



# Kubernetes Policy WG Session

KubeCon Barcelona 2019



Howard Huang, Huawei  
Erica Von Beulow, RedHat

# WG Overview

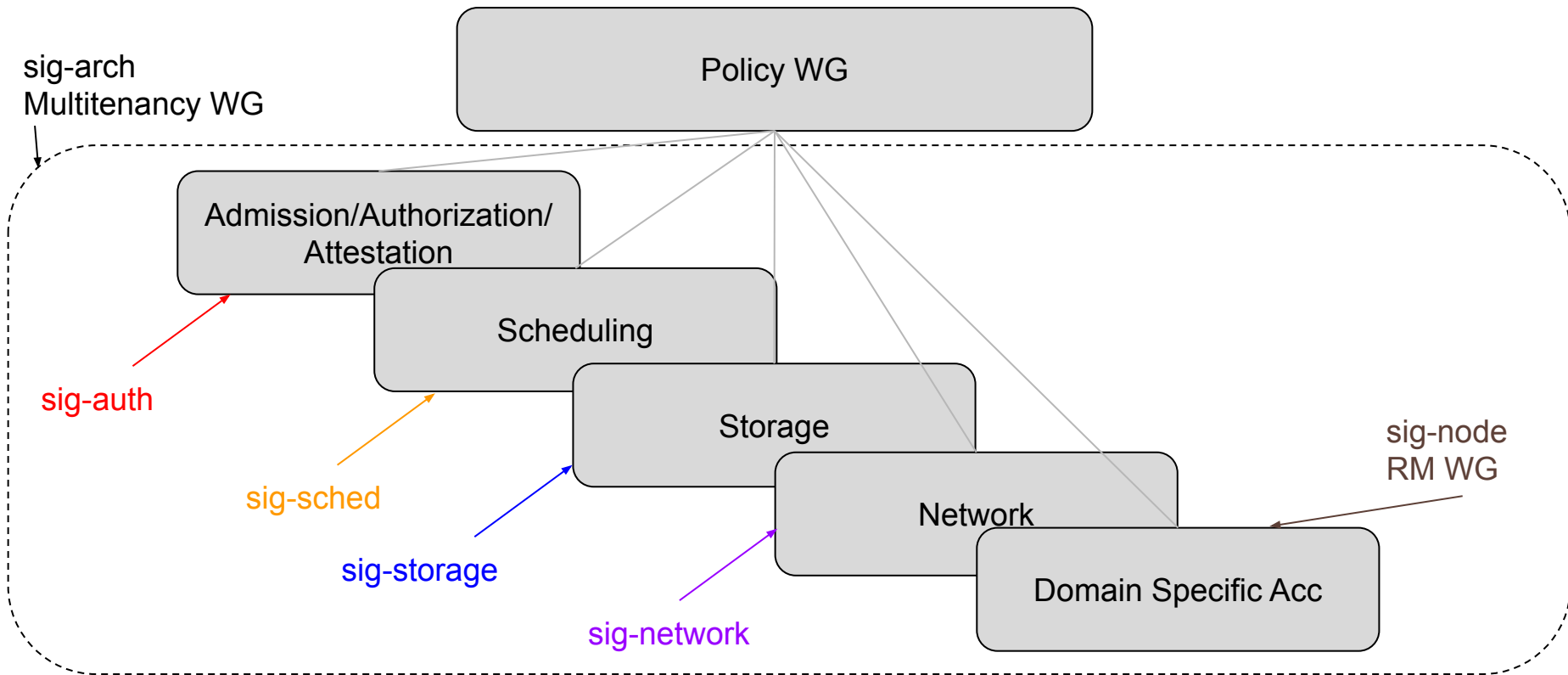
# Motivation (from Brian G)

- Kubernetes The Policy Framework
- Policies impose permissions, quotas, constraints, requirements, defaults, etc. on other resources
- What patterns should we adopt going forward?
  - Built in vs extensions
  - Extension using DSLs vs APIs
  - Domain-specific (scheduling policy) vs resource-specific (pod restriction)
  - Conventions across policy types: whitelists, blacklists, profiles, defaults, etc.
  - Cluster-level vs namespace-level
  - Policies vs component flags
- How do we provide policy defaults?

