FILE INTEGRITY MONITOR AT SCALE

Go scale or die trying

Davide Barbato

Topic

- What Why How
- The journey of building a FIM at scale

FIM?

- **Def**: Change detection mechanism
- **Type**: Agent-based or agent-less

Features

- Detection of unauthorized changes
- Who/What/When a change happens
- File and folder continuous monitoring

FIM: why?

- PCI compliance (10.5.5, 11.5)
- McAfee Change Manager
- Because security.

Our needs

- Still passing the PCI compliance audit
- BeyondCorp’s Zero Trust Model
- Open Source
- Automation
- FIM as a Service

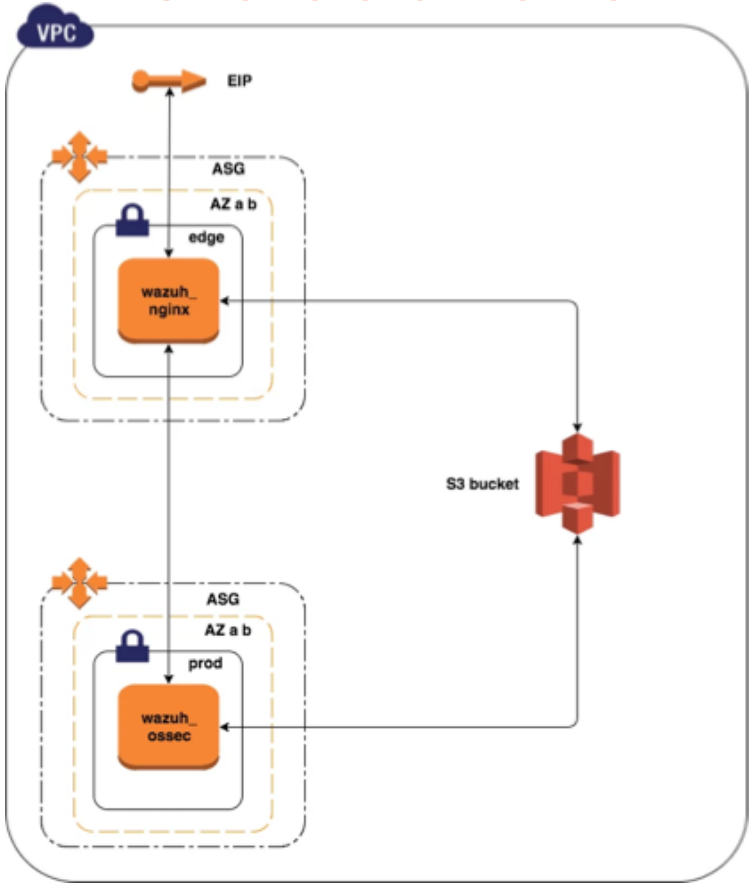
Our needs: solutions

- Still passing the PCI compliance audit (…meh...)
- BeyondCorp's Zero Trust Model Cloud (AWS)
- Open Source OSSEC
- Automation Wazuh
- FIM as a Service IaC (Terraform)

FIM: Wazuh-OSSEC

- **Wazuh** (REST APIs)
  - Nodejs app
  - Wrapper for OSSEC (using *system()*)
  - Support authentication (*htpasswd*)
  - Support SSL/TLS
- **OSSEC** (FIM)
  - Agent-based
  - Support profiles
  - Flexible & customizable
  - Active-response

FIMv1 in the cloud: architecture



FIMv1: features

- Wazuh 1.2
- TLS 1.2 (strong ciphers)
- Multi-AZ
- Dynamic configuration through S3 bucket
- Data backup every 6h on S3 (EFS not working)
- A custom web-interface (**wazuh-wui**)
- Shipping logs through rsyslogd (in JSON)
- delete_agents script
- **post-hooks**

FIM trick: post-hooks

- It’s a way to extend the userdata script by adding extra commands to execute at boot time
- It’s defined by the user and it’s called by the “main userdata” at the end of its execution (just before the cleanup)
- Flexibility++

