



EDGE X FOUNDRY™

DevOps Working Group

Thursday August 8, 2019

Agenda

Time	Topic	Owner
10 Min	DevOps WG Update (Fuji)	James Gregg
10 Min	Snyk Demo Recap and Discussion	All
10 min	Other Business: Multiple Topics	
10 Min	Opens	All

Attendees

Participants (12)

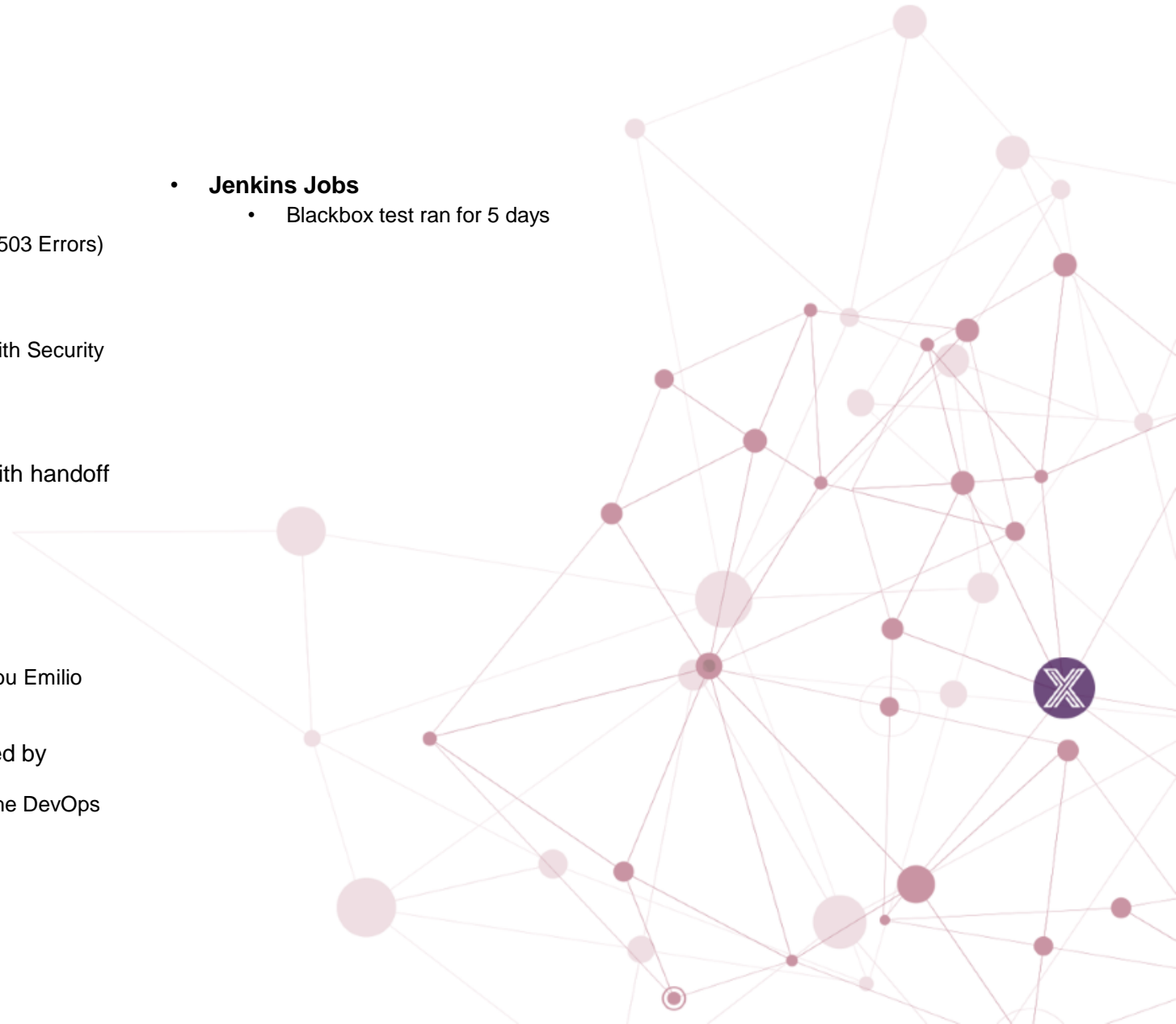
Find a participant

- JG James Gregg (Intel) (Me)
- C Colin Hutchinson
- 12062253082
- Eric Ball - Linux Foundation
- E- Ernesto - Intel
- JP Jeremy Phelps
- JW Jim White
- LG Lenny Goodell (Intel)
- LR Lisa Rashidi-Ranjbar
- R Ricardo
- TC Trevor Conn
- V Vishwas



