Critical Infrastructure-as-Code (CIaC)

Matthew R. Backes

S4x2020

01/21/2020

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Rights and Markings Statements

This material is based upon work supported by the USD NON-LINE under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the USD NON-LINE.

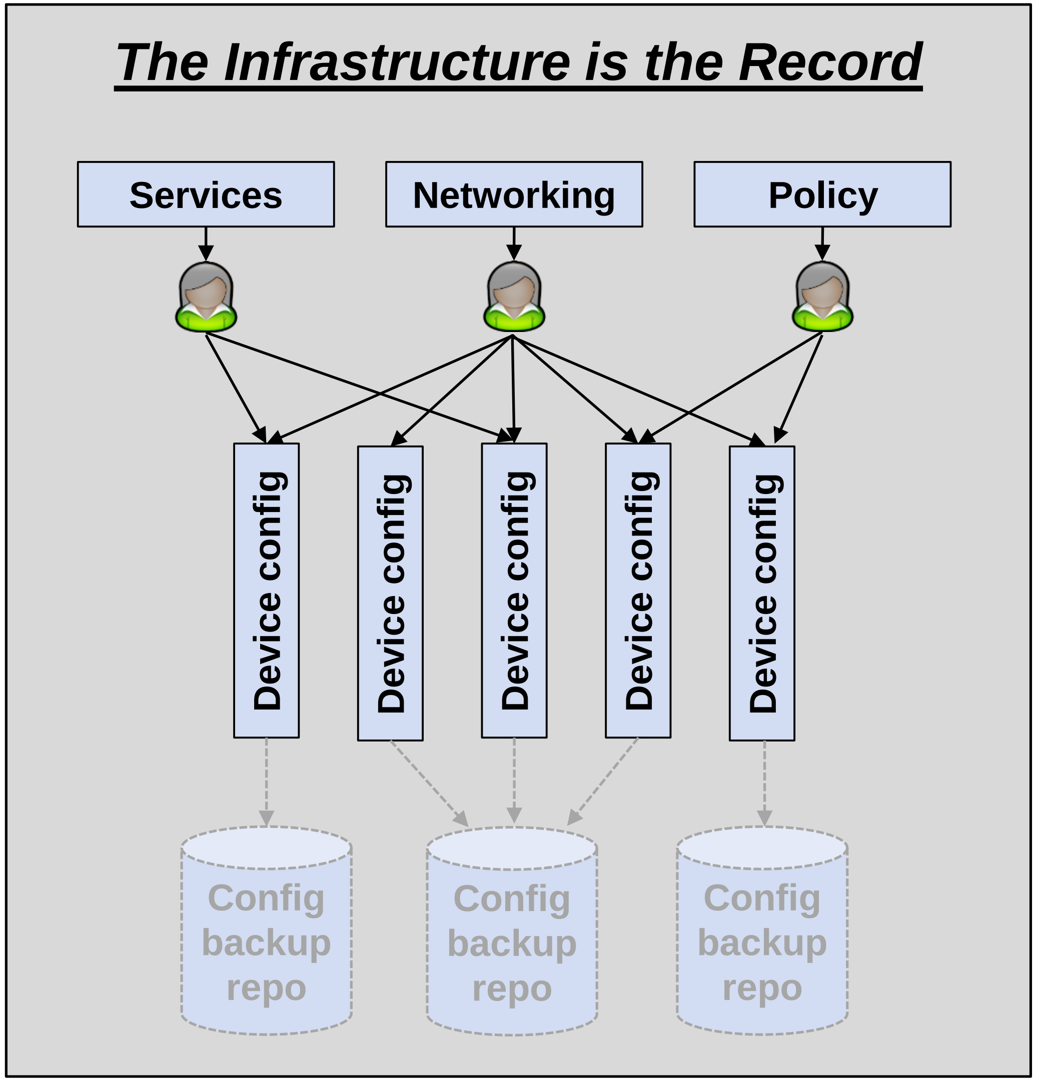
Asserting Control by Improving Manageability

