Azure Mobile App Service & Xamarin

Ge.Mobi 19 Giugno 2018

Giampaolo TUCCI

www.trilogik.it

Blog: https://www.informaticapressapochista.com/it/

Agenda & Livello

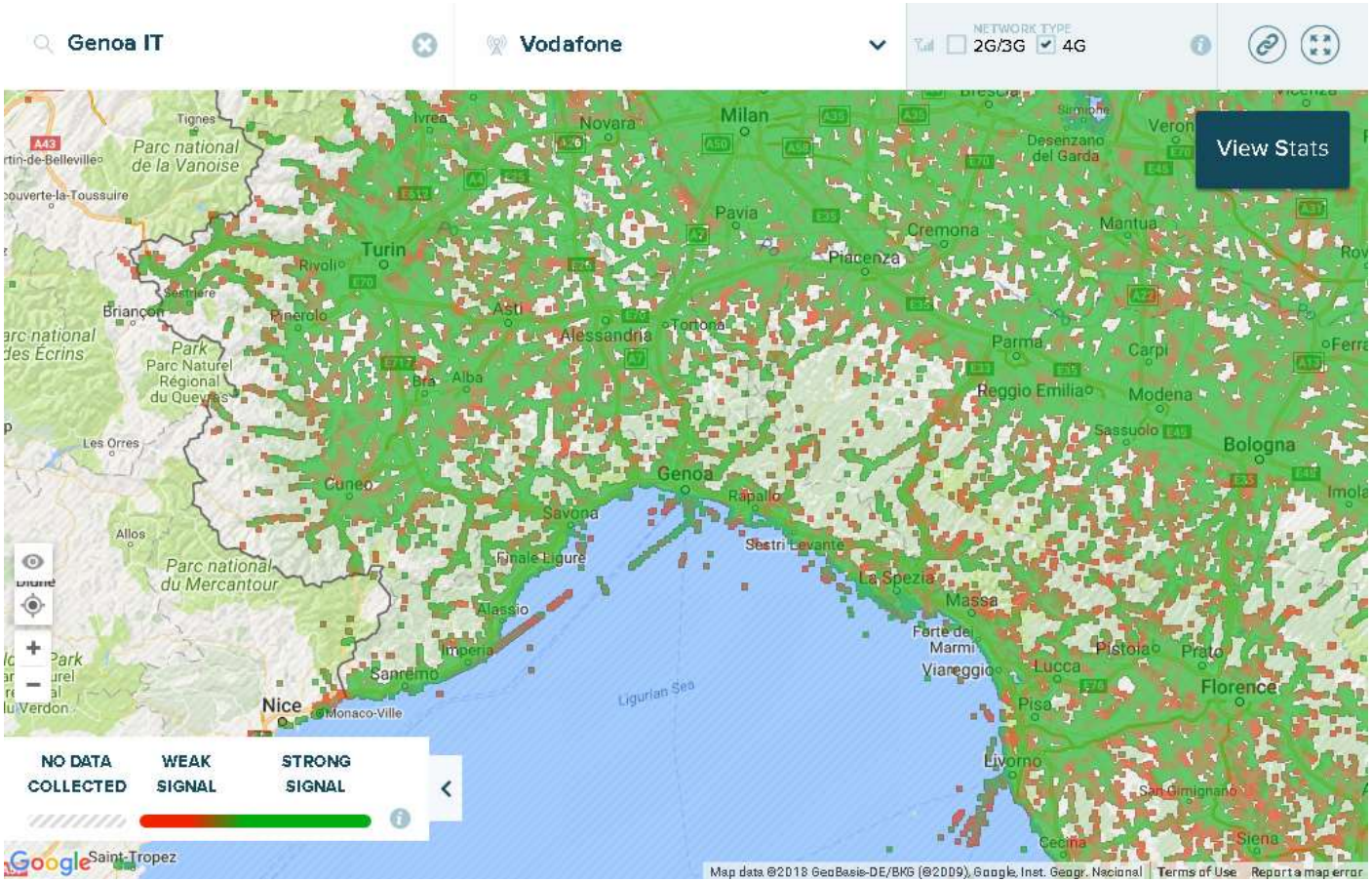
Cosa vedremo in questo intervento ??

