Infra Security Automation in CI/CD: Impossible is Nothing

Spyros Manglis – Davide Cioccia

ING

#whoarewe

Spyros Manglis

Security Engineer @ ING

Davide Cioccia

Security Engineer @ ING

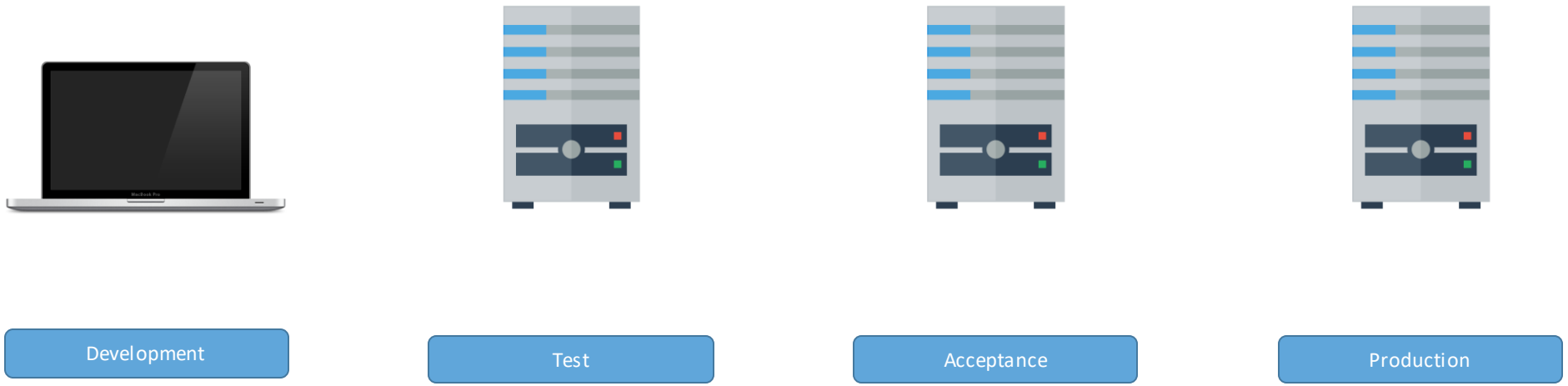
Trainer @ defdeveu

ING

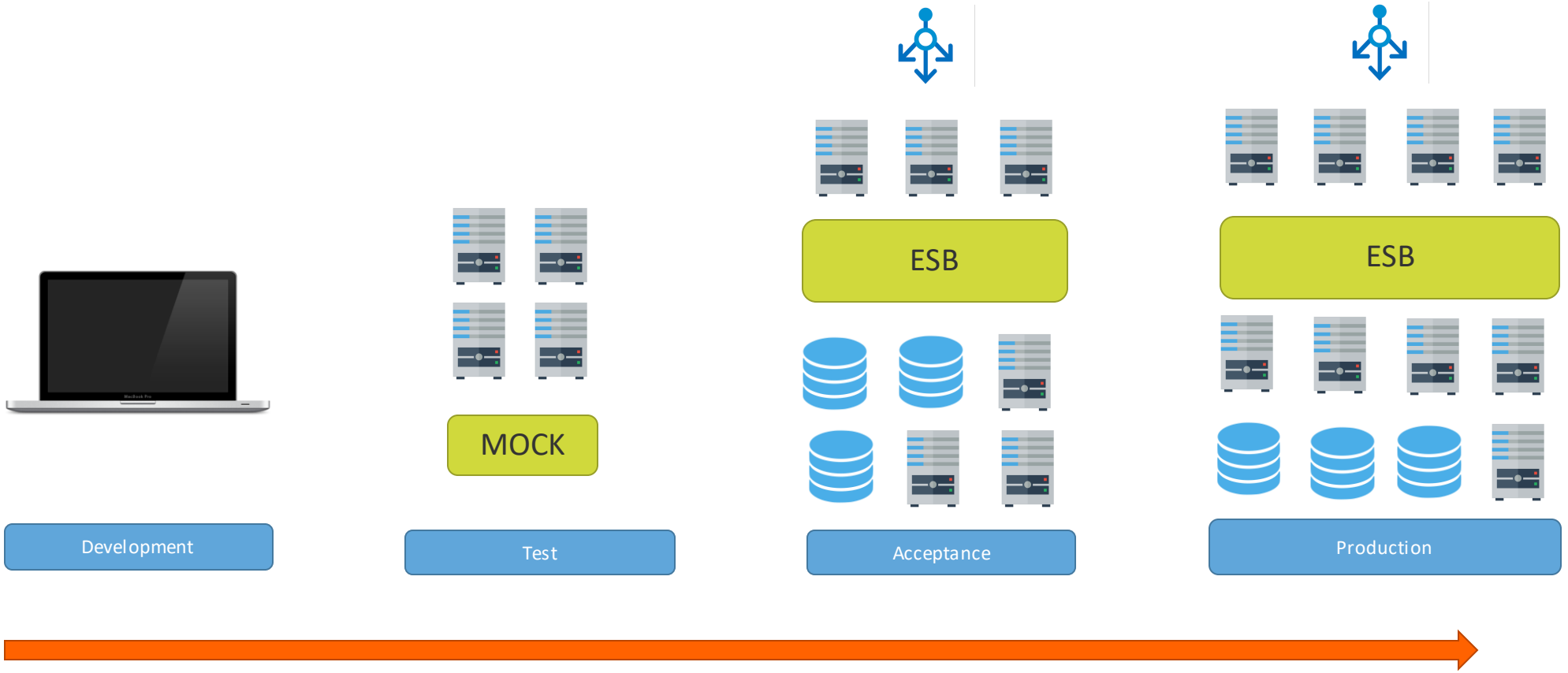
What's this all about?