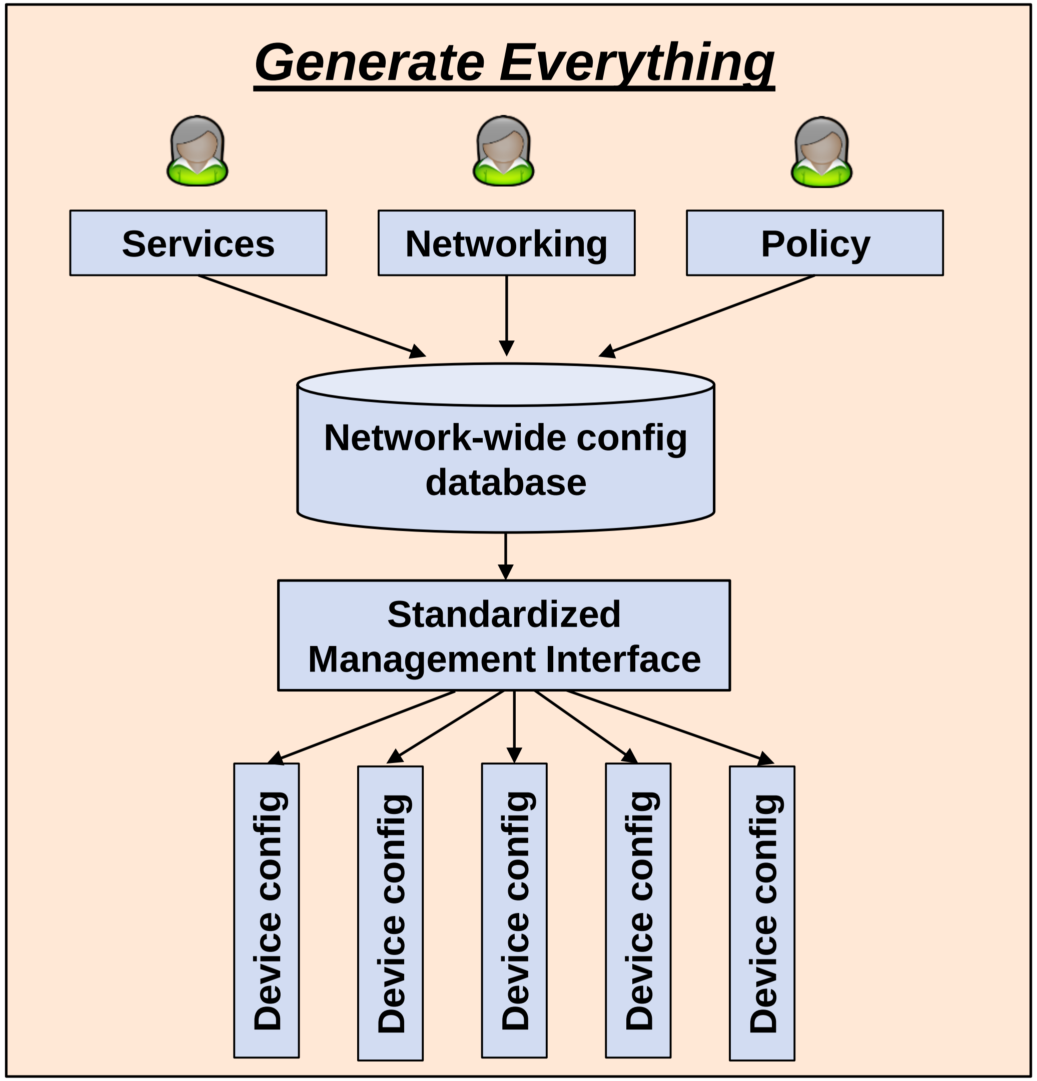
- **Safety requires positive control. Positive control requires integrity. Integrity requires well-managed systems. Cybersecurity underpins these properties.**
- **Current approaches to CM are *ad hoc* and vendor-specific**
  - **System configurations need to be well-understood and manageable by the end users**
  - **False dichotomy between active and passive methods**
- **We need to make our management approaches:**
  - **Principled**
  - **Systematic**
  - **Operator/Engineer-friendly**
  - **Enable identification, protection, detection, response, and recovery**

We need to elevate configuration management as a first-class objective

CM: Configuration Management

Configuration Management Philosophies





Impetus

- Systems in operation are rarely well-managed across the system lifecycle
- Dynamic hardware-in-the-loop testbed reveals configuration management nightmare that could benefit from systematic code-based approach
- Could we do this similarly to how modern IT infrastructure is managed?
- "Critical infrastructure": any device or machine reachable via IP or serial networks
- Purpose of talk:
  - Share lessons learned
  - Propose a way forward
  - Solicit feedback

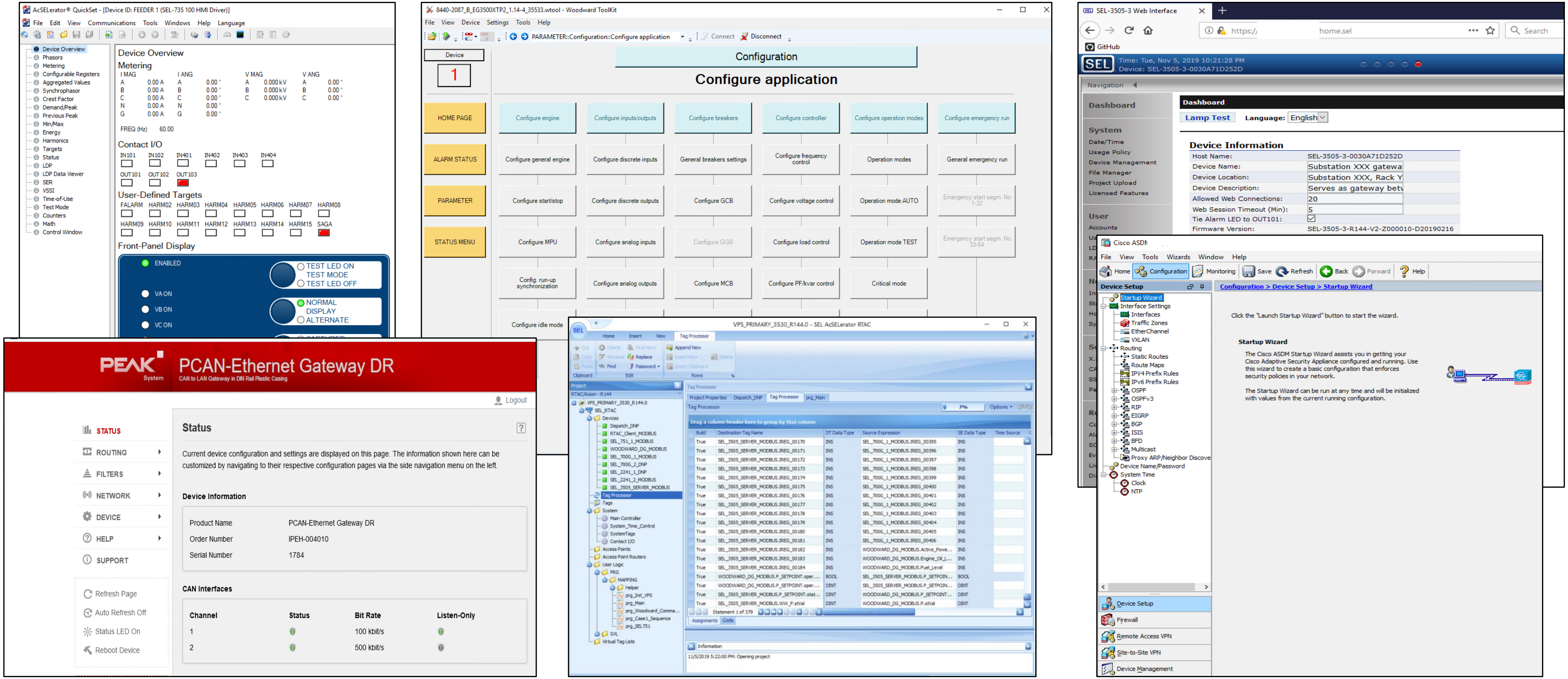
Unknown baselines and unmanaged devices leave control systems in a fragile state that is less than weakly defensible

