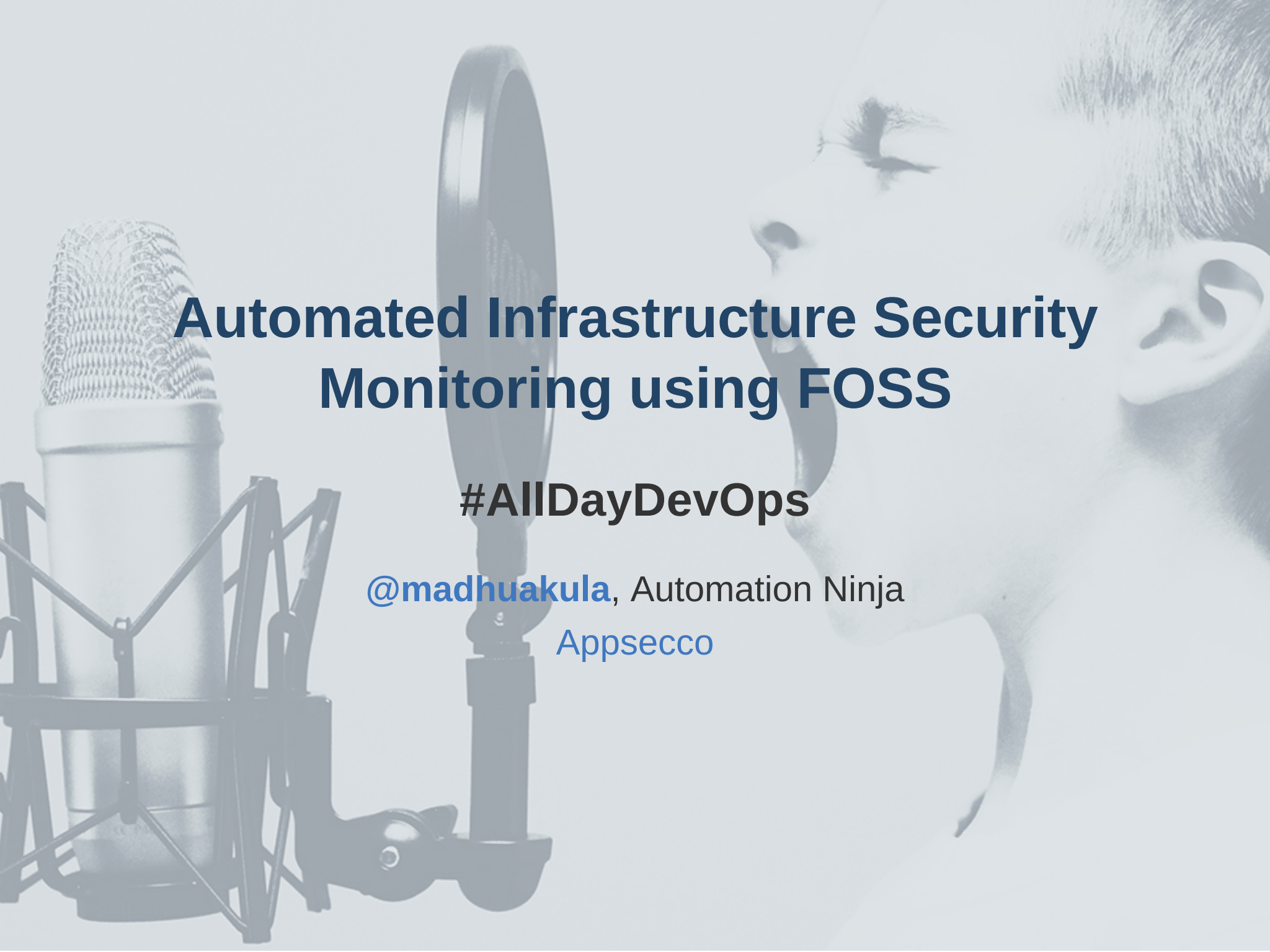


Automated Infrastructure Security Monitoring using FOSS

#AllDayDevOps

@madhuakula, Automation Ninja

Appsecco

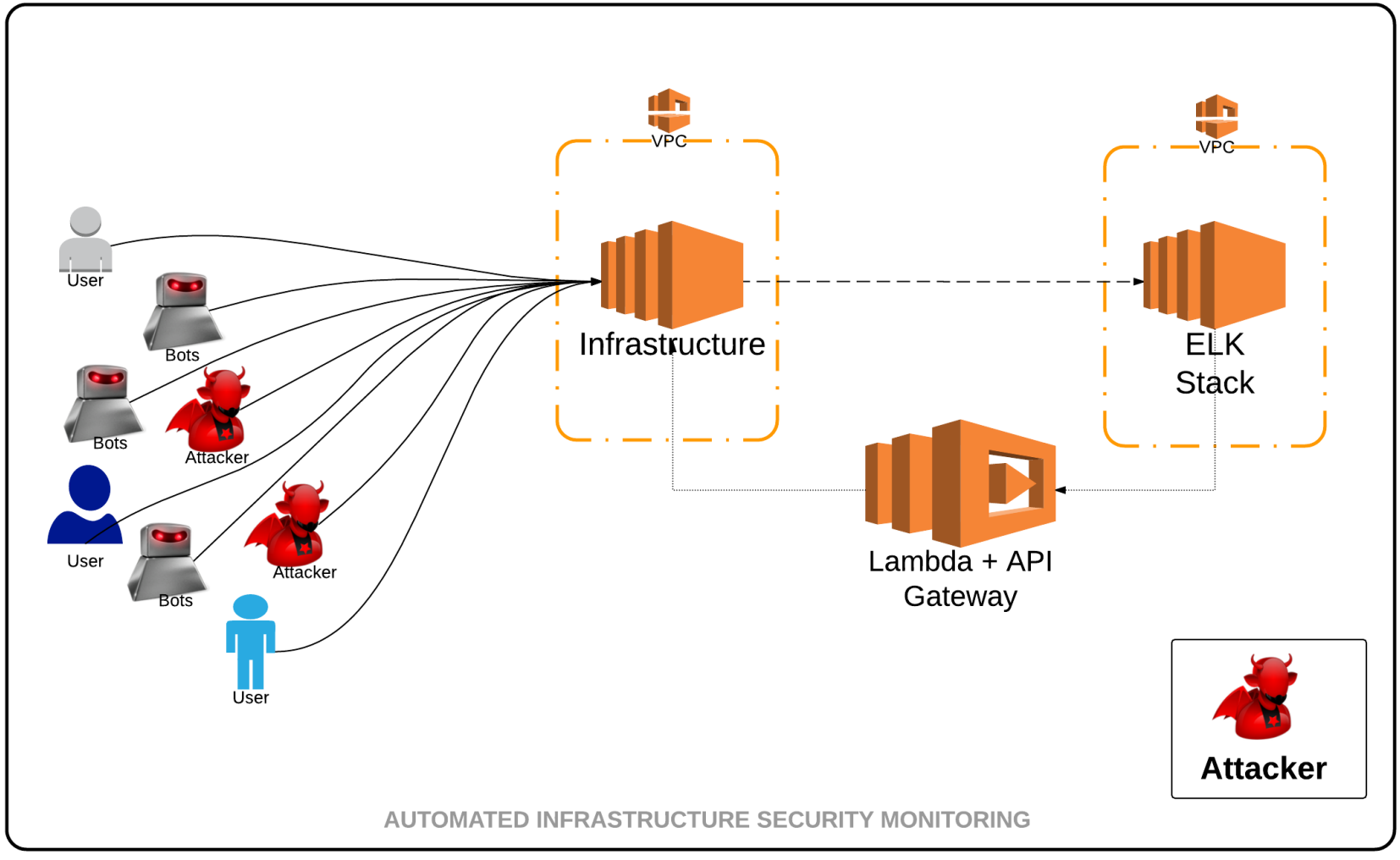
About Me !

- Automation Ninja at Appsecco
  - Appsecco is a specialist application security company
- Interested in Security, DevOps & Cloud
- Found bugs in Google, Microsoft, Yahoo, etc
- Never ending learner!
- Follow (or) Tweet to me @madhuakula

What we are covering today?

- ELK stack to analyse and visualise logs in near real-time
- ElastAlert to create rules to automatically defend against SSH bruteforce attacks
- AWS Lambda to do this, since our infra is hosted on AWS
- Python based Chalice framework for using AWS Lambda

Architecture



Automated Defence Demo

http://bit.ly/addo-aism

AWS Lambda - Chalice Code

```
@app.route('/[redacted]/ip/inframonitor/{ipadd}')
def ip_address(ipadd):
    network_acl_id = 'acl-[redacted]' # Subsititue your AWS VPC Network ACL ID
    acl_action = 'deny'
    protocol_num = 6 # TCP protocol (http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml)
    acl_rule_num = 99 # Any rule above 1 and below 100 is acceptable
    port_num = 22 # Default SSH port

    if app.current_request.context["source-ip"] == allowed_ip: # To ensure only whitelisted IP can make API call
        vpc2conn = boto.vpc.VPCConnection(aws_access_key_id, aws_secret_access_key, is_secure=True, host=None, port=None
            , proxy=None, proxy_port=None, proxy_user=None, proxy_pass=None, debug=0, https_connection_factory=None,
            region=None, path='/', api_version=None, security_token=None, validate_certs=True, profile_name=None)
        network_acl_entry = vpc2conn.create_network_acl_entry(network_acl_id, acl_rule_num, protocol_num, acl_action,
            ipadd + '/32', egress=False, icmp_code=-1, icmp_type=-1, port_range_from=port_num, port_range_to=port_num)
        return {'ip address': ipadd }
    else:
        return {'error': 'Not Authorised'}
```