- Present our challenge
- Share our knowledge and experience on custom infrastructure security automation
- Share the framework we created in details
- Future work: embed the tool in the CI/CD pipeline
- Q&A

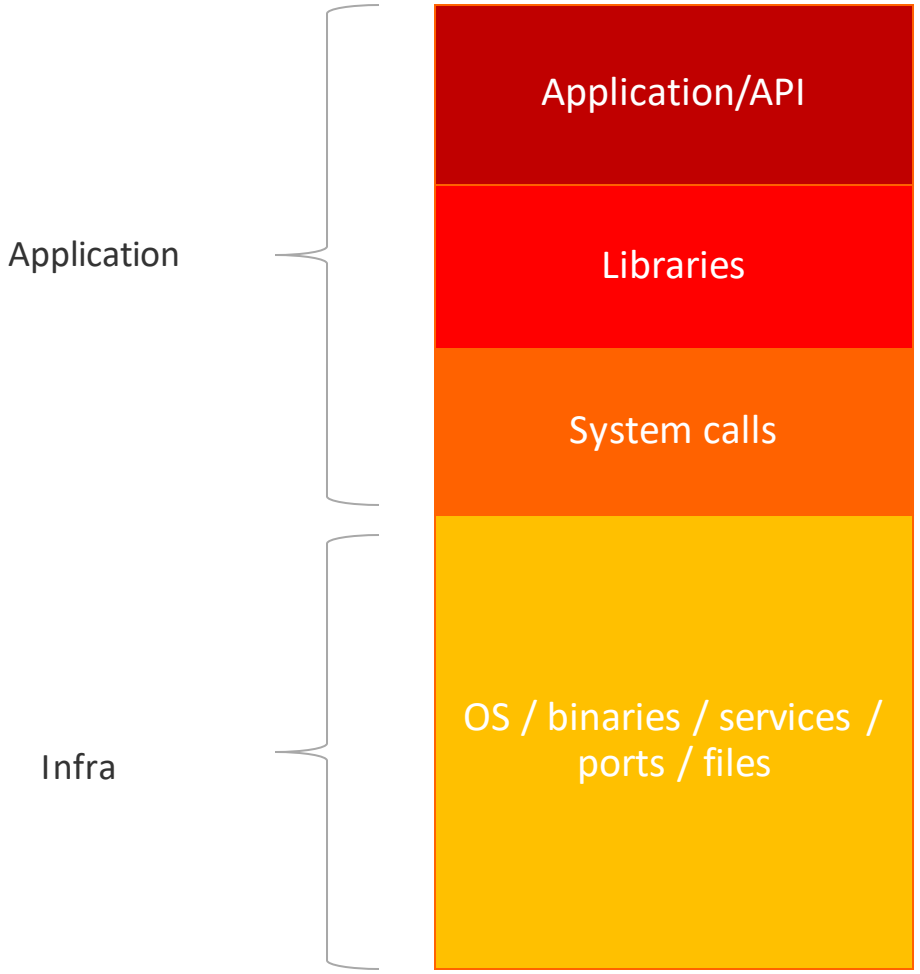
A simple DTAP environment for one service



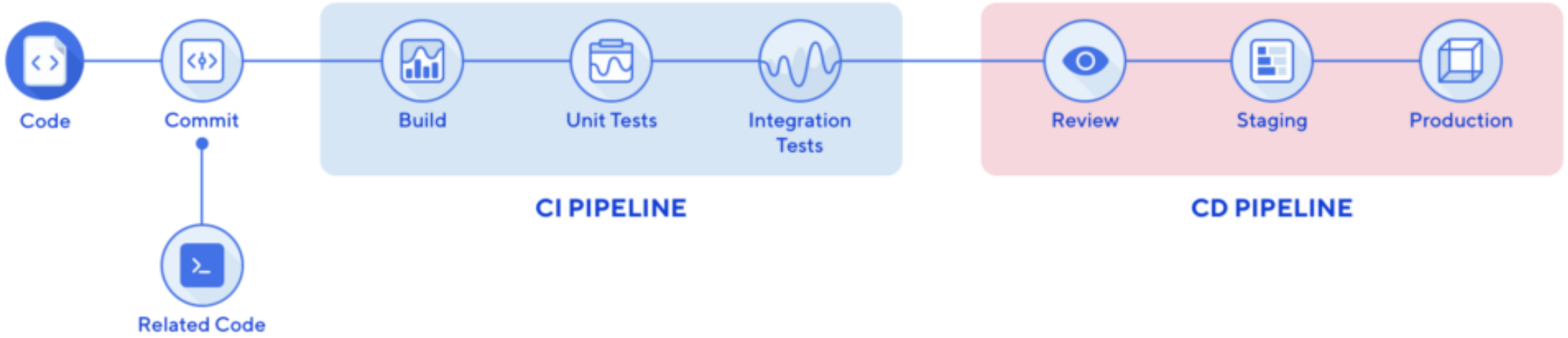
The ING DTAP environment for one service