# Motivation (from ourselves)

- Policy are needed and designed all over the place in kubernetes
- Policy description are domain specific in nature:
  - Not only in the sense Brian G meant (Kubernetes' domain), but also in a larger context of usage (audit, security, storage, network, AI...), vertical adoption (finance, telco, pharma,...), languages, ...
  - Usually out of scope for WG description
- Policy semantic and control mechanism is universal
  - Policy semantic: the underlying description of the policy description
  - Policy control mechanism: life cycle of policy itself, and life cycle of elements defined in policy

# Overview (SIG Relationship)

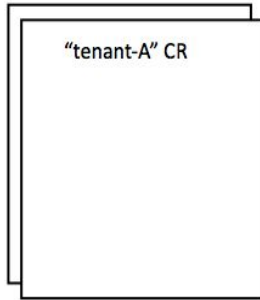


# WG Work Items

# Policy WG Work Items Overview

- Running list of interested items
  - Scheduling Policy Proposal (**pending**)
  - Metadata Policy: (**pending**)
  - Multi-tenancy: <https://github.com/kubernetes-sigs/multi-tenancy>
  - Gatekeeper:  
<https://docs.google.com/document/d/1A1-Q-10Mw3Q0Ds1wT6eqfLTagcGmgzAJAjJihi03T48/edit#heading=h.rosd3aktkpys>
- New Area Exploration
  - Policy as type system
  - Policy formal verification
- Case Studies

# WG Running List 2019 - Multi-Tenancy Policy



## Minimal Base version workflow:

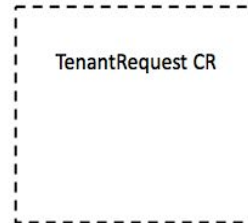
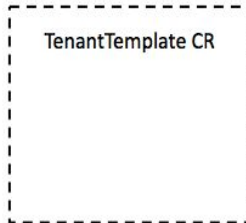
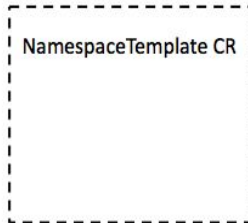
*KubectI create -f newtenant.yaml*

Kind:  
**tenant**  
Spec:  
*Tenant\_name*  
*Admin\_contact*  
<NamespaceTemplateList>