https://github.com/appsecco/alldaydevops-aism

Security for our AWS Lambda

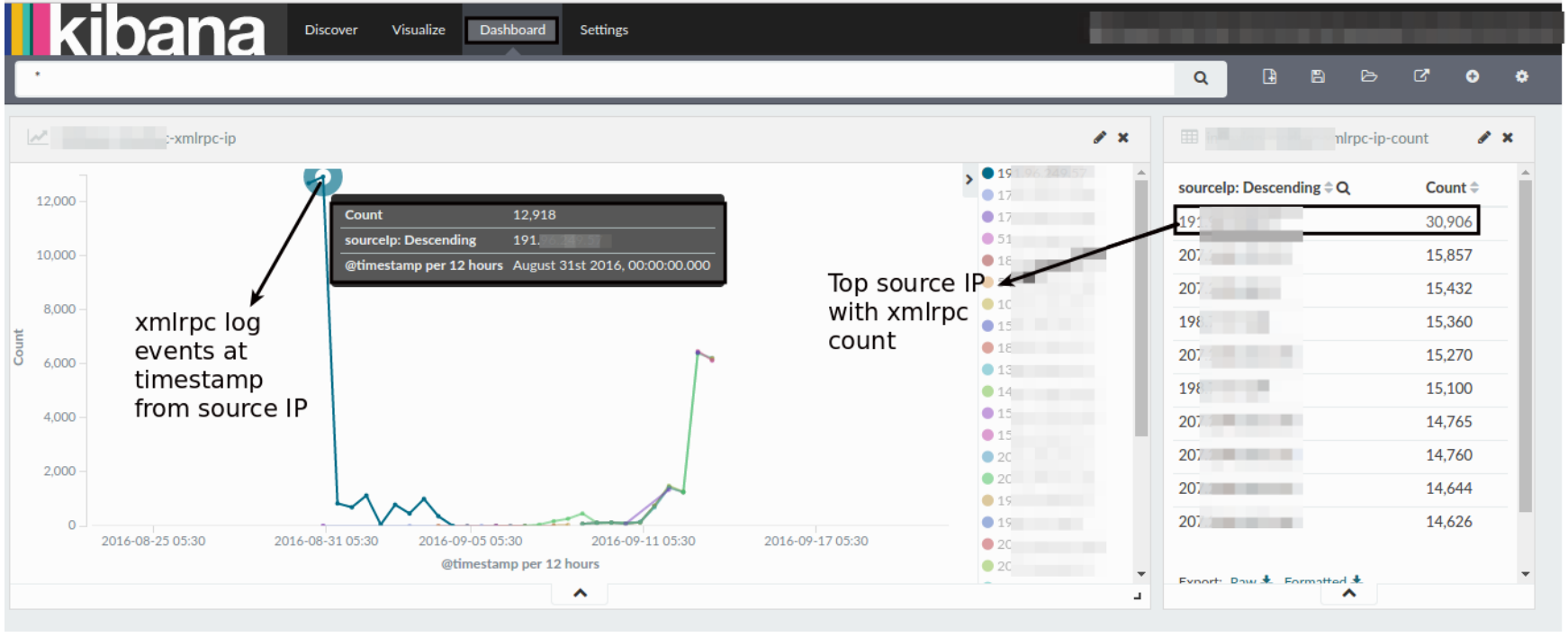
We are primarily doing the following two things

1. A *sufficiently random token* to protect the request when we post the IP address from ElastAlert
2. *Whitelist* the IP address of the host where the `HTTP POST` request originates from

Use Cases for Automated Defence

1. Automated Defender (Attack Alerts + Automated Firewall)
2. Security Analytics + Reports
3. Near real-time Centralised Log Monitoring

Attack Scenario : Wordpress XML-RPC



https://blog.appsecco.com/analysing-attacks-on-a-wordpress-xml-rpc-using-an-elk-stack-3bf25a7e36cc

Needs Improvement

- More attack signatures required
- For example OSSEC Wazuh Ruleset
- Improve the ElastAlert Alerter custom code
- Any suggestions from your side

Alternatives to our stack

Stack

- Elastic
- Graylog
- TICK Stack
- Prometheus + Grafana

Serverless

- AWS Lambda
- Azure Functions
- Cloud Functions

Our assumptions

- You are already monitoring in near real-time using the ELK stack
- You are under attack for a specific service
- You have configured ElastAlert for your alerting

In Summary

- We created attack threshold rules in ElastAlert
- We created an AWS Lambda endpoint to be able to modify AWS VPC Network ACLs
- We have a real-time blocking system infinitely scalable

References

- Blog Post
- Elastic
- Elast Alert
- AWS Lambda
- Chalice

COMPANIES WHO MADE THIS EVENT POSSIBLE

Sponsors for All Day DevOps 2016

Sonatype

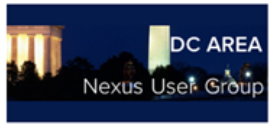
SAUCELABS

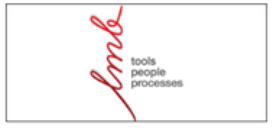
Signal Sciences

cloudbees

XebiaLabs

SUPPORTERS OF ALL DAY DEVOPS





Thanks

@madhuakula | @appseccouk | http://appsecco.com