Challenges with Managing Infrastructure

- **It’s not managed**
  - **“I’ve commissioned a good bit of [field device type], never had anybody doing backup on the settings files.”**
- **Machine and device heterogeneity**
- **As-builts are commonly the only source of documentation (if they exist)**
- **Fragile infrastructure: access mechanisms are primitive**
- **Many dependencies: vendor software**
- **Configuration drift:**
  - **“…Those entities typically did not incorporate configuration changes into baselines due to *overlooking a manual component* of the workflow process.” – FERC[1]**

[1] https://www.ferc.gov/legal/staff-reports/2019/2018-report-audits.pdf

Configuration is Cumbersome



AcSELerator® QuickSet - [Device ID: FEEDER 1 (SEL-735 100 HMI Driver)]

Device Overview

Metering

I MAG		I ANG		V MAG		V ANG	
A	0.00 A	A	0.00 °	A	0.000 kV	A	0.00 °
B	0.00 A	B	0.00 °	B	0.000 kV	B	0.00 °
C	0.00 A	C	0.00 °	C	0.000 kV	C	0.00 °
N	0.00 A	N	0.00 °				
G	0.00 A	G	0.00 °				

FREQ (Hz) 60.00

Contact I/O

IN101	IN102	IN401	IN402	IN403	IN404
□	□	□	□	□	□

OUT101	OUT102	OUT103
□	□	🟥

User-Defined Targets

FALARM	HARM02	HARM03	HARM04	HARM05	HARM06	HARM07	HARM08
□	□	□	□	□	□	□	□
HARM09	HARM10	HARM11	HARM12	HARM13	HARM14	HARM15	SAGA
□	□	□	□	□	□	□	🟥

Front-Panel Display

8440-2087_B_EG3500XTP2_1.14-4_35533.wtool - Woodward ToolKit

File View Device Settings Tools Help
PARAMETER::Configuration::Configure application
Connect
Disconnect
Device
1
HOME PAGE
ALARM STATUS
PARAMETER
STATUS MENU

Configuration

Configure application

SEL-3505-3 Web Interface

Time: Tue, Nov 5, 2019 10:21:28 PM
Device: SEL-3505-3-0030A71D252D

Dashboard

Language: English

Device Information

Host Name:	SEL-3505-3-0030A71D252D
Device Name:	Substation XXX gatewa
Device Location:	Substation XXX, Rack Y
Device Description:	Serves as gateway betv
Allowed Web Connections:	20
Web Session Timeout (Min):	5
Tie Alarm LED to OUT101:	☑
Firmware Version:	SEL-3505-3-R144-V2-Z000010-D20190216

PEAK System

PCAN-Ethernet Gateway DR

CAN to LAN Gateway in DIN Rail Plastic Casing

Status

Current device configuration and settings are displayed on this page. The information shown here can be customized by navigating to their respective configuration pages via the side navigation menu on the left.

Device Information

Product Name	PCAN-Ethernet Gateway DR
Order Number	IPEH-004010
Serial Number	1784

