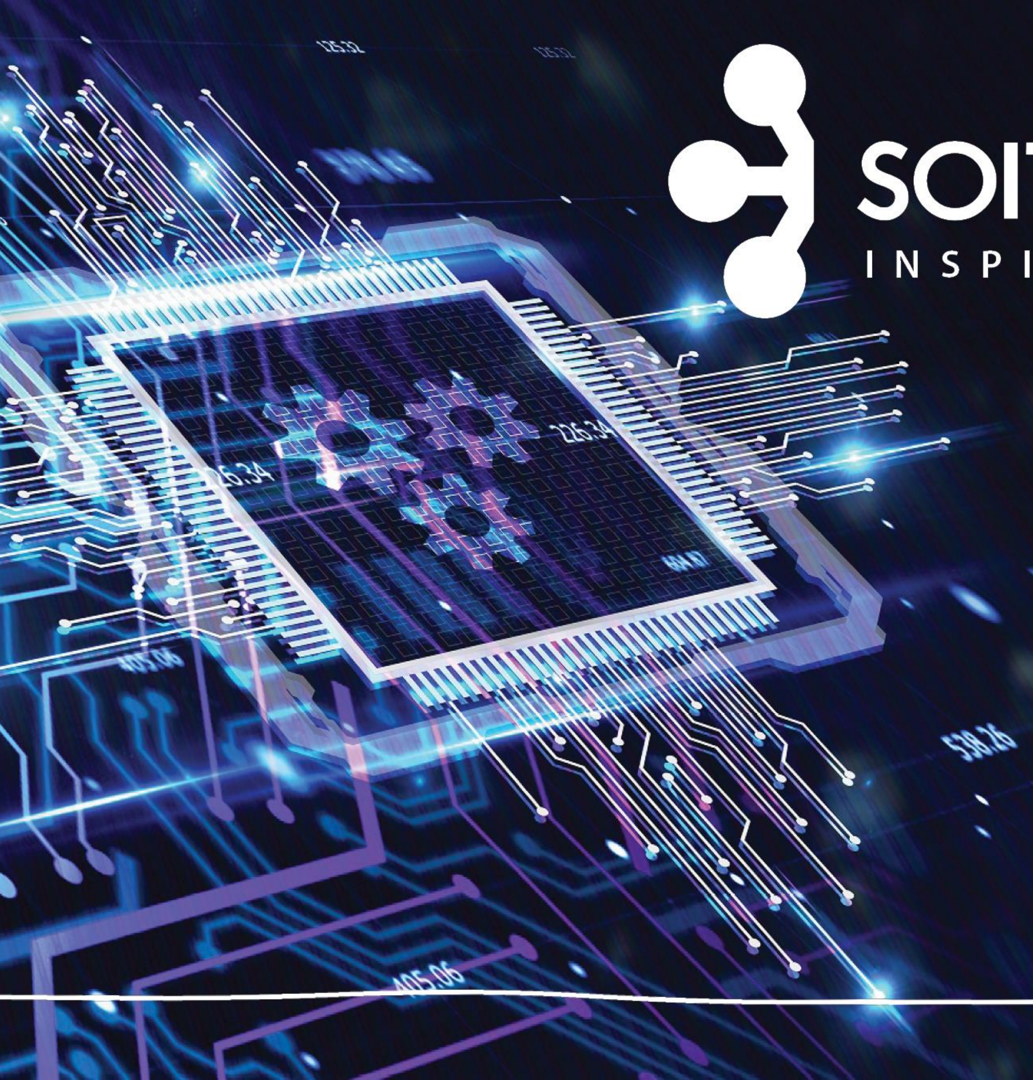


SOITRON* INSPIRÁRIUM

3. 11. 2022

9:00 - 10:00 hod.

Automatizácia: Trend v správe sieťovej infraštruktúry

AGENDA

▪ Presentation	40 min
▪ Discussion and summary	15 min
▪ Microphones are muted	
▪ You can write questions to Q&A	

Part #1/5

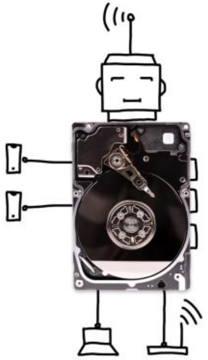
Infrastructure Automation

- Infrastructure automation benefits
- Some facts...
  - Model-Driven Programmability
  - Device-Level vs. Controller-Level Management
  - Imperative and Declarative Method
  - Service Models
- Service Delivery
- Use Cases

Infrastructure Automation

Advantages

- Reduced possibility of (human) mistakes
- Operational efficiency is considerably increased
- Repeatability
- Lower operating cost
- Network downtime is decreased
- More effective staff
- No expert-level staff requirements



Disadvantages

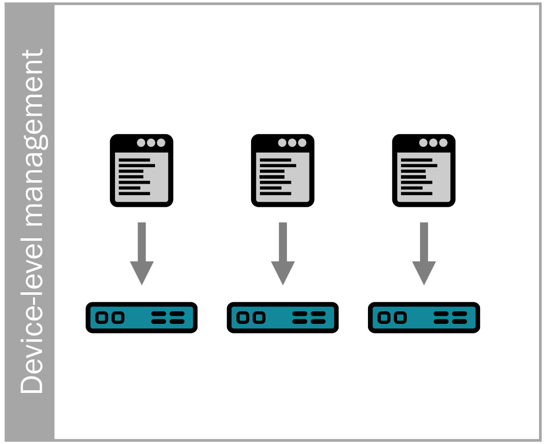
- Longer initialization curve
- Customization is required
- Complexity in modern networks
- Losing the dominance of technology
- Change of thinking

