Akademia Górniczo-Hutnicza

im. Stanisława Staszica w Krakowie

Wydział Informatyki, Elektroniki i Telekomunikacji

Alarm

Specyfikacja protokołów

Mateusz Gaweł

Jędrzej Dąbrowa

Mateusz Konieczny

Przemysław Kurc

Wykonał: Jędrzej Dąbrowa

Opis komunikacji i specyfikacja jej protokołów

W działaniu systemu wyróżniono 3 główne kanały komunikacji (implementowane w ramach prac nad projektem) i odpowiadający im protokoły oraz 1 protokół zewnętrzny, którego działanie jest ukryte i implementowane przez usługę zewnętrzną:

Protokoły główne:

- **REST API służące do komunikacji z serwerem, inicjowanej przez aplikacje klienckie**

  Protokołem komunikacji jest tutaj HTTP. Treść requestu HTTP to tekstowa reprezentacja formatu JSON, do którego serializowane są obiekty języka Java.
  Obiekty te zaimplementowane będą w formie POJO, a ich dystrybucja odbywać się będzie poprzez osobny plik JAR, współdzielony przez kod serwera oraz aplikacji klienckiej na system Android.

  Istotne jest, aby klient najpierw zarejestrował w serwerze token uzyskany w ramach Google ID - token ten zostanie przechowany w bazie danych wraz z danymi użytkownika i będzie służyć do późniejszego uwierzytelniania zapytań przychodzących do serwera. W każdym następnym zapytaniu do serwera, klient obowiązkowa załącza swój token. Jeśli token nie zostanie poprawnie zwalidowany po stronie serwera, zapytanie nie zostanie obsłużone i klient otrzyma informację o błędzie uwierzytelniania.

- **Komunikacja z bazą danych: protokół tcp, obsługiwany przez mechanizm JDBC**

  Serwer komunikuje się z bazą danych poprzez sterownik JDBC MongoDB, który jako protokół komunikacji wykorzystuje pojedynczy socket TCP. Według założeń odnośnie infrastruktury, baza danych oraz serwer aplikacji umieszczone będą na tym samym serwerze wirtualnym, jednak schemat komunikacji nie narzuca w tej kwestii ograniczeń - przeniesienie bazy na innego hosta fizycznego/wirtualnego wiązać się będzie jedynie z niewielką zmianą konfiguracyjną aplikacji serwera.

- **Komunikacja serwera aplikacji z serwerem dostępowym usługi Google Cloud Messaging**