CAN Interfaces

Channel	Status	Bit Rate	Listen-Only
1	●	100 kbit/s	●
2	●	500 kbit/s	●

VPS_PRIMARY_3530_R144.0 - SEL AcSELerator RTAC

Tag Processor

Build	Destination Tag Name	DT Data Type	Source Expression	SE Data Type	Time Source
True	SEL_3505_SERVER_MODBUS.IREG_00170	INS	SEL_700G_1_MODBUS.IREG_00395	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00171	INS	SEL_700G_1_MODBUS.IREG_00396	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00172	INS	SEL_700G_1_MODBUS.IREG_00397	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00173	INS	SEL_700G_1_MODBUS.IREG_00398	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00174	INS	SEL_700G_1_MODBUS.IREG_00399	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00175	INS	SEL_700G_1_MODBUS.IREG_00400	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00176	INS	SEL_700G_1_MODBUS.IREG_00401	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00177	INS	SEL_700G_1_MODBUS.IREG_00402	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00178	INS	SEL_700G_1_MODBUS.IREG_00403	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00179	INS	SEL_700G_1_MODBUS.IREG_00404	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00180	INS	SEL_700G_1_MODBUS.IREG_00405	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00181	INS	SEL_700G_1_MODBUS.IREG_00406	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00182	INS	WOODWARD_DG_MODBUS.Active_Powe...	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00183	INS	WOODWARD_DG_MODBUS.Engine_Oil_L...	INS	
True	SEL_3505_SERVER_MODBUS.IREG_00184	INS	WOODWARD_DG_MODBUS.Fuel_Level	INS	
True	WOODWARD_DG_MODBUS.P_SETPOINT.oper....	BOOL	SEL_3505_SERVER_MODBUS.P_SETPOIN...	BOOL	
True	WOODWARD_DG_MODBUS.P_SETPOINT.oper....	DINT	SEL_3505_SERVER_MODBUS.P_SETPOIN...	DINT	
True	SEL_3505_SERVER_MODBUS.P_SETPOINT.stat...	DINT	WOODWARD_DG_MODBUS.P_SETPOINT...	DINT	
True	SEL_3505_SERVER_MODBUS.WW_P.stVal	DINT	WOODWARD_DG_MODBUS.P.stVal	DINT	

Statement 1 of 379

Information

11/5/2019 5:22:00 PM: Opening project

Cisco ASDM

Configuration > Device Setup > Startup Wizard

Click the "Launch Startup Wizard" button to start the wizard.

Startup Wizard

The Cisco ASDM Startup Wizard assists you in getting your Cisco Adaptive Security Appliance configured and running. Use this wizard to create a basic configuration that enforces security policies in your network.

The Startup Wizard can be run at any time and will be initialized with values from the current running configuration.