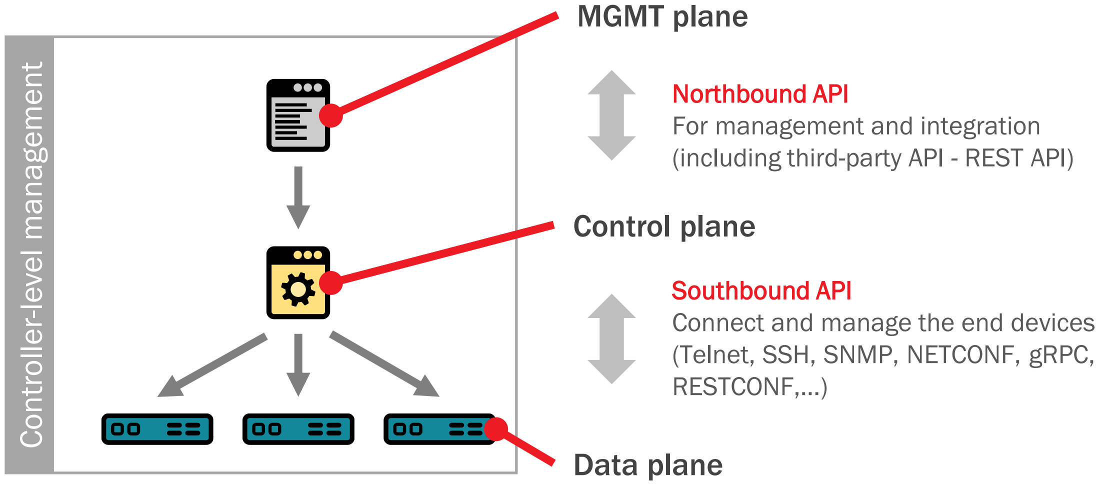
A bit of history (Model-Driven Programmability)

	SNMP	NETCONF	gRPC	RESTCONF
Year	~1980	~2006/2011	2015	2017
Standard	IETF	IETF	Google	IETF
Transport	UDP	SSH	HTTP	HTTP
Resources	OIDs	Paths	ProtoBuf	URLs
Encoding	BER	XML	Binary	JSON, XML,...
Data Modeling	SMI/MIB	YANG	YANG	YANG
	Limited	NW only		Universal

Model-Driven Programmability provides mechanism to install, manipulate and delete configuration (not only network devices)

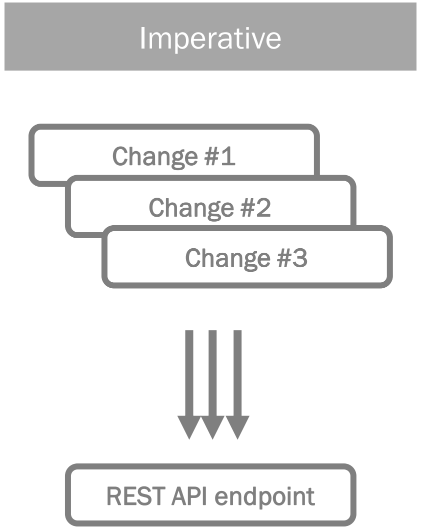
Device-level vs. Controller-level Management

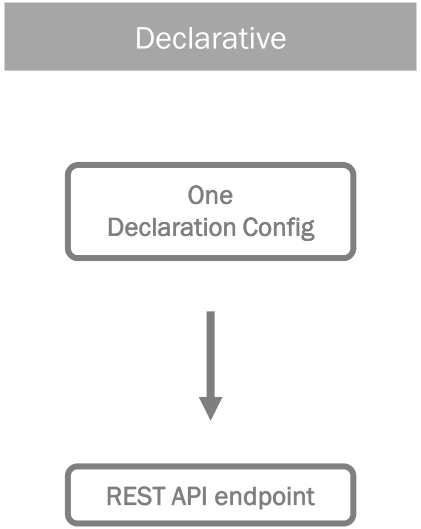



Controller-level management
MGMT plane
Northbound API
For management and integration
(including third-party API - REST API)
Control plane
Southbound API
Connect and manage the end devices
(Telnet, SSH, SNMP, NETCONF, gRPC, RESTCONF,...)
Data plane

A **Network Controller** is a centralized software platform dedicated to managing the configuration and operational data of network devices

Imperative (Procedural) vs. Declarative method (engineering point of view)





Tells the target system exactly **how** to do something.

What you want to do.

Imperative (Procedural) vs. Declarative method (in the language of managers ;))

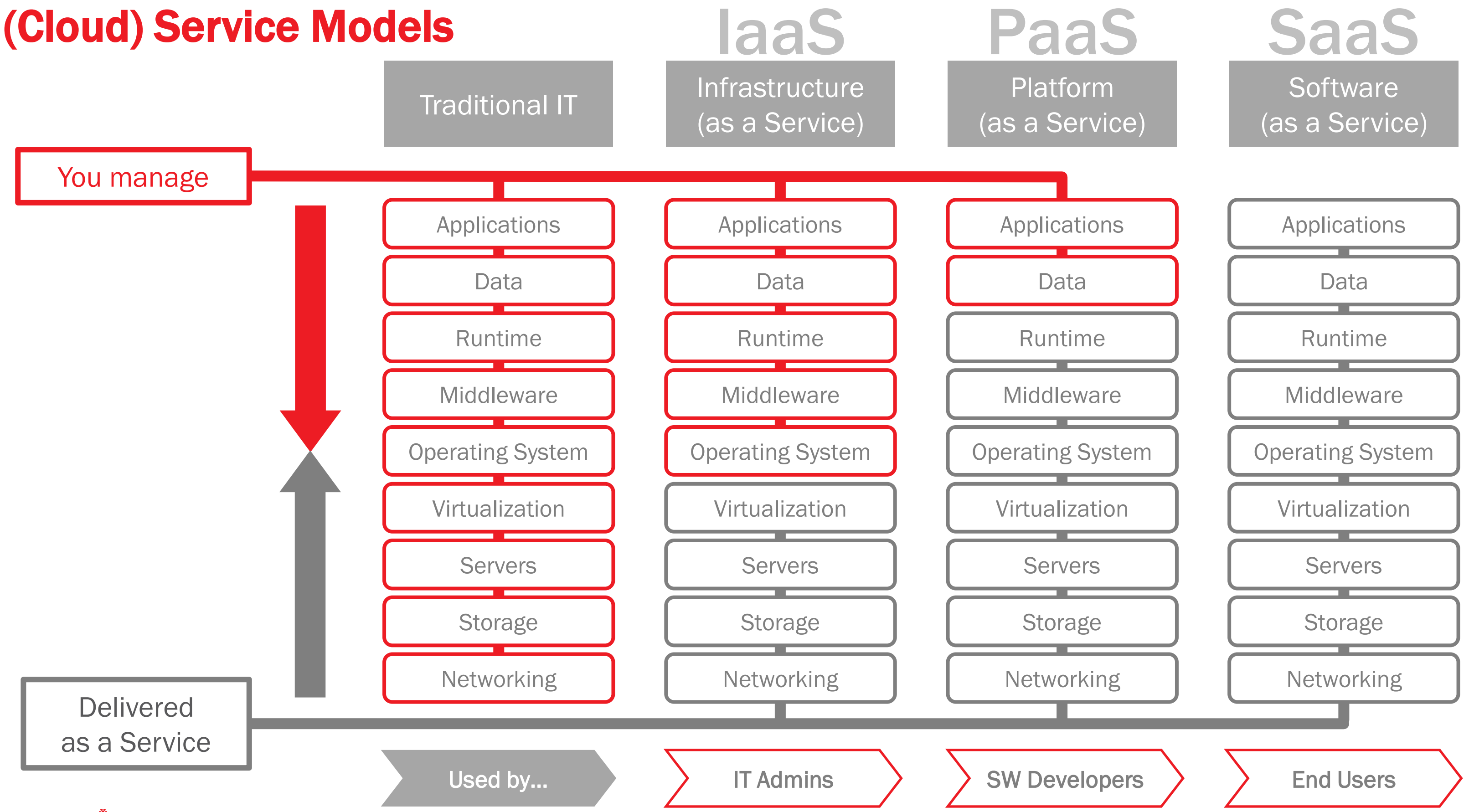
Imperative

Declarative

Tells the target system exactly **how** to do something.

