Google Cloud Messaging

Seminar

Kevin Joseph

July 18, 2016

College of Engineering, Trivandrum

Table of contents

1. Introduction to Cloud

2. Communication Between Cloud and Device

3. Push Technology

4. Cloud To Device Messaging (C2DM)

5. Google Cloud Messaging

6. Firebase Cloud Messaging(FCM)

7. Conclusion

Introduction to Cloud

Cloud Computing

Cloud computing is a general term for the delivery of hosted services over the Internet.

Cloud computing enables companies to consume computer resources as a utility – just like electricity – rather than having to build and maintain computing infrastructures in-house.

Types of Cloud Services

- Software As A Service (SAAS)
- Platform As A Service (PAAS)
- Infrastructure As A Service (IAAS)

Communication Between Cloud and Device

Types of Communication

- **Pulling**: The request for the transmission of information is initiated by the receiver or client.
- **Pushing**: The request for a given transaction is initiated by the publisher or central server.

Push Technology

Push

Push services are often based on information preferences expressed in advance. This is called a **publish/subscribe** model. A client "subscribes" to various information "channels" provided by a server; whenever new content is available on one of those channels, the server pushes that information out to the client.

General Usage: Synchronous Conferencing, instant messaging, email, etc

Hosted Push Services

Push notification services are available from several providers which support automated push notification campaigns:

- Apple Push Notification Service
- Google Cloud Messaging

Cloud To Device Messaging (C2DM)

C2DM

Features:

- When messages are received on the Android client, the system will wake up the application via an Intent broad- cast, and pass the message data
- Developers are encouraged to send short messages, essentially notifying the mobile application that updated information can be retrieved from the server
- Maxi- mum number of messages that can be sent is approxi- mately 200,000 per day

Disadvantages:

- Certain features, such as sending messages to multiple clients, is not supported.
- The development environment and API could be better.
- In certain cases quite a big increase in response times for some requests was seen.

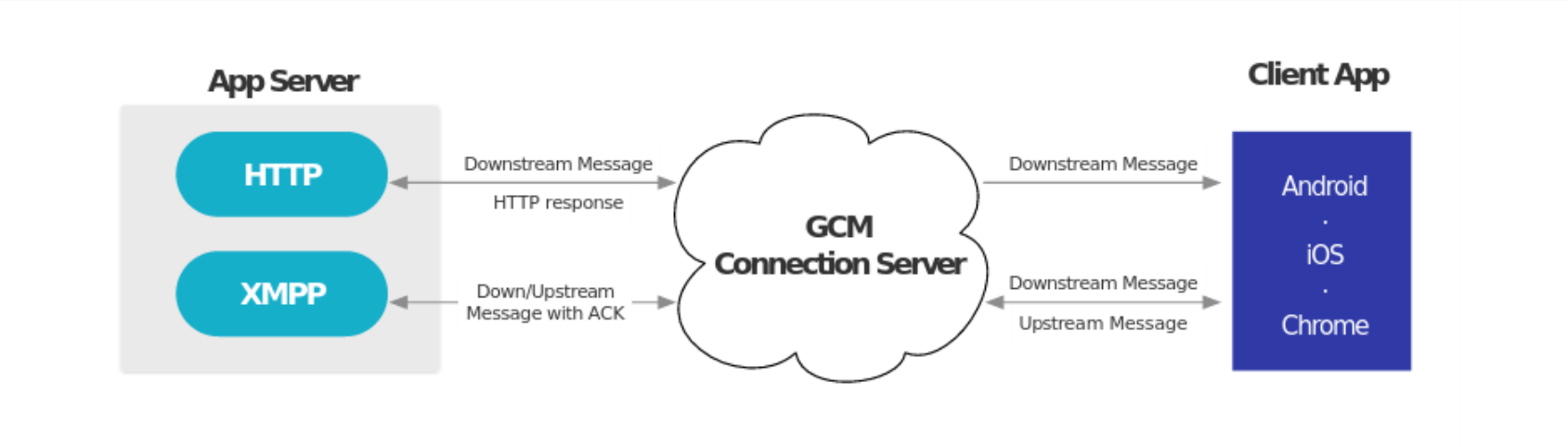
Google Cloud Messaging

GCM

Google Cloud Messaging (GCM) is a free service that enables developers to send messages between servers and client apps. This includes downstream messages from servers to client apps, and upstream messages from client apps to servers.