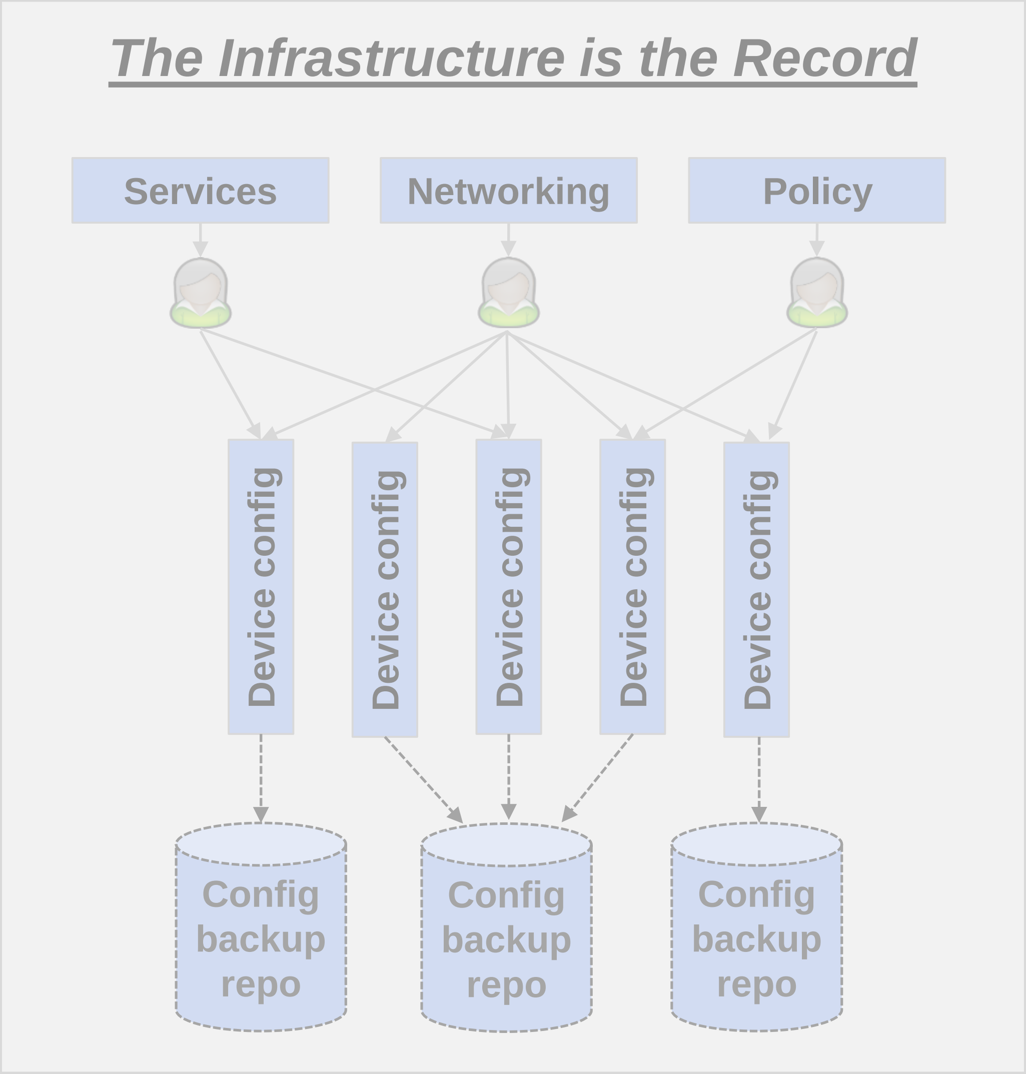
Configuration Management Improvements

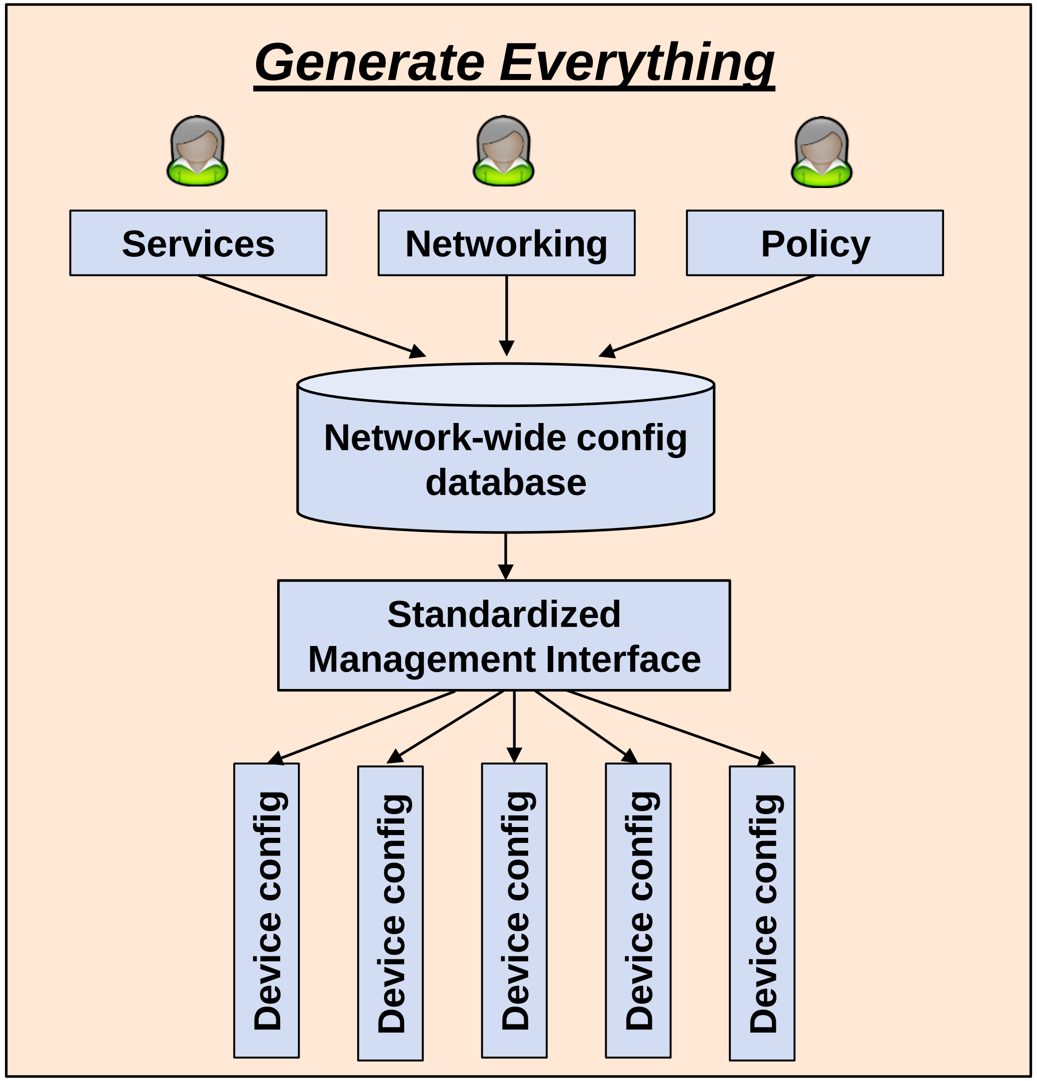
Status Quo	Desired End State
Almost always require another piece of software	**Reduced complexity** – Eliminate application software; minimize the TCB
Retrieve from multiple interfaces, each w/ distinct functionality	**Standardized interface** – A single, well-defined interface, e.g., NETCONF
Cannot be retrieved securely	**Secured Comms** – Public key SSH
Proprietary, non-human-readable file formats	**Interoperable Data Formats** – YAML, JSON
Cannot quickly be compared via text processing tools	**Revision Controlled with secured fallback** – git, diff, RAUC

TCB: Trusted Computing Base
SSH: Secure Shell

YAML: Yet Another Markup Language
JSON: Javascript Object Notation
RAUC: Robust Auto-Update Controller

