Nephio for Intent-Driven Security Automation

Author: Rahul Jadhav, Project SE-RAN – The Security-Enhanced Radio Access Network (5GSec.com), Accuknox Inc.

The Security Architecture of 5G relies on advanced techniques that involve disaggregation of RAN, the democratization of the RAN control plane where multiple vendors can now push their workloads in the control plane and consume events and provide control operations, Network Slicing that allows various applications to operate on the same physical resources, and the ability to deploy part of the components in cloud native architectures.

This is a game changer since this would allow service providers to improve their feature release velocity, benefitting end users. However, the biggest challenges/trade-offs here are, firstly, how to automate the deployment itself and secondly, how to ensure the security of the applications, user data, and privacy concerns.

Nephio's R1 Release highlights

Nephio's R1 release focussed primarily on making sure that the underlying automation layer has strong foundational elements. Nephio decided to use k8s as an orchestration solution for domain automation and then further bridge the gaps from Day2 manageability perspectives. For example, the use of kpt for managing Configuration as Data greatly simplifies managing the Kubernetes deployments by ensuring that the configuration data is the source of truth and stored separately from the live state. Furthermore, Nephio's prebuilt packages allow for ease of experimentation as well as provide out-of-the-box elements for telco deployments such as 5G core network functions, User & Control plane capacity handling.

Intent-Driven Security Automation: Why bother?

Security automation has traditionally been a difficult task. Part of the problem lies in the fact that organizations try to deploy tools and engines rather than focus on the security problem at hand. The security threat model might significantly differ from deployment to deployment even though the underlying deployment modules remain the same. For example, the threat model for private 5G is very different from that of public 5G and hence the security model differs as well. The aim for any organization should be to state its security goal/intents and the underlying tooling should be able to convert these goals/intents into actionable elements. These actionable elements could be in the form of policies/rules depending on certain policy engines and could differ from deployment to deployment. If the intent could not be satisfied, the automation should have a way to highlight to the user that there is a gap between what the intent was and what could be achieved. More importantly, when the deployments change at runtime, the security elements

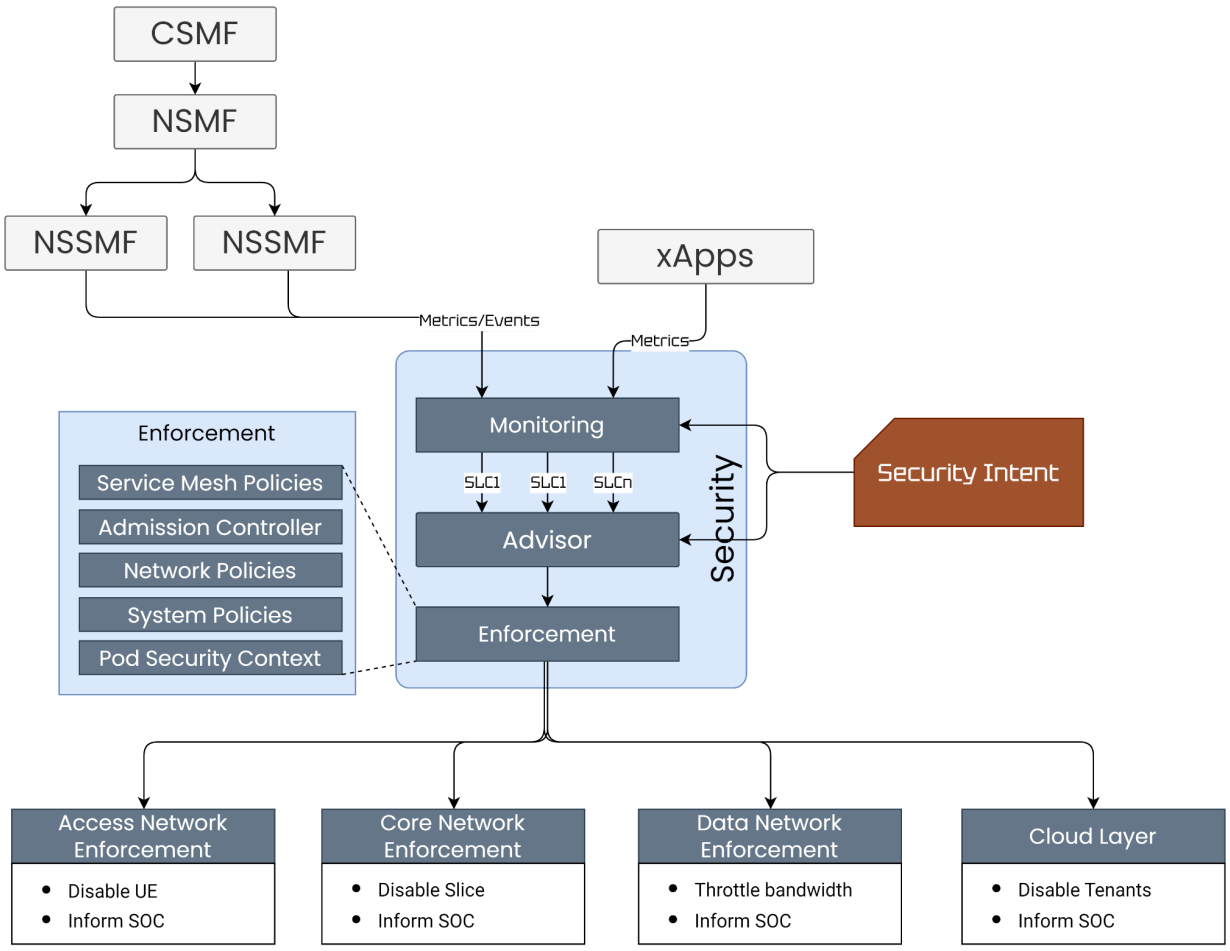
should automatically adjust to the change in the deployments and should not ideally require human intervention.
A recent Nephio article about security in the context of enhanced O-RAN Networks sheds more light on specific 5G Control and Data Plane security and possible solution space.

