Advanced beyond SSH: Automating with APIs

Nick Russo
NETWORK ENGINEER

@nickrusso42518 www.njrusmc.net

Agenda

- Why do we need APIs?
- NETCONF RPCs (and a bit on YANG)
- RT management with NETCONF
- RESTCONF HTTP requests
- Nornir + RESTCONF!

Application Programming Interface

set of operations built for standardized management of network devices. These operations are easily consumed by programmers and utilize structured data.

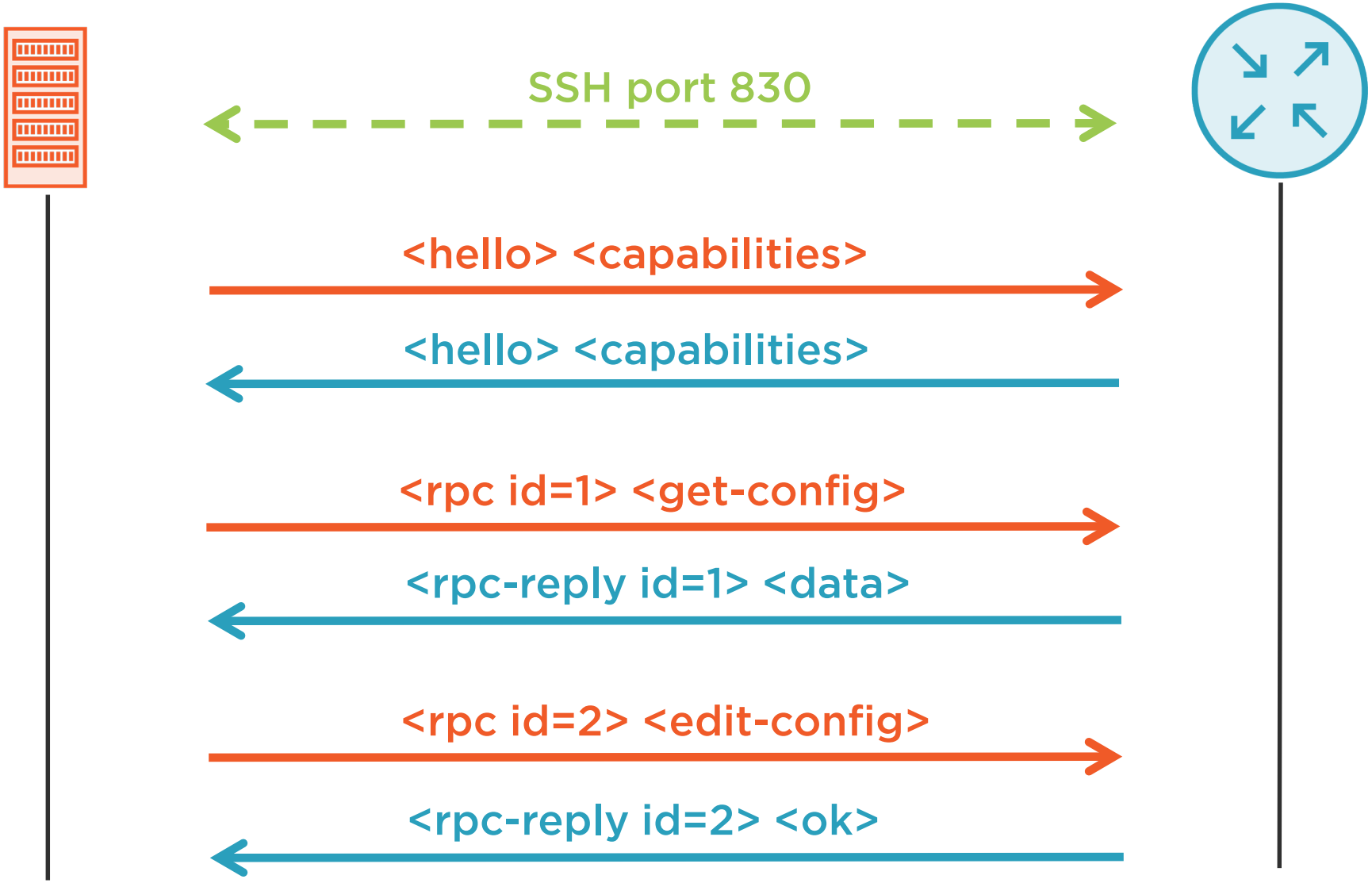
Who Cares?

