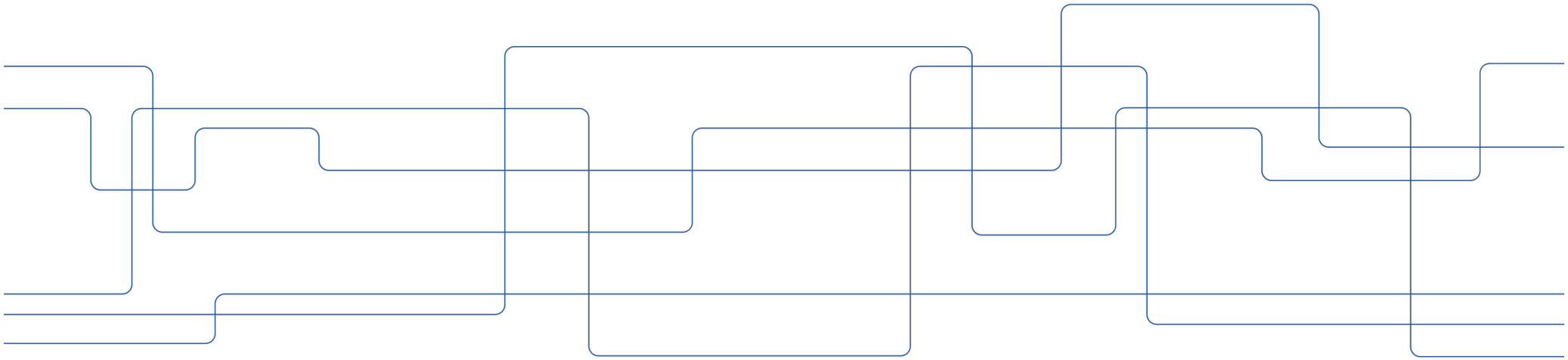
KTH ROYAL INSTITUTE
OF TECHNOLOGY

KTH/IT Automation and Sectigo API

How KTH/IT uses Sectigo API to enable automation



Who

- **Enrico Pelletta**, IT System Architect at KTH/IT. Main area: Web-Infrastructure including Proxy/LB and Internet Domains Service.
- **Rikard Warney**, Sys. Admin. At KTH/IT. Main area: Linux, Ansible and automation solutions.

Why

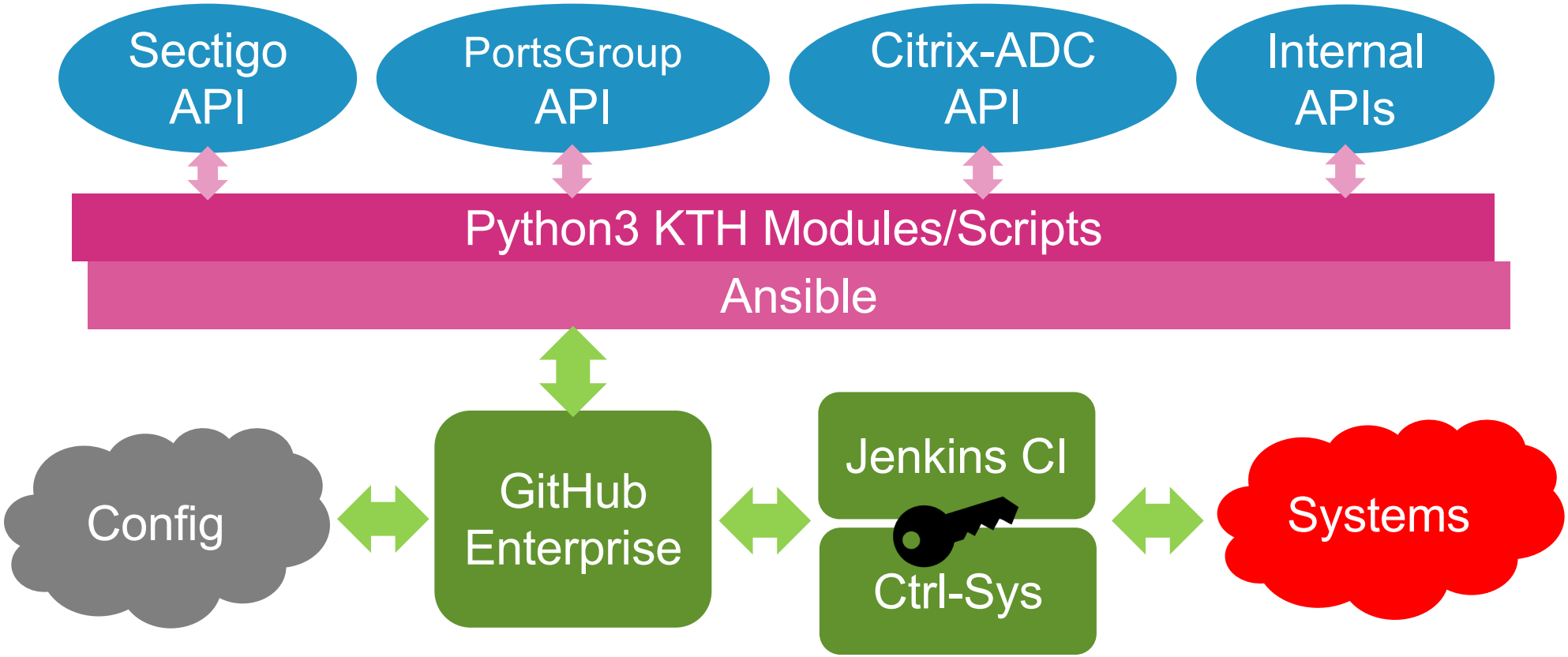
We daily deal with **many repetitive** jobs that consume some of our **time** and, **most critically**, might cause **incidents** and/or impact system performance and **security**.

