

The background of the image is a vibrant, abstract composition of overlapping, translucent geometric shapes. These shapes, primarily in shades of blue and purple, create a sense of depth and movement, resembling a stylized architectural structure or a complex data visualization. The lines are sharp and angular, contributing to a modern, tech-oriented aesthetic.

# AWS re:Inforce

JULY 26 – 27, 2022 | BOSTON, MA

TDR233

# Building a chatbot for security operations

Byron Pogson (he/him)

Senior Security Solutions Architect  
Amazon Web Services

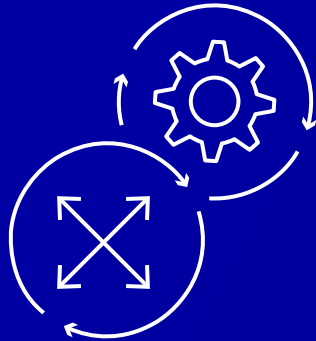
Aarthi Kannan (she/her)

Security Partner Solutions Architect  
Amazon Web Services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# AWS Well-Architected



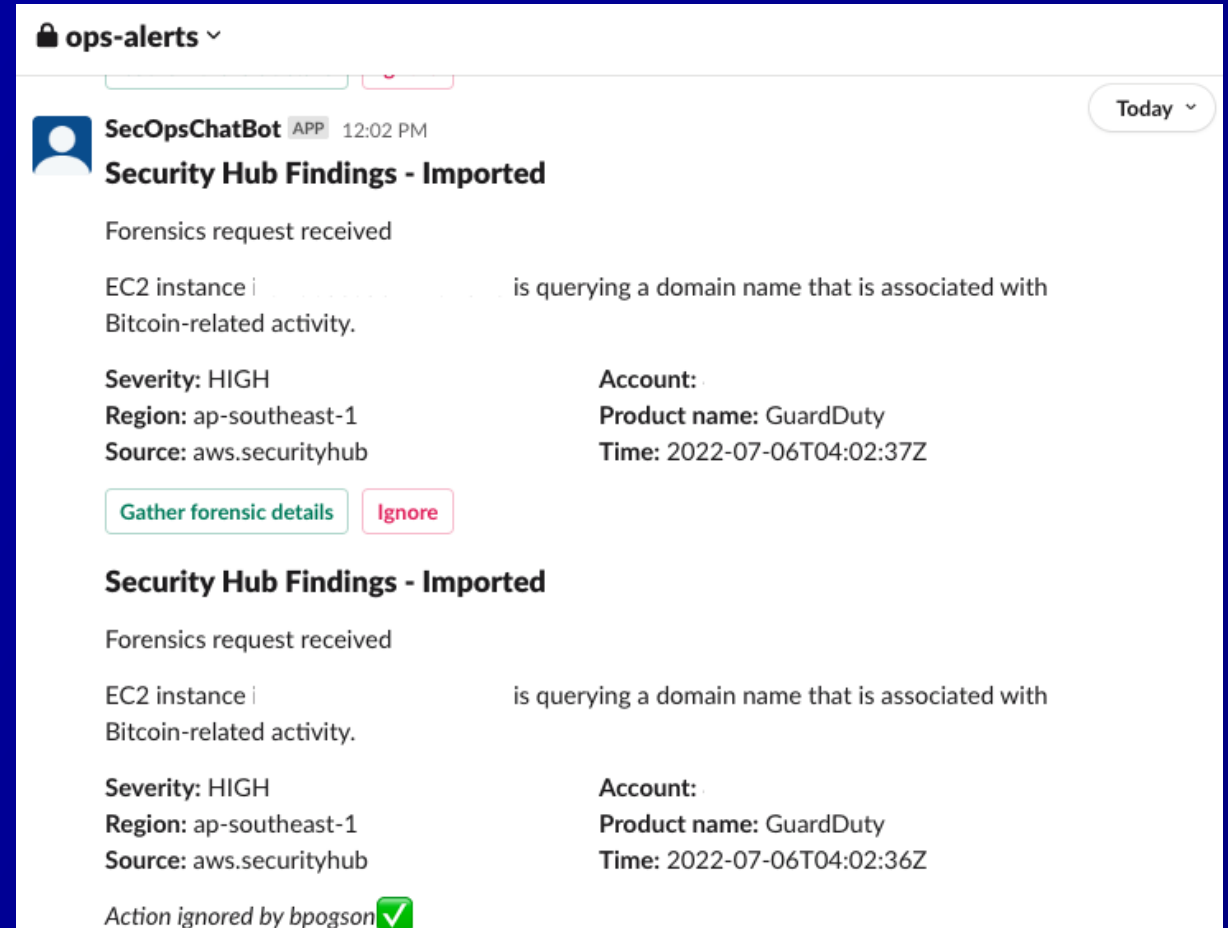
Automate security best practices



Keep people away from data

# A chatbot can help

- The platform we're using
- Visibility
- Take action
- Shorten response time