DevOps WG Update

- **Container Scanning**
 - Clair server has now been landed by Linux Foundation on AWS.
 - Troubleshooting with Daniel Nunez to debug issues with the endpoint (503 Errors)
- **Static Code Analysis Tools**
 - Checkmarx, SonarCloud and Snyk output from egex-go scan shared with Security WG
 - Coverity scan and output still WIP
- **Audit of GitHub repos / teams** of main edgexfoundry org completed with handoff to Lisa
 - Review and identify repo owners
 - Discuss in next DevOps WG
 - Changes to add repo owners will require TSC approval
- **Build Performance Optimizations** (new scope)
 - log archiving optimization made to the Jenkins freestyle jobs - Thank you Emilio
- **Proposal for Shared Infrastructure** under LF Edge umbrella introduced by Andrew Grimberg
 - Need clarifications for the list of questions that were not addressed in the DevOps WG
- **Jenkins Jobs**
 - Blackbox test ran for 5 days



Snyk Demo / Discussion

Feedback from Demo

```

~/git/gohack$ snyk test --file=go.mod --print-deps
github.com/rogpeppe/gohack @ 0.0.0
├── github.com/rogpeppe/go-internal/dirhash @ v1.0.0
├── github.com/rogpeppe/go-internal/modfile @ v1.0.0
│   ├── github.com/rogpeppe/go-internal/module @ v1.0.0
│   └── github.com/rogpeppe/go-internal/semver @ v1.0.0
├── github.com/rogpeppe/go-internal/semver @ v1.0.0
├── github.com/rogpeppe/go-internal/module @ v1.0.0
│   └── github.com/rogpeppe/go-internal/semver @ v1.0.0
├── github.com/rogpeppe/go-internal/semver @ v1.0.0
├── golang.org/x/tools/go/vcs @ #90fa682c2a6e
├── gopkg.in/errgo.v2/fmt/errors @ v2.1.0
└── gopkg.in/errgo.v2/errors @ v2.1.0
    
```

```

Introduced through: node@6.14.2
From: node@6.14.2
Introduced by your base image (node:6.14.2-slim)
Fixed in: 6.15.0

Organisation: acme-co
Package manager: deb
Target file: Dockerfile
Docker image: docker-gooF
Base image: node:6.14.2-slim
Licenses: enabled

Tested 252 dependencies for known issues, found 534 issues.

Base Image      Vulnerabilities  Severity
node:6.14.2-slim  168              98 high, 59 medium, 11 low

Recommendations for base image upgrade:

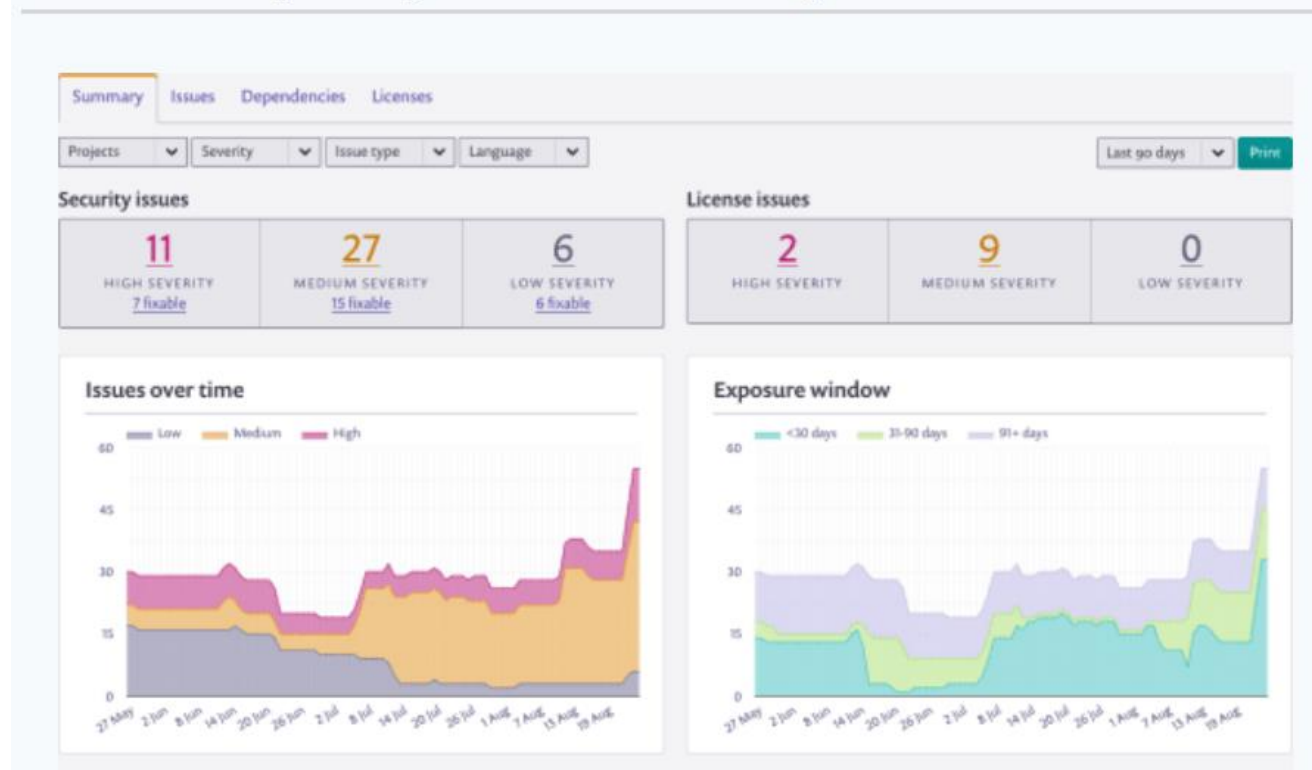
Minor upgrades
Base Image      Vulnerabilities  Severity
node:6.17.0-slim  101              60 high, 34 medium, 7 low

Major upgrades
Base Image      Vulnerabilities  Severity
node:12.6.0-slim  87               47 high, 33 medium, 7 low

Alternative image types
Base Image      Vulnerabilities  Severity
node:dubnium-slim  87              47 high, 33 medium, 7 low
node:lts-jessie-slim  102             53 high, 41 medium, 8 low

Pro tip: use '--exclude-base-image-vulns' to exclude from display Docker base image vulnerabilities.
To remove this message in the future, please run `snyk config set disableSuggestions=true`
    
```