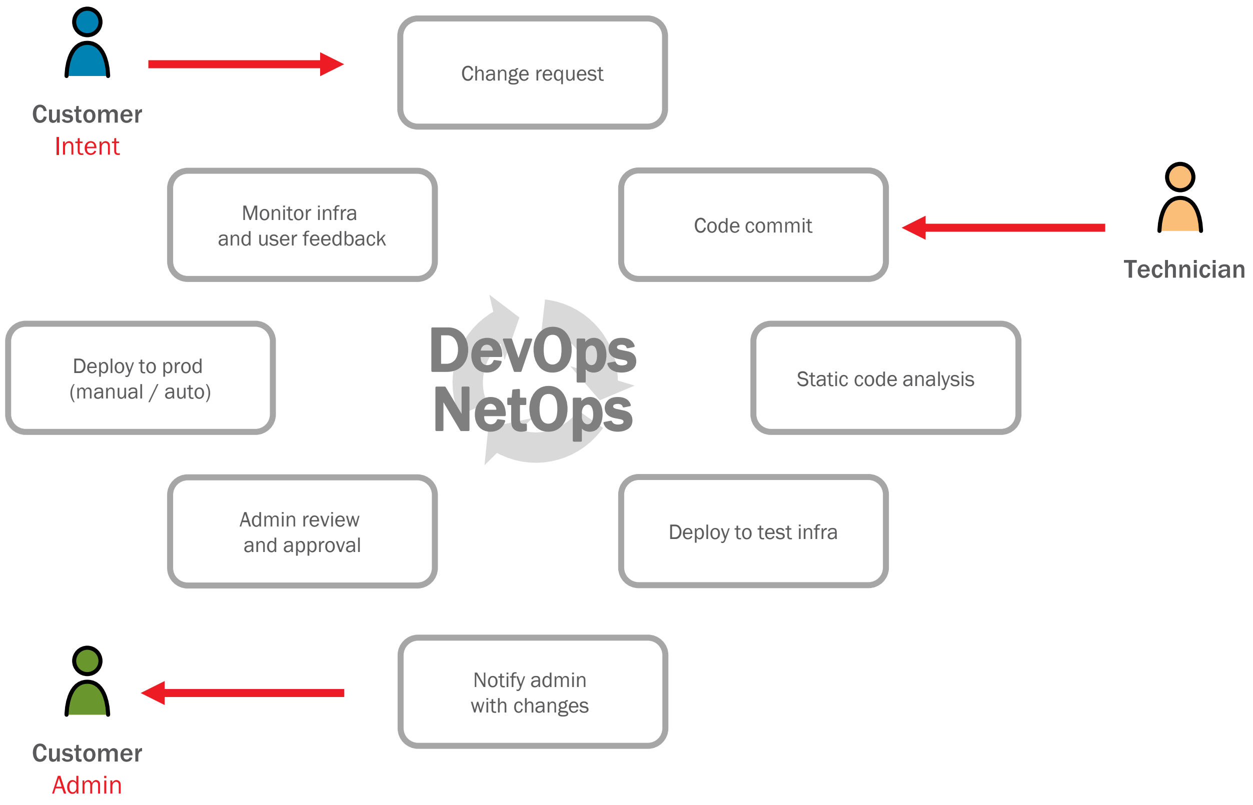
What you want to do.

(Cloud) Service Models

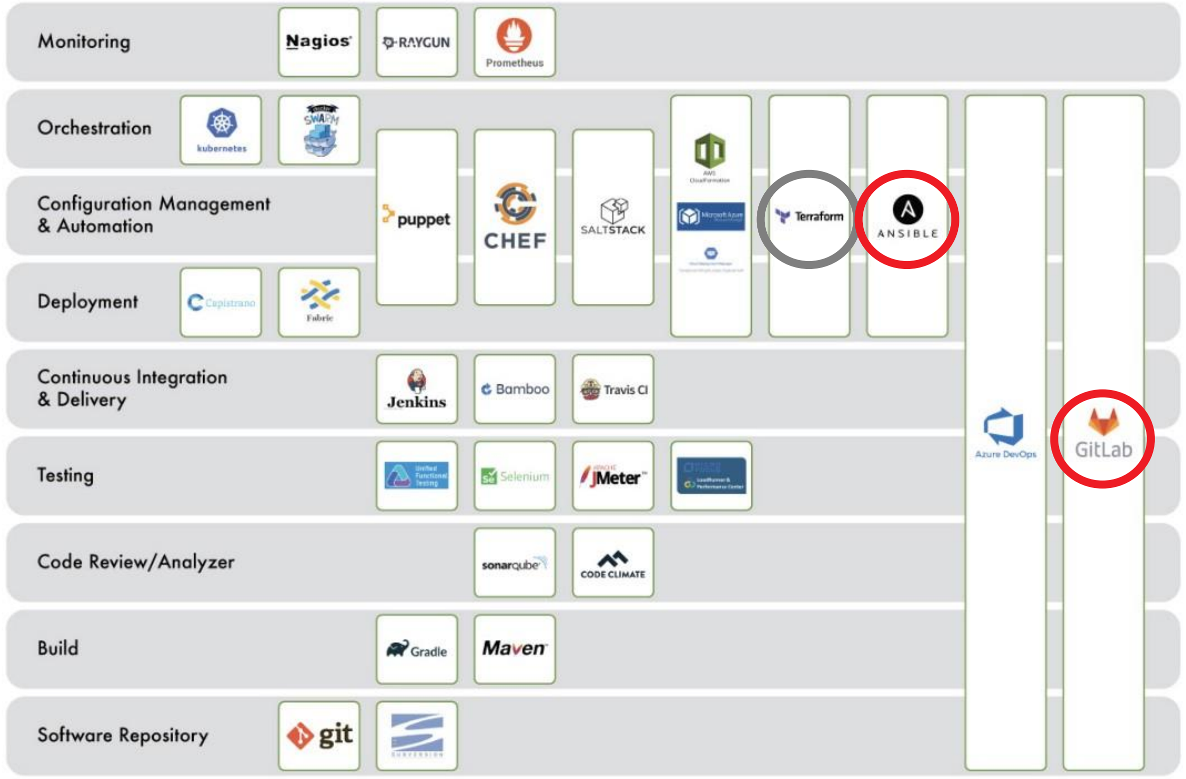


SOITRON*

Service Delivery Workflow



Service Delivery Tools (very short review)