Simplified implementation

Structured data

Standardized abstraction

Introducing NETCONF



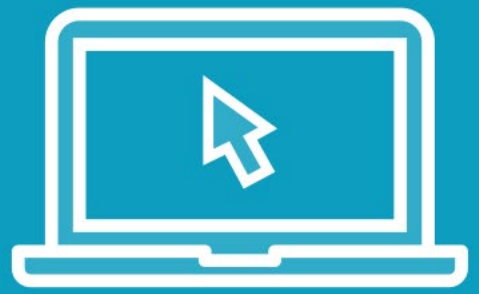
```
// ios-types
typedef asn-ip-type {
  type string {
    pattern "complex regex";
  }
}

// native VRF model
list export {
  description "lots of text";
  key "asn-ip";
  leaf asn-ip {
    type ios-types:asn-ip-type;
  }
}
```

- C-style modeling language
- What must an RT look like?
- We store export RTs in a list
- Each element has one key
- The value should conform to the type defined above

```
<export>
  <asn-ip>65000:1</asn-ip>
</export>
<import>
  <asn-ip>65000:1</asn-ip>
  <asn-ip>65000:2</asn-ip>
</import>
```

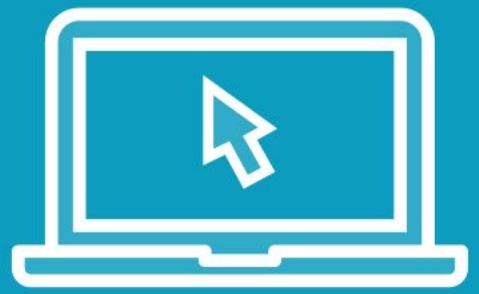
- List named "export"
- Elements named "asn-ip" with values conforming to typedef
- List named "import" with similar values