An overview of your organization's vulnerability status



LFtools log publishing optimization

- Leverage existing docker image that includes lftools to push logs to Nexus
 - Eliminates need of installing lftools for every build
 - Saves time and cleans up logs by eliminating installation messages
- Build time reduction
 - ARM: ~4-5 minutes
 - X64: ~1-2 minutes
- Bug - SysInfo log missing SAR system information
<https://github.com/edgexfoundry/ci-management/issues/459>

API Endpoints (Lisa)

How do we want to report versioning long term based off a tag?

- Issue: We need a way for services to identify their version at runtime.
 - Right now we are injecting this at build time using the VERSION file.
 - We have seen that we often forget to update the VERSION file manually so this is problematic.

Options:

Previously discussed “git describe”

- The downside to “git describe” is that without pushing a tag it will return the commit SHA.
- Recommendation
 - Use “git-semver” utility

How do we want to report versioning long term based off a tag? (continued)

In Edinburgh we implemented git-semver a tool to manage the version in github repos to address these small gaps. Git itself does not infer what the next version should be based off semantic versioning but git-semver does.

We can use git semver utility to inject the version into the build in the Makefile.

Right now app-service-configurable has a Makefile set up where we can pop in the git semver command.

<https://github.com/edgexfoundry-holding/app-service-configurable/blob/master/makefile>

Simply change the "cat ./VERSION" command to "git semver" this will return the version of the build.

