Microsoft

Reactor

STOCKHOLM

This session will commence shortly

We are constantly striving to create excellent content
and would appreciate if you could take this brief survey.

Survey Link: https://aka.ms/Reactor/Survey

Please enter the event code 12455 at the start of survey

Speaker Slide:

Massimo Bonanni

Azure Technical Trainer @ Microsoft

I spend my time to help customers to empower their Azure skills to achieve more and leverage the power of Azure in their solutions.
I'm also a technical speaker both for local and international events and a user-group guy.
I founded Aa couple of communities in Italy and collaborated with most of the Italian communities.
Finally, I is also passionate about biking, reading, and dogs!!

Centralize your configurations with Azure App Configuration .. and not only!!

Massimo Bonanni

Azure Technical Trainer @ Microsoft

meetup.com/Microsoft-Reactor-Stockholm/

Our Code of Conduct

Microsoft is dedicated to empowering every person and every organization on the planet to achieve more.

This includes Microsoft Reactor events where we seek to provide a respectful, friendly, professional experience for everyone, regardless of gender, sexual orientation, physical appearance, disability, age, race or religion.

We do not tolerate any behaviour that is harassing or degrading to any individual, in any form. Individuals are responsible for knowing and abiding by these standards. We encourage everyone to assist in creating a welcoming and safe environment.

Be aware of others

Be friendly and patient

Be welcoming and respectful

Be open to all questions and viewpoints

Be understanding of differences

Be kind and considerate to others

Azure App Configuration provides a service to centrally manage application settings and feature flag

App Configuration Key features

 Key-Value store	• Stores configuration data as key-value pairs
 Point-in-time snapshot	• Maintains a record of changes made to key-value pairs • You can reconstruct the history of any key-value within the previous seven days
 Feature management	• Decouples feature release from code deployment • Enables quick changes to feature availability on demand AKA "feature flags"
 Security	• Encrypt using customer-managed keys • Using private endpoints • Integrate with Azure Managed Identity and Azure KeyVault

App Configuration benefits

A fully managed service that can be set up in minutes

Flexible key representations and mappings

Tagging with labels

Point-in-time replay of settings

Dedicated UI for feature flag management

Comparison of two sets of configurations on custom-defined dimensions

Enhanced security through Azure-managed identities

Encryption of sensitive information at rest and in transit

Native integration with popular frameworks

Point-in-time snapshot

Azure App Configuration keeps records of the precise times when a new key-value pair is created and then modified.

These records form a complete timeline in key-value changes.

An App Configuration store can reconstruct the history of any key value and replay its past value at any given moment, up to the present.

With this feature, you can “time-travel” backward and retrieve an old key value.

App Configuration provider for ASP.NET Core

App Configuration's .NET Client is implemented as configuration provider.

1) Use the package:

```
dotnet add package Microsoft.Azure.AppConfiguration.AspNetCore
```

2) Configure the provider like any other configuration providers:

```
public static IHostBuilder CreateHostBuilder(string[] args) =>
    Host.CreateDefaultBuilder(args)
        .ConfigureWebHostDefaults(webBuilder =>
            webBuilder.ConfigureAppConfiguration(config =>
            {
                var settings = config.Build();
                var connection = settings.GetConnectionString("AppConfig");
                config.AddAzureAppConfiguration(connection);
            }).UseStartup<Startup>());
```

How much?