Utilizzo dei servizi Azure Mobile App Service per creare una App dotata di:

- Offline sync (cioè sincronia di una database Sqlite locale che equipaggia il dispositivo mobile con un database Sql Server, o una parte di questo)
- Autenticazione con un provider esterno

Il livello dell'intervento è Entry-Level (semplice introduzione sull'argomento).

Perchè offline sync ??



Se una app enterprise-grade ha la necessità continua di interagire con un set di dati conservati remotamente (es.: Sql Server) è impensabile lavorare esclusivamente in modalità connessa.

Problemi

- Copertura rete 4G non totale
- Situazioni geografiche sfavorevoli (palazzi, muri, etc)
- Contratto SIM che può limitare il traffico una volta superate determinate soglie quindi occorre limitare il traffico generato e gestire i casi di mancanza di connettività

Fonte: opensignal.com

Perchè usare un’autenticazione tramite provider esterno ??

- Per evitare che l’utilizzatore debba inventarsi e gestire ulteriori account (userid e password)
- Per evitare che il servizio di assistenza del software debba gestire problemi di autenticazione (es: password persa)
- Maggiore sicurezza
- In questo momento è molto “cool”

Perchè usare Azure Mobile App Service + SDK ?

Azure Mobile App Service permette di implementare un backend ad alta resilienza e basso costo di setup e gestione.

Inoltre il servizio dispone di un SDK dedicato, *Azure Mobile SDK*, che facilita l'utilizzo dei suoi servizi.

L'accoppiata dei due permette di implementare, in modo facile, una app con le seguenti caratteristiche.

- Accesso ai dati via REST/Json in modo trasparente
- Offline sync (sincronia di una database Sqlite locale con un database Sql Server, o sezionamento di questo)
- Authentication via OAuth e l'ausilio di diversi IdP (Facebook, Google, Twitter, MSA Microsoft Account Identity)
- Push notification
- ……

Perchè usare Azure Mobile App Service + SDK ?

Tutto quello che è possible implementare con gli Azure Mobile App Service + SDK si può ottenere anche utilizzando le usuali Web API interrogate lato client con una implementazione di HttpClient.

La differenza è che usando gli Azure-Amiconi (Azure + SDK) lo sviluppo e il deploy della soluzione è più veloce ed efficace !

Come usare gli Azure App Service ?

L'SDK Mobile App Service, da utilizzarsi lato client, è disponibile oltre che per .Net Standard (Xamarin, etc.) anche per Android, IoS e librerie Javascript.

Il backend ospitato può essere scritto con NodeJS (Javascript) o Asp.net (C#).

Nel seguito si userà Xamarin .Net Standard e il backend scritto in .Net, ma tutti gli ambienti indicate sopra offrono le medesime funzionalità.

L’esempio: Focac-Book

Per descrivere il funzionamento degli Azure App Service esporrò i punti principali nell’ambito dell’implementazione della app “Focac-Book” che permetterà condividere in modo social i posti migliore dove si mangia la focaccia genovese, assegnando ad essi anche un voto !

I dati saranno salvati su un database Sql Server posto su Azure.

Demo

Creare servizio Azure App Service + Sql Server

Azure “Quick Start”: cenni su Quick Start, Application settings, Logging

Tabella Sql Server

Nome Campo	Note
NomeUtente	Varchar(50) L'utente segnalatore
DataOra	Datetime Data e ora rilevamento
Luogo	Varchar(200) Luogo in cui si è gustata la focaccia
Voto	Int Un voto da 1 a 5 per meglio descrivere la propria esperienza di assaggio

Focac-Book – Prima revisione senza offline-sync e nessuna autorizzazione/autenticazione

Modello backend

```
public class FocaccePost : EntityData
{
    public string NomeUtente { get; set; }
    public string Luogo { get; set; }
    public DateTime DataOra { get; set; }
    public int Voto { get; set; }
}

public abstract class EntityData : ITableData
{
    public string Id { get; set; }
    public byte[] Version { get; set; }
    public DateTimeOffset? CreatedAt { get; set; }
    public DateTimeOffset? UpdatedAt { get; set; }
    public bool Deleted { get; set; }
}
```