## Advanced Full version workflow (WIP):

*KubectI create -f newtenant.yaml*

Kind:  
**tenantrequest**  
Spec:  
*tenantTemplateInstance*





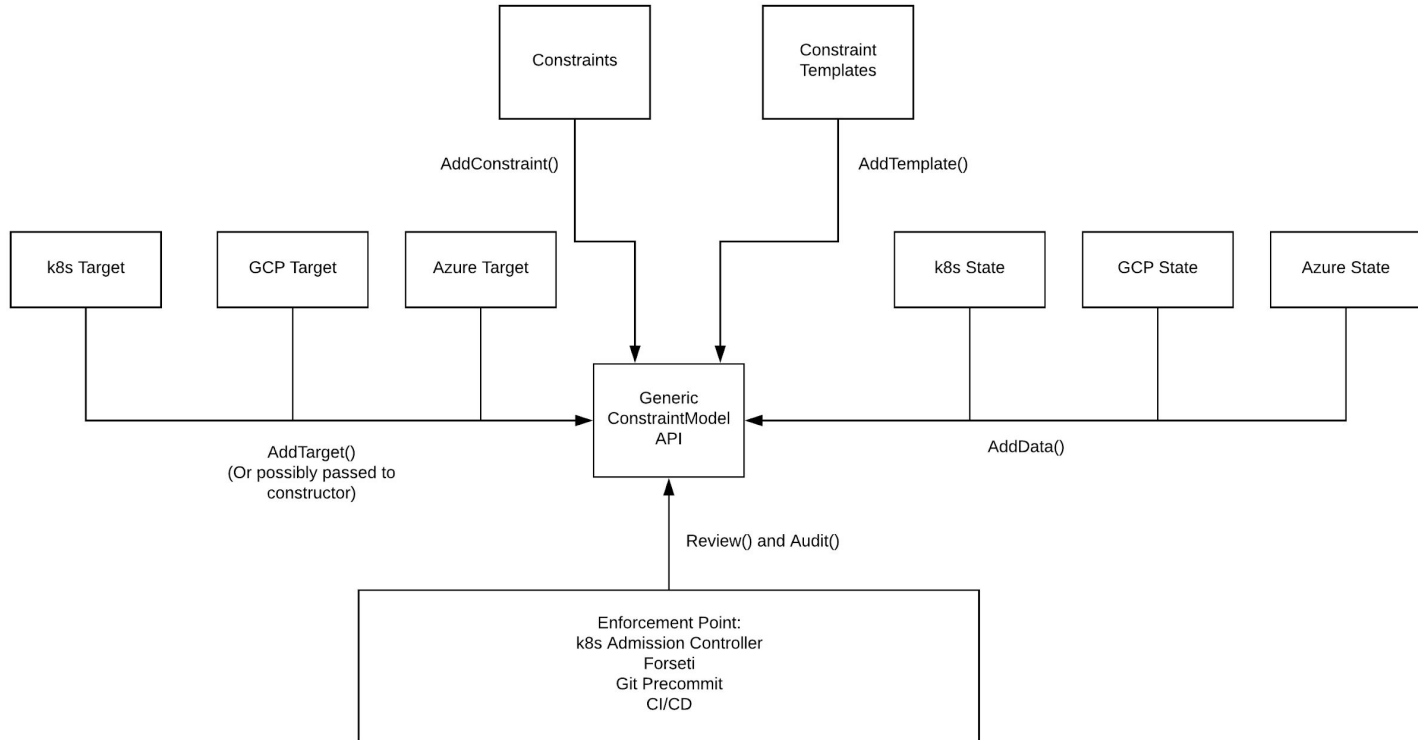
# WG Running List 2019 - Multi-Tenancy Policy

- Self-service Namespace Creation
  - “kubectl create ns” by tenant admins without going over an indirect way through Tenant CRD and Tenant CRD controller.
- Cluster-scoped Resources
  - the tenant admins may have permissions to create cluster scoped resources like PodSecurityPolicy
- In a nutshell, help solving the CR population problem in the context of multi-tenancy

# WG Running List 2019 - Multi-Tenancy Policy

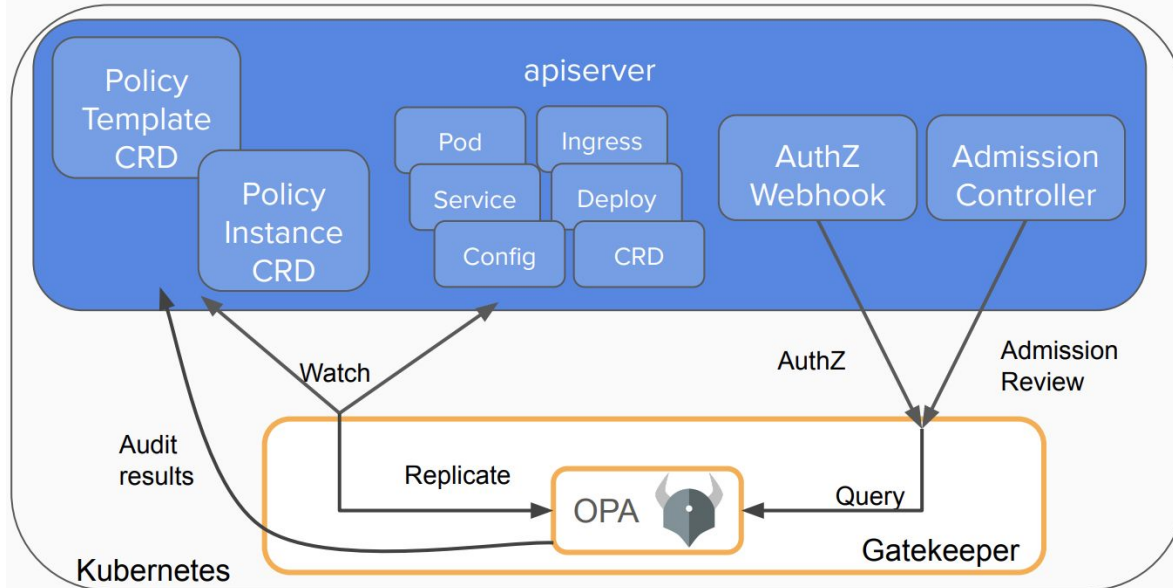
- **Proposal** : Policy Engine -> Policy Compiler -> Tenant Policy object -> Resource Population (ns, podsec, network, rbac, ....)
- **Example** : OPA -> Gatekeeper (Tenant Policy Object -> Resource Population) -> General Kubernetes Cluster
- **Problem**: how to define the constraint for a population (when do we hit a wall and stop)