1 free instance per subscription

1,02€ per day

0,051€ per each 10.000 requests

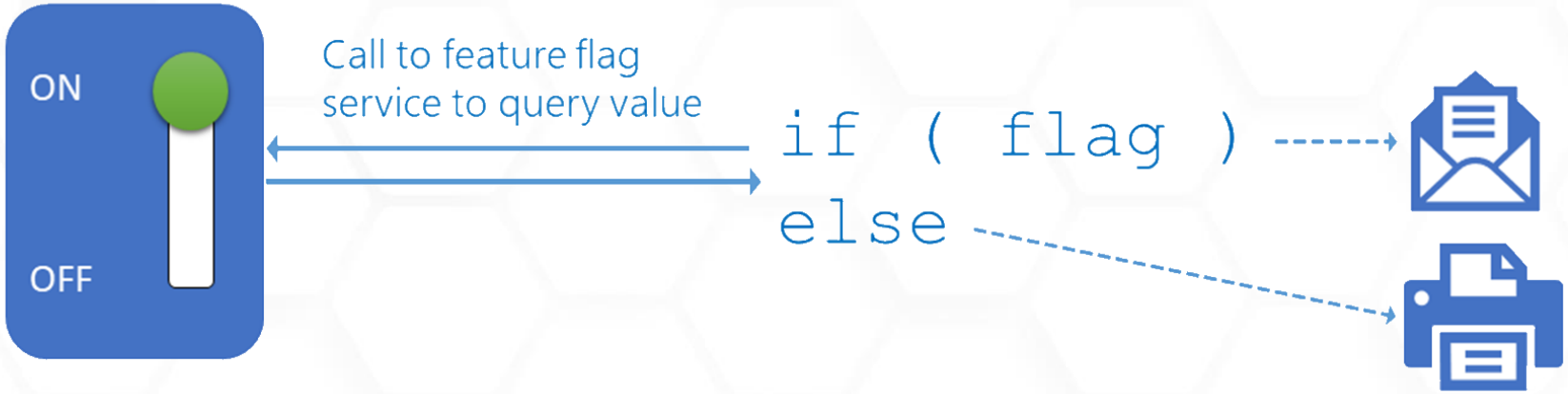
200.000 requests per day in daily charge

Photo by Markus Spiske on Unsplash

Web Site Configuration

What are Feature Flags (FF)?

Feature management is a modern software-development practice that decouples feature release from code deployment and enables quick changes to feature availability on demand.



View a feature flag as an ON | OFF switch for a specific feature.
You can deploy a solution to production that includes both an email and a print feature.
If the feature flag is set (ON), you'll email, else you'll print.

Feature Management – Basic Concepts

Feature flag	A feature flag is a variable with a binary state of *on* or *off*. The feature flag also has an associated code block. The feature flag's state triggers whether the code block runs.
Feature manager	A feature manager is an application package that handles the life cycle of all the feature flags in an application. The feature manager also provides additional functionality, including caching feature flags and updating their states.
Filter	A filter is a rule for evaluating the state of a feature flag. Potential filters include user groups, device or browser types, geographic locations, and time windows.

Enable feature flags in an ASP.NET Core app

The .NET Core Feature Management libraries provide a great support for ASP.NET application.

1) Add reference to **`Microsoft.FeatureManagement.AspNetCore`** and **`Microsoft.FeatureManagement`** packages.

2) Enable Feature Management in the startup phase:

```
using Microsoft.FeatureManagement;

public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddFeatureManagement();
    }
}
```

Use feature flags in an ASP.NET Core app

The .NET Core Feature Management libraries inject the `IFeatureManager` in your controller:

```
public class HomeController : Controller
{
    private readonly IFeatureManager _featureManager;

    public HomeController(IFeatureManager featureManager)
    {
        _featureManager = featureManager;
    }
}
```

You can use `IFeatureManager` to check if a block of code must be execute or not:

```
IFeatureManager featureManager;
...
if (await featureManager.IsEnabledAsync(nameof(MyFeatureFlags.FeatureA)))
{
    // Run the following code
}
```

Use feature flags in an ASP.NET Core app

The .NET Core Feature Management libraries also provide action/controller filters and TagHelpers.

```
using Microsoft.FeatureManagement.Mvc;

[FeatureGate(MyFeatureFlags.FeatureA)]
public class HomeController : Controller
{
    ...
}
```

```
@addTagHelper *, Microsoft.FeatureManagement.AspNetCore

<feature name="FeatureA">
    <p>This can only be seen if 'FeatureA' is enabled.</p>
</feature>
```

```
using Microsoft.FeatureManagement.Mvc;

[FeatureGate(MyFeatureFlags.FeatureA)]
public IActionResult Index()
{
    return View();
}
```