Automation Use Cases

Unified policy distribution with approval process

- **Credentials**
- DNS, NTP, SNMP **settings**
- New device **provisioning**

Migration to new infra

- Every **deploy is consistent** – using migration process agreed before. Also - input validation and automation is used (eliminating **human errors**)
- Lowering **maintenance window** duration (configs are prepared before, deployed quickly during window without many GUI clicks)
- In case of problem input config can be edited and **redeployed quickly** (or rollbacked)

Input validation before pushing to production

- Input is **structured** -> it is possible to make various **validation** checks (JSON schemas validators, CI/CD pipeline, etc.)
- Naming **unification** (problem: every admin use different style)
- Infrastructure **templates** – design is maintained across many environments

Bonus

#1 network state is abstracted in the repository – this can be used as **documentation** or for testing

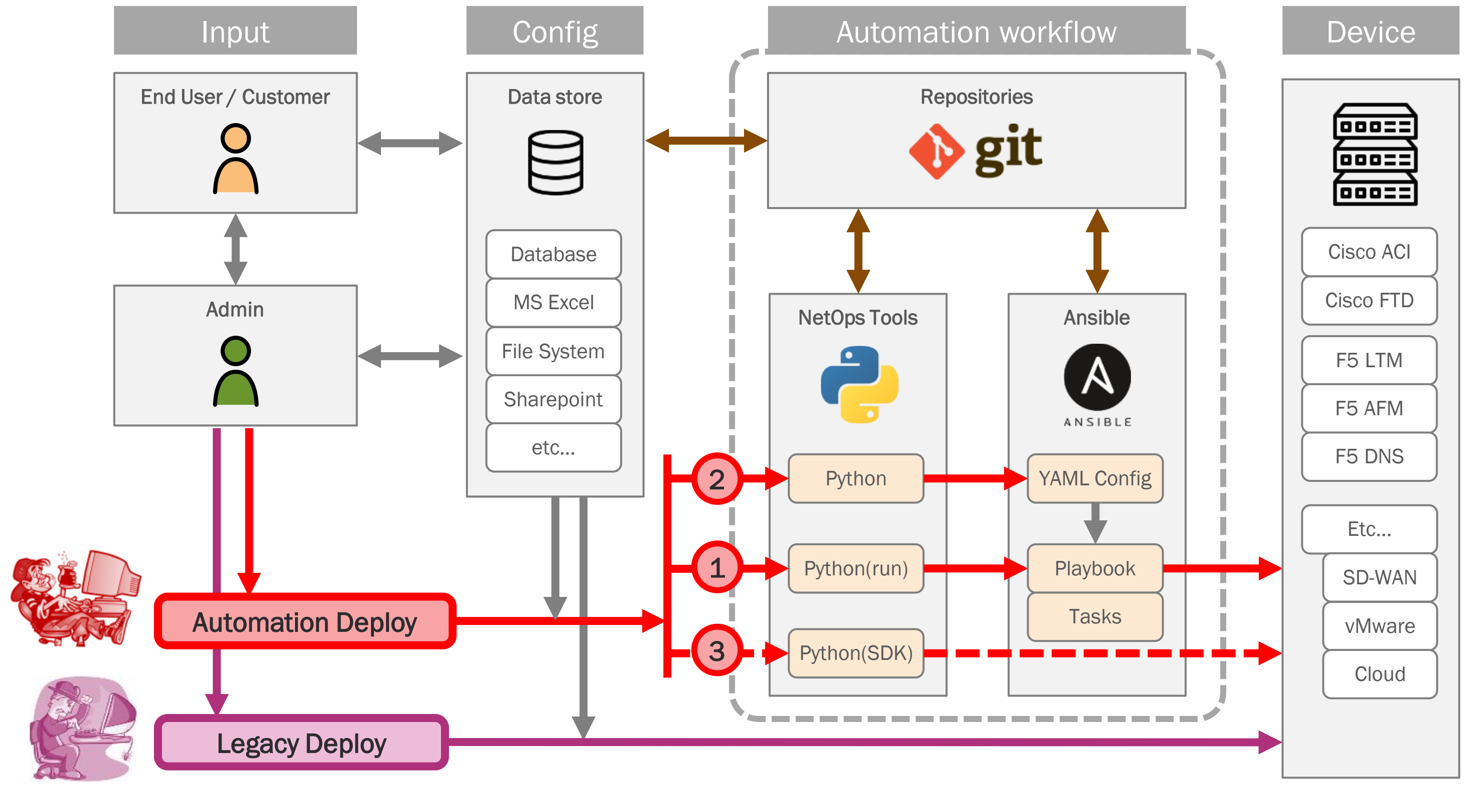
#2 **migration workflow** can be used later/again for deployment of **new services**