Slide 11

GT1 Giampaolo Tucci; 29/05/2018

Dopo prove e modifiche….. Struttura tabella finale Sql Server

Nome Campo	Note
Id	Chiave (Azure) che deve essere una Guid
Version	Timestamp, cioè revisione riga (Azure) Per gestione concorrenza
CreatedAt	Data creazione riga (Azure) Per ottimizzare performance accesso alla tabella (partizionamento orizzontale contenuti)
UpdatetAt	Ultima data modifica della riga (Azure) Ottimizzazione performance sync offline
NomeUtente	L'utente segnalatore
DataOra	Data e ora rilevamento
Luogo	Luogo in cui si è gustata la focaccia
Voto	Un voto da 1 a 5 per meglio descrivere la propria esperienza di assaggio

Demo

Prima revisione app senza offline-sync (online-sync) e senza autenticazione

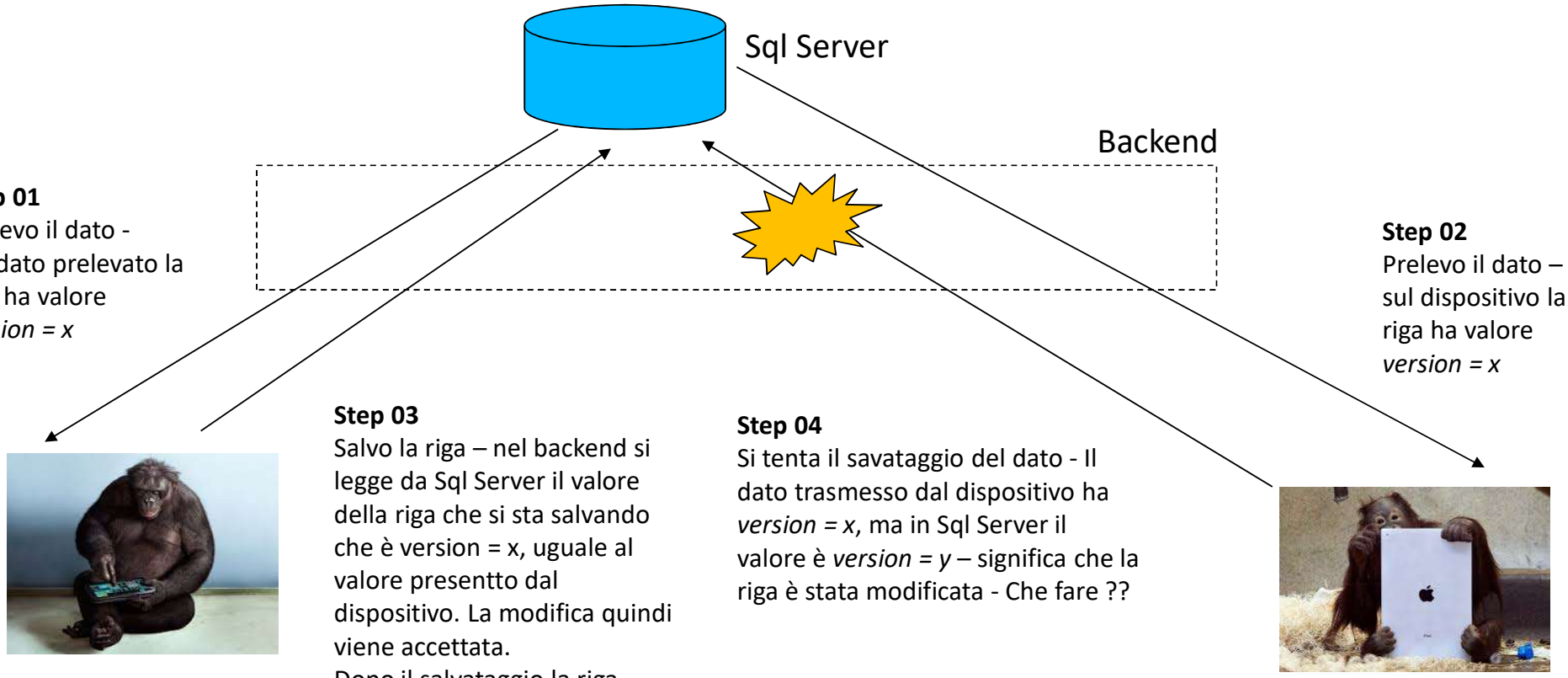
Deploy App Service + Test + Debug

Perchè non possiamo essere amici ?
Gestione della concorrenza

Slide 14

GT1 Giampaolo Tucci; 29/05/2018

Gestione della concorrenza (Optimistic Concurrency)



