



OWASP

Open Web Application  
Security Project

# When Serverless Met Security... Serverless Security & Functions-as-a-Service

Niels Tanis - Veracode

# About me

- Niels Tanis
  - Security Researcher
  - Background in:
    - .NET development
    - Pen tester
    - Security Consultancy
    - CSSLP

**VERACODE**



**OWASP**  
Open Web Application  
Security Project

## SERVERLESS ECONOMIC IMPACT



Daniel Stori {turnoff.us}



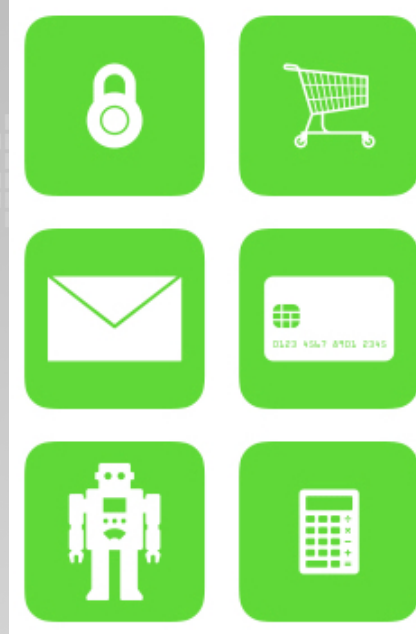
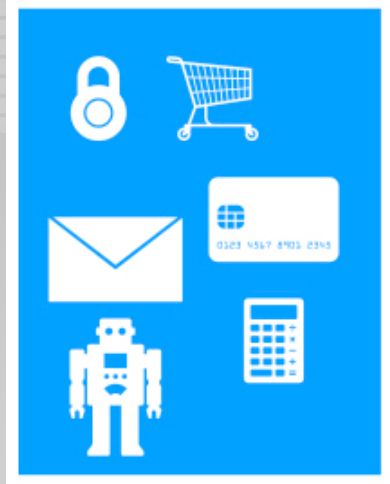
**OWASP**  
Open Web Application  
Security Project

# Agenda

- Serverless Security - Functions-as-a-Service (FaaS)
  - Overview
  - Benefits
  - Downsides
- Conclusion
- Q&A



# Monolith - MicroServices - FaaS



<https://dzone.com/articles/introduction-to-serverless-computing>

# What is Serverless?

- Full abstraction of servers
- Instant, scalable and event-driven
- Pay-per-use
- ‘Cloud is an operating system Serverless is its native code!’ (Erik Peterson, QCON)



IaaS	PaaS	Serverless	SaaS
Application	Application	Application	Application
Data	Data	Data	Data
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
OS	OS	OS	OS
Virtualization	Virtualization	Virtualization	Virtualization
Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage
Network	Network	Network	Network

You manage

By platform



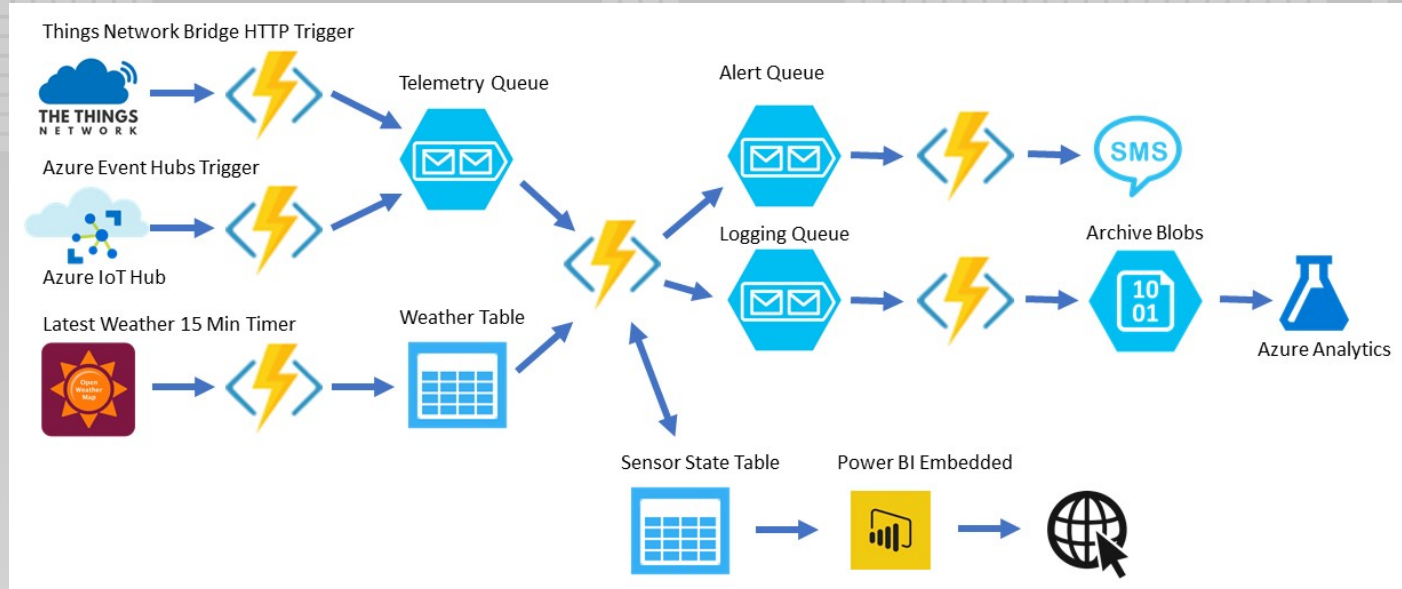
**OWASP**  
Open Web Application  
Security Project