Components

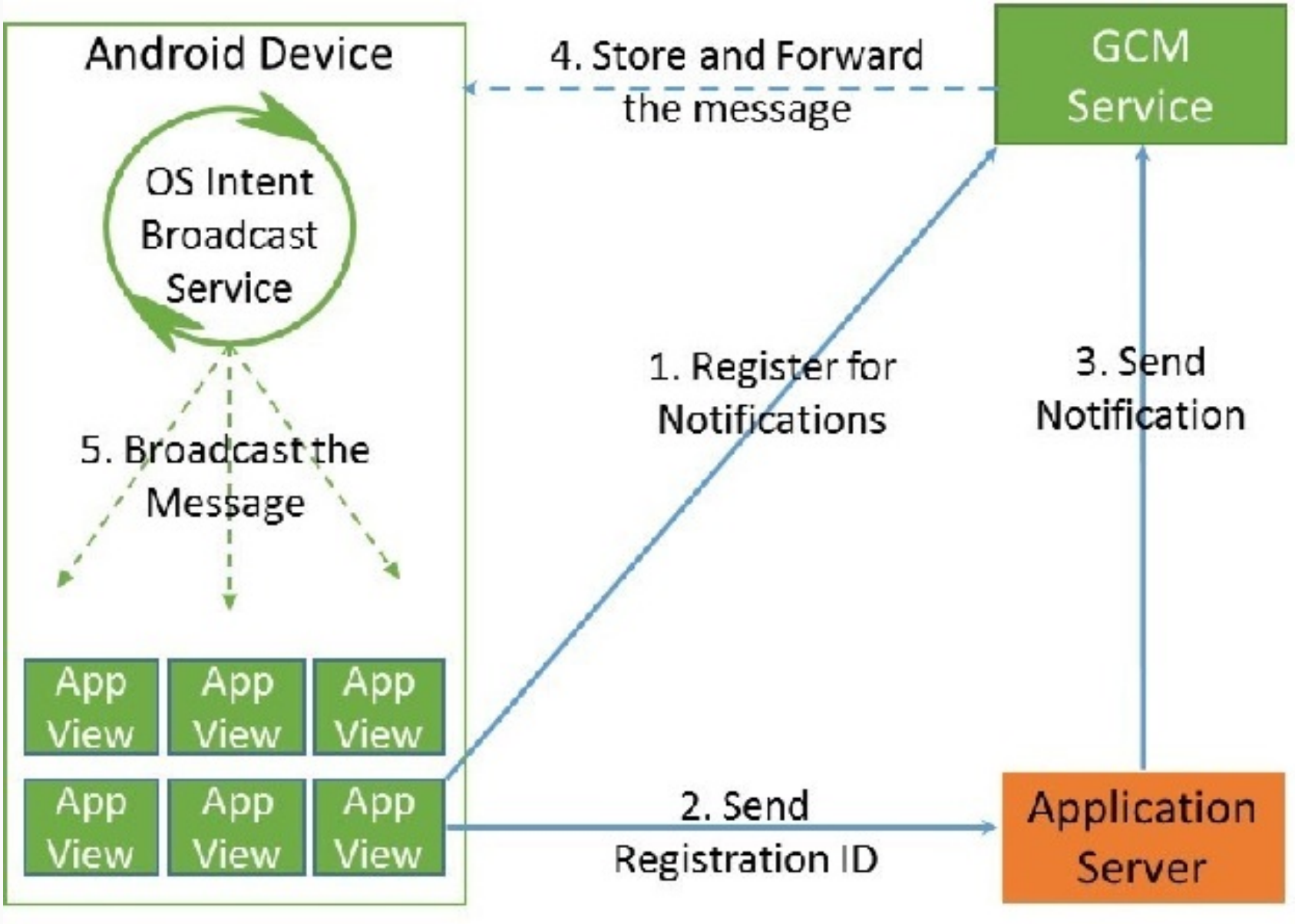
- GCM Connection Servers
- Client App
- App Server



