Devnexus

https://devnexus.com/

Atlanta Feb 2020

Why this presentation?

- I used to avoid conferences, but experience has taught me that attending them from time to time is beneficial.

Why Devnexus?

- Low Cost
- Held Annually in Atlanta
- Event provides the ability to find good content on a variety of topics relevant to Seatec.

History

- Originally Atlanta Java Users Group had a conference named Java One. Due to name issues with Java, they took a break in 2003 and came back as DEVNEXUS in 2004
- It was named as a variation of DEVOXX (https://en.wikipedia.org/wiki/Devoxx)
- DEVNEXUS is reported to be the largest Java conference in North America. Brussels DEVOXX is considered the best in the World.

Sponsors

- Special Sponsors: IBM, Red Hat, Google Cloud
- Platinum Sponsors: Heroku, Jfrog, Oracle, Home Depot
- 19 Gold Sponsors
- 28 Silver Sponsors including DELTA Airlines.

Conference Topics are in 12 different areas

- Architecture, Open Java, Cloud Technology
- Other, Tools, Security,
- Java Platform, (Unobtanium),
- Agile, JVM Languages, Core Java
- Cloud Infrastructure

Typical Day at Devnexus

- Breakfast
- Morning Keynote (1 session)
- 2 morning sessions (12 different session topics to chose)
- Lunch
- 3 afternoon sessions (12 different session topics to chose)
- Evening Keynote (1 session)

Attendees

- Authors, Technology Evangelists, Architects, Executives
- Developers, Vendors, Open Source Contributors, Consultants
- Software developers employed at major companies with large scale software solutions

Is it worth it?

- There are some exceptional seminars.
- It is beneficial to see the new trends and find out what other people are doing and how they approach problems.
- Networking with the developer community.

Past Trends at DEVNEXUS

- Microsoft was the big sponsor
- Cloud, Docker and Kubernetes
- Spring and Spring Boot
- Everyone was hyped and wanted to go to Micro-Services Oriented Architecture
- Evolutionary Software Architectures
- Security Sessions were low level technical security topics

Current Trends at DEVNEXUS

- Microsoft is gone and IBM is the big sponsor
- Common usage of Docker and Spring.
- Choosing carefully when to go from monolith to microservices and how to do it
- DevSecOps Security by Equifax executives as requiring culture, process, and cross department teamwork from security being part of sprint stories, SLC, Security KPI, deployment pipelines, and incident response. Developing a security plan.

Trends DEVNEXUS continued..

- Cloud Agnostic cloud deployments with Kubernetes (Open Shift)
- Faster Boot times with Quarkus and/or GraalVm (JIT or AOT compiles)
- API security with Keycloak is becoming common
- Considerations in multi-tenant approaches with Spring
- Evolving the Java language to make it less verbose and more modern
- Seemed to be less Android

Keycloak

OldWay SAML (verbose & complicated to implement)

Oath2/OIDC typically Base64 Token can be encrypted & signed.

PEP – Policy Enforcement Point

PDP – Policy Decision Point

Where to do PDP?

- This is typically best done in a centralized location such as Keycloak
- My previous research was use Keycloak when developing and later you can transistion to Okta or other vender as needed
- Conference was very pro-Keycloak.
- Keycloak seems to be widely used rather than commercial options
- Still remember that Red Hat is an event sponsor

Where to do PEP?

- Monolith can perhaps just embed it
- Microservices can have it at API Gateway, at the component, or both
- Kubernetes has a POD or Side Car that sort of act like a proxy with the service

Access Control Policy Approaches

- fine grain properties for users/groups very complex,
- Role Based RBAC (user/role/permission),
- Attribute based ABAC resource/environment/user),
- Rule base RUBAC (example JS code in configuration)

DevSecOps

- Equifax VP operations and VP development co-presented
- Breaches: 43% small company; 16% public; 15% Healthcare; 10% Financial
- Culture: Hire right people, software life cycle, security KPI
- Pipeline SAST (Static Application Security Testing), SCA (Software Composite Analysis), container analysis, DAST (Dynamic Application Security Testing), code signed
- Creating a Security Culture and implementing a 12 month Security Improvement plan

DevSecOps (continued 2)

- SIEM (Security Info & Event Monitoring)
- Intrusion Detection & Prevention (User Behaviorial Patterns)
- Day Zero Attack & Prevention (method of protection until have fix)
- OWASP Top 10
- Rotate passwords, keep them out of software
- Refresh Infrastructure frequently
- Rebuild! Never Patch!
- Encrypt Data

DevSecOps (continued 3)

- Tokenize and store key data
- Remove unnecessary data
- Defined Security Responsible party on each team
- Threat model included on stories

Multi-Tenancy with Spring

- Single DB shared tables (i.e., Table Column with tenantId)
- Single DB Distinct Tables (i.e., Table Name Pre-fix or Post-fix with tenant identifier)
- Unique DB per Customer
- What if Customer needs DB access to query DB data?
- Back up / Restores that are per customer?
- Mix of metrics and SLA requirements?
- Functional Variations?

Multi-tenancy (continued 2)

- Request has some type of opaque token
- Call may be made to AuthManager to determine Authorization endpoint (maybe keycloak realm?)
- The Authorization End point would then be able to map to something like a tenantId
- The tenant Id can then be used to access appropriate DB, DB table, or table row based upon the approach.

Multi-tenancy Postgres(continued 3)

- Isolated (Multiple DB run on a single cluster; scale limited)
- Semi-isolated (Multiple Schemas single DB within cluster; cross schema issues can result i.e., ids may not be globally unique?)
- Shared (Shared tables in single Schema in single DB; issues with per-customer data and backups)
- Above are Postgres specific options in addition to prior options
- Mixed approach may make sense for different areas of the application

Other Sessions Attended

- Quarkus for fast deploys, low memory, fast loads (native, JVM, or traditional JVM)
- Agile: BDD, Waterfall Agile, Error Scenarios in Acceptance Testing, test at lowest possible level
- Machine Learning (tensor flow)
- Efficient Memory Java (70% of bottlenecks are memory)
- Java future changes
- Quantum Computing
- GraphQL

Other Sessions (Continued 2)

- Migrating from Monolith to Microservice
- Running cloud agnostic with Kubernetes and maven plugins
- Venkat sessions

Questions?