# Functions-as-a-Service (FaaS)

- FaaS != Serverless
- FaaS is key building block
- Stateless & Ephemeral
- Single Responsibility
- Scalable & Event driven



# Example Waste Management System



<https://github.com/gloveboxes/Waste-Management-Azure-Function-Based>

# Security benefits of Serverless

- Servers are maintained by vendor
- No server to be compromised?
  - ‘Gone in 60 Milliseconds’ - Rich Jones
- Denial of Service is mitigated?



# Denial-of-Service

- Network DoS mitigated
- What gets executed? Let's limit it!
- Denial-of-Service Wallet

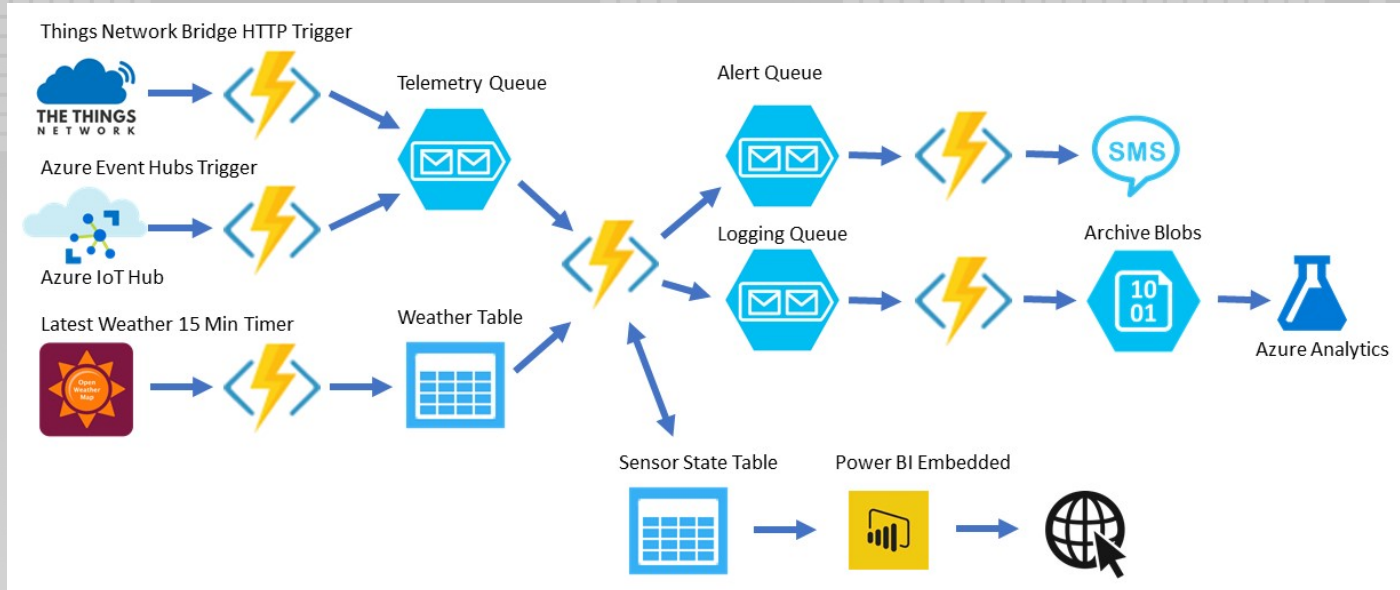


# Attack Surface

- App shattered across platform
- Lot of complexity
- Inner- and outer attack surface



# Waste Management System



<https://github.com/gloveboxes/Waste-Management-Azure-Function-Based>

# Monitoring

- What has happened?
- Logging and correlation
- What are you monitoring/logging



# Developed Code == Vulnerabilities

- Developed in various languages/technologies
- Old ‘fashioned’ vulnerabilities
  - SQL Injection
  - Remote Code Execution
  - Log Injection



# Third Party Libraries

- Simple Azure Function in C# - 10 lines
  - 50k lines for Azure Functions Host
  - 120k lines for Newtonsoft.JSON
- Vulnerability found/published
- Malicious/compromised package






**Gary Bernhardt**

@garybernhardt

Volg Je nu



An NPM package with 2,000,000 weekly downloads had malicious code injected into it. No one knows what the malicious code does yet.