What’s on each server: the stack

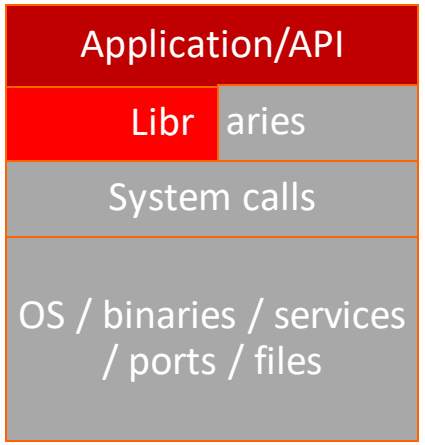


CI/CD secure pipeline and stack assessment coverage



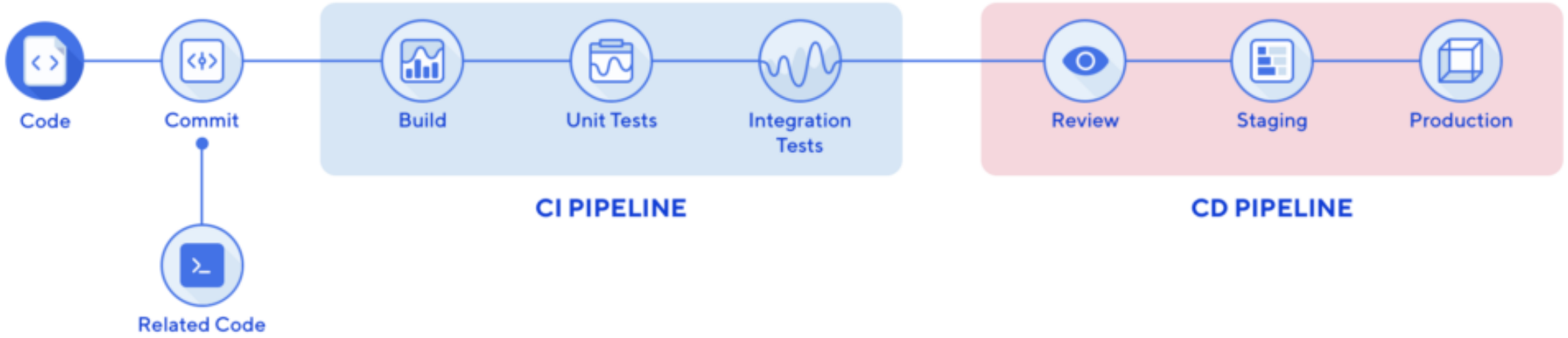
Real time SAST in IDE

Incremental SAST



ING

CI/CD secure pipeline and stack assessment coverage



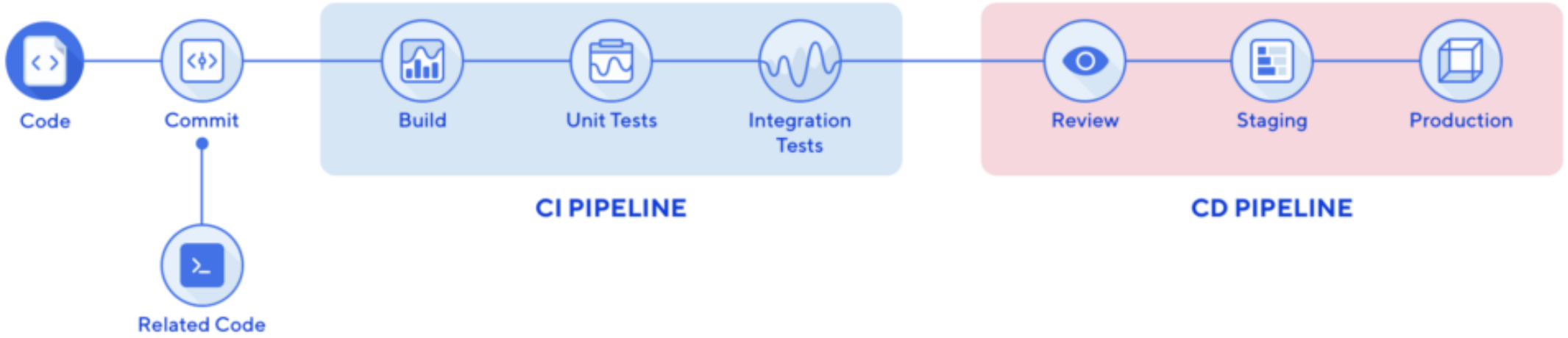
Real time SAST in IDE

SCA scanning

Incremental SAST

Application/API
Libraries
System calls
OS / binaries / services / ports / files

CI/CD secure pipeline and stack assessment coverage



Real time SAST in IDE

Incremental SAST

SCA scanning

IAST

DAST

Application/API

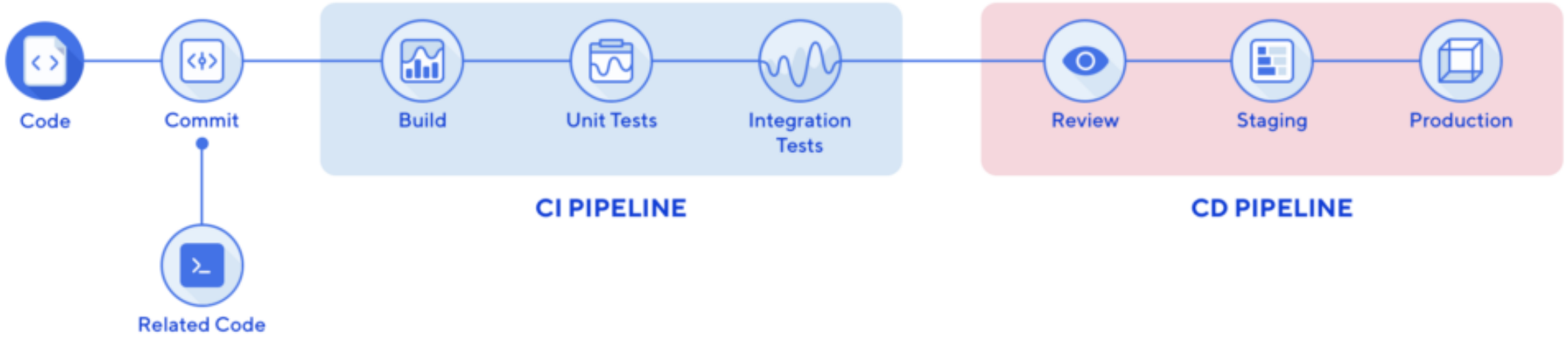
Libraries

System calls

OS / binaries / services / ports / files

ING

CI/CD secure pipeline and stack assessment coverage



Real time SAST in IDE

Incremental SAST

SCA scanning

IAST

DAST

Manual Application and Infra Pentest

Application/API

Libraries

System calls

OS / binaries / services / ports / files

ING


SAY "MANUAL TESTER" AGAIN.
I DARE YOU! I DOUBLE DARE YOU!
imgflip.com

The challenge

Problems:

- Manual infrastrucure security testing overhed
- Machines, VM images, Docker images are only assessed on provisioning
- Infrastructure incremental changes are not checked
- VA tools not valuable
- 2000 applications = 10x 2000 machines =  testers

Question:

- How can we check the security of the deltas on infra level?

#whoarewe

Spyros Manglis

Security Engineer @ ING

Davide Cioccia

Security Engineer @ ING

Let’s automate

ING

Why Ansible

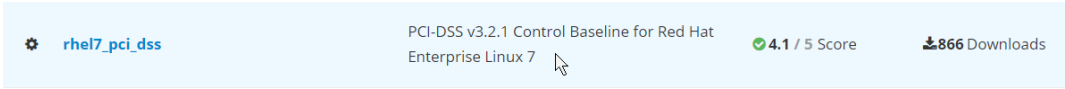
- Agentless
- Ssh/wirnm
- Keeps state
- Extensible
- Easy to scale
- Running based on facts (preconditions)