Slide 15

GT1 Giampaolo Tucci; 29/05/2018

…and the winner is….. la gestione concorrenza lato client

La gestione della concorrenza può essere facilemente implementata lato server, ma spesso è necessario presentare all'utilizzatore la scelta di quale versione debba vincere, cioè quale sia la modifica vincente.

L'utilizzo dell'SDK Mobile App Service permette di ottenere, lato client, la versione del server che provoca l'incompatibilità tramite l'utilizzo di un errore *412-Precondition Failed*.

Lato client è possibile ad entrambe le versioni (quella in Sql Server e quella in salvataggio) ed è possible decidere quale debba vincere.

Demo

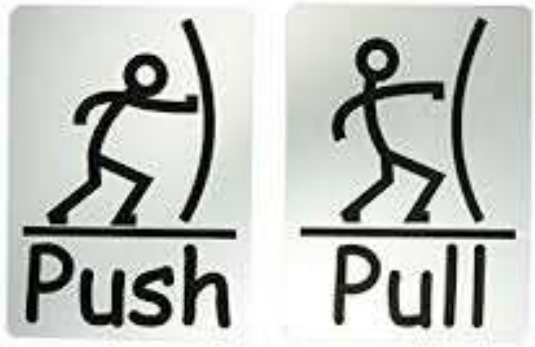
Modifiche concorrenti

Offline-sync

La sincronizzazione offline permette di scaricare i dati (o parte di essi) sulle tabelli locali (operazione detta di pull), e di eseguire su questa modifiche CUD, anche in assenza di connettività.

Solo all'avvio della procedura di sincronizzazione (push) le modifiche saranno riportate sul dabase Sql Server nel backend.

Push & Pull



Alcune caratteristiche offerte dall'SDK Mobile App.

- L'ordine con cui avvengono le modifiche sul database locale è lo stesso con cui vengono riportate le modifiche sul backend: questo per salvaguardare l'integrità e la semantica dei dati.
- L'operazione di pull è sempre incrementale (si basa su UpdateAt) e paginate (di default 50 tuple alla volta) per ottimizzare l'utilizzo della banda e I tempi di attesa.
- Push implicito

..l'annoso problema dei record cancellati..

Ovvero: Come è possible cancellare un item in una serie arbitraria di dispositivi se tutti questi operano in offline sync ??

Infatti se un record viene cancellato per mano di un client questo non sarà più presente lato backend. Gli altri dispositive come fanno a sapere che è stato cancellato ?

Semplice: Soft Delete !!

Demo

Offline-sync

Soft-Delete

Perchè non possiamo essere amici (parte 2) ?

Nel caso dell'offline sync le possibilità di clash causati da modifiche concorrenti aumentano.

Possibilità offerte dall'SDK.

CancelAndUpdateItemAsync
Cancella l'aggiornamento in corso (client->backend) e aggiorna il record locale con quanto passato come argomento.
CancelAndDiscardItemAsync
Cancella l'aggiornamento in corso.
UpdateOperationAsync
Aggiorna il record locale con quanto passato come argomento e aggiunge le eventuali modifiche eseguite alla tabella CUD: quindi al prossimo Push verranno riproposte le operazioni qui eseguite sul record.

Slide 22

GT1 Giampaolo Tucci; 29/05/2018

Demo

Concorrenza

Autenticazione vs Autorizzazione

L’autenticazione è il processo che permette di identificare l’utilizzatore in modo sicuro.

Questo processo nel passato veniva svolto tramite l’ausilio di una userid e una password: il concetto che sottende questa attività è che solo chi conosce la relativa password è in grado di sostenere di essere realmente l’utente specificato.

Ora i sistemi più moderni dovrebbero supportare un processo diverso, chiamato OAuth, che utilizza attori esterni nel processo di autenticazione.

Il termine autorizzazione indica invece in generale il permesso o la facoltà di compiere determinate azioni: nel nostro caso che tipo di dati posso scaricare come offline sync e posso modificare.