 Tweet vertalen



**I don't know what to say. · Issue #116 · dominictarr...**

@dominictarr Why was @right9ctrl given access to this repo? He added flatmap-stream which is entirely (1 commit to the repo but has 3 versions, the latest one... [github.com](https://github.com)

18:44 - 26 nov. 2018



**OWASP**  
Open Web Application  
Security Project

# EventStream NPM

- 2M installs (a week)
- 5k packages depend on it
- <https://github.com/dominictarr/event-stream/issues/116>
- Targets bitcoin wallet to steal keys



# Storing Secrets

- Environment variables
- Use platform vendor service
- ‘Secrets at Scale’ - Ian Haken of Netflix



# Encryption of data

- Protecting data in transit and at rest
- Most vendors do ‘transparent’ encryption for data at rest.
- Consider ‘Client-Side Encryption’ in transit

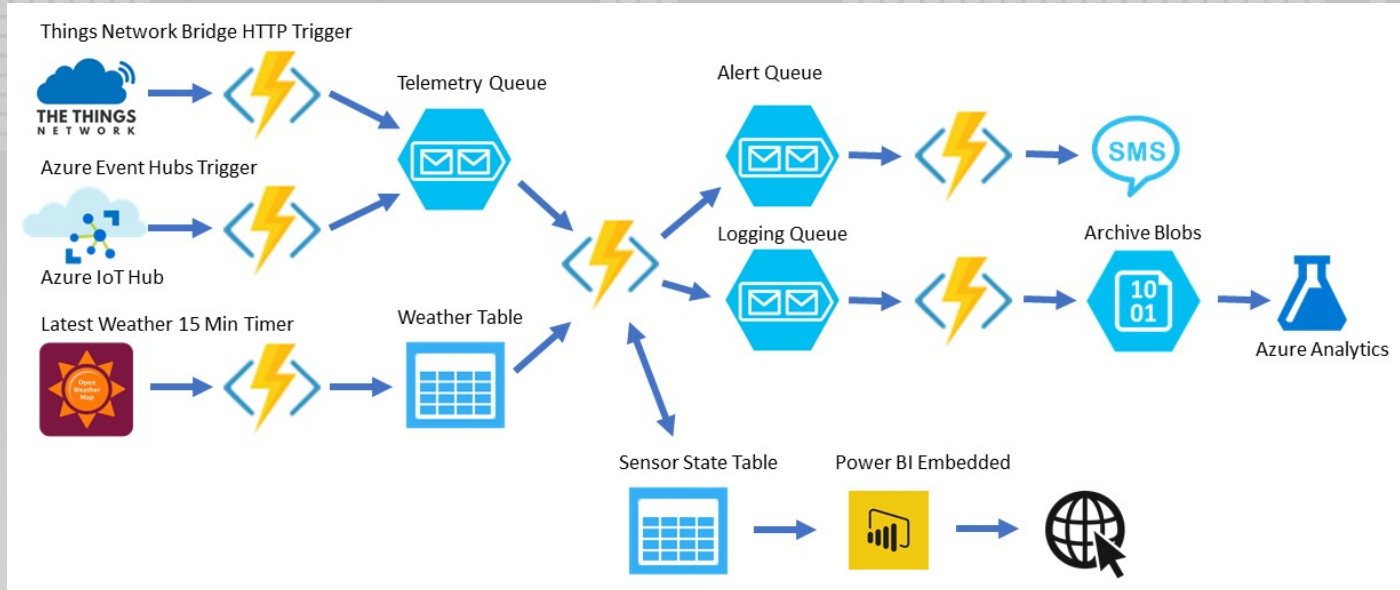


# Least Privilege

- Fit for purpose privileges
- Review or audit them over time



# Waste Management System



<https://github.com/gloveboxes/Waste-Management-Azure-Function-Based>

# Least Privilege

- Fit for purpose privileges
- Review or audit them over time



# AuthN & AuthZ

- Leverage platform specific API gateways & services



# Software Supply Chain

- Automation is king!
- Deployment as code
- Separate different environments
  - Development
  - Staging
  - Production



# Conclusion

- Easy to create & hard to keep track of!
- Threat modelling
- Compartmentalise
- Monitoring and logging
- Automate delivery and configuration



# Thanks! Questions?

- ▶ ntanis at veracode.com
- ▶ <https://twitter.com/nielstanis>



# Links

- Serverless Security and Things That Go Bump the Night - <https://www.infoq.com/presentations/serverless-security>
- Storing Secrets at Scale - <https://www.youtube.com/watch?v=15H5uCj1hIE>
- Gone in 60ms - <https://www.youtube.com/watch?v=YZ058hmLuv0>