Part #2/5

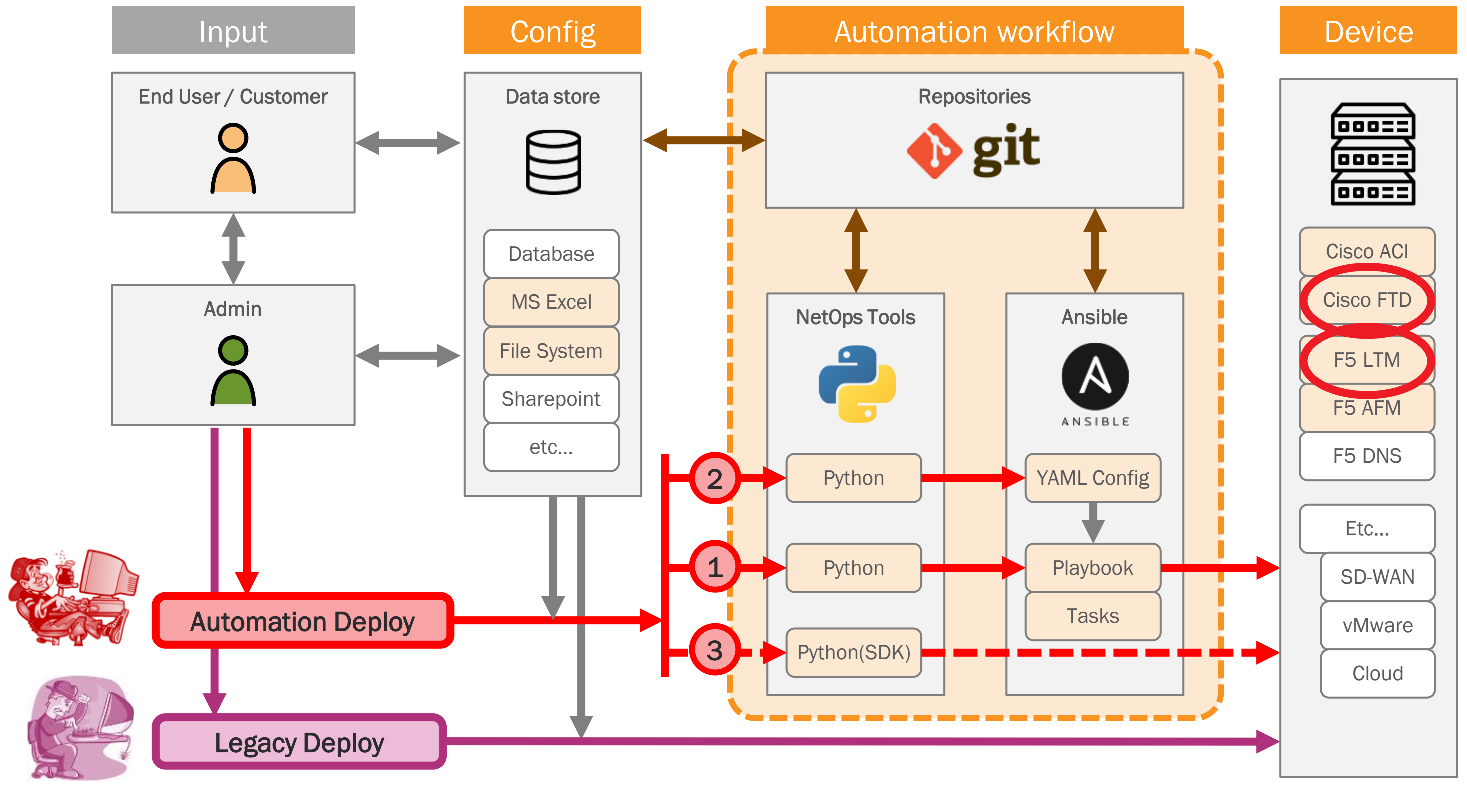
LAB intro

- Automation workflow
- Git and GitLab

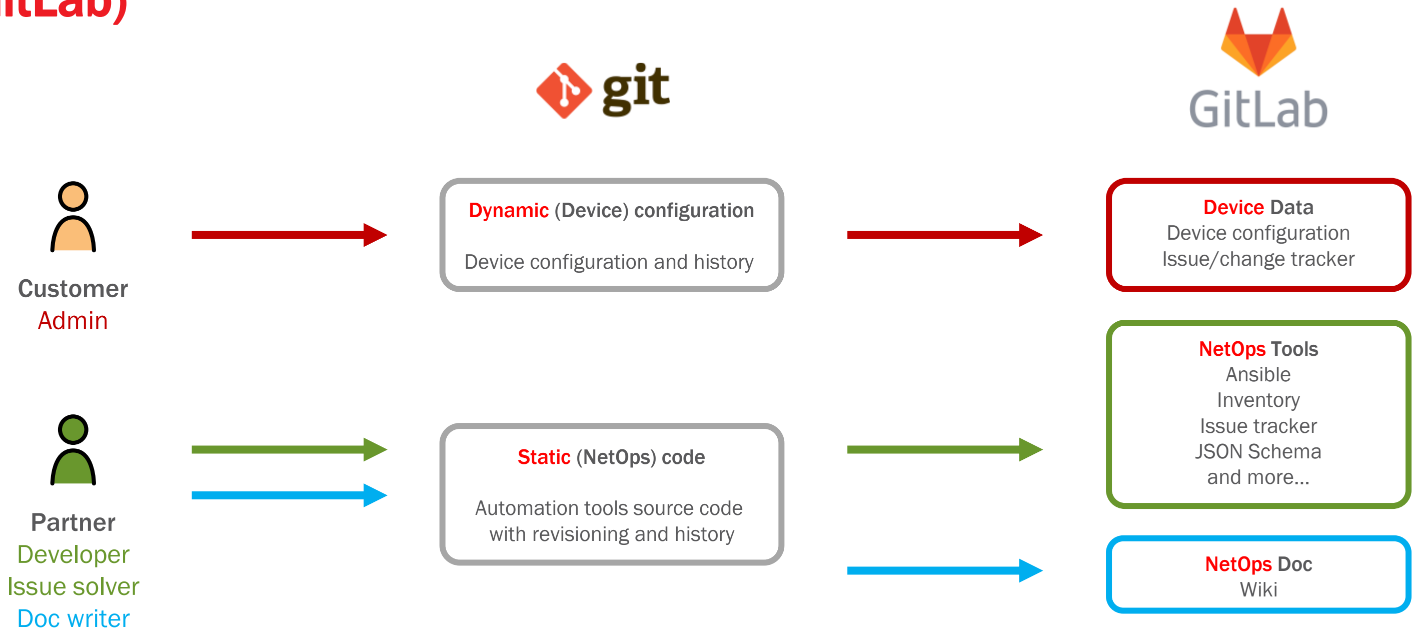
Automation workflow



Automation workflow



Git (and GitLab)



Git is a free and open-source **versioning control system**.

GitLab is a Git-based **platform** with lot of powerful features.