Open source community

https://galaxy.ansible.com/home

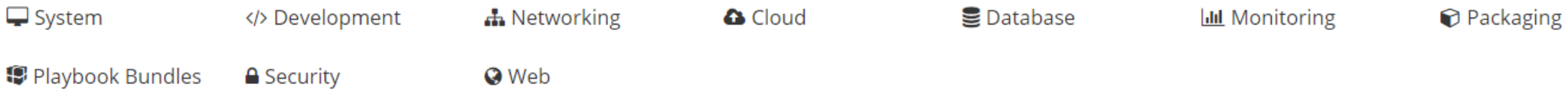
Well known authors

You can share yours



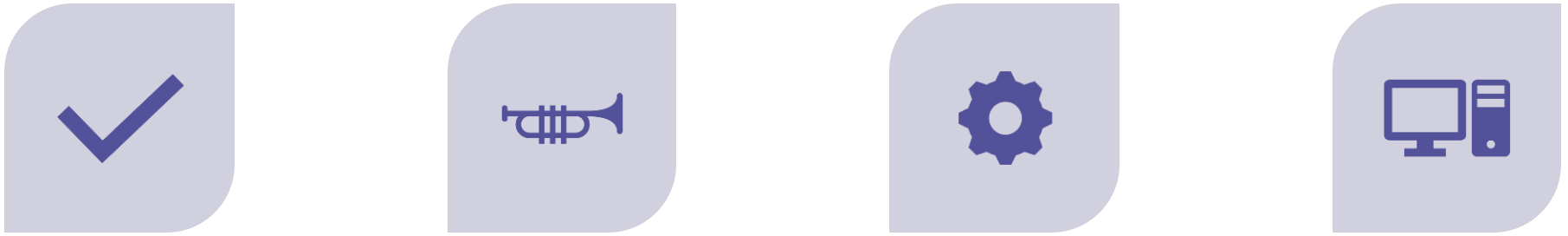
Most Popular

Playbook for everything



ING

Ansible use cases



PROVISIONING ORCHESTRATION CONFIGURATION DEPLOYMENT

How is Ansible mostly used in security

- STIG (Security Technical Implementation Guide)
- PCI DSS (Payment Card Industry – Data Security Standards)
- Networking Hardening
- Internal Security Standards
- Threat Hunting (0 days)

Why not other tools?

- Can't ABC do the XYZ
- Developers and Ops can understand
- Developers and Ops can use

ING

Why not other tools?

- I can take advantage of all those

ING

The process

1. Understand our infrastructure landscape and how it changes
2. Adopt well know industry standard (PCI DSS, MITRE, STIG)
3. Define threats for Windows and Linux machines based on the STRIDE model
4. Map the threats with the MITRE ATT&CK, STIG
5. Define tests (create playbooks) based on the threats.
6. Assess the base images (at provisioning)
7. Incremental scans on changes

The process

Bussiness Q&A

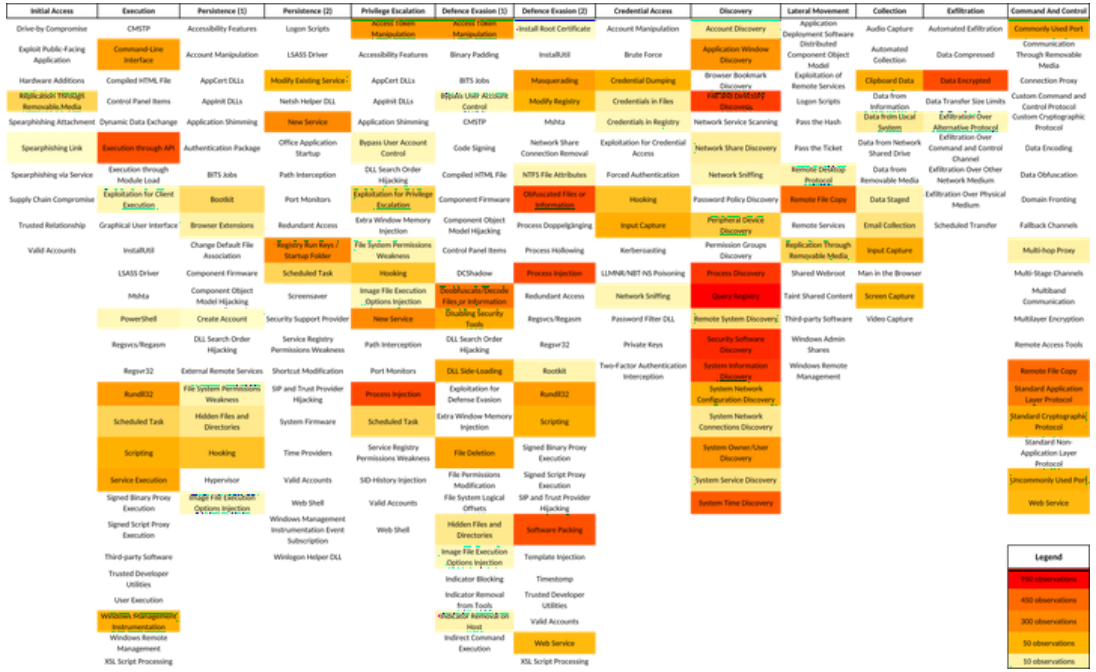
Questions
When these default images change?
How these default images change?
What does the change process look like?
How often changed?
What testing was done on the changes?
Which optional packages do you require the most?
What are the possible external threats?

The process
Technical Q&A

Questions
Do you require extra disk
Config files
File permissions
Nessus scan
Additional Network scans
Installed packages
Custom Builds / Deploys
CronJobs
Internal Builds / Artifactory
Installed Patches
Tools
Certificates
Ports
Critical folders
Groups/User
Possible external threats?
SEM-A

How is been used within ING

- Define infrastructure security standards
  (guided always STIG, PCI DSS, MITRE ATT&CK …etc)