By automating process, we free ourself for annoying routines and strength systems (and sometime do something funny...)

Many regular processes/routines includes **interaction with CA** (Sectigo). Thus Sectigo-API **enables** automating processes.

Sectigo API alone is not enough, a **set of APIs and tools** are required.

KTH automation ecosystem



Sectigo API

- Sectigo provides a REST API:
  - Simple, well documented (make sure you got latest!), organized in sections (minor overlaps!).
  - We have used Sectigo API for **3 years**. No surprises; not sudden API changes breaking stuff etc…
  - All you need is an API user/token. Get proper permission it may be the real issue here… As well to properly handle the key

- Test with CURL (!!!). You need **NO** programming super-power to Go!

```
$ curl 'https://cert-manager.com/api/domain/v1' -i -X GET \
  -H 'login: batman' \ -H 'password: Gotham123!' \
  -H 'customerUri: sunet'
```

- KTH/IT uses Python3 and a small module (class) we have created based on ***requests***.

Python3 API: Example

```
def get_domain_dcv_info(self, domain):
    """
    Get Domain info from Sectigo API
    :param domain: domain-name
    :return: dict or NULL
    """

    headers = {
        'Content-Type': 'application/json',
        'login': self.user,
        'password': self.passwd,
        'customerUri': self.customerUri
    }

    data = { 'domain': domain }

    resp = requests.post(self._url('dcv/v2/validation/status'),
                         headers=headers,
                         data=json.dumps(data))

    if resp.status_code == 200:
        return resp.json()
    else:
        return None
```

Automated Process for Web and Domains Srvs

- Proxy/LB certificates check/update/install:
  - **OV** and **EV** multidomain certificates.
  - **RSA** and **ECC.**
  - About **500** sites names (including aliases).

- Domains Inventory Status-Validation:
  - About **100** guest internet-domains.
  - Check status including Sectigo CA registration.

- Domain Control Validation (DCV): check-status and update:
  - DCV-DNS method

The Good, the Bad, and the Ugly

- The **GOOD** is that works! We have used Sectigo API to automate processes for 3 years, and we are happy. We have enabled service enhancements otherwise impossible with little cost/effort. Example:
  - Support of lots of site-name aliases (.se, .nu. .org, .info, .net, .com, etc… etc…) and also "*kungligatekniskahögskolan.se*".
  - EV and OV certificates.
  - RSA and ECC.

- The **BAD** is that "*sometime*" (rarely…?) it does not work.
  - Example: Anchor certificate requirement change cause block of certificate issuing.
  - Sectigo API performance time-to-time have a bad day… You might get unexpected delays.
  - Game rules keep changing. Ex: DCV-HTTP to DCV-DNS. Is it a real issue? You swear a bit, but it might be easier than you expect! Using automation, at least…

- The **UGLY** is about API key permissions and thus its protection.
  - Current Sectigo/GÉANT/Sunet solution does not apparently provide the permission granularity we would really like. Some processes must therefore remain semi-automatic (**DCV**) or even manual (register/unregister).

Ansible integration – Why?

- Integrate Sectigo API and configure, for example, Apache with the same tools you use to get certificates
  - This allows us to define a webhost in the same place as it's certificte
- No need to write code to integrate and manage servers – Ansible already solves this.

Ansible integration – How?

We use four components:

1. Sectigo's REST api
2. Custom Ansible module (Python)
   – are used to create domains, and Users in Sectigos CertManager
3. Certbot
   – Used to download created domains using the created user
4. Ansible
   – The thing that connects everything together

Ansible integration – Playbook Example

```
- name: ”Check if domain exists”
  # Runs a script in library/acme_domain.py
  acme_domain:
    name: ”{{ item.name }}”
    description: ”{{ item.description }}”
...
- name: ”Check if ACME account exist”
  acme_user:
    name: ”{{ sectigo_acme_user }}”
...
- name: ”Check if domains are mapped to user”
  acme_user_domains:
    name: ”{{ sectigo_acme_user }}”
```