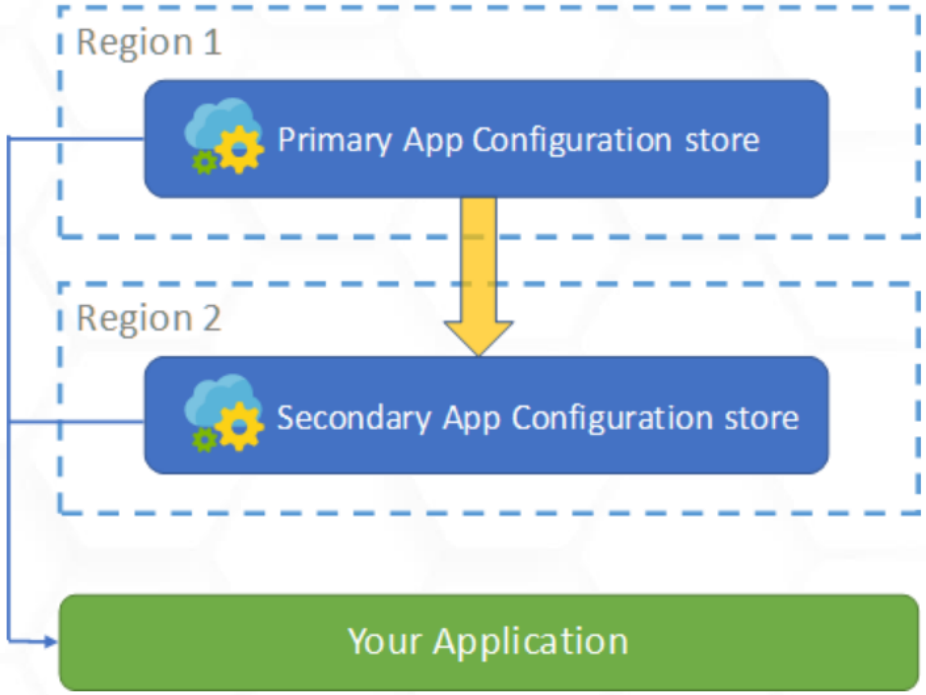
Feature Flags

Resiliency and disaster recovery

Azure App Configuration is a regional service.

Now, you don't have any automation for replication between different regions.

Your application loads its configuration from both the primary and secondary stores.