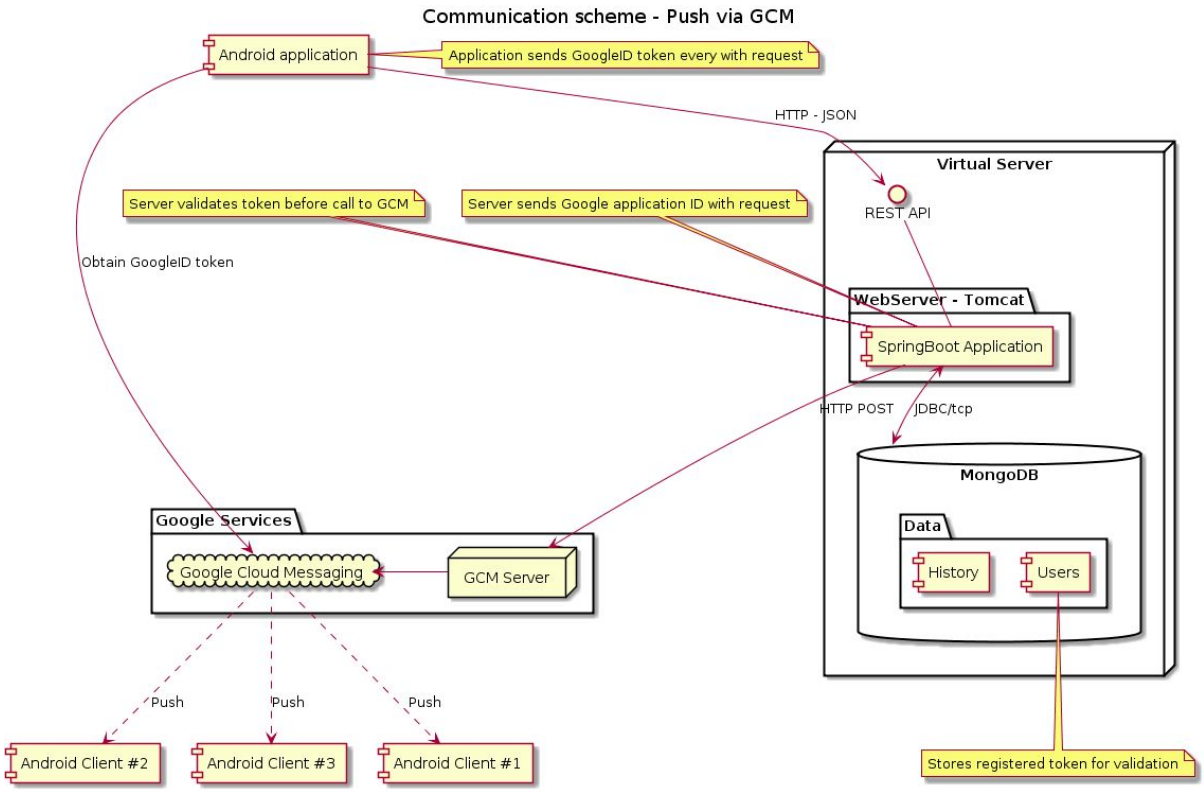
  Po wykonaniu operacji niezbędnych dla potwierdzenia tożsamości użytkownika chcącego wysłać wiadomość do innych, serwer zleca wysłanie wiadomości do aplikacji w systemie Android. Dokonuje tego poprzez request HTTP POST do serwera dostępowego GCM, który następnie przekazuje zlecenie dalej do chmury Google i dalej do użytkowników końcowych. Usługa GCM wymaga uwierzytelnienia serwera aplikacji - w tym celu serwer przekazuje swoją tożsamość Google Application ID, która tworzona jest w panelu administracyjnym usługi GCM i przechowywana na serwerze w ramach plików konfiguracyjnych. Po poprawnym uwierzytelnieniu serwera aplikacji, serwer GCM przyjmuje zlecenie przesłania wiadomości do aplikacji klienckich. Częścią tej wiadomości jest temat ("*topic*"), na który rejestrują się aplikacje. W ten sposób klienci otrzymują tylko powiadomienia w ramach interesujących ich grup.

Protokół zewnętrzny:

- ***Push-notifications*** **realizowane w ramach usługi Google Cloud Messaging oraz systemu Andoid**

  Ostateczne przekazanie wiadomości z serwerów chmury GCM do poszczególnych urządzeń Android odbywa się poza kontrolą twórców systemu Alarm i jest realizowane całkowicie przez protokoły firmy Google. Z perspektywy twórców aplikacji Alarm, wymagane jest jedynie utworzenie konta Google, uzyskanie w jego ramach tożsamości Google Application ID oraz poprawne użycie tej tożsamości w aplikacji serwera podczas korzystania z API GCM.

Całość komunikacji między komponentami systemu została przedstawiona na schemacie poniżej:


Communication scheme - Push via GCM
Android application
Application sends GoogleID token every with request
HTTP - JSON
Virtual Server
REST API
Server validates token before call to GCM
Server sends Google application ID with request
Obtain GoogleID token
WebServer - Tomcat
SpringBoot Application
HTTP POST
JDBC/tcp
MongoDB
Data
History
Users
Google Services
Google Cloud Messaging
GCM Server
Push
Push
Push
Android Client #2
Android Client #3
Android Client #1
Stores registered token for validation