Slide 24

GT1 Giampaolo Tucci; 29/05/2018

Autenticazione OAuth: server o client flow ??

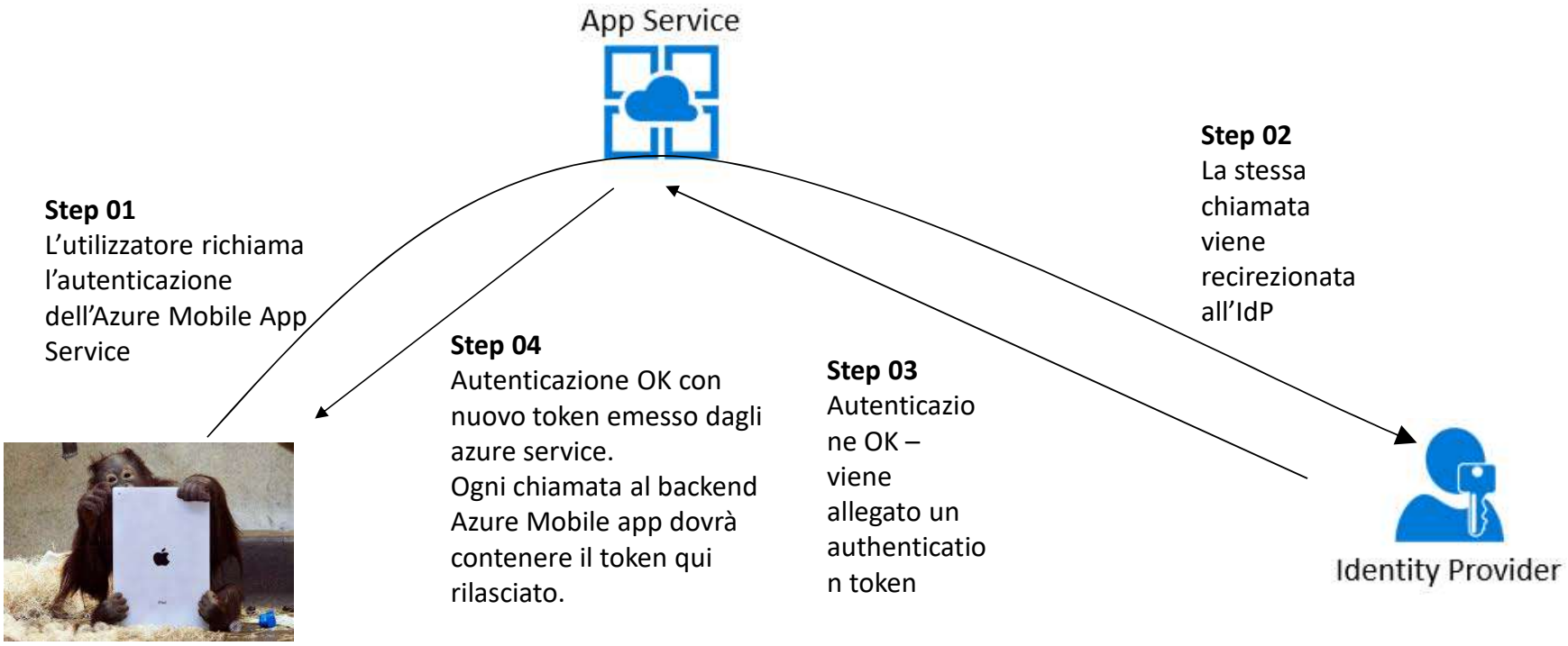
In un’autenticazione OAuth ci sono tre attori che vengono coinvolti.

- Il client, che è l’applicazione che deve accedere alle risorse (la nostra App).
- La risorse, che offrono le loro funzionalità a utenze autenticate (mobile backend).
- L’identity provider (IdP) che è il servizio responsabile dell’autenticazione dell’utilizzatore del client.