FIM trick: post-hooks

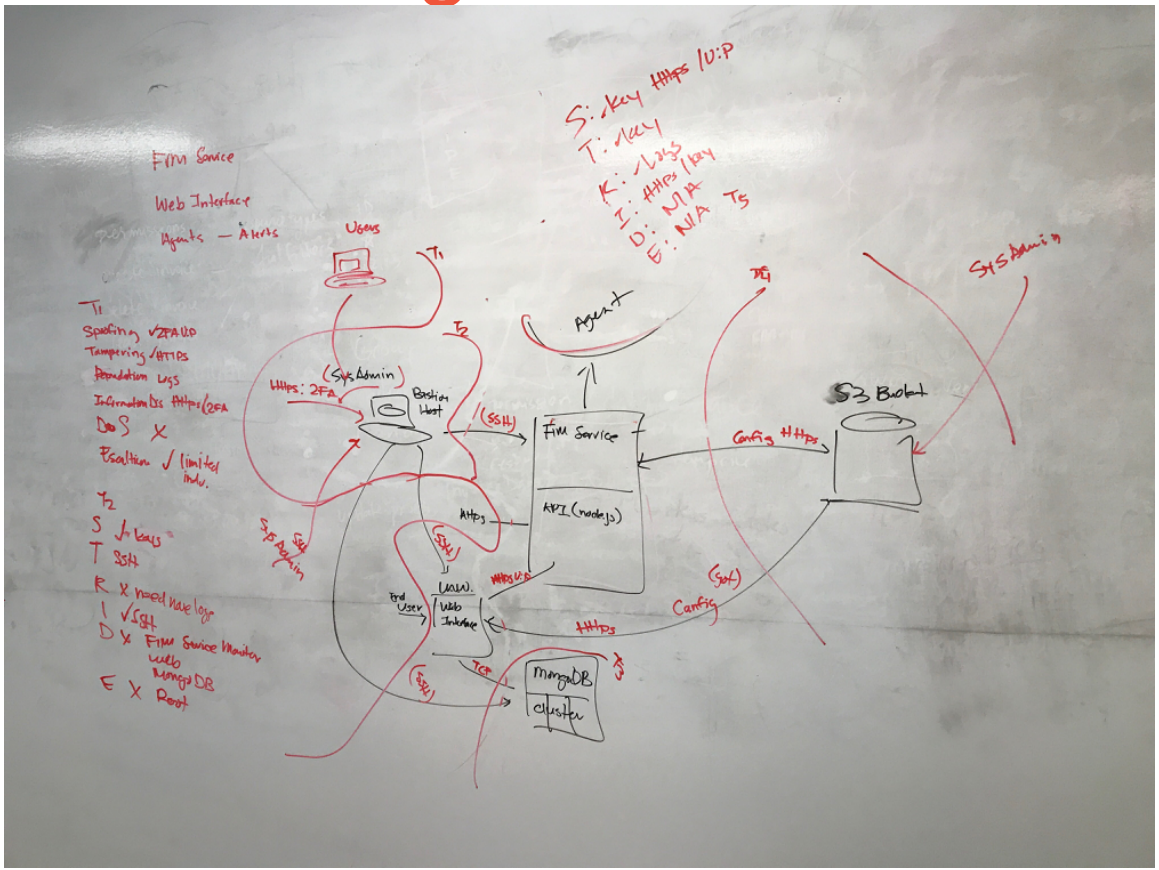
userdata

```
35
36  /etc/init.d/wazuh-manager restart
37
38  /etc/init.d/wazuh-api restart
39
40  if [ -e confs/post-hooks-ossec ]; then
41      echo "Running post-hooks..."
42      bash confs/post-hooks-ossec
43  fi
44
45  rm -rf confs/
```

post-hooks-ossec

```
1   OSSEC_DIR="/var/ossec"
2
3   mv confs/ossec.conf $OSSEC_DIR/etc/
4   mv confs/local_rules.xml $OSSEC_DIR/rules/
5
6   $OSSEC_DIR/bin/ossec-control restart
7
8   mv confs/22-nxlog-feye.conf /etc/rsyslog.d/
9
10  if ! [ -d /root/certs ]; then
11      mkdir /root/certs
12  fi
13
14  mv confs/*.crt /root/certs
15  mv confs/*.key /root/certs
16  /etc/init.d/rsyslog restart
17
18  # remove wazuh' ssl support since we're moving SSL on nginx (it causes conflict)
19  sed -i "s/config.https = \"yes\";/config.https = \"no\";/" $OSSEC_DIR/api/config.js
20
21  # restart wazuh
22  node_pid=`pidof node`
23  kill -9 $node_pid && node $OSSEC_DIR/api/app.js &
```