Credentials

- **Sender ID**: The sender ID is used in the registration process to identify an app server that is permitted to send messages to the client app.
- **API Key**: An API key saved on the app server that gives the app server authorized access to Google services. In HTTP, the API key is included in the header of POST requests that send messages. In XMPP, the API key is used in the SASL PLAIN authentication request as a password to authenticate the connection
- **Application ID**: The client app that is registering to receive messages.
- **Registration Token**: An ID issued by the GCM connection servers to the client app that allows it to receive messages.

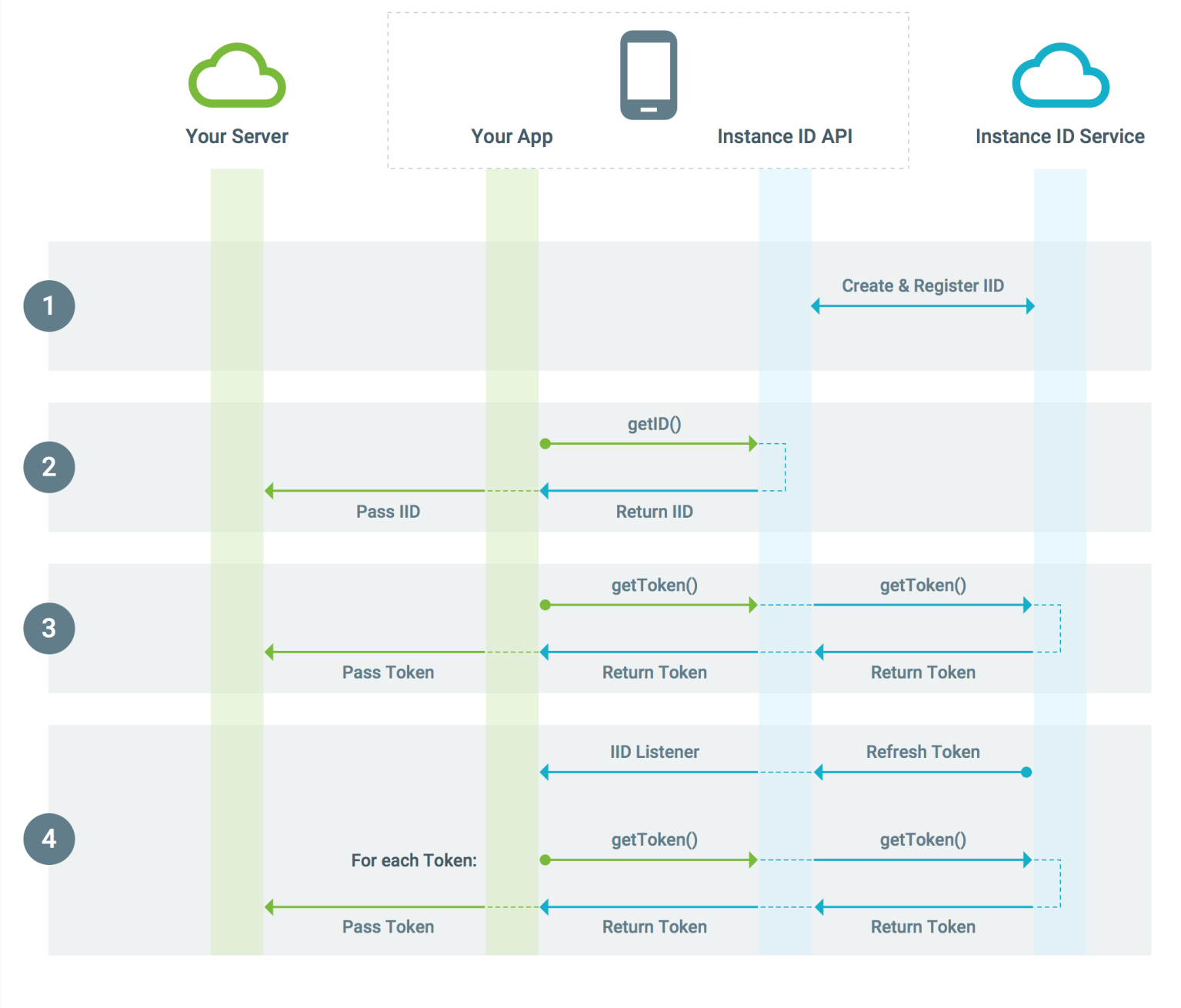
Steps


Android Device
OS Intent Broadcast Service
5. Broadcast the Message
App View
App View
App View
App View
App View
App View
4. Store and Forward the message
GCM Service
1. Register for Notifications
3. Send Notification
2. Send Registration ID
Application Server

Registering Client Apps

- The client app must obtain a registration token using the Instance ID API.
- Instance ID provides a unique ID per instance of your apps.
- Instance ID provides a simple API to generate security tokens that authorize third parties to access your app's server side managed resources.

Instance ID Lifecycle



Android Implementation

Get ID

```
String iid = InstanceID.getInstance(context).getId();
```

Generate Token

```
String authorizedEntity = PROJECT_ID; // Project id from Google Developer Console
String scope = "GCM"; // e.g. communicating using GCM, but you can use any
                      // URL-safe characters up to a maximum of 1000, or
                      // you can also leave it blank.
String token = InstanceID.getInstance(context).getToken(authorizedEntity,scope);
```

Refresh Token

```
public class MyInstanceIDService extends InstanceIDListenerService {
  public void onTokenRefresh() {
    refreshAllTokens();
  }

  private void refreshAllTokens() {
    // assuming you have defined TokenList as
    // some generalized store for your tokens
    ArrayList<TokenList> tokenList = TokensList.get();
    InstanceID iid = InstanceID.getInstance(this);
    for(tokenItem : tokenList) {
      tokenItem.token =