We can pilot this with app-service-configurable when we set up the pipeline to build the service.

makefile

```
.PHONY: build test clean docker

GO=CGO_ENABLED=1 go

APPVERSION=$(shell cat ./VERSION)

# This pulls the version of the SDK from the go.mod file. If the SDK is the only
# required module,
# it must first remove the word 'required' so the offset of $2 is the same if there
# are multiple required modules
SDKVERSION=$(shell cat ./go.mod | grep 'github.com/edgexfoundry/app-functions-sdk-
go v' | sed 's/require//g' | awk '{print $$2}')

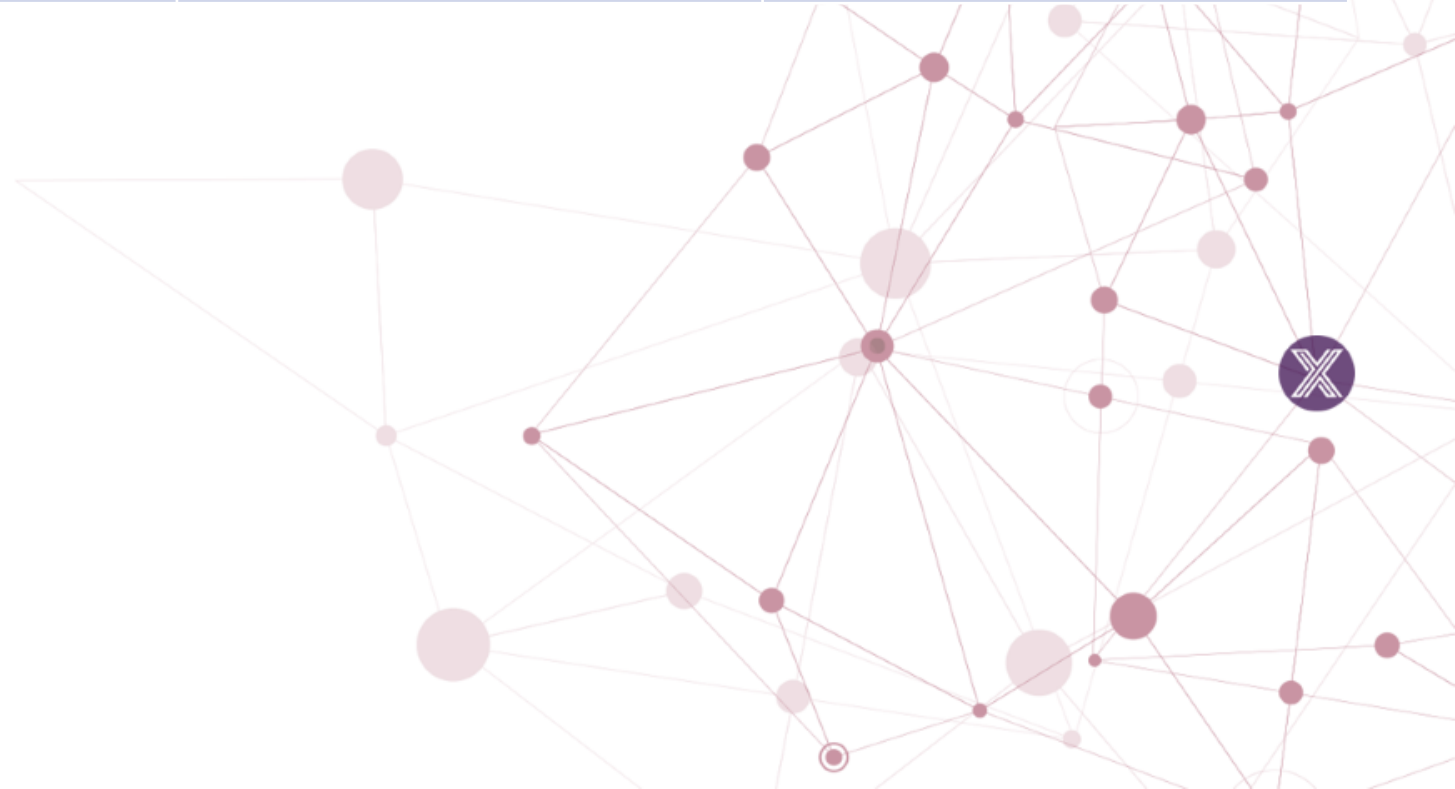
MICROSERVICE=app-service-configurable
GOFLAGS=-ldflags "-X github.com/edgexfoundry/app-functions-sdk-
go/internal.SDKVersion=$(SDKVERSION) -X github.com/edgexfoundry/app-functions-sdk-
go/internal.ApplicationVersion=$(APPVERSION)"

GIT_SHA=$(shell git rev-parse HEAD)

build:
    $(GO) build $(GOFLAGS) -o $(MICROSERVICE)
```


