International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Incorporating Google Cloud Based Pub/Sub in Implementation of Identity Concentrated Encryption

Shilpashree K S[1], Harshavardhan L[2]

[1, 2]Department of CS&E, BGSIT, BG Nagar

Abstract: ***In a content-based publisher subscriber system it is very challenging to provide security mechanisms. This paper presents an approach making use of the cloud environment at the middleware. Google Cloud Pub/Sub brings the scalability, flexibility, and reliability of enterprise message-oriented middleware to the cloud. The identity based cryptography is used for the authentication of publishers and subscribers.***

Keywords: Google cloud, content-based, security, identity-based encryption.

1. Introduction

The publish/subscribe (pub/sub) communication paradigm has gained high popularity because of its inherent decoupling of publishers from subscribers in terms of time, space, and synchronization. Publishers inject information into the pub/sub system, and subscribers specify the events of interest by means of subscriptions.

Published events are routed to their relevant subscribers, without the publishers knowing the relevant set of subscribers, or vice versa. This decoupling is traditionally ensured by intermediate routing over a broker network. In more recent systems, publishers and subscribers organize themselves in a broker-less routing infrastructure, forming an event forwarding overlay. Content-based pub/sub is the variant that provides the most expressive subscription model, where subscriptions define restrictions on the message content. Its expressiveness and asynchronous nature is particularly useful for large-scale distributed applications such as news distribution, stock exchange, environmental monitoring, traffic control, and public sensing. Not surprisingly, pub/sub needs to provide supportive mechanisms to fulfill the basic security demands of these applications such as access control and confidentiality. Access control is achieved as events are delivered to authorized subscribers.

In the past, most research has focused only on providing expressive and scalable pub/sub systems, but little attention has been paid for the need of security. Existing approaches toward secure pub/sub systems mostly rely on the presence of a traditional broker network. These either address security under restricted expressiveness, for example, by using only keyword matching for routing events or rely on a network of (semi-)trusted brokers. Fine grained access was not provided in scalable manner. Clustering was made based on the subscriptions.

Based on these results, this paper incorporates Google Cloud Pub/Sub that brings the scalability, flexibility, and reliability of enterprise message-oriented middleware to the cloud. By providing many-to-many, asynchronous messaging that decouples senders and receivers, it allows for secure and highly available communication between independently written applications. Google Cloud Pub/Sub delivers low-latency, durable messaging that helps developers quickly integrate systems hosted on the Google Cloud Platform and externally. This paper has adapted identity-based encryption (IBE) mechanisms [1], [2] 1) to ensure that a particular subscriber can decrypt an event only if there is a match between the credentials associated with the event and the key; and 2) to allow subscribers to verify the authenticity of received events.

2. System Model and Background

2.1 Content-Based Publish/Subscribe

For the routing of events from publishers to the relevant subscribers, we use the content-based data model. The event space, denoted by OMEGHA, is composed of a global ordered set of d distinct attributes (Ai): OMEGHA={A1, A2,….., Ad}. Each attribute Ai is characterized by a unique name, its data type, and its domain.

The data type can be any ordered type such as integer, floating point, and character strings. The domain describes the range [Li, Ui] of possible attribute values. A subscription filter f is a conjunction of predicates, i.e., f ={Pred1 ^ Pred2 …. ^ Predj}. Predi is defined as a tuple (Ai, Opi, vi), where Opi denotes an operator and vi a value. The operator Opi typically includes equality and range operations for numeric attributes and prefix/suffix operations for strings.

An event consists of attributes and associated values. An event is matched against a subscription f if the values of attributes in the event satisfy the corresponding constraints imposed by the subscription. It is considered that, pub/sub in a setting where there exits no dedicated broker infrastructure. Publishers and subscribers contribute as peers to the maintenance of a self-organizing overlay structure. To authenticate publishers, we use the concept of advertisements in which a publisher announces beforehand the set of events which it intends to publish.

2.2 Identity-Based Encryption

While a traditional PKI infrastructure requires to maintain for each publisher or subscriber a private/public key pair which has to be known between communicating entities to encrypt and decrypt messages, identity-based encryption [3]

provide a promising alternative to reduce the amount of keys to be managed. In identity-based encryption, any valid string which uniquely identifies a user can be the public key of the user. A key server maintains a single pair of public and private master keys. The master public key can be used by the sender to encrypt and send the messages to a user with any identity, for example, an e-mail address.