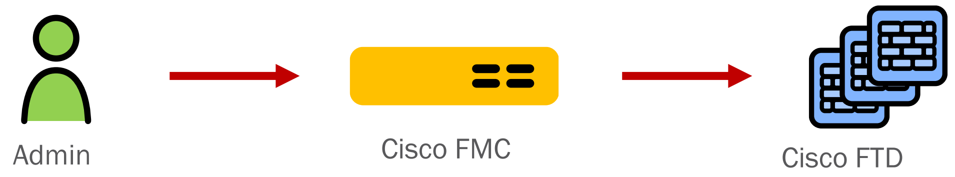
Part #3/4

LAB – Firewall

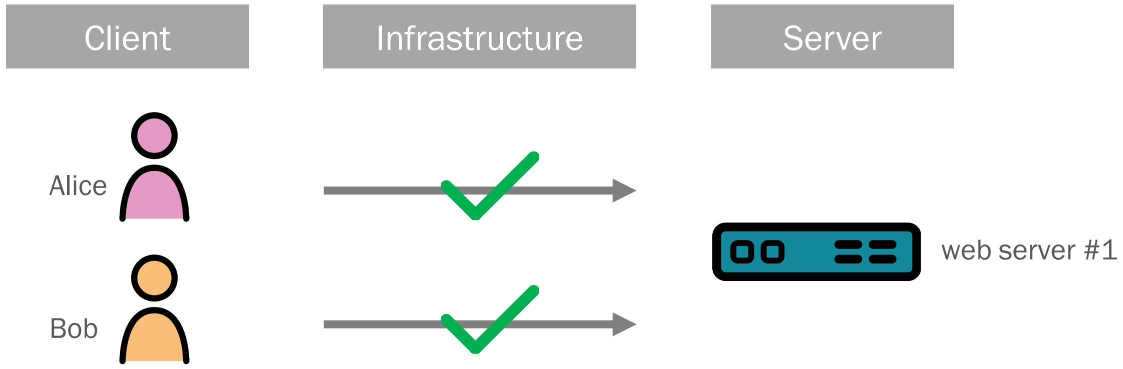
- Cisco FMC/FTD
- NetOps principles
- LAB: FW rules automation

Cisco FTD automation intro

- Traditional configuration methods
  - GUI (FTD or FMC), CLI show commands
- (REST API) Modern configuration management
  - FTD – REST API, full config with ansible
  - FMC - REST API, basic config with ansible/terraform
- (SDK) “fmcapi” python package
  - Cisco community package on GitHub – easier API scripting
  - https://github.com/marksull/fmcapi



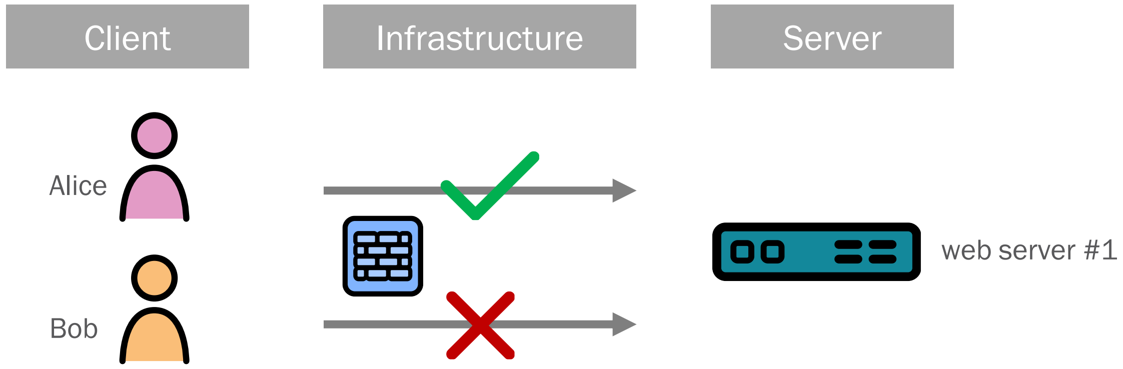
LAB1: FW automation



Situation:

Clients can connect to web server

LAB1: FW automation



Change request:
Only Alice can connect to web server.

Solution:
Disable connection on FW

Part #4/5

LAB - Application Delivery Controller

- F5 BIG-IP automation
- Imperative vs. Declarative method
- JSON Schema
- LAB: f5 LTM automation