App Configuration events

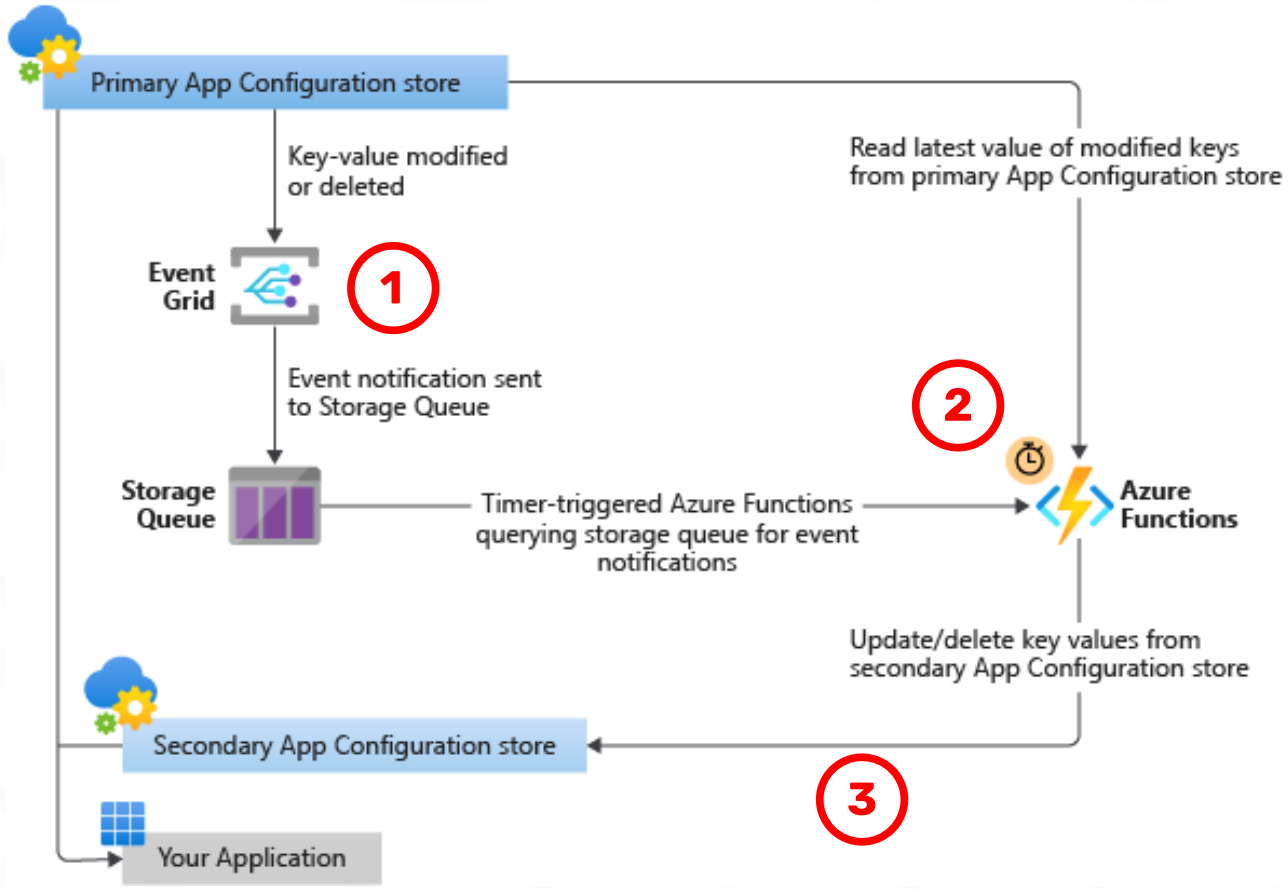
Azure App Configuration emits the following event types:

Event type	Description
Microsoft.AppConfiguration.KeyValueModified	Raised when a key-value is created or replaced.
Microsoft.AppConfiguration.KeyValueDeleted	Raised when a key-value is deleted.


```
{
  "id": "46bd183f-7335-40eb-ab3a-8e396483e7a0",
  "topic": "/subscriptions/60504780-44c1-4d01-b84f-baca1d9239ed/resourceGroups/…./primaryappconfig",
  "subject": "https://primaryappconfig.azconfig.io/kv/myConfigValue?label=%00&api-version=1.0",
  "data": {
    "key": "myConfigValue",
    "label" : "myLabel",
    "etag": "1NQRz8iYl4jZKR46Zy3sDml7Vy5"
  },
  "eventType": "Microsoft.AppConfiguration.KeyValueModified",
  "dataVersion": "1",
  "metadataVersion": "1",
  "eventTime": "2020-12-07T13:01:01.3961595Z"
}
```