Demo



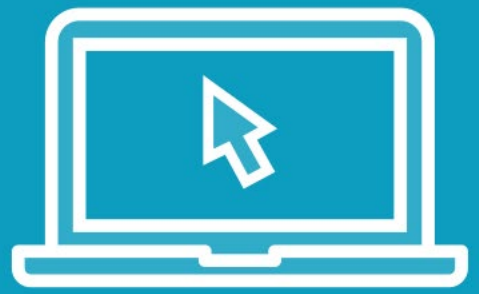
VRF "get-config" RPC with ncclient

Demo



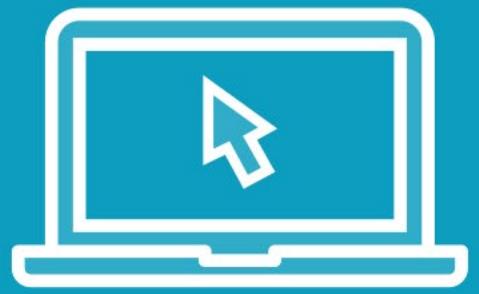
VRF "edit-config" RPC with ncclient

Demo



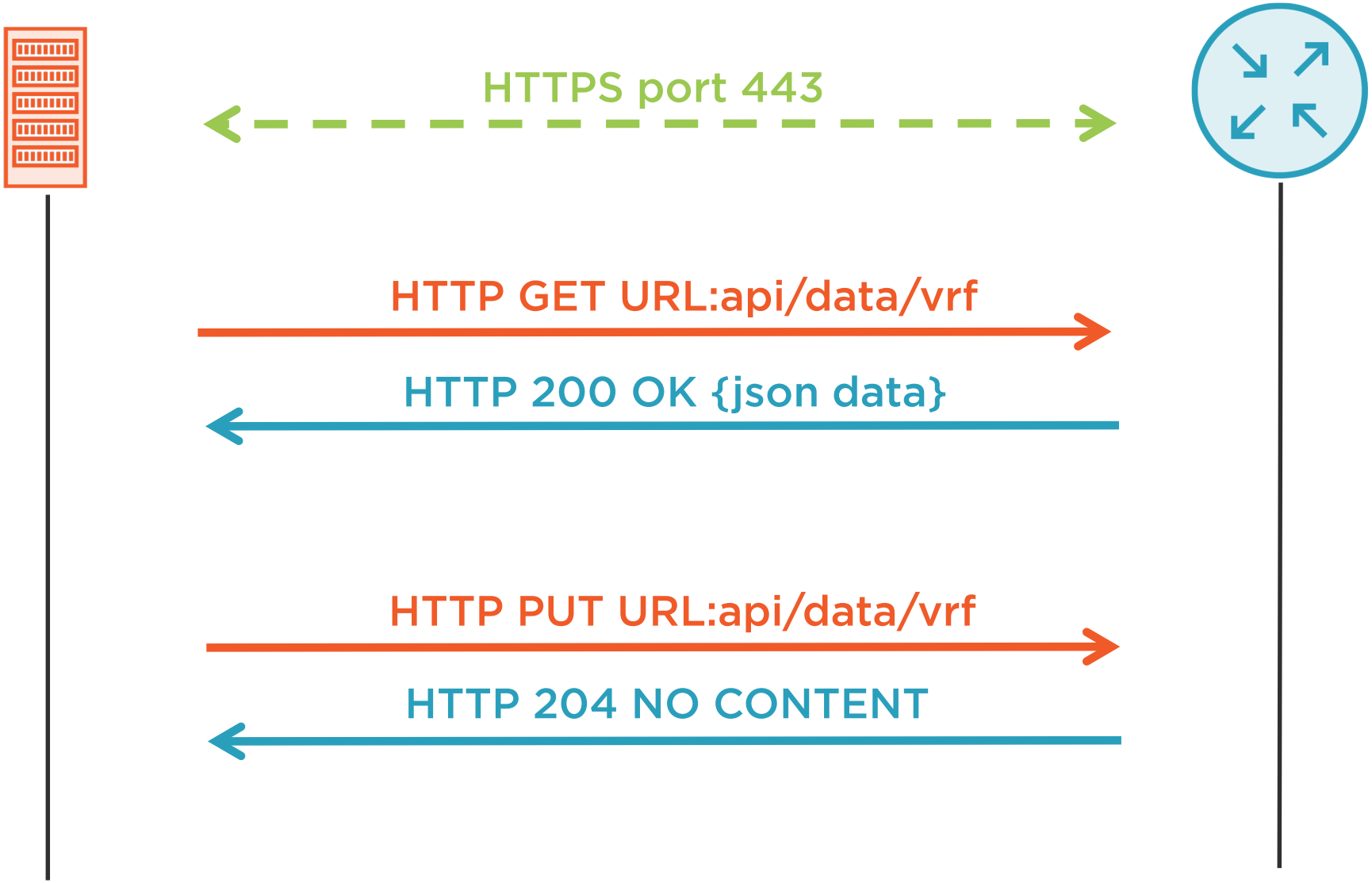
NETCONF "get_config" with Nornir

Demo



NETCONF "edit_config" with Nornir

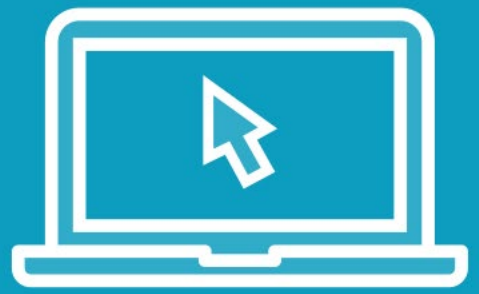
Introducing RESTCONF



```
"export": [
  {
    "asn-ip": "65000:1"
  }
],
"import": [
  {
    "asn-ip": "65000:1",
    "asn-ip": "65000:2"
  }
]
```

- List named "export"
- Elements named "asn-ip" with values conforming to typedef
- List named "import" with similar values

Demo



RT management via RESTCONF + Nornir

Comparing NETCONF and RESTCONF

NETCONF	RESTCONF
SSH transport, port 830	HTTP transport, port 443
Custom RPCs	HTTP operations
XML only	XML or JSON
RFC 6241 and 6242	RFC 8040
Nornir support!	Nornir support!