Internal Threat model

ThreatID	Threat	Description	Mitigations	Mitigations sufficient?	Tests
INF-T-WND-1	Attacker adds or edits system users or groups	An attacker edits system, service or regular users accounts gain persistence An attacker adds new low or high privileged account to gains persistence An attacker edits or adds a system group.			Done
INF-T-WND-2	Attacker installs malicious package/software	An attack installs custom package with vulnerabilities to aid him with gaining persistence or elevate privileges as a stepping stone towards further attacks or lateral movement.			Done
INF-T-WND-3	Attacker exploits known vulnerability of a running service	An attacker exploits a vulnerability of a services installed and/or running on the system to get foothold, elevate privileges or continue further attacks. Such as: network services, usb, thunderbolt, smartcard, ...	Keep your system up-to-date, check for new discovered vulnerabilities		done
INF-T-WND-4	Attacker exploits known vulnerability on installed software in order to hurt Confidentiality, Integrity or Availability	Attacker exploits a known vulnerability in the installed software on the system to elevate privileges and put himself in position to further his attack. (Think: buffovs, privesc, unquoted service path, ...)			done
INF-T-WND-5	Attacker steals and reuses credentials and continues lateral movement	Attacker find credentials that aid him in furthering his attack from configuration files, memory, logs, network traffic, etc. Can be combined with WND-3			Done
INF-T-WND-6	Attacker disables system security controls	An attacker disables system/os security controls such as ASLR, DEP in order to weaken the system in his further malicious actions such as exploiting a vulnerability such as buffer overflow on an application present on the system in order to elevate his privileges and continue his attack.			done

ThreatID	Threat	Description	Mitigations	Mitigations sufficient?	Tests	CIO Security Scope
INF-T-LNX-1	Attacker manipulates confidential application related data	Applications that are hosted on the server handle confidential data. Attacker that is unauthorised gains access to the data and enables him to manipulate it in any way possible. This could mean reading application configuration files, databases config files, gains access to source code to understand business logic, decompiles application, ...	SEM-E will trigger alert on sensitive file modification.	NO, because the teams specifies the files to monitor (can miss). Does SEM-I notify read activities?	Done	Yes
INF-T-LNX-2	Attacker hosts a rouge application to the web/app server to steal credentials and confidential info	Attacker uploads/writes rouge application with replica functionalities as the ones genuine ones on the same web/app server and lures victims to use his application instead. This could also mean taking down or changing the endpoint/urls of the genuine application.	SEM-A		Done	Yes
INF-T-LNX-3	Attacker extracts confidential data from application logs	Attackers gets read access to all application logs of all log levels; He/She then looks for confidential and sensitive information in the log files.	CIO Security guidelines does not allow to write sensitive info in the log files	No, teams that do not go through the SCR/LA procedure can bypass the check	Done	Yes
INF-T-LNX-4	Attacker fills up disk space by flooding the application logs	Attacker realises that the logs are being stored on a local/remote disk with limited capacity and tries to fill the disk up by triggering logging functionality on the hosted application.			Done	Yes
INF-T-LNX-5	Attacker manipulates application logs	Attacker has gained write access to the applications logs; he/she uses that to cover or clear his malicious previously executed activities by selectively deleting or modifying existing logs.			Done	Yes

ING ISTG (Infrastructure Security Testing Guide)

Mapping Threats to Ansible tasks

1. Information leaks in application properties files.

Application `.properties` and `.config` file could leak sensative information such as access credentials to local JavaKeyStore files. The following example shows how to locate such leaks.

- First, locate all the readable `.properties` files on the server.

```
find / -readable -type f -iname *.prop* 2>/dev/null > readable-properties-files.list
```

- Next, identify if sensitive information is leaked in any of these files.

```
for file in $(cat readable-properties-files.list); do printf "\nChecking File $file:\n" && grep -iE "pass|key|ntlm|hash|user" --color=always $file;done > properties-files-check.list
```

This results to the following disclosure:

```
com.ibm.ssl.trustStore=${user.root}/keys/TWSClientTrustFile.jks
com.ibm.ssl.trustStorePassword={xor}somevalue==
```

Steps

- Attacker find where the tomcat instance is located by searching through the server. Example: `find / -name "tomcat"` `find / -name "tcserver"` `find / -name "server.xml"`
- After finding out the location of the tomcat server, the attacker proceeds and look for applications logs for the targeted application by observing the log configuration on the server and on the application.
  Example, Given the target application is named *BigTransactionsAPI*, the attacker tries to locate the logs by: `find /location/logs/usually/go -type f -iname "*BigTransactionsAPI*.log"`
- Attacker then checks or confirms if he has read access to the logs by `ls -la` the log files and observing file permisions. *(This can be combined in one comand)*
- Once confirmed, attacker digs through the log files to find/steal confidential, sensitive information. Example: `grep -E "account|creditcard|password|username|iban|phonenumber" /var/log/tcserver/logs/bigtransactionsapi-application.log`

Result

- Depending on the CIA Rating of the application in some cases loging sensitive and confidential data is allowed. However, this should be taken good care of. The access to the logs should be limited to few NPA accounts at most.
- C3 and C4 regarded data are the ones attackers will be targing and they should be not available on an application that has lower C rating
- Accessing log files should trigger a SEM-A alert
- The logs found should comply with retention policy

INF-T-LNX-3.md 3.34 KB Edit | Web IDE | Replace | Delete

INF-T-LNX-3

Attacker extracts confidential data from application logs

Threat desription

Attacker has gaing read access to all application logs of all log levels. He/She proceeds further and digs through the log files looking for confidential and sensitive information.