Infrastructure-as-Code (IaC) Goals and Practices

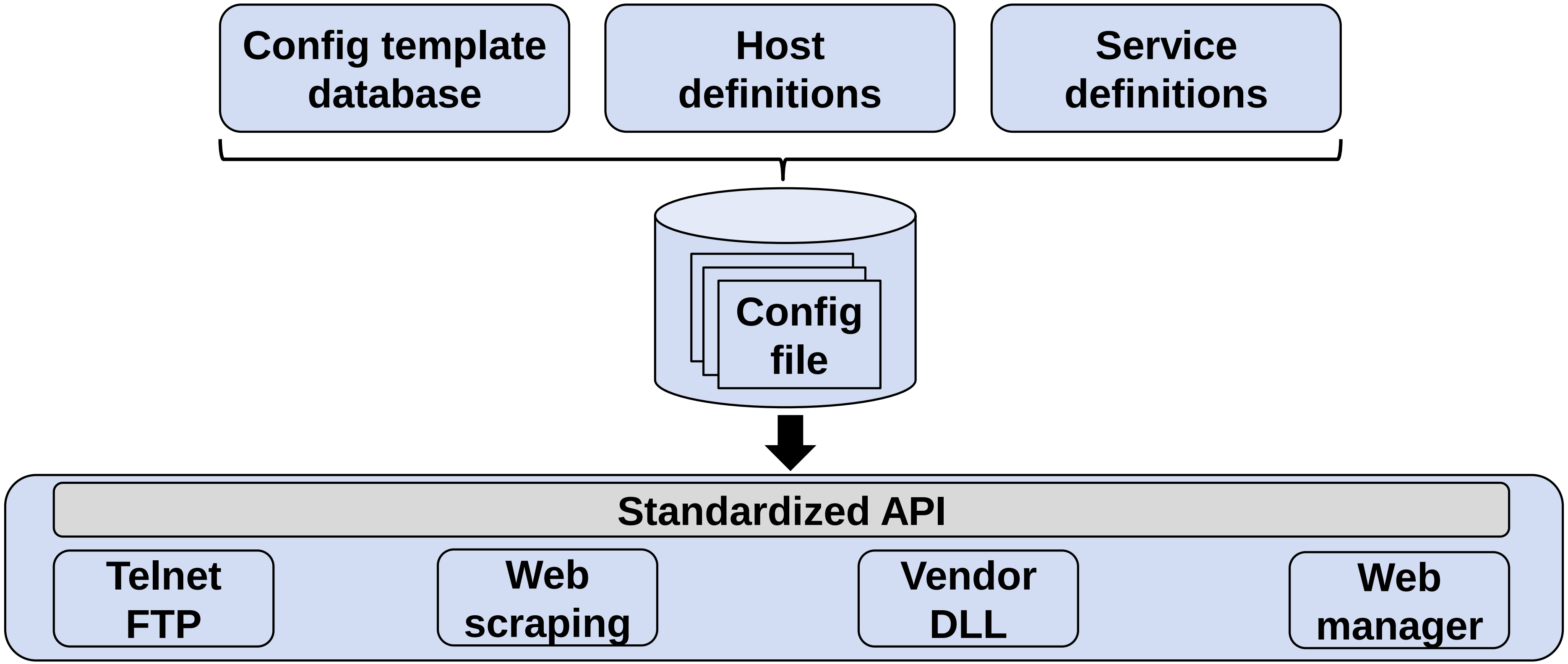




Control System Configuration Management Deployment

ANSIBLE

JSON



Config template database

Host definitions

Service definitions

Config file

Standardized API

Telnet FTP

Web scraping

Vendor DLL

Web manager

Protection Relays

RTUs

Genset controllers

Serial-to-Ethernet Converters

API: Application Programming Interface
RTU: Remote Terminal Unit

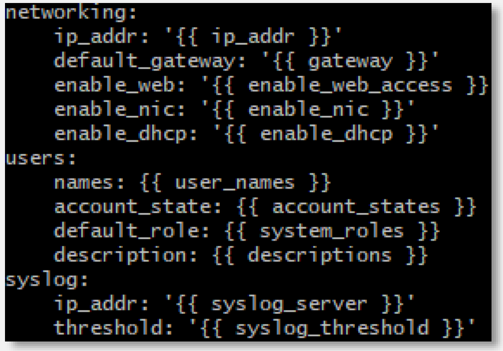
FTP: File Transfer Protocol
DLL: Dynamical Link Library

Defining a Configuration Workflow: Leveraging Ansible

Standardize Templates → Define Environment → Define Device and Service Functionality → Generate Artifacts

ANSIBLE

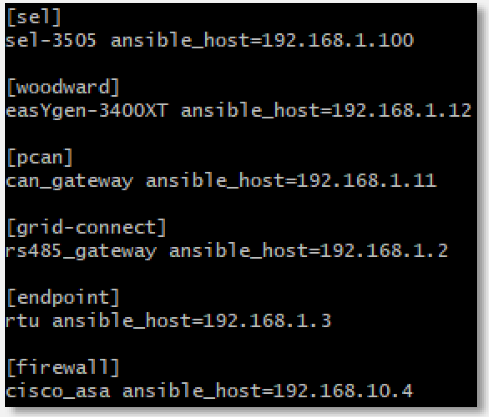
Configuration template



```
networking:
    ip_addr: '{{ ip_addr }}'
    default_gateway: '{{ gateway }}'
    enable_web: '{{ enable_web_access }}'
    enable_nic: '{{ enable_nic }}'
    enable_dhcp: '{{ enable_dhcp }}'
users:
    names: {{ user_names }}
    account_state: {{ account_states }}
    default_role: {{ system_roles }}
    description: {{ descriptions }}
syslog:
    ip_addr: '{{ syslog_server }}'
    threshold: '{{ syslog_threshold }}'
```

jinja2

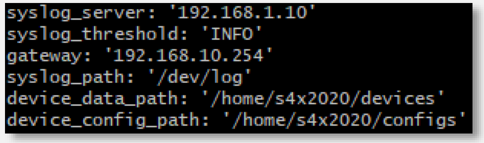
Host inventory



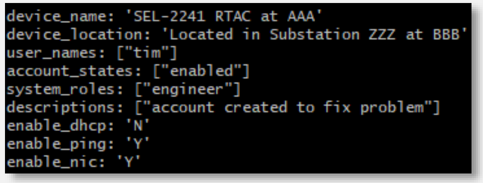
```
[sel]
sel-3505 ansible_host=192.168.1.100

[woodward]
easYgen-3400XT ansible_host=192.168.1.12

[pcan]
can_gateway ansible_host=192.168.1.11

[grid-connect]
rs485_gateway ansible_host=192.168.1.2

[endpoint]
rtu ansible_host=192.168.1.3

[firewall]
cisco_asa ansible_host=192.168.10.4
```