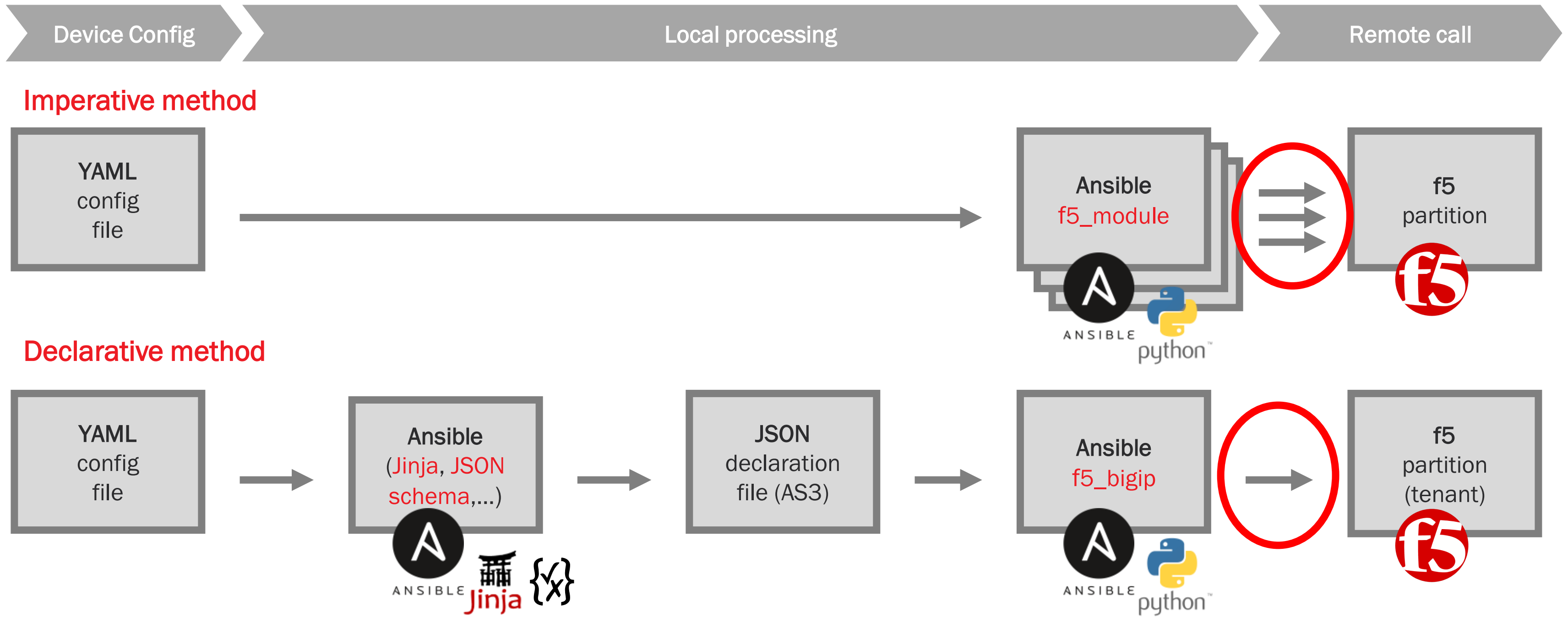
F5 BIG-IP automation intro

- **Traditional BIG-IP configuration methods**
  - CLI and GUI (Web API) – not useful for automation
- **SDK (Software Development Kit)**
  - F5 SDK (Python) – client library to access various (most popular) f5 products and services
- **REST API (Automation Tool Chain)**
  - **Declarative Onboarding (DO)** - initial configuration (license, module provision, network, users,…)
  - **Application Services 3 (AS3)** – configuring application services
  - **Telemetry Streaming (TS)** – for streaming statistics (device, VS, pool,…) to external application
- **Ecosystems (3rd party) Integration (using SDK and/or REST API)**
  - Ansible
  - Terraform
  - Cisco ACI

F5 AUTOMATION TOOLCHAIN

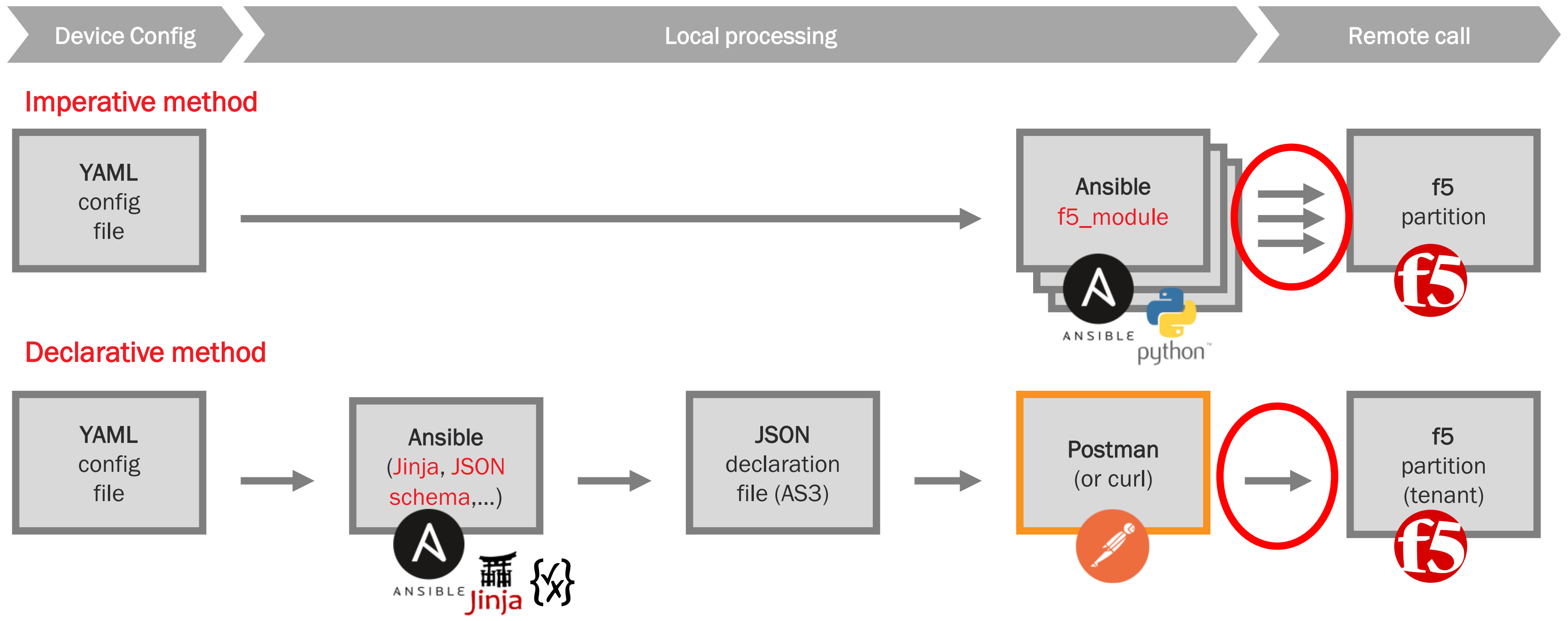
Source: clouddocs.f5.com

F5 BIG-IP automation workflow (imperative vs. declarative deep dive)



SOITRON*

F5 BIG-IP automation workflow (imperative vs. declarative deep dive)



F5 resources

Ansible Collections

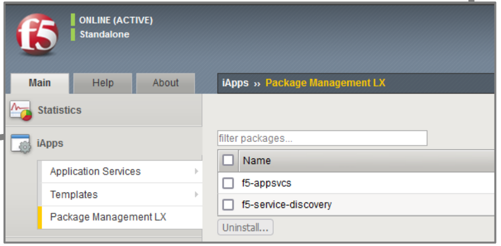
- Imperative (f5_modules)
  https://galaxy.ansible.com/f5networks/f5_modules
- Declarative (f5_bigip)
  https://galaxy.ansible.com/f5networks/f5_bigip

AS3

- AS3 documentation
  https://clouddocs.f5.com/products/extensions/f5-appsvcs-extension/latest/userguide/

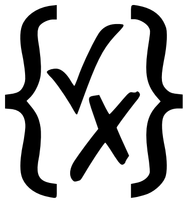
F5 appsvsc extension

- RPM package https://github.com/F5Networks/f5-appsvcs-extension
- + Postman collection
- + JSON schema



JSON schema

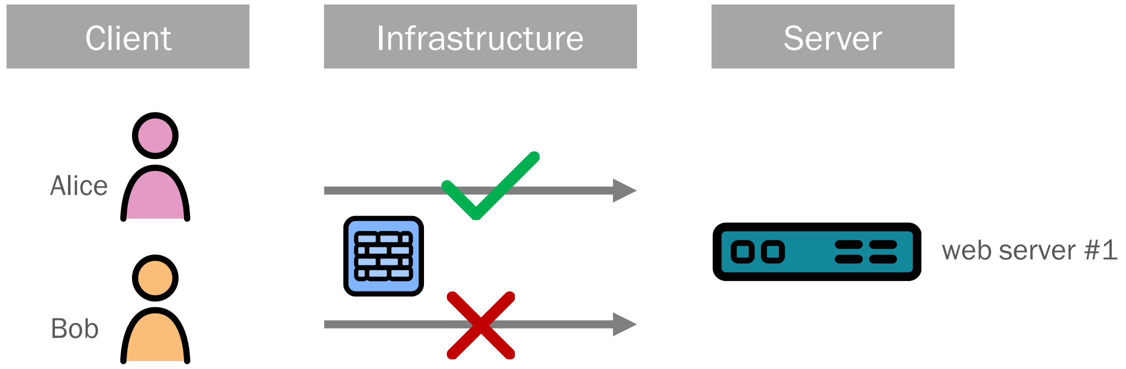
- **Describes** your existing data format(s)
- **Validates** input data
- Vocabulary can be public or private
- Use-case:
  - Automated testing
  - Ensuring quality of client submitted data
  - Real-time documentation