The screenshot displays the AWS OpsAlerts chat interface. At the top, there's a header with a lock icon and the text 'ops-alerts'. Below this, a chat message from 'SecOpsChatBot' (labeled 'APP') is shown, timestamped '12:02 PM'. The message title is 'Security Hub Findings - Imported'. The main text of the message describes a 'Forensics request received' from an 'EC2 instance' that is 'querying a domain name that is associated with Bitcoin-related activity'. The message includes metadata: 'Severity: HIGH', 'Region: ap-southeast-1', 'Source: aws.securityhub', 'Account:', 'Product name: GuardDuty', and 'Time: 2022-07-06T04:02:37Z'. Below the message, there are two buttons: 'Gather forensic details' (in green) and 'Ignore' (in red). The message is repeated below a section header 'Security Hub Findings - Imported'. At the bottom, a status message reads 'Action ignored by bpogson' followed by a green checkmark icon.

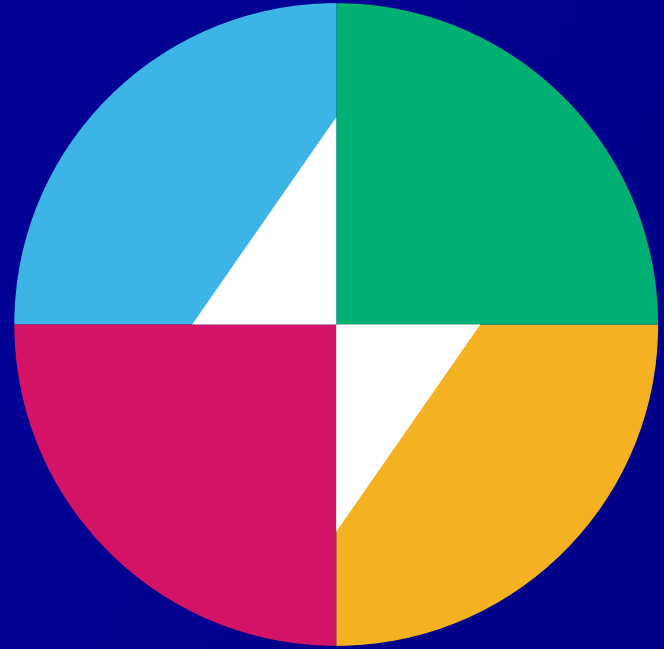


# Basic architecture



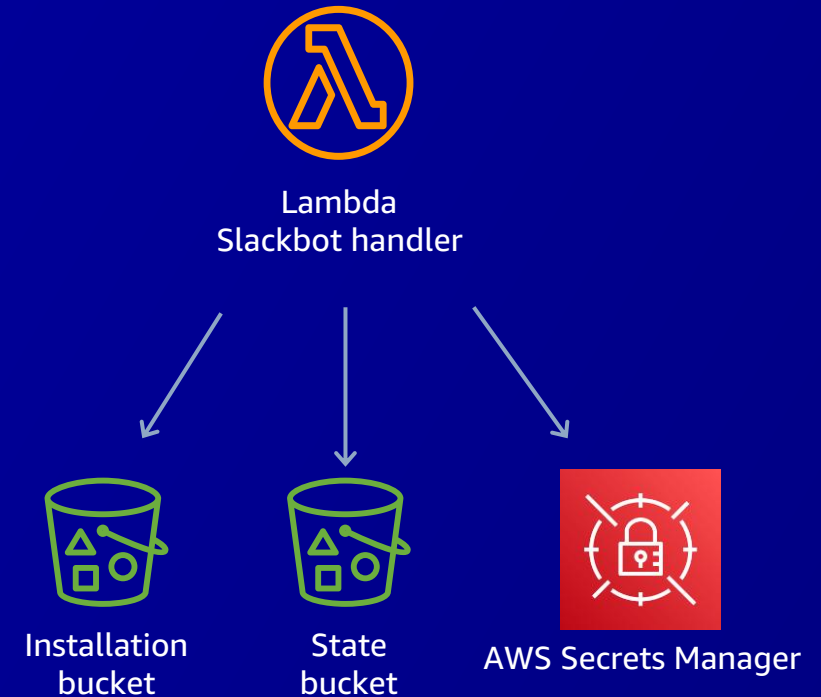
# Slack Bolt

- Framework for creating Slack applications
- Handles OAuth
- Annotations for interactivity
- Easy API access