Tests

Assuming that the attacker has gained limited or root access to the server, he/she can move forward and look for confidential and sensitive data in application logs. The application logs can differ but most commonly we have java and scala applications running on tomcat server. However, there are other flavours of applications and depending on the application type and the server type application log files will differ. The steps explained below are applicable mostly to tomcat application logs but the same approach should be adopted and followed for other/similar envrionments as well.

The default permissions are:

```
<mapping>
        <directory>${dist.install.dir}/log</directory>
    <filemode>750</filemode>
    <username>apprun</username>
    <groupname>applog</groupname>
</mapping>
```


ING

Playbooks

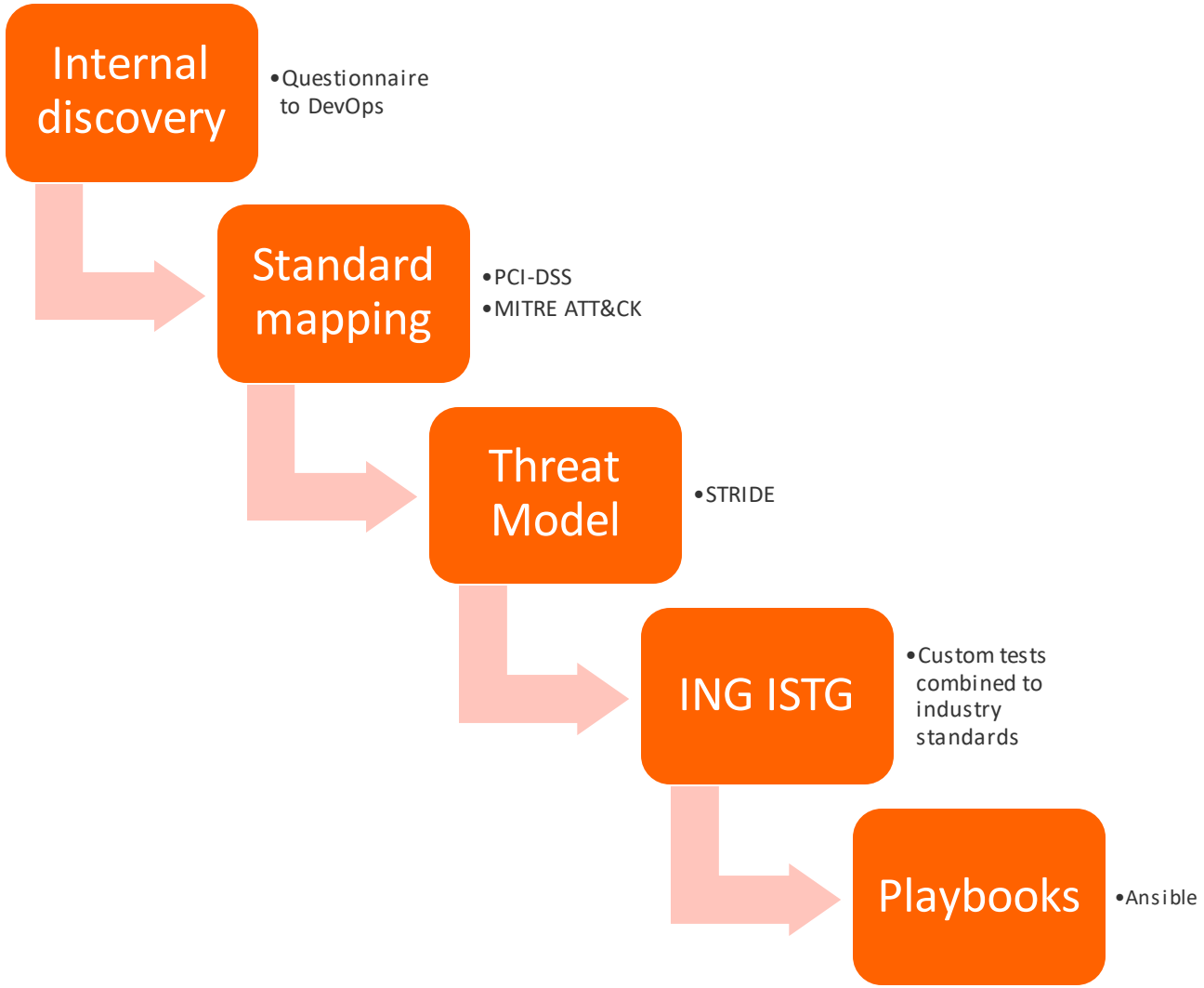
Ansible in action

```yaml
# Attacker extracts sensitive data
---
- name: ING | Check for sensitive data in the readable lo
  find:
    paths: /
    file_type: file
    recurse: yes
    hidden: yes
    patterns:
      - '(?i)^.*{{ application }}.*\.logs$'
    use_regex: yes
    contains: '^(.*firstname|.*user|.*pass|.*address|.*st
    creditcard|.*surname|.*iban|.*phone|.*ntlm|.*hash|.*k

  register: secret_app_logs
  check_mode: yes
  ignore_errors: yes

- set_fact:
    app_secret_logs: "{{ secret_app_logs.files | to_json
      gr_name, mode: mode, path: path }\") }}"

- debug:
    var: app_secret_logs
  run_once: True
```

```yaml
# Attacker manipulates application logs
---
- name: ING | Check for readable - writable application log files
  find:
    paths: /
    file_type: file
    # follow: yes
    recurse: yes
    hidden: yes
    use_regex: yes
    patterns:
      - '(?i)^.*{{ application }}.*\.logs$'
  register: application_logs
  check_mode: yes
  ignore_errors: yes


- set_fact:
    app_logs_writeble: "{{ application_logs.files | to_json | from_json |
      [?ends_with(mode, '2') == `true`|| ends_with(mode, '3') == `true`||
      == `true`||ends_with(mode, '7') == `true`].{gr_name: gr_name, mode:
      }}"

- debug:
    var: app_logs_writeble
  run_once: True

- set_fact:
    app_logs_readable: "{{ application_logs.files | to_json | from_json |
      [?ends_with(mode, '4') == `true`|| ends_with(mode, '5') == `true`||
      == `true`||ends_with(mode, '7') == `true`].{gr_name: gr_name, mode:
      }}"

- debug:
    var: app_logs_readable
  run_once: True
```