        iid.getToken(tokenItem.authorizedEntity,tokenItem.scope,tokenItem.options);
      // send this tokenItem.token to your server
    }
  }
};
```

Uninstalled Client App Unregistration

- The app server sends a message to GCM connection server addressed to an uninstalled instance.
- The GCM connection server sends the message to the GCM client on the device.
- The GCM client on the device receives the message and detects that the client app has been uninstalled.
- The GCM client on the device informs the GCM connection server that the client app was uninstalled.
- The GCM connection server marks the registration token for deletion.
- The app server sends a message to GCM for the same instance.
- The GCM returns a NotRegistered error message to the app server.
- The app server should delete the registration token.

GCM Connection Servers

The server side of GCM consists of two components:

- Connection Servers
- Application Servers

HTTP Server

To send a message, the application server issues a POST request.A message request is made of 2 parts: HTTP header and HTTP body. The header must contain authorization and content-type.

Example

```
Content-Type: application/json
Authorization: key=AIzaSyZ-1u...0GBYzPu7Udno5aA

{
  "to": "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1..."
  "data": {
    ...
  },
}
```

Request Format

- **Send To Sync:** This is the smallest possible message

```
{ "to" : "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...
```

- **Message with payload notification message:**

```
{ "notification": {
            "title": "Portugal vs. Denmark",
            "text": "5 to 1"
          },
  "to" : "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...
}
```

- **Message with payload Data message:**

```
{ "data": {
            "score": "5x1",
            "time": "15:10"
          },
  "to" : "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...
}
```

XMPP Server

The Google Cloud Messaging (GCM) Cloud Connection Server (CCS) is an XMPP endpoint that provides a persistent, asynchronous, bidirectional connection to Google servers. The connection can be used to send and receive messages between your server and your users' GCM-connected devices.

