang, Lei Dong, Hui Peng, Hong Chen, Deying Li, and Cuiping Li, *Achieving efficient and secure range query in two-tiered wireless sensor networks*, 2014 IEEE 22nd International Symposium of Quality of Service (IWQoS), May 2014, pp. 380–388.