YAML

Variables:
- Global
- Vendor
- Device



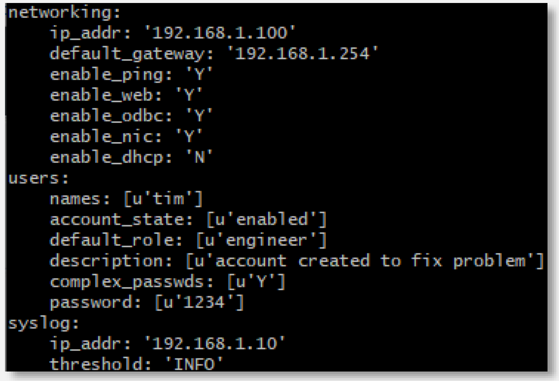
```
syslog_server: '192.168.1.10'
syslog_threshold: 'INFO'
gateway: '192.168.10.254'
syslog_path: '/dev/log'
device_data_path: '/home/s4x2020/devices'
device_config_path: '/home/s4x2020/configs'
```



```
device_name: 'SEL-2241 RTAC at AAA'
device_location: 'Located in Substation ZZZ at BBB'
user_names: ["tim"]
account_states: ["enabled"]
system_roles: ["engineer"]
descriptions: ["account created to fix problem"]
enable_dhcp: 'N'
enable_ping: 'Y'
enable_nic: 'Y'
```

YAML

Generated Configuration File

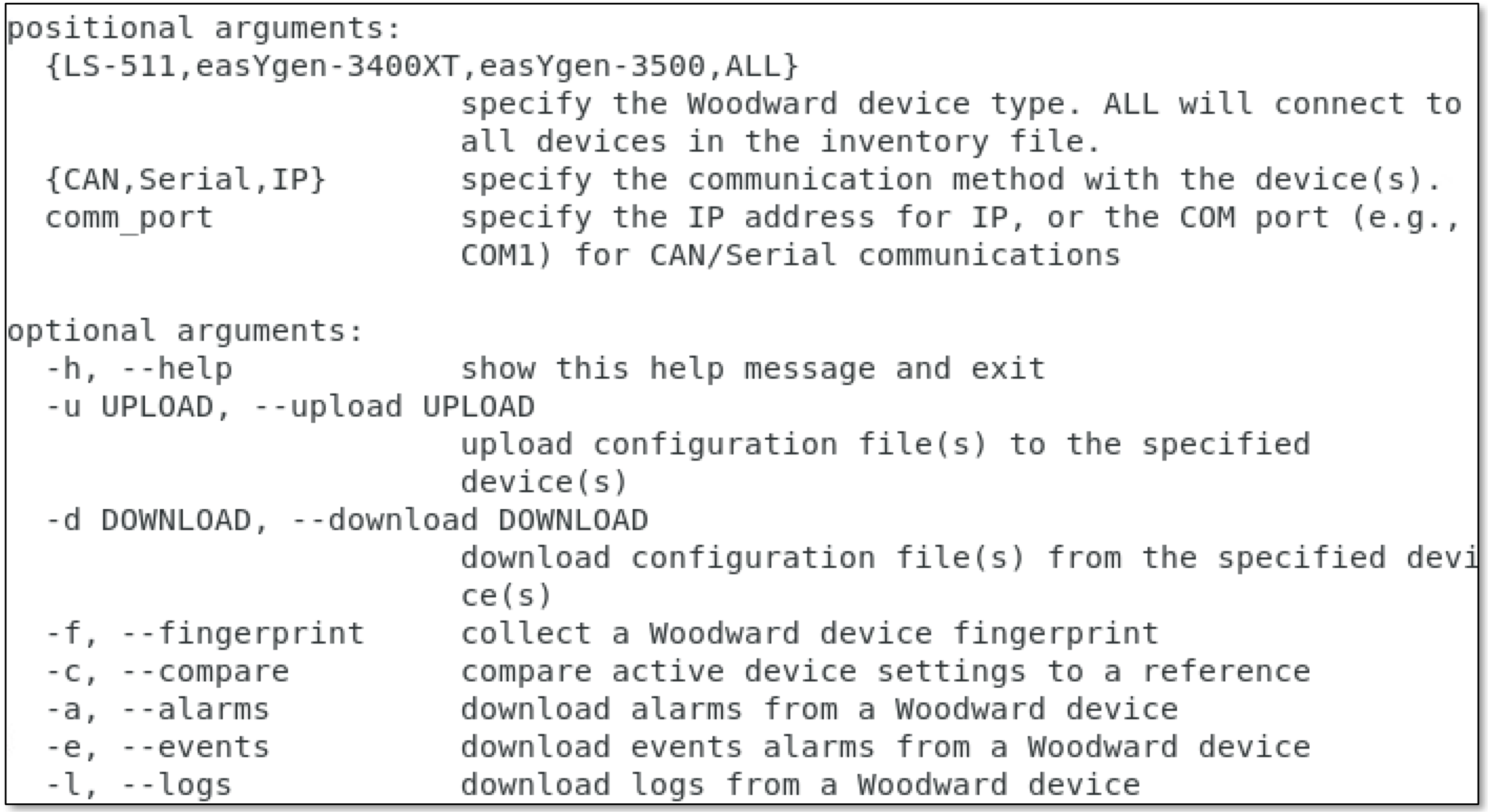


```
networking:
    ip_addr: '192.168.1.100'
    default_gateway: '192.168.1.254'
    enable_ping: 'Y'
    enable_web: 'Y'
    enable_odbc: 'Y'
    enable_nic: 'Y'
    enable_dhcp: 'N'
users:
    names: [u'tim']
    account_state: [u'enabled']
    default_role: [u'engineer']
    description: [u'account created to fix problem']
    complex_passwds: [u'Y']
    password: [u'1234']
syslog:
    ip_addr: '192.168.1.10'
    threshold: 'INFO'
```