Request Format

- **Send To Sync:** This is the smallest possible message

```
<message id="">
  <gcm xmlns="google:mobile:data">
  {
        "to":"REGISTRATION_ID",
  }
  </gcm>
</message>
```

Request Format

- **Message with payload notification message:**

```
<message id="">
  <gcm xmlns="google:mobile:data">
  {
        "to":"REGISTRATION_ID",
      "notification": {
          'title': 'Portugal',
          'text': 'test'
        },
        "time_to_live":"600"
}
  </gcm>
</message>
```

Request Format

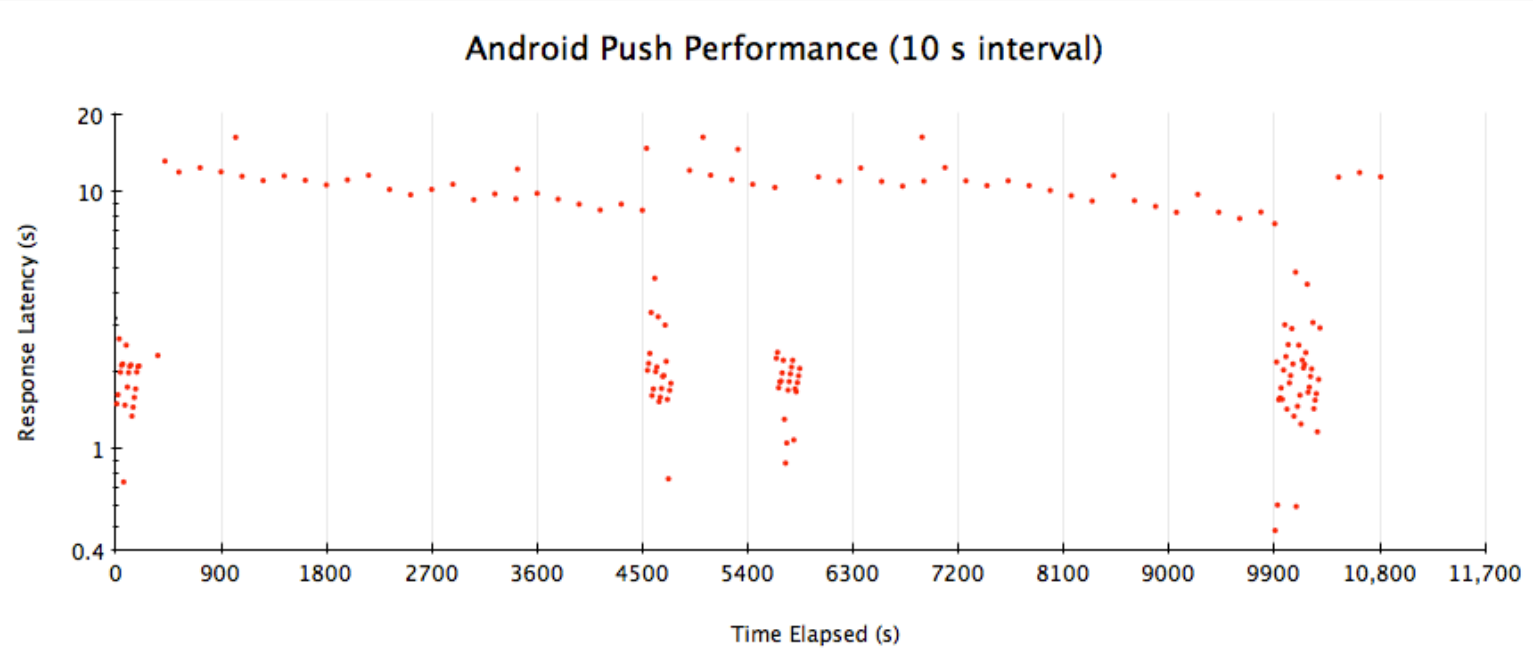
- **Message with payload Data message:**

```
<message id="">
  <gcm xmlns="google:mobile:data">
  {
       "to":"REGISTRATION_ID",
        "message_id":"m-1366082849205"
        "data":
        {
          "hello":"world",
        }
        "time_to_live":"600",
        "delay_while_idle": true/false,
        "delivery_receipt_requested": true/false
  }
  </gcm>
</message>
```