Slide 25

GT1 Giampaolo Tucci; 29/05/2018

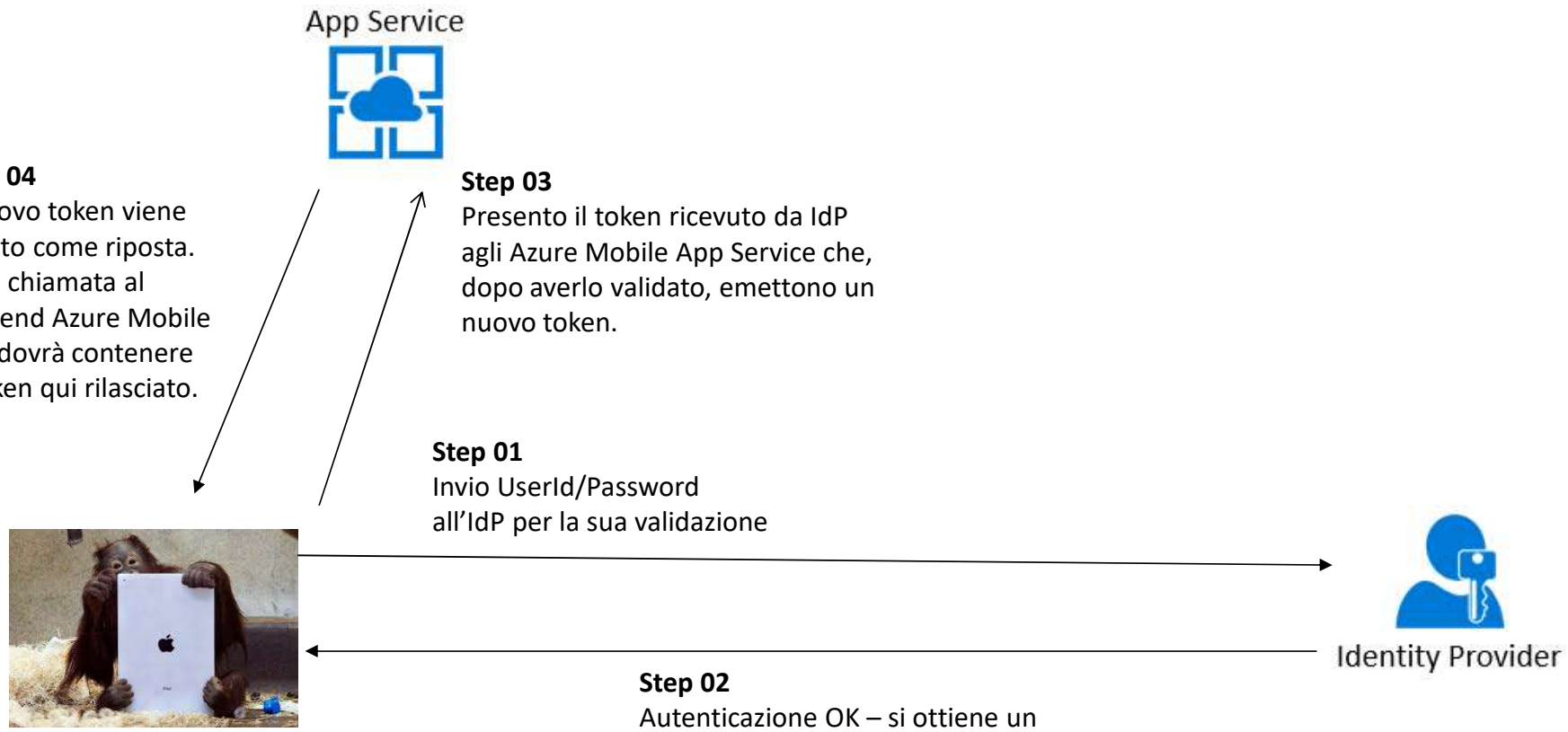
Server-Flow Authentication



Slide 26

GT1 Giampaolo Tucci; 29/05/2018

Client-Flow Authentication



Slide 27

GT1 Giampaolo Tucci; 29/05/2018

Server-Flow: che Identity Provider usare ?

Nativamente Azure Mobile App Service supporta i seguenti.

- Azure Active Directory
- Facebook
- Google
- Twitter
- Microsoft

Usando uno di questi IdP aggiungere l'autenticazione a una app è molto veloce.
Esiste però un'ulteriore possibilità: la custom authentication.

Con questa modalità nel codice del backend è possibile «coniare» un token come se fosse generato a seguito di autenticazione con IdP.