Threat Modeling



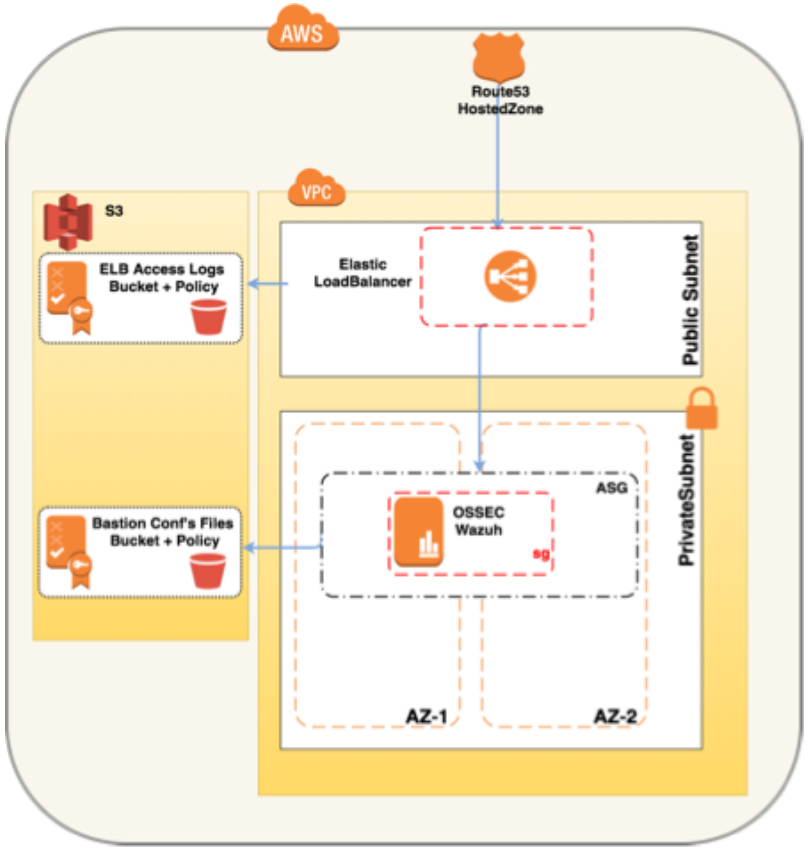
FIMv1: problems

- ELB does **NOT** support UDP
- Too much tied to our infrastructure
- Manual checks for EC2s (you know, no ELB…)
- IAM policies too permissive (*post-hooks* can be used for attacker persistence)

FIMv2

- Wazuh 3.1!!
- TCP instead of UDP (= ELB instead of nginx)
- Removed all the manual ASG/EC2 checks
- Stricter IAM role/bucket policies
- Default **no** SSL (offloading to the ELB)
- nxlog agent for log forwarding (+flexibility)
- ASG events notification (via Slack)
- Modular approach
- Chaos Monkey support (through *chaos_monkey* tag)

FIMv2: architecture



```
provider "aws" {
  region = "${var.aws_region}"
}

provider "vault" {
  address = "${var.vault_addr}"
}

module "ossec" {
  source = "/path/to/module/ossec_wazuh"

  vpc_id = "vpc-e568d681"

  project = "wazuh3"
  squad = "SET"

  subnet_pub_ids = "subnet-99a104c1,subnet-e1a93c97"
  subnet_priv_ids = "subnet-e2a93c94,subnet-e6a104be"

  wazuh_name = "${var.wazuh_name}"

  wazuh_passwd = "${data.vault_generic_secret.wazuh_creds.data["password"]}"
  wazuh_username = "${data.vault_generic_secret.wazuh_creds.data["username"]}"

  s3_bucket_name = "${data.template_file.s3_bucket_name.rendered}"
  s3_replica_region = "${var.s3_replica_region}"

  upload_path = "${path.root}/upload/ossec"

  keypair_name = "${var.keypair_name}"

  ossec_ami_id = "${data.aws_ami.amzn_ossec_ami.image_id}"

  bastion_host_sg_id = "${aws_security_group.bastion_guac_elb_sg.id}"

  sns_topic = "arn:aws:sns:eu-west-1:XXXXXXXX:SlackNotify"

  ssl_certificate_id = "arn:aws:acm:eu-west-1:XXXXXXXX:certificate/yyyyyyyy-xxxx-zzzz-vvvv-kkkkkkkkkkkk"

  chaos_monkey = "true"
}
```