ING

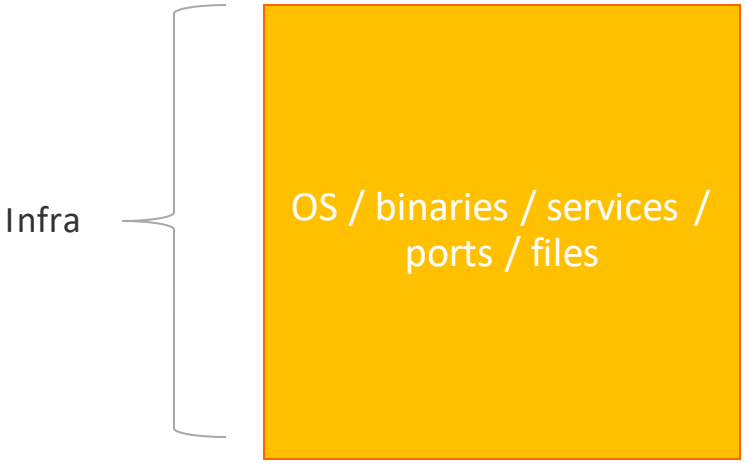
Recap


Internal discovery
•Questionnaire to DevOps
Standard mapping
•PCI-DSS
•MITRE ATT&CK
Threat Model
•STRIDE
ING ISTG
•Custom tests combined to industry standards
Playbooks
•Ansible

How is been used within ING

Define the `Default` Stack

- Map your systems, get the default stack
  - Can be used for hardening
  - Can be used for provisioning
  - Can be used to keep state



ING

How is been used within ING

Define the `Changes`

- Map the changes
- Changes per application stack (php, javascript, java …etc)
- What extra component are used from teams

ING

How is been used within ING

What about Pentesters

How is been used within ING

An example of what we look for

- Vulnerable binaries on the machine
- DoS vulnerabilities
- Privilege escalation techniques
- Sensitive information stored in clear text
- File and Folder Permissions
- Cronjobs
- Report (simple example {take it from our current report})
- ….

Which security problem(s) we can solve

- Catch the log hanging fruits
- Replicate 0 Days
- Enforce configuration
- Automate the obvious

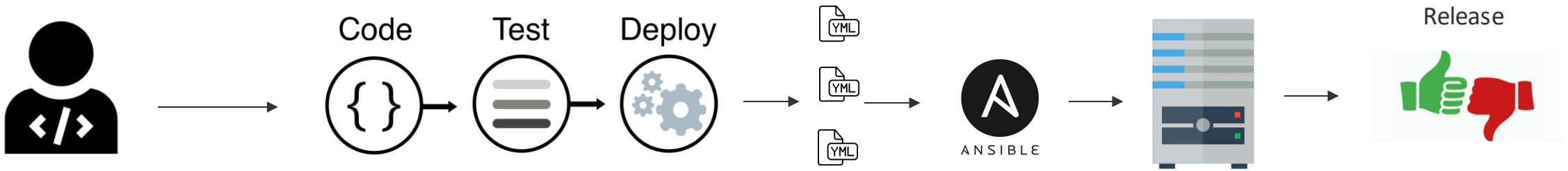

LIVE DEMO
I TOO LIKE TO LIVE DANGEROUSLY
makeameme.org

Future work

- Release the playbooks as open source project
- Embed in Azure CI/CD pipeline
- Assign Risk score to tasks
- Automate the approval/disapproval

How it will look in the near future

CI/CD pipeline (future)



ING

Sit Back and Enjoy