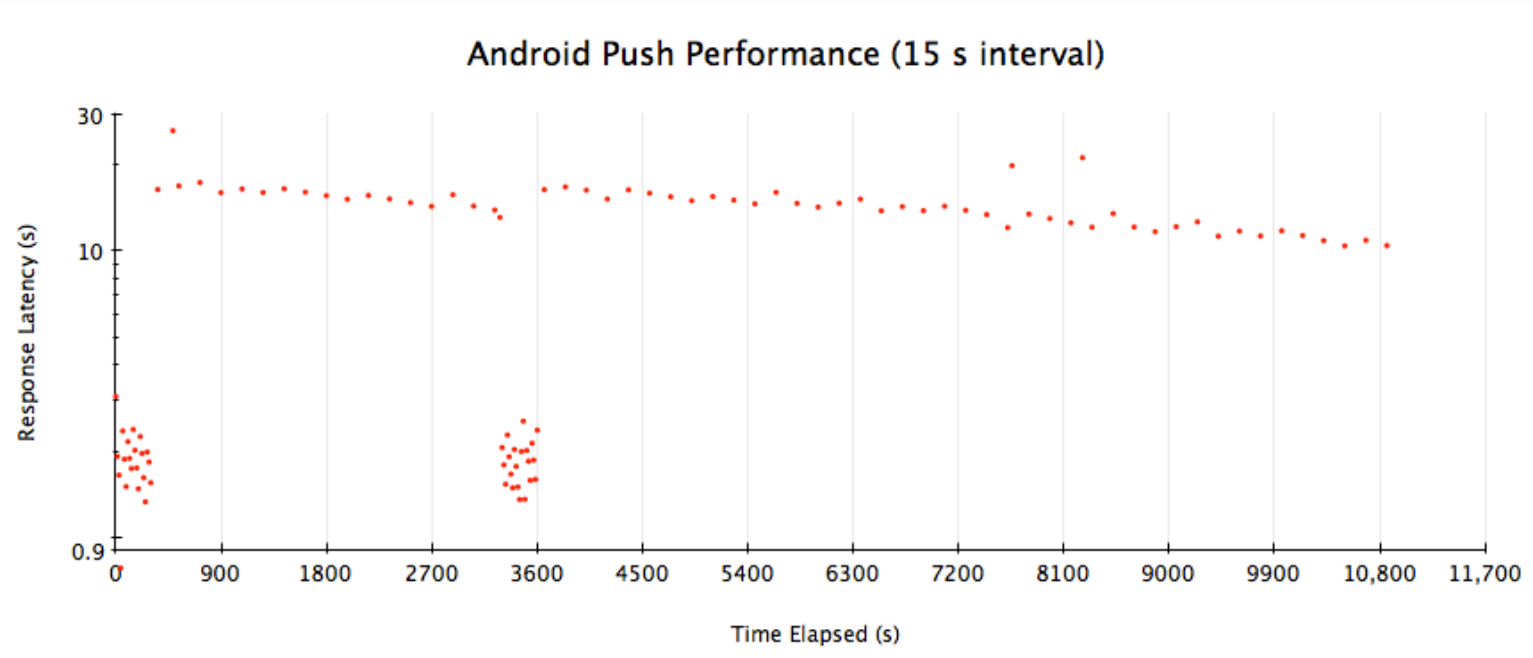
Performance Analysis

- In GCM, push notifications to the same device are throttled using token bucket scheme.
- The bucket holds 20 tokens and is replenished at a rate of once every 180 seconds.
- The bucket appears to be completely replenished (randomly) every 0 to 90 minutes.
- When the bucket is out of tokens, push notifications are dropped instead of queued.

