Case 2 discussion

I2NSF virtual meeting

September 6th 2017

Rafael Marín López and Gabriel López Millán

University of Murcia

Case 2: no IKEv2 in the NSF

- The idea starts from this text in RFC 4301:
  - "…but other automated key distribution techniques MAY be used".
- Our I-D: SDN-based **automated** key distribution techniques
- This approach is "more compliant" with the SDN paradigm
  - Data plane → IPsec stack in the NSF
  - Control plane → Key management in the SDN

Implementation issues

- Assuming we use NETCONF, NETCONF server is tight to the NSF
- We do not think there is an issue: the same is implemented by any other key management technique
- Under kernel standpoint there is an API to interact with. The NETCONF server will use this API instructed by the controller. We follow the API (PK_KEYv2 or XFRM) rules and parameters
- Regarding policies, all access control system is inbuilt in the NETCONF protocol and specified in the YANG model
- In general, policies are managed by the SDN controller

Chicken and egg problem

- NETCONF uses SSH/TLS SA
- One single SSH/TLS SA will be used anyway to protect configuration and monitoring for any other service in the NSF (regardless of the IPsec management)
- A constrained device will usually need anyway a constrained interface and SA (e.g. DTLS) for management

NSA for attack

- Regardless of the IPsec management, the controller will **always** be a target for attack. Strong security measurements are required to protect the controller anyway (SDN paradigm dictates that, not particular to our I-D)
- If the controller reboots it can still gather information from the NSFs (in fact the NSFs will realize that something happened with the controller) and act accordingly (e.g. rekey all the IPsec SA with fresh information)
- No need to store permanently actual keys in the SDN controller beyond the time they are randomly generated.
- In any case, we must add in the security considerations of the I-D the implications in case 2.

Nonces reuse

- Our I-D shows a SDN-based **automated key management**
- The controller always sends fresh information including keys, nonces or any other required parameter as any other KMP will do.
- When the NSF restarts, the kernel will have SAD empty.
- The controller **never** sends old previous keys (nor even have them stored)

Latency issues

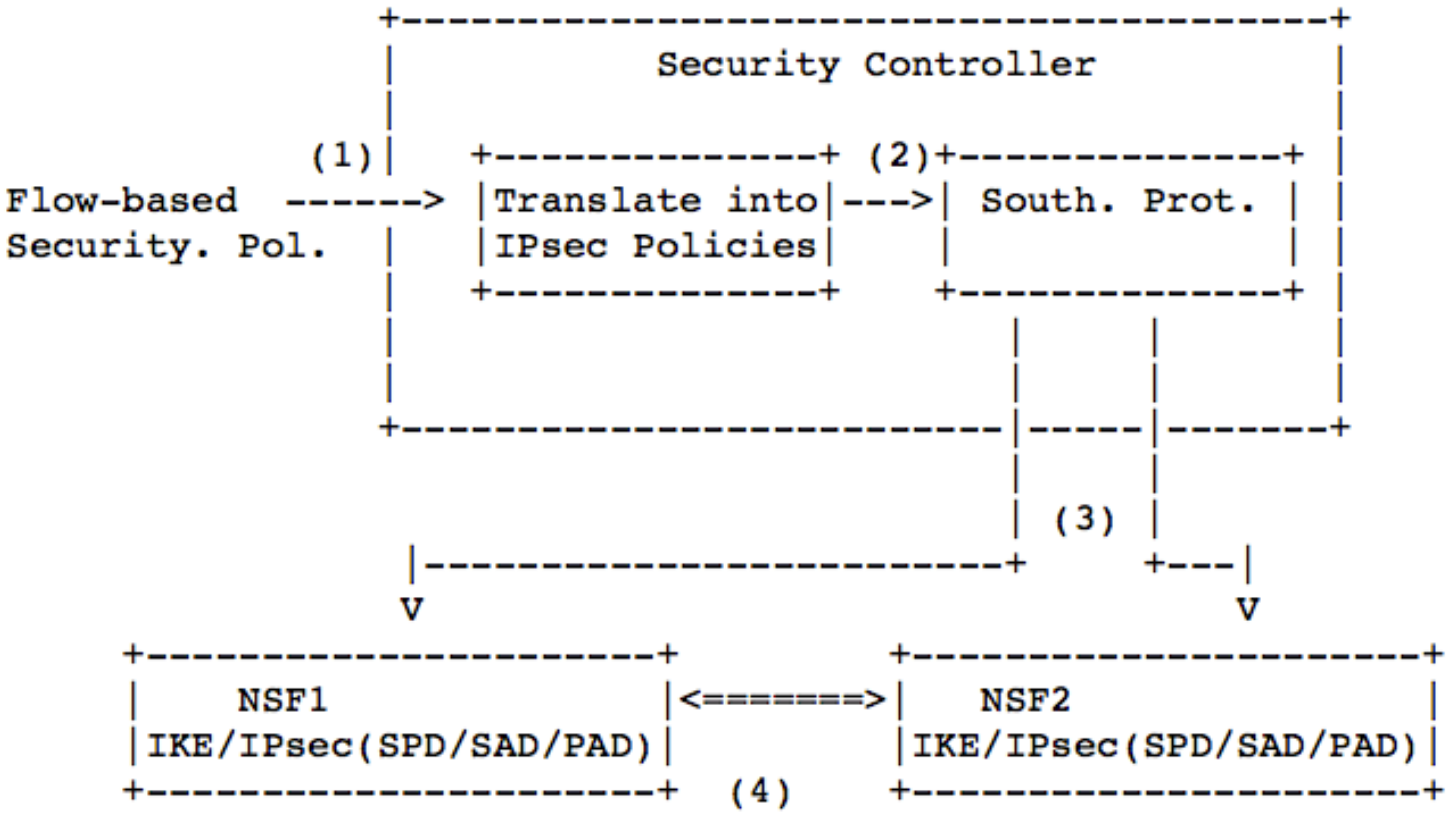
- Latency is a key aspect in many of the applications that use SDNs management (regardless of the SDN-based IPsec management)
- Between the NSFs and the controller we can assume there is a management network
- IKEv2 negotiation happens between two NSFs over the data network
- Why should we assume the latency in the management network (NSF←→Security controller) is worse than data network?
- We have tested this (e.g. new SA trigger) and we have not observed any issue at the moment

Backup slides

I2NSF virtual meeting

September 6th 2017

Case 1 with a single controller



Case 2 with a single controller