In questo modo è possibile usare qualsiasi IdP, implementando il reltivo accesso nel backend, o anche scrivere autonomamente nuovi metodi di autenticazione.

Qui nel seguito useremo Facebook.

Slide 28

GT1 Giampaolo Tucci; 29/05/2018

Server-Flow: Task List

Se si vuole usare un IdP supportato il processo di configurazione è praticamente uguale per tutti.

1. Accedere alla console developer dell'IdP scelto
2. Creare una nuova App e abilitare l'utilizzo dell'autenticazione Oauth come web application)
3. Configurare la parte impostando l'URL del redirect del mobile app service e quindi prelevare gli identificativi proposti (*App Id* e *App Secret*) che saranno usati da Azure per verificare la correttezza dei dati ricevuti dall'IdP.
4. Accedere al portale Azure
5. Impostare i dati ottenuti dall'IdP nel Mobile App service.
6. Proteggere i controller del backend da accessi non autenticati (attributo [Authorize])

L'URL del redirect sarà sempre nella seguente forma.
<URL Azure Mobile App>/.auth/login/<Provider scelto>/callback

Esempio
https://focac-book.azurewebsites.net/.auth/login/facebook/callback

Slide 29

GT1 Giampaolo Tucci; 29/05/2018

Demo

Configurare l'IdP e Azure Mobile App

Autenticazione Server-Flow

Server-Flow: Downsides

- Per il processo di autenticazione si usa una WebView (nelle versioni più recenti *Custom Chrome Tab*), che quindi può offrire una scarsa esperienza di utilizzo
- L'autenticazione server-flow nell'SDK Azure è stato recentemente profondamente modificato per soddisfare nuovi requisiti di sicurezza: potrebbe succedere la stessa cosa nel futuro per analoghe esigenze.
- Difficoltà nel gestire il logout: possono rimanere nel browser del dispositivo cookie o password memorizzate che forzano l'autenticazione anche se si è eseguito esplicitamente il logout
- Se nel nostro dispositivo abbiamo per esempio già la app dell'IdP non è possibile riutilizzare l'accesso già ottenuto per questa.

Slide 31

GT1 Giampaolo Tucci; 29/05/2018

Client-Flow: Task List

Anche in questo caso se si vuole usare come IdP uno tra quelli proposti nativamente dagli Azure Mobile App il processo di configurazione è praticamente uguale per tutti.

1. Accedere alla console developer dell'IdP scelto
2. Creare una nuova App e abilitare l'autenticazione Oauth
3. Inserire i dati della app che vuole utilizzare l'SDK native e la relativa autenticazione: generalmente il package name e l'hash key della firma utilizzata
4. Prelevare i dati identificativi e la secret proposti dal portale (*App Id* e *App Secret*).
5. Accedere al portale Azure e impostare app id e secret nel Mobile App service.
6. Proteggere i controller del backend da accessi non autenticati (attributo [Authorize])

Slide 32

GT1 Giampaolo Tucci; 29/05/2018

Demo

Autenticazione Client-Flow

Client-Flow: Downsides

- Occorre studiare un nuovo SDK
- Occorre avere l'SDK disponibile per il client che si sta creando
- In generale la soluzione è più difficile da implementare ma anche quella che permette una migliore esperienza utente
- Generalmente interazione con account già inseriti nella app dell'IdP se installata sul dispositivo (per esempio: se si è installato il facebook app allora si può riusare l'account e non dover reinserire userid/password)

Slide 34

GT1 Giampaolo Tucci; 29/05/2018

Ehmm….. Ma io chi sono ??

Semplice: Chiedilo a /.auth/me !!!

L’autenticazione avviene con l’asusilio di un SDK (Azure Mobile o native dell’IdP) e alla fine si ottiene solo un token rilasciato da Azure.

Per tale motivo non si ha l’evidenza dello userid utilizzato per acceder all’IdP.

Esempio: Nel caso più semplice la mail usata per accedere all’IdP.

Come fare ??

Slide 35

GT1 Giampaolo Tucci; 29/05/2018

Demo

Chi sono io ?? (lato client)

Chi sono io ?? (lato server)

Autorizzazione: quando la mail non è sufficiente.....