Work Review

Helpdesk Ticket #	Description	Details	Status
75648	Dedicated Clair server for EdgeX	Pending decision on strategy for K8s + cost / availability of resources with LF	WIP
IT-17020	Status Update Requested: Clair Server		WIP
IT-17019	Permissions request for SonarCloud	Need to change Quality Profile / Quality Gate	WIP



Backlog Review

EdgeX Foundry Project

Repositories 89 Packages People 56 Teams 90 Projects 14

DevOps WG Updated 15 hours ago

Filter cards + Add cards Fullscreen Menu

Icebox

- regex for isReleaseStream is overly broad
edge-global-pipelines#26 opened by dweomer
- Snap builds should use unbuffer for better output logging
ci-management#369 opened by anonymous64
- edgeXSemver leaves command to caller but doesn't allow specifying the version/image to use
edge-global-pipelines#15 opened by dweomer bug

New Issues

- Nexus sys-info build log is missing some system information
ci-management#459 opened by soda480
- New build automation for device-bacnet-c
ci-management#451 opened by jamesgregg enhancement Fuji
- Update snapcraft inside docker to use new setup from snapcraft
ci-management#449 opened by anonymous64 enhancement
- New Build Automation needed for application-service-configurable
ci-management#447 opened by jamesgregg enhancement Fuji
- New build automation needed for device-gps service - <Place holder pending move from holding after TSC approval >
ci-management#446 opened by jamesgregg
- Refactor docker push scripts to be more generic
ci-management#435 opened by Iranjbar enhancement
- repository: add description and topics
git-semver#2 opened by dweomer enhancement
- Slack integration with Jenkins Pipeline for Clair reporting.
ci-build-images#54 opened by jamesgregg enhancement Geneva

Release Backlog

- Delete repo - jenkins_pipeline_presentation (FUJI)
edge-global-pipelines#32 opened by jamesgregg
- https://github.com/edgexfoundry/holding/go-mod-core-security ON-HOLD (FUJI)
May not be needed - Trevor will check and advise
Added by jamesgregg
- Placeholder for snap global function in edge-global-pipelines (FUJI)
US4581
US4580
Added by jamesgregg

In Progress

- Clair Server landing - LF Infrastructure or AWS decision and architecture / SS with TSC approval (@jamesgregg)
ci-management#439 opened by jamesgregg enhancement Fuji

QA/Code Review

- WIP: Add sigul signing to golang binaries
ci-management#318 opened by JPWKU
- Changes requested

New Work for DevOps User Stories

- App Service Configurable – build automation (Jenkins)
 - version endpoint on SDK injected during the build

Device Services - current/coming

Open Source connectors (Current):

- Modbus (TCP/RTU)
- Virtual Device
- Random
- Grove C - ARM only
- SNMP
- MQTT

Open Source connectors (Future):

- BACnet (IP & MSTP) (Aug)
- OPC UA (Aug)
- Bluetooth (Aug)
- ONVIF Cameras (Oct)

Commercial connectors (Current):

- MQTT
- Modbus (TCP/RTU)
- BACnet (IP & MSTP)
- OPC UA
- GPS
- MEMS

Commercial connectors (Future):

- CAN SocketCAN, CANopen (Sept)
- EtherNet/IP (Sept)
- EtherCat (Sept)
- PROFINET (Sept)
- Siemens s5/s7 (Sept)

Meeting Minutes

- There were some optimizations being made to the BB testing but the job should have had a timeout set after 72 hours, so the fact that it ran for 5 days was an anomaly. Robin will check with the team and get back on the setting to ensure it has a timeout set up for the job going forward.
- Snyk demo was impressive from Tom / Rob yesterday in Security WG meeting
 - +1 from James, Jim, Trevor
 - Need to look into cost for Enterprise version with reporting and ensure that what was saw in the demo was included in the OSS version
 - Look into enabling the Jenkins Pipeline as a POC in Sandbox for additional evaluation
 - James to follow up with Rob / Snyk
- API Versioning discussion
 - We will set up the new app service configurable as a Pipeline job using git-semver as a POC
 - Lisa - Bring back a flow diagram which illustrates the way things work so we can visualize the flow.
 - Set up the new Pipeline job for POC
- Lftools optimization / enhancement
 - Emilio shared the work he completed to optimize the Jenkins freestyle jobs
 - Introduction of a minor bug re: SAR should be fixed by EOW
- New device services expected to land by end of Aug

Edinburgh Retrospective

What went right?