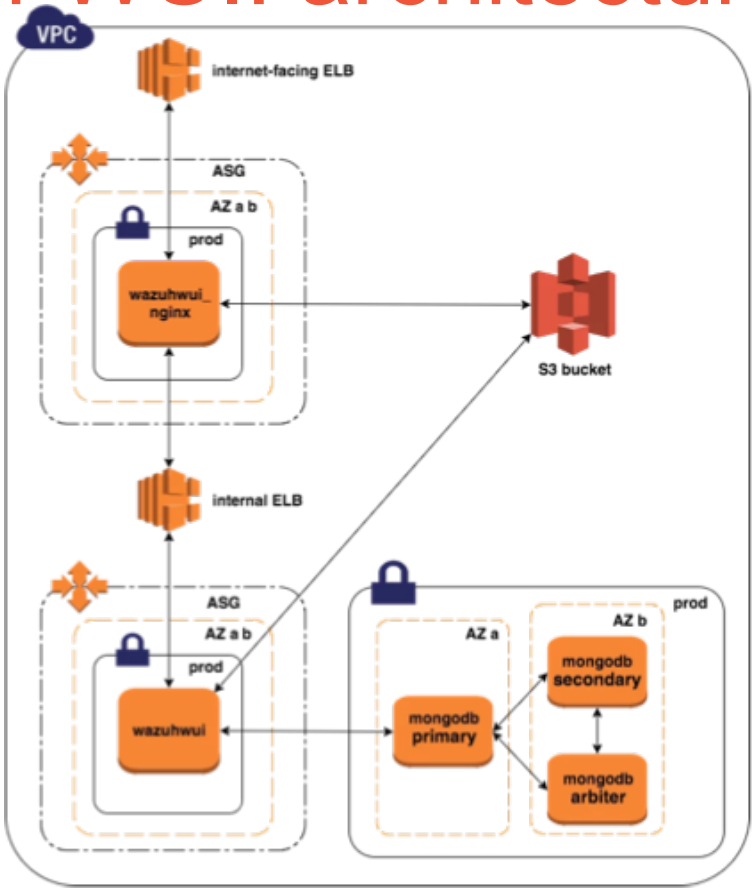
```
module "cloudwatch_dashboard" {
  source = "/path/to/module/cloudwatch_ddos_dashboard"

  dashboard_name = "${var.wazuh_name}-ddos"

  asg_name = "${module.ossec.wazuh_asg_name}"
}

module "inspector" {
  source = "/path/to/module/inspector_automation"

  name = "${var.wazuh_name}"

  inspector_tag = "wazuh-${var.wazuh_name}"

  template_duration = 3600

  cloudwatch_sched_rule = "rate(7 days)"
}

resource "aws_route53_record" "www" {
  zone_id = "XXXXXXXXXXXXX"
  name    = "${var.dns_record}"
  type    = "A"

  alias {
    name    = "${module.ossec.lb_dns}"
    zone_id = "${module.ossec.lb_zone_id}"
    evaluate_target_health = false
  }
}
```

```
module "wazuh_wui" {
  source = "/path/to/module/wazuh_wui"

  vpc_id = "vpc-e568d681"

  project = "wazuh3"
  squad = "SET"

  subnet_pub_ids = "subnet-99a104c1,subnet-e1a93c97"
  subnet_priv_ids = "subnet-e2a93c94,subnet-e6a104be"

  wazuh_name = "${var.wazuh_name}"
  wazuhwui_password = "${var.wui_passwd}"

  s3_bucket_name = "${module.s3_repl.s3_bucket}"

  upload_path = "${path.root}/upload/wazuhwui"

  availability_zones = "${var.az}"

  keypair_name = "${var.keypair_name}"

  instance_profile = "${module.iam_roles.wazuh_profile_name}"

  ssl_certificate_id = "arn:aws:acm:eu-west-1:XXXXXXXXXXXX:certificate/YYYYYYYY-ZZZZ-WWWW-NNNN-BBBBBBBBBBBB"
}
```

Wazuh-WUI

- Web interface to manage agents
- Easy to use
- Can do everything the APIs can do (still need to implement everything…)
- Written in python (tornado) + semantic-ui (front-end)

Wazuh-WUI: architecture



FIM: deploying agents

- Automatic deploy thanks to the API
- Linux & Windows (bash & powershell)
- Can be deployed in userdata script
- Can be deployed with Puppet/Chef/Ansile/Packer/you name it

Future

- Kubernetes
- Health check dashboard
- WAF (both wazuh and wazuh-wui)
- Tags “flexibility”
- EC2 spot instances
- Wazuh-WUI moves to DynamoDB/Aurora
- Wazuh-WUI native auth (…maybe...)
- Wazuh-WUI integrates with Vault for Wazuh API creds

Links

- https://wazuh.com/
- https://github.com/Cimpress-MCP/terraform
- dbarbato@cimpress.com

Questions?