JSON Schema is a vocabulary that allows you to **annotate** and **validate** JSON (or YAML) documents

Source: json-schema.org

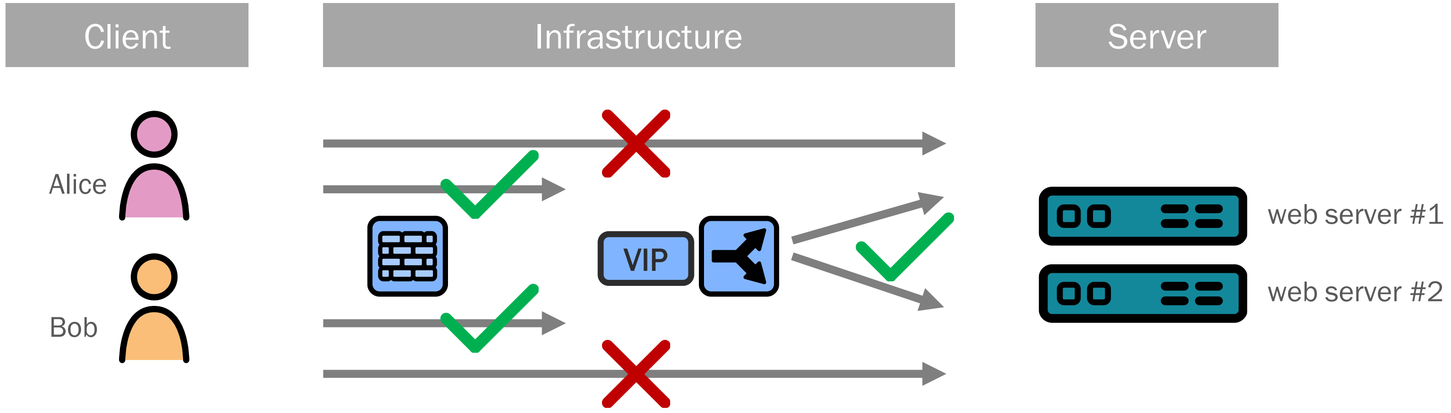
LAB2: f5 automation



Situation:

Only Alice can connect to web server

LAB2: f5 automation



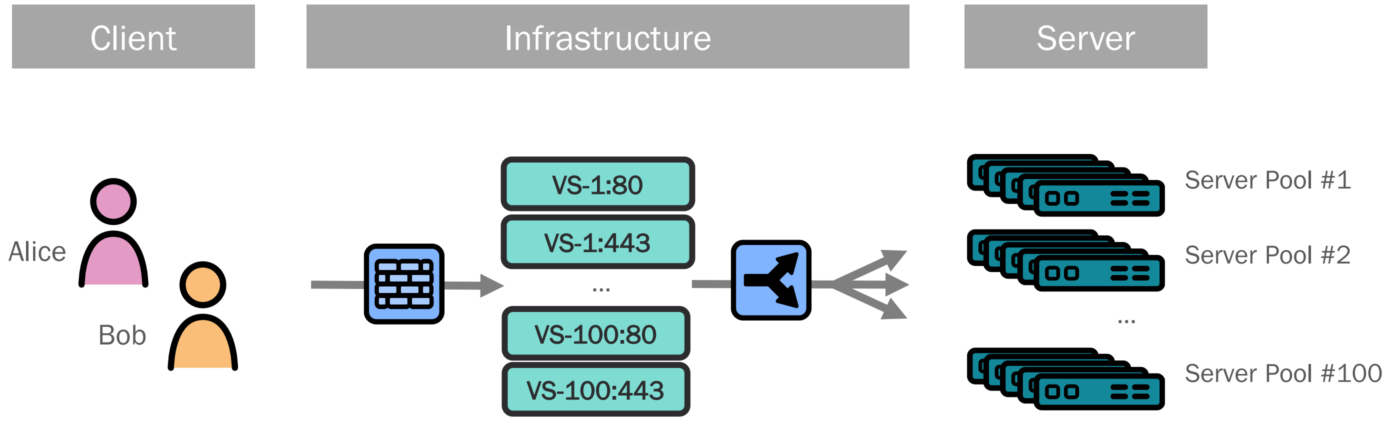
Change request:

- Add server #2, Add LB
- Disable direct access to servers

Solution:

- Disable direct connection on FW
- Create virtual server on LB

LAB3: How about a more complex configuration?



What's about 2x100 VSs and 100 POOLs with 5 MEMBERs each?

Deployment time ~1 minute ;)

Part #5/5

Summary

- Why to use Infrastructure automation?
- Our experience and customer's feedback
- Every process can be somehow automated

Summary

Why to use Infrastructure automation

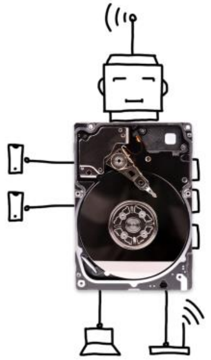
- Reduced possibility of (human) mistakes
- Repeatability
- Lower operating cost

Our experience and customer's feedback

- Reduced possibility of (human) mistakes
- Speed of configuration and/or migration process
- Config validation and unification/standardization
- Documentation (source of truth)

We can do much more…

- Problem well defined = problem half solved
- Every process can be somehow automated
- Any use-case is possible



ANKETA A DISKUSIA

Praha

Soitron s.r.o.
Pekařská 621/7
155 00 Praha 5

tel.: +420 266199918

Bratislava

Soitron, s.r.o.
Plynárenská 5
829 75 Bratislava 25

tel.: +421 258224111

e-mail: marketing@soitron.com
web: www.soitron.com

Soitron Inspirárium:
www.soitron.sk/riesenia-a-sluzby/soitron-inspirarium

Martin Kyrc
Network System Engineer

Roman Panenka
Network System Engineer

SOITRON*

SOITRON