- Communication of when the release was scheduled, was very clear.
- LF and DevOps team cooperating together seemed to work well but with pressure added on top of everyone
- Most artifacts were ready to release in the beginning. It didn't seem like every repo was affected with issues (no extra work)
- Edinburgh Staging view was very helpful
- Great communication and collaboration between Intel DevOps team members and great prioritization
- Prioritization and Organization of the work (assignments and splitting up the work early on in the code release) was helpful

What went wrong?

- EdgeX-UI repos were late code drop
- Lack of a UI WG
- Communication around details was lacking from WG leads
- Need clear definition and understanding from WG leads that have different / independent release cycles
- JSD was introduced in the middle of the release and introduced issues that impacted communications with LF RE
- Availability and Competing priorities of Eric Ball / RE impacted release work
- Branches cut early caused extra work for both developers and DevOps

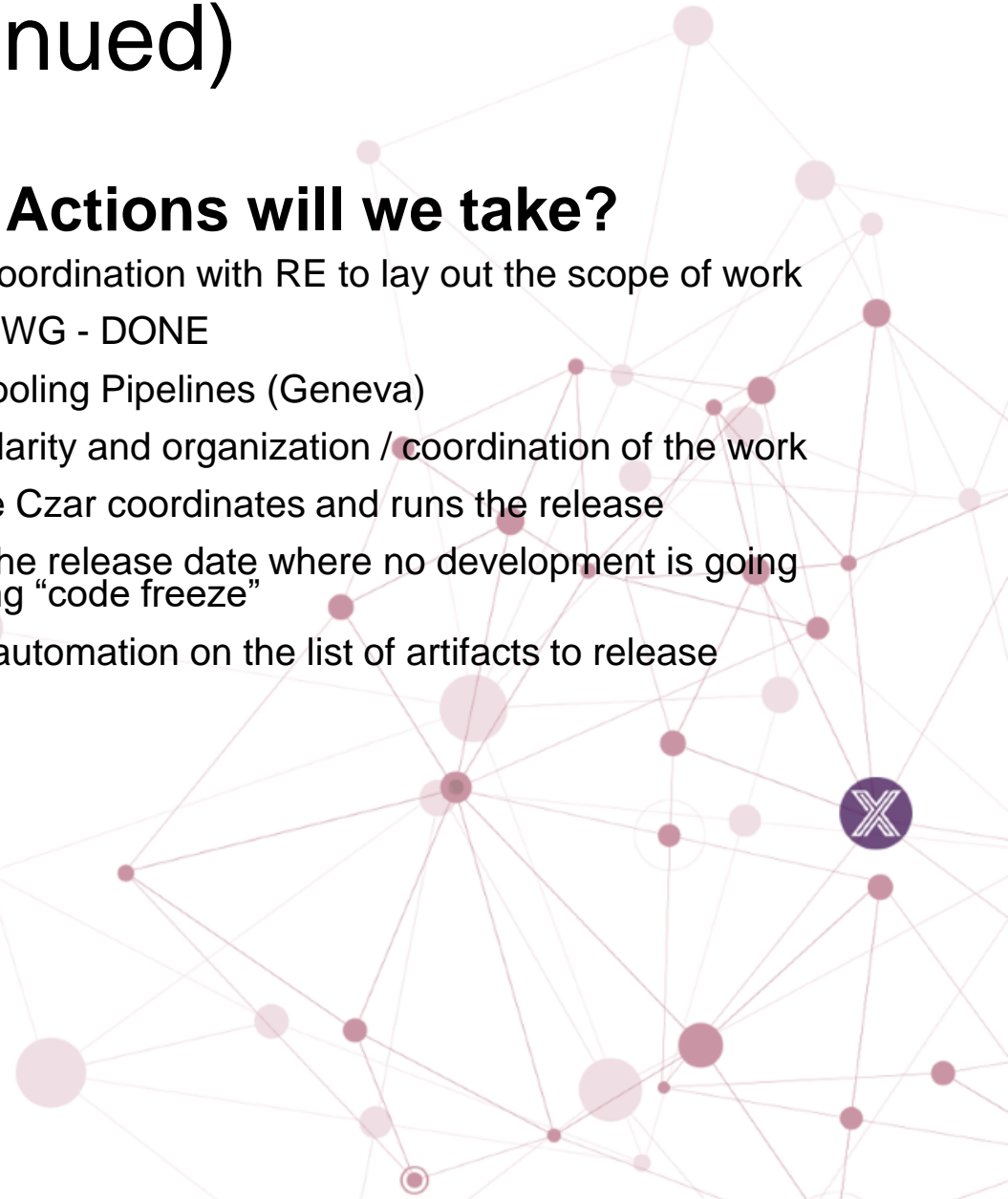
Edinburgh Retrospective (continued)