# Slack Bolt – Initial configuration

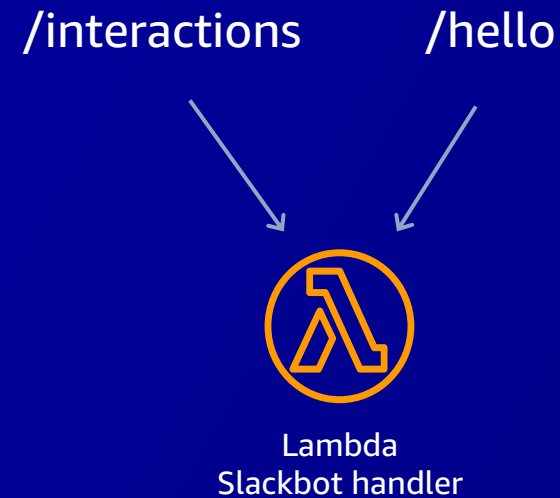
```
app = App(  
    process_before_response=True,  
    oauth_flow=LambdaS3OAuthFlow()  
)  
  
def slack_lambda_handler(event, context):  
    slack_handler = SlackRequestHandler(app=app)  
    return slack_handler.handle(event, context)
```



# Slack Bolt – Event handlers

```
@app.event("app_mention")
def handle_app_mentions(body, say):
    say(get_hello_message())
```

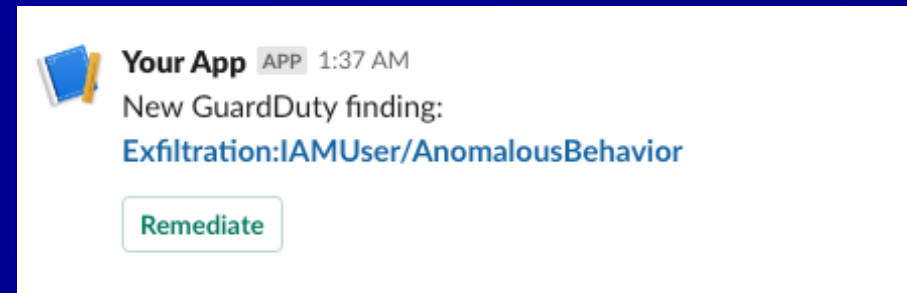
```
@app.command("/hello")
def handle_hello_command(ack):
    ack(get_hello_message())
```





# Slack Bolt – Views

```
{
  "blocks": [
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "New GuardDuty finding:\n*<example.com|Exfiltration:IAMUser/AnomalousBehavior>*"
      }
    },
    {
      "type": "actions",
      "elements": [
        {
          "type": "button",
          "text": {
            "type": "plain_text",
            "emoji": true,
            "text": "Remediate"
          }
        }
      ],
      "style": "primary",
      "action": "remediate_guardduty_finding"
    }
  ]
}
```



# Slack Bolt – View interaction

```
{
  "type": "button",
  "text": {
    "type": "plain_text",
    "emoji": True,
    "text": "Remediate"
  },
  "style": "primary",
  "value": json.dumps(data),
  "action": "remediate_guarddduty_finding"
}
```

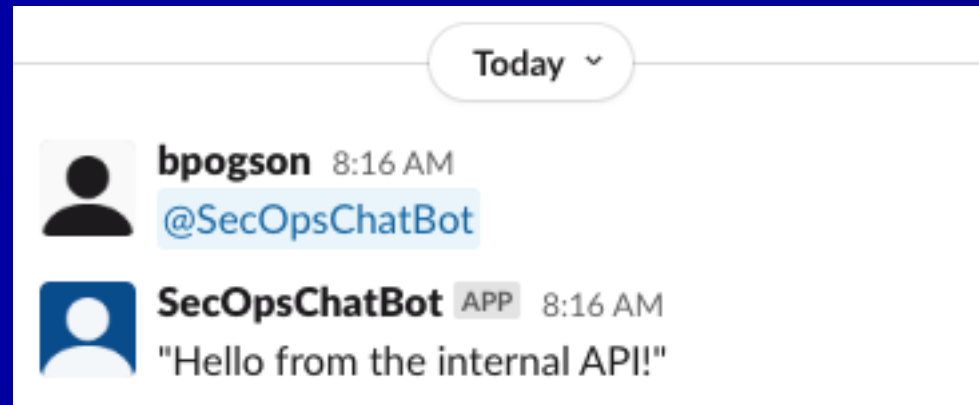
```
@app.action("remediate_guarddduty_finding")
def handle_remediation_action(ack, respond, payload, body):
    """Handles a pipeline rejection event from Slack
```

Args:

ack (Ack): The acknowledgement to interact with from Slack  
respond (Respond): The respond object from the Slack context  
payload (Payload): The payload from the Slack event  
body (Body): The Slack body containing the full message  
"""

```
    remediate_finding()
```

# Intro module



# What would you like to automate?



# Future considerations

- Identity implications
- Multi-account
- Multi-Region
- Deployment mode – single tenant or multi-tenant
- Operations – deployment pipeline, ownership



# Thank you!