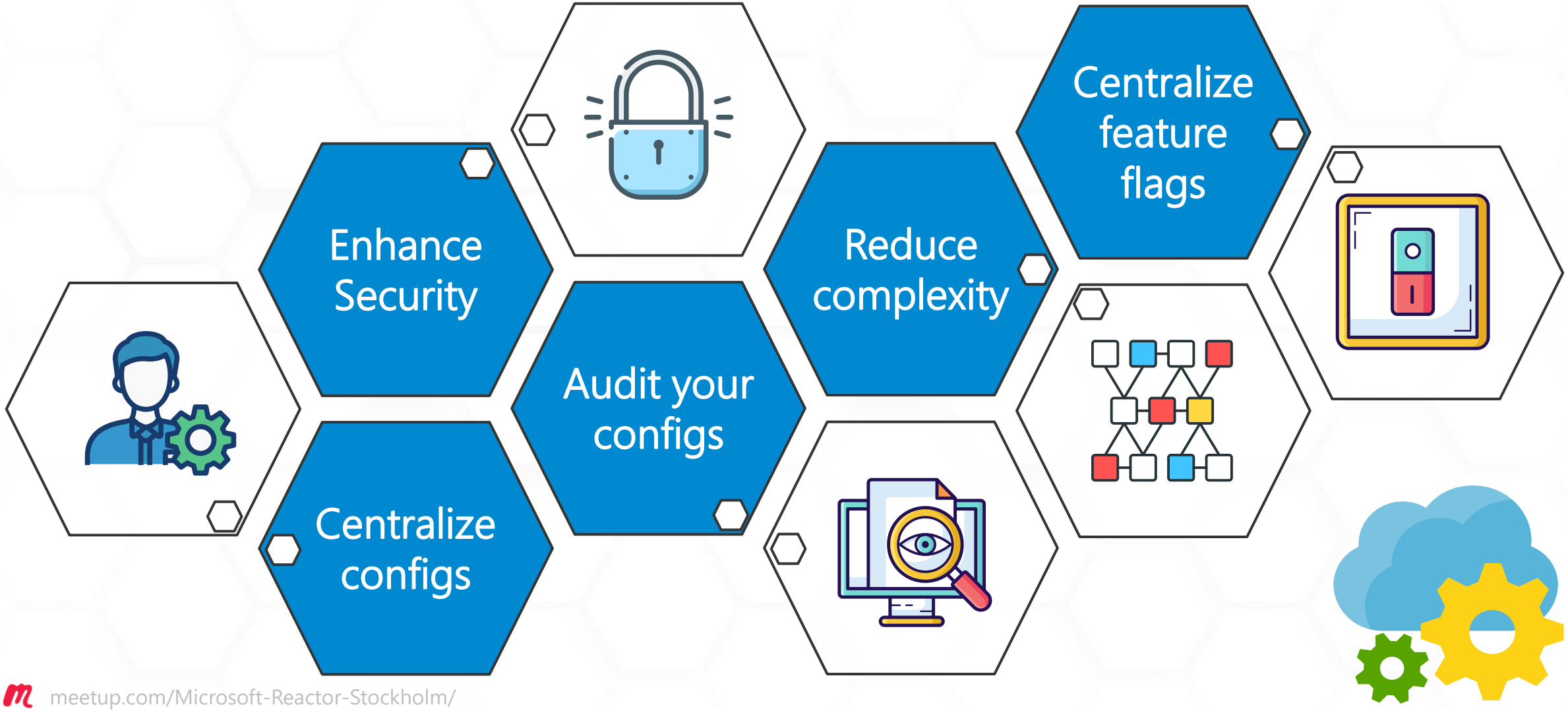
Back up App Configuration with Functions

1. Use Event Grid to route key-value modified and deleted events to a queue.
2. Use a time triggered Azure Function to read latest value from App Configuration and events from queue.
3. Add, remove or update keys in the secondary App Configuration according to what the function reads from primary.



App Configuration Replication with Azure Functions

Why use Azure App Configuration?



Thanks for your attention!!!!!

Massimo Bonanni

Azure Technical Trainer

massimo.bonanni@microsoft.com
@massimobonanni

Connect with me on LinkedIn

linkedin.com/in/massimobonanni/

meetup.com/Microsoft-Reactor-Stockholm/

Join our community

meetup.com/Microsoft-Reactor-Stockholm/

@MSFTReactor

http://www.youtube.com/c/MicrosoftReactor

ReactorStockholm@microsoft.com

Microsoft Reactor at Epicenter,
Master Samuelsgatan 36, 5th floor,
111 57 Stockholm Sweden

Questions? ReactorStockholm@microsoft.com

References

- Azure App Configuration documentation
  https://docs.microsoft.com/en-us/azure/azure-app-configuration/
- What is Azure App Configuration?
  https://docs.microsoft.com/en-us/azure/azure-app-configuration/overview
- Web Site App Configuration GitHub Repo
  https://github.com/massimobonanni/azure-att-demo
- App Configuration Syncro with Azure Functions GitHub Repo
  https://github.com/massimobonanni/AzureFunctionsSamples

Photo by Jaredd Craig on Unsplash