# WG Running List 2019 - OPA Gatekeeper Project



# WG Running List 2019 - OPA Gatekeeper Project

## OPA Gatekeeper v3.0



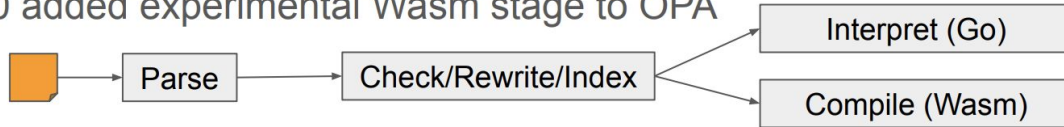
- Validating admission. Audit. CI/CD.
- Policy templates (Rego) and instances
- Policies stored in CRDs
- Audit results stored on policy CRDs
- [Full architecture](#)
- Google, Microsoft, Redhat, CBA, Styra
- "Gatekeeper" donated by Replicated
- Built with kubebuilder

# WG Running List 2019 - OPA Gatekeeper Project

## v0.11: Native Integrations: WebAssembly progress

- WebAssembly (Wasm) is an instruction format for virtual machines
  - Provides a safe/efficient/portable runtime for policy evaluation
  - Goal: enable library embeddings of OPA policies in any language/runtime

- v0.10 added experimental Wasm stage to OPA



- v0.11 expands the fragment of Rego supported by the Wasm stage
  - All types of rules (ordered/unordered, default, partial sets/objects) now supported
- Example: [open-policy-agent/contrib/wasm](#) (CDN example)



# WG New Area Exploration - policy formal verification

- Background Knowledge
  - SMT can be thought of as a form of the constraint satisfaction problem and thus a certain formalized approach to constraint programming
  - an SMT instance is a formula in first-order logic, where some function and predicate symbols have additional interpretations, and SMT is the problem of determining whether such a formula is satisfiable
  - A predicate is a binary-valued function of non-binary variables. Example predicates include linear inequalities (e.g.  $3x + 2y - z \geq 4$ ) or equalities involving uninterpreted terms and function symbols (e.g:  
 $f(f(u, v), v) = f(u, v)$ )

# WG New Area Exploration - policy formal verification

$$(\sin(x))^3 = \cos(\log(y) \cdot x) \vee b \vee -x^2 \geq 2.3y \wedge (-b \vee y < -34.4 \vee \exp(x) > \frac{y}{x})$$

where

$$b \in \mathbb{B}, x, y \in \mathbb{R}$$

CVC4 ABOUT NEWS PEOPLE DOWNLOAD PUBLICATIONS COPYRIGHT

the smt solver

## ABOUT

- 1 About CVC4
- 1.1 Features
- 1.2 Documentation
- 1.3 BibTex Entry
- 1.4 Downloads
- 1.5 Copyright
- 1.6 Technical Support
- 1.7 Third-party applications
- 1.8 Publications
- 2 History of CVC
- 3 Authors
- 4 Acknowledgments

## RECENT POSTS

- CVC4 1.7 released (April 9, 2019)
- CVC4 1.6 released (June 25, 2018)
- CVC4 1.5 released (July 10, 2017)
- 2015 competition results (November 24, 2015)
- CVC4 at Vienna Summer of Logic (July 28, 2014)

FIND US ON GOOGLE

Z3Prover / z3

Code Issues 137 Pull requests 8 Projects 0 Wiki Insights

The Z3 Theorem Prover

10,819 commits 8 branches 14 releases 121 contributors View license

Branch: master New pull request Create new file Upload files Find File Clone or download