What Ideas would help next time?

- Jenkins Pipelines branch defined in Jenkinsfile
- Do Not cut the branch early
- Release from master
- Don't allow PRs to master during code freeze (unless bug fix)
- Release Czar manages the release and acts as coordinator
- Need better visibility as to WIP during release. What does DONE look like?
- Set clear expectations for FUJI ahead of time
- Have a solid and well defined scope for executing the release.
- Shorten the release timeframe
- Actually Freeze Code
- Don't pull in late code drops

What Actions will we take?

- Better coordination with RE to lay out the scope of work
- New UI WG - DONE
- Better tooling Pipelines (Geneva)
- Better clarity and organization / coordination of the work
- Release Czar coordinates and runs the release
- Shrink the release date where no development is going on during "code freeze"
- Create automation on the list of artifacts to release





EDGE X FOUNDRY™

Fuji Planning

Scope Discussions

Fuji – DevOps

In

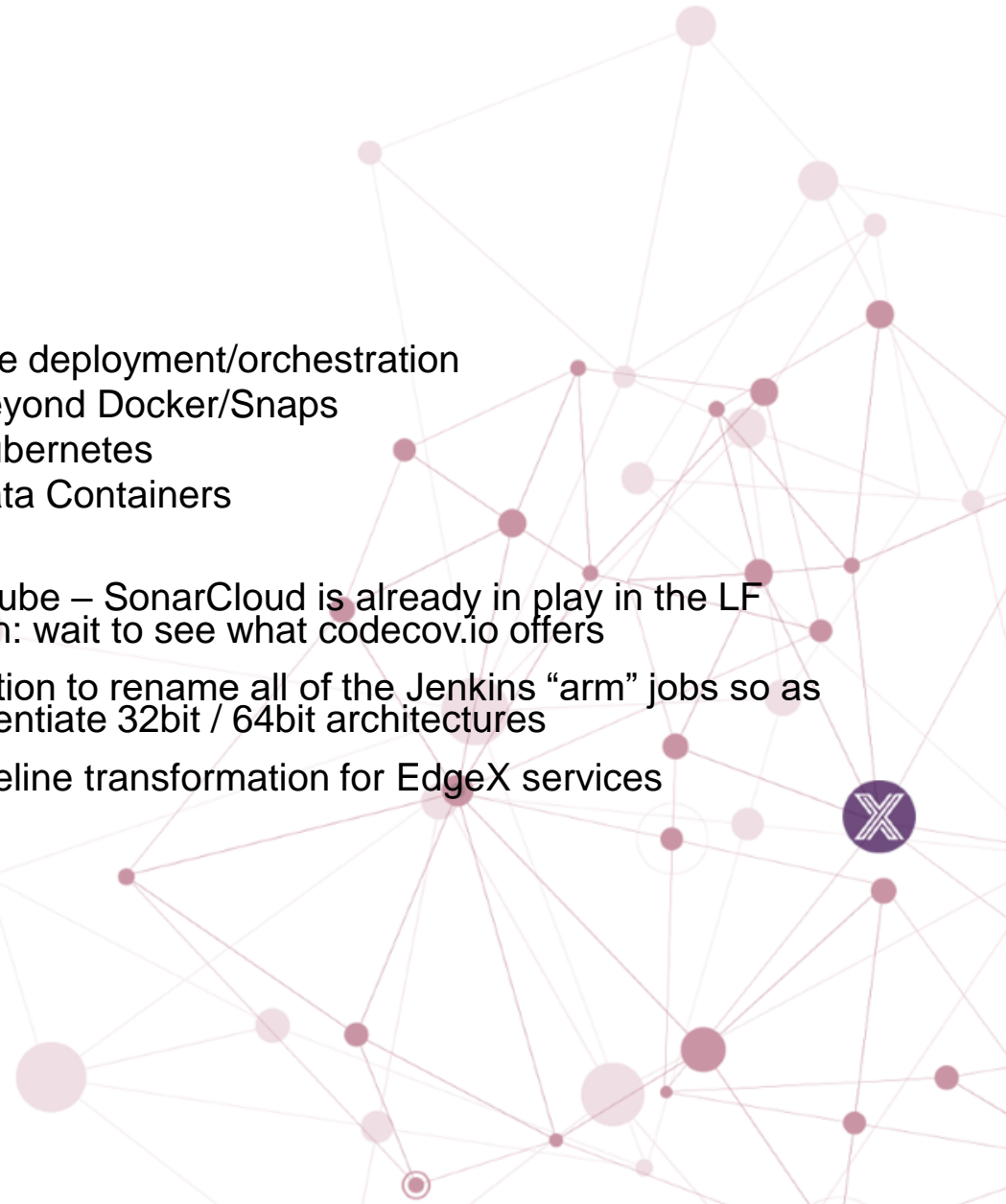
- Static code analysis tool identified and integrated into the EdgeX Jenkins Pipeline for Docker image scanning (Clair Server)