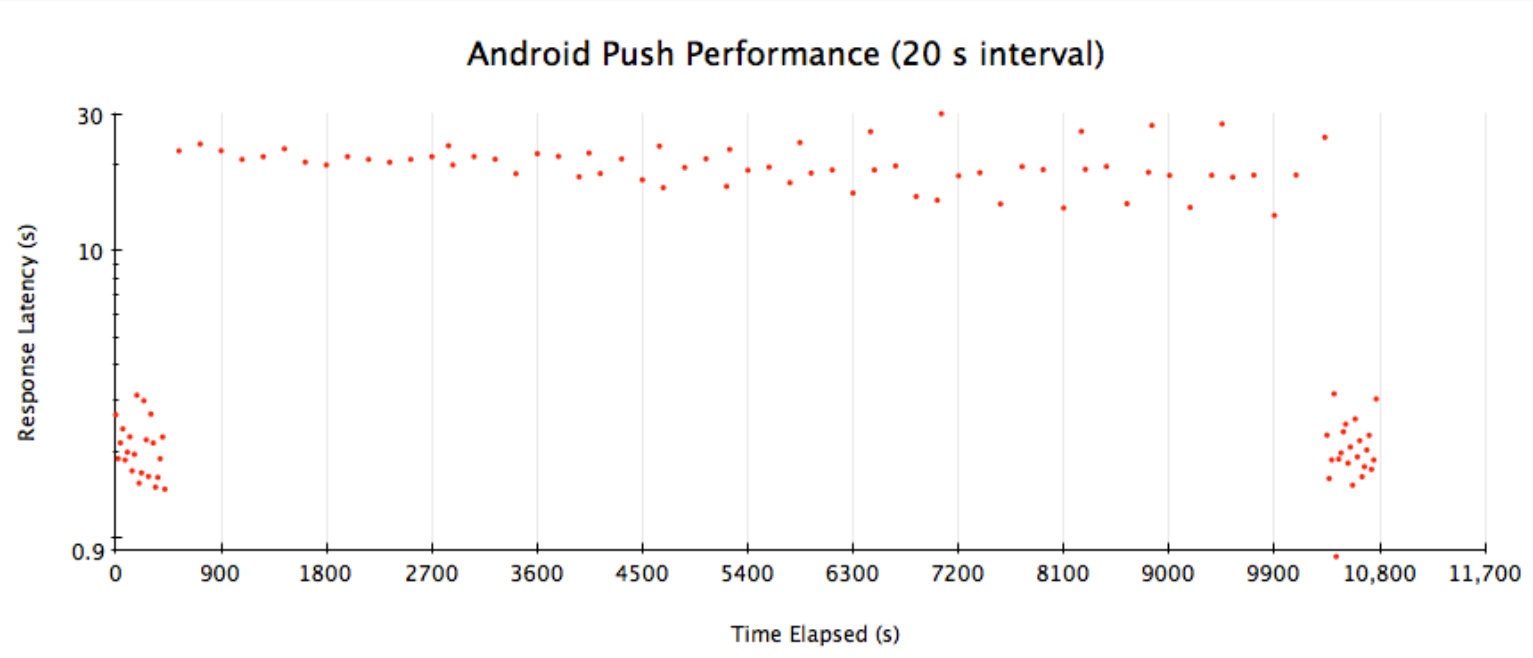
Performance Analysis



Performance Analysis

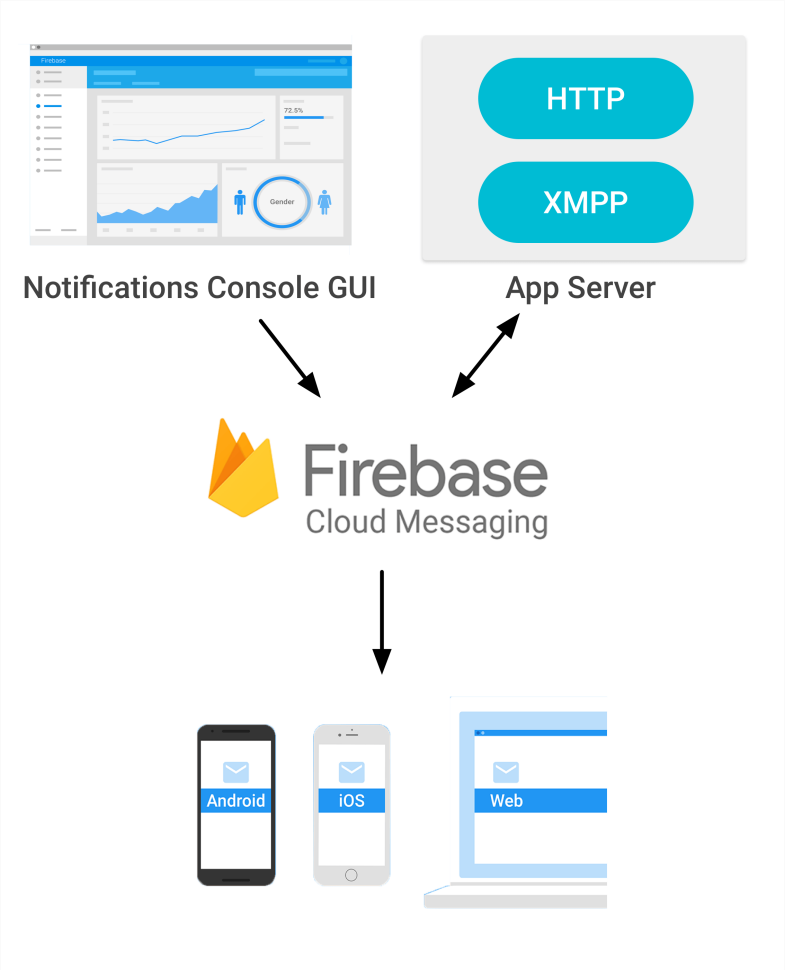


Performance Analysis



Firebase Cloud Messaging(FCM)

FCM architecture



FCM

- Firebase Cloud Platform inherits GCMs core infrastructure but simplifies the client development. Developers no longer needs to write their own registration or subscription retry logic. FCM includes a web console called Firebase Notifications which allows users to send notifications without an App server.
- Firebase covers all that mobile developers need to build, maintain and scale a mobile app on all platforms (even on iOS): from Storage and databases to innovative tools like Remote Config and Test Lab.
- Firebase also presents some performance improvements, it has an average response latency on 250ms and 95% delivery rate which is a huge improvement when compared to 10s and 40% on GCM.

Conclusion

Questions?

References I

Cloud computing: Wiki.
https://en.wikipedia.org/wiki/Cloud_computing.

Google cloud messaging.
https://developers.google.com/cloud-messaging/.

P. B.Dhivya, G.Lakshmiprabha.
Gcm service driven communication with an android application in cloud computing.
International Journal of Emerging Technology and Innovative Engineering, March 2015.

Q. Hassan.
Demystifying cloud computing.
The Journal of Defense Software Engineering, December 2014.

References II


G. G. Jarle Hansen, Tor-Morten Grnli.
Towards cloud to device push messaging on android: Technologies, possibilities and challenges.
Int. J. Communications, Network and System Sciences, May 2012.


P. M. K. Naresh Kumar N (M.Tech).
Gcm service driven communication with an android application in cloud computing.
International Journal of Engineering Research and Technology, May 2013.


A. M. D. Yavuz Selim, Yilmaz Bahadir Ismail.
Google cloud messaging (gcm): An evaluation.
Globecom 2014 - Symposium on Selected Areas in Communications: GC14 SAC Internet of Things.