Nikola Björner Merge pull request #2282 from AIGo/locami-static-linking Latest commit 112e13e 15 hours ago

cmake	Change from BINARY_DIR to PROJECT_BINARY_DIR	7 days ago
contrib	Fix bug in qprofdiff	4 months ago
doc	Change from BINARY_DIR to PROJECT_BINARY_DIR	7 days ago
examples	Change from BINARY_DIR to PROJECT_BINARY_DIR	7 days ago

The Rosette Language

ABOUT DOWNLOAD DOCS APPLICATIONS THANKS

## About Rosette

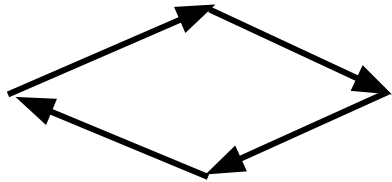
Rosette is a solver-aided programming language that extends Racket with language constructs for program synthesis, verification, and more. To verify or synthesize code, Rosette compiles it to logical constraints solved with off-the-shelf SMT solvers. By combining virtualized access to solvers with Racket's metaprogramming, Rosette makes it easy to develop synthesis and verification tools for new languages. You simply write an interpreter for your language in Rosette, and you get the tools for free!

```
#lang rosette

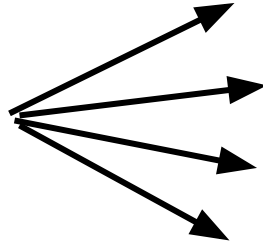
(define (interpret formula)
  (match formula
    [(^ ,expr ...) (apply && (map interpret expr))]
    [(v ,expr ...) (apply || (map interpret expr))]
    [(~ ,expr) (! (interpret expr))]
    [lit (constant lit boolean?)])
```

# WG New Area Exploration - policy formal verification

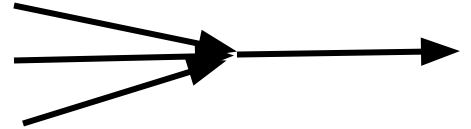
**Construct a policy symbolic graph for each kubernetes domain**



**networking**



**Multi-tenancy**



**Security**



# WG New Area Exploration - policy formal verification

- Possible new open source project for policy fv starting this year either under the banner of kube wg-policy or cncf sig-security
- Collaboration involving AWS, Styra, and many others in the community
- Keep an eye on the slack channel or ping us via email (**zhipengh512@gmail, evb@redhat.com**) if you are also interested

# WG New Area Exploration - policy as type system

*Together, these concepts*

1. Identity
2. Outcome Set
3. State
4. Rules

*enable us to define a policy in a way that is consistent and automatable.*

# WG New Area Exploration - policy as type system

Proposed long term vision:

## **1- Strong type system for Kubernetes resources**

- Better specifications and validation with a formal type system
- Algebraic types:
  - Allows you to define more complex resource types (e.g. "pod"+"configmap", union types)
  - Compositional transformations and admission chains

# WG New Area Exploration - policy as type system

Proposed long term vision:

## **2- Policy Hooks at key points**

- Lifecycle: Admission, deletion
- Network traffic in and pod of pods
- Pod start up and down
- API calls - webhook not quite enough

# WG New Area Exploration - policy as type system

Proposed long term vision:

## **3- Capabilities**

- Pod "leases"
- Delegation, access control

# WG New Area Exploration - policy as type system

Proposed long term vision:

## **4- Kubernetes as the “now”**

- Flattened view with explicit consistency bands
- Pipeline of transformations to facilitate managing clusters
- Favor “compiled” over runtime interpretation

# WG New Area Exploration - policy as type system

- **Everything in namespaces**

- Doesn't have to be the same "namespace"
  - e.g. "organization" concept over users
  - Needs to be - every resource is in one and only one namespace (or zone or class or whatever)