Explore SAST for true static code analysis to include additional tooling such as Fortify / Coverity





- Code and artifact signing with semantic versioning
- Fix Documentation – edgex-go
 - Create a new repo for edgex-docs
- Build Performance Optimizations
 - Pipelines for EdgeX Foundry base build images
 - Basebuild images managed locally within Nexus
 - Leverage PyPi Proxy for local pip dependencies
 - ARM builds – optimization leveraging different high CPU build nodes / OS (ARM Team)

Out

- Alternate deployment/orchestration
 - Beyond Docker/Snaps
 - Kubernetes
 - Kata Containers
 - ...
- SonarQube – SonarCloud is already in play in the LF Decision: wait to see what codecov.io offers
- Suggestion to rename all of the Jenkins “arm” jobs so as to differentiate 32bit / 64bit architectures
- Full Pipeline transformation for EdgeX services



EdgeX DevOps Commitments (Fuji)

Scope of Work	
Add static artifact analysis into the EdgeX Jenkins Pipeline (analysis of Docker /runtime artifacts, not the source code)	
Add code and artifact signing with semantic versioning	
Conduct build performance optimizations by: <ul style="list-style-type: none"> • Adding Pipelines for EdgeX Foundry base build images • Allow base build images to be managed locally within Nexus • Leverage PyPi Proxy for local pip dependencies 	
Explore static code analysis like Checkmarx, Coverity, GuardRails, Synk, SonarQube	

- Clair Server landing no longer at risk for Fuji
 - LF committed to implement on AWS and fund with expected completion next week
- gitsemver along with lftools used for artifact signing and semantic versioning
- Jenkins build performance optimizations for base build images completed
- All base build images will now be stored in Nexus (Snapshot):10003
- PyPi enabled as part of Edinburgh scope
- Initial review of GuardRails showed that the product was identifying issues which were not applicable for microservices architecture



EDGE X FOUNDRY™

Edinburgh Release

Release Planning

Edinburgh Dates

- Freeze Date – May 28
- Release Date – June 20
- Press Release – July 11
- Dot Release – July 22





EDGE X FOUNDRY™

Past / Future Agenda Topics

WW27	No Meeting – US Holiday
WW28	
WW29	
WW30	
WW31	Shared Infrastructure LF Edge Umbrella Projects
WW32	Fuji Update / Performance Optimization Iftools
WW33	
WW34	
WW35	
	Athens Project – proxy server for go package dependencies
	Community Involvement