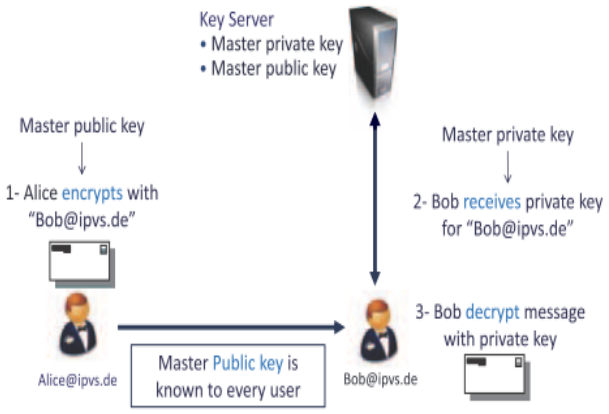

Figure 1: Identity based encryption

To successfully decrypt the message, a receiver needs to obtain a private key for its identity from the key server. Fig. 1 shows the basic idea of using identity-based encryption. We want to stress here that although identity-based encryption at the first glance appears like a highly centralized solution, its properties are ideal for highly distributed applications. A sender needs to know only a single master public key to communicate with any identity. Similarly, a receiver only obtains private keys for their own identities. Furthermore, an instance of central key server can be easily replicated within the network. Finally, a key server maintains only a single pair of master keys and therefore, can be realized as a smart card, provided to each participant of the system.

3. Approach Overview

Google Cloud Pub/Sub brings the scalability, flexibility, and reliability of enterprise message-oriented middleware to the cloud. By providing many-to-many, asynchronous messaging that decouples senders and receivers, it allows for secure and highly available communication between independently written applications. Google Cloud Pub/Sub delivers low-latency, durable messaging that helps developers quickly integrate systems hosted on the Google Cloud Platform and externally.

Figure 2: Google Cloud Pub/Sub.

publisher application creates and sends messages to a topic. Subscriber applications create a subscription to a topic to receive messages from it. Communication can be one-to-many, many-to-one and many-to-many

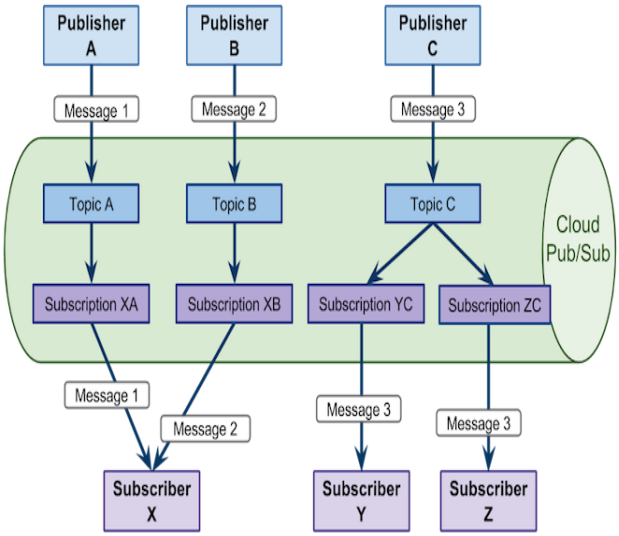

Figure 3: Message flow in Cloud Pub/Sub

3.1 Common scenarios

Here are some classic use cases for Google Cloud Pub/Sub:

- **Balancing workloads in network clusters.** For example, a large queue of tasks can be efficiently distributed among multiple workers, such as Google Compute Engine instances.
- **Implementing asynchronous workflows.** For example, an order processing application can place an order on a topic, from which it can be processed by one or more workers.
- **Distributing event notifications.** For example, a service that accepts user signups can send notifications whenever a new user registers, and downstream services can subscribe to receive notifications of the event.

- **Refreshing distributed caches.** For example, an application can publish invalidation events to update the IDs of objects that have changed.
- **Logging to multiple systems.** For example, a Google Compute Engine instance can write logs to the monitoring system, to a database for later querying, and so on.
- **Data streaming from various processes or devices.** For example, a residential sensor can stream data to backend servers hosted in the cloud.
- **Reliability improvement.** For example, a single-zone Compute Engine service can operate in additional zones by subscribing to a common topic, to recover from failures in a zone or region.

3.2 Benefits and features

- **Unified messaging:** Durability and low-latency delivery in a single product
- **Global presence:** Connect services located anywhere in the world
- **Flexible delivery options:** Both push- and pull-style subscriptions supported
- **Data reliability:** Replicated storage and guaranteed at-least-once message delivery
- **End-to-end reliability:** Explicit application-level acknowledgement
- **Data security and protection:** Encryption of data on the wire and at rest
- **Flow control:** Dynamic rate limiting implemented by the Pub/Sub system
- **Simplicity:** Easy-to-use REST/JSON API

3.3 Pub/Sub concepts and data flow

Here is an overview of the components in the pub/Sub system and how data flows between them:

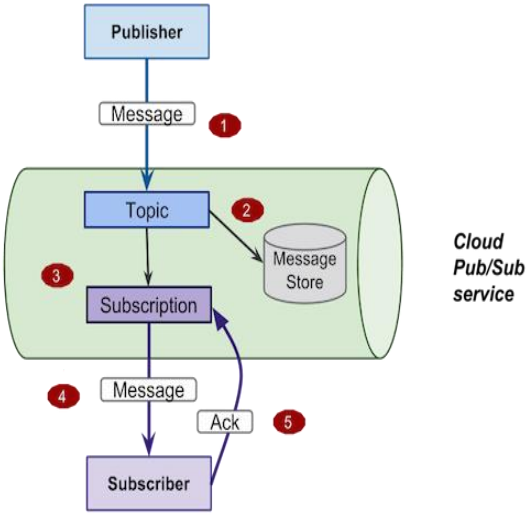

Figure 4: Pub/Sub data flow

1. A publisher application creates topic in the Google Cloud Pub/Sub service and sends messages to the topic. A message contains a payload and optional attributes that describe the payload content.
2. Messages are persisted in a message store until they are delivered and acknowledged by subscribers.
3. The Pub/Sub service forwards messages from a topic to all of its subscriptions, individually. Each subscription receives messages either by Pub/Sub *pushing* them to the subscriber's chosen endpoint, or by the subscriber *pulling* them from the service.
4. The subscriber receives pending messages from its subscription and acknowledges each one to the Pub/Sub service.
5. When a message is acknowledged by the subscriber, it is removed from the subscription's queue of messages.

3.4 Publisher and subscriber endpoints

Publishers can be any application that can make HTTPS requests to googleapis.com which is an App Engine app, a web service hosted on Google Compute Engine or any other third-party network, an installed app for desktop or mobile device, or even a browser.

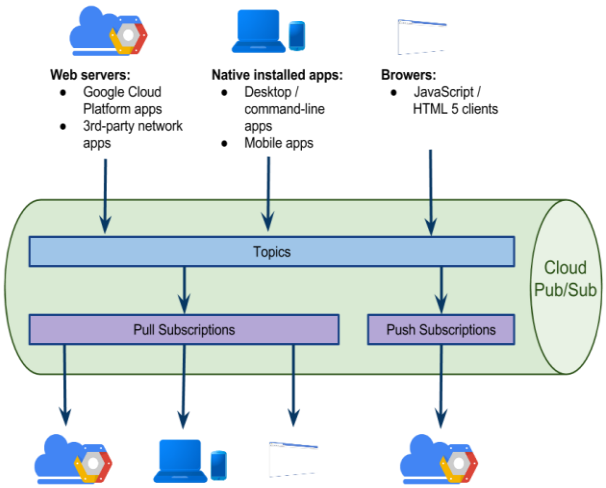

Figure 5: Publisher Subscriber end points

Pull subscribers can also be any application that can make HTTPS requests to googleapis.com. Currently, push subscribers must be Webhook endpoints that can accept POST requests over HTTPS.

4. Conclusion

This paper presents techniques from identity based encryption 1) to ensure that a particular subscriber can decrypt an event only if there is a match between the credentials associated with the event and its private keys and 2) to allow subscribers to verify the authenticity of received events. Google Cloud Pub/Sub brings the scalability, flexibility, and reliability of enterprise message-oriented middleware to the cloud. By providing many-to-many, asynchronous messaging that decouples senders and receivers, it allows for secure and highly available communication between independently written applications. Google Cloud Pub/Sub delivers low-latency, durable messaging that helps developers quickly integrate systems hosted on the Google Cloud Platform and externally. This approach is not recommended for production use.

References

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," Proc. IEEE Symp. Security and Privacy, 2007.
[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data,"
[3] D. Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing," Proc. Int'l Cryptology Conf. Advances in Cryptology,2001.
[4] M. Nabeel, N. Shang, and E. Bertino, "Efficient Privacy Preserving Content Based Publish Subscribe Systems," Proc. 17th ACM Symp. Access Control Models and Technologies, 2012.
[5] H. Khurana, "Scalable Security and Accounting Services for Content-Based Publish/Subscribe Systems," Proc. ACM Symp.
[6] Applied Computing, 2005.
[7] M. Ion, G. Russello, and B. Crispo, "Supporting Publication and Subscription Confidentiality in Pub/Sub Networks," Proc. Sixth Int'l ICST Conf. Security and Privacy in Comm. Networks (SecureComm),2010.

Author Profile

Shilpashree K S received the B.E degree in Computer Science from VTU Karnataka in 2013, and currently he is a post graduate student pursuing M.Tech in Computer Science and Engineering from B.G.S Institute of Technology under Visvesvaraya Technological University Karnataka. She has presented 2 papers in National Conferences; her main research interests include computer networks, cloud computing and wireless sensor networks. She is currently doing his project in Cloud Computing.

Harshavardhan L is working as an assistant professor in BGSIT, BG Nagar, Visvesvaraya Technological University, Karnataka. His bareas of interest are algorithm, Encryption, Computer Graphics etc.