How do Nephio's fundamentals enable Intent-Driven Security Automation?

One of the fundamental visions of Nephio is to separate the intent from the execution. While it may be argued that this segregation already exists in some shape or form if you use k8s based deployment model, the challenge is to separate Intent from the actual k8s resource. Kubernetes already allows one to specify a security policy that can be deployed across deployments. However, the security policy is usually tied to the policy engine. E.g., NetworkPolicy is implemented by the CNI, and the admission controller policies are implemented by the corresponding controllers such as Kyverno or Gatekeeper.

Decoupling security intent from the actual policies/rules allows:

1. Dynamically implement policies/rules given the specific engines available in the target environment
2. It enables a defense-in-depth mechanism since multiple engines could handle the same intent under the hood.
3. Multiple policies or engines may be used to satisfy a given intent.
4. The alerts/telemetry that is generated should have the context of the intent.



Sample Use-cases/Scenarios

Do not allow egress access for KPIMon xApp

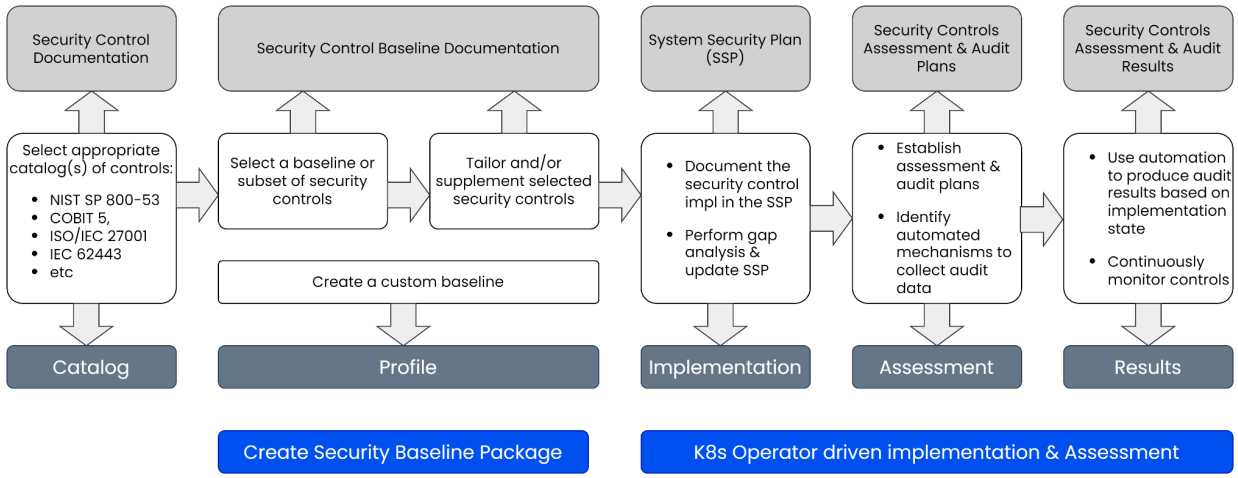
This security use-case can be fulfilled in multiple ways i.e., using a Network policy or using a Pod Security Context, or using a system's hardening tool such as KubeArmor. The Pod Security Context or a KubeArmor policy could be used to limit egress network access by any process within the workloads.

Protect volume-mounted sensitive assets in *onos-kpimon* xApp

In the context of Kubernetes, the sensitive assets can be protected by ensuring it has the right access level from the API perspective. Further, the access can be controlled at the file system level ensuring that only the allowed part of the sensitive assets are exposed to the internal processes. This provides a multi-level security, first using API security engines such as Service Mesh providers and secondly, using systems hardening enforcers such as Pod Security Control.

How does this align with other industry initiatives?

One such initiative is OSCAL which provides a set of formats expressed in XML, JSON, and YAML. These formats provide machine-readable representations of control catalogs, control baselines, system security plans, and assessment plans and results.
The specification is pretty exhaustive but it lacks real-world end-to-end implementation. There are parts of the solutions that already leverage OSCAL specs, for example, the use of OSCAL for publishing assessment results in a standard format. Nephio can provide the ability to instantiate or leverage the OSCAL spec to most of its extent since Nephio provides the right level of abstraction as well as specific instantiation using the domain automation model.



Summary

Nephio's domain automation framework provides a great way to extend it for holistic security needs not only by providing specific instantiation of achieving the automation using the k8s + Configuration-as-Data model but also by ensuring that multiple community members and vendors can collaborate openly. The final aim is to reduce the security risks in any deployment by providing an automated security blueprint instantiation along with the actual deployment.