YAML

Configurations can be expressed in human-readable format!

Interface Operations



```
positional arguments:
  {LS-511,easYgen-3400XT,easYgen-3500,ALL}
                        specify the Woodward device type. ALL will connect to
                        all devices in the inventory file.
  {CAN,Serial,IP}       specify the communication method with the device(s).
  comm_port             specify the IP address for IP, or the COM port (e.g.,
                        COM1) for CAN/Serial communications

optional arguments:
  -h, --help            show this help message and exit
  -u UPLOAD, --upload UPLOAD
                        upload configuration file(s) to the specified
                        device(s)
  -d DOWNLOAD, --download DOWNLOAD
                        download configuration file(s) from the specified devi
                        ce(s)
  -f, --fingerprint     collect a Woodward device fingerprint
  -c, --compare         compare active device settings to a reference
  -a, --alarms          download alarms from a Woodward device
  -e, --events          download events alarms from a Woodward device
  -l, --logs            download logs from a Woodward device
```

Specifying device and communication method

Defined a minimum set of operations to interact with device configuration and logged info

Device Translators

Vendors	Schweitzer Engineering Laboratories		Woodward	Grid Connect PCAN
Hardware Devices				
	telnetlib ftp	selenium	pythonnet automation.dll	requests telnetlib
Existing Config Mechanisms	• Text files available via ftp • Telnet CLI running SEL ASCII • No SSH	• Can upload files via a browser • Can obtain and parse the firewall ruleset	• Library with methods for automating config • Provides config compare feature	• One provided a REST-like API • Saves settings in .ini format • Provides settings compare feature

CLI: Command Line Interface

CIaC: Lessons Learned

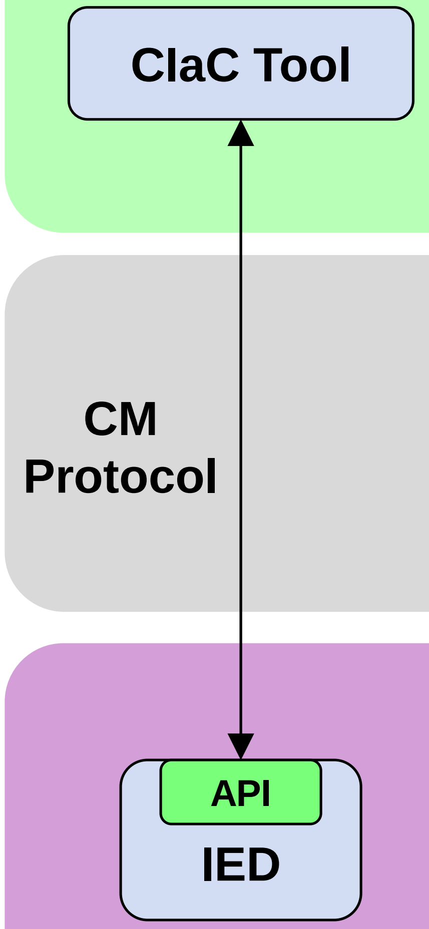
Tread Carefully:

- Interaction mechanisms are not always secure, and are rarely standardized
- Configuration file parsers can be extremely fragile
- Misconfiguring network parameters may affect entire device functionality
- Enabled management interfaces expose additional (sometimes extreme) risk
- Different firmware versions of same device can have different interaction mechanisms

Moving Forward:

- Work is needed by vendors to support secure and safe CM functionality
- Existing IaC tools can be used, but will likely require modification
- Community needs to come together to define requirements and build interoperable toolsets and APIs

Three *Separate* Functions



ClaC Tool requirements:

- Scriptable interface
- Externalized configuration
- Minimize software dependencies

ANSIBLE

HashiCorp Terraform

CM Protocol requirements:

- RPC Paradigm
- Secure access
- Well-defined CM functions (‘get-config’)

NETCONF

OpenFlow

API requirements:

- Simple (RESTful) and secure
- Configs are human-readable
- Event and log retrieval

JUNOS

IED: Intelligent Electronic Device

RPC: Remote Procedure Call

Building on a Solid Foundation

Defensible systems require positive control. We must enforce simplicity and regain control through better manageability

This enables defenders to:

- Facilitate configuration control boards
- Backup and revision-control all device configurations
- Easily declare network-wide policies
- Automatically generate firewall and application whitelist rules
- Provide a way to easily manage Digital Twins – configurations are easily reproduced
- IT/OT integration…we can speak a common language for configuration management

Questions?

- **Thanks!**