- **Immutable labels or annotations**

- Keep context, allow chains of validations

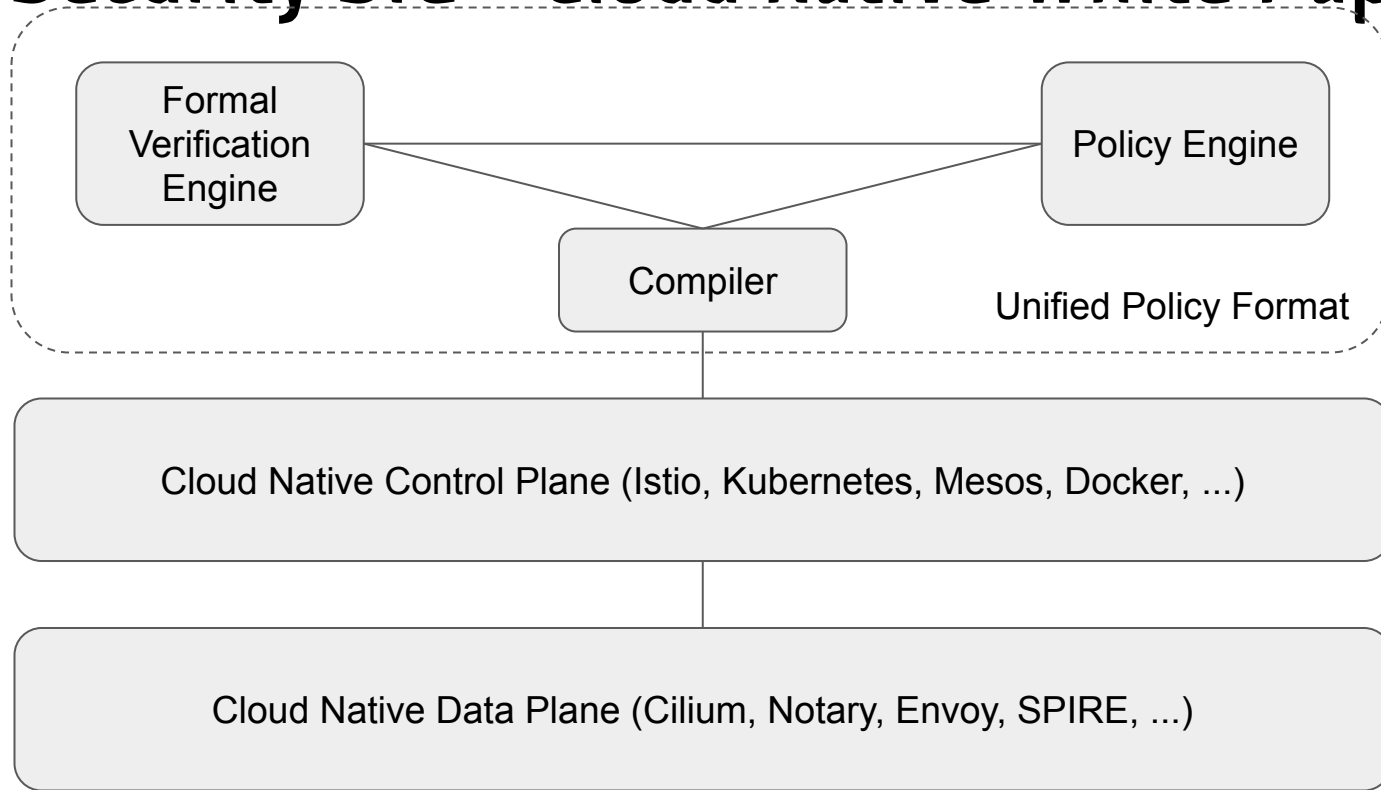
- **Improved ownership**

- Cross-namespace
- "Object pairs" or other way to easily tie lifecycles together

# CNCF Wide Collaboration



# CNCF Security SIG - Cloud Native White Paper



**Semantic + Control = Architecture**

# Contact and Contribute

# WG Facts

- Feel free to join the weekly meeting or leave a note on the meeting minute doc (<https://goo.gl/auTfy2> ) if you have more interesting topics or projects could be used for **case studies** !
- Find us at **#wg-policy on slack**, propose any new interesting idea like we talked here for futuristic open source study !
- Add **label wg-policy for your KEPs** if it is policy related !

**Thank you !**

**Q & A**