Securing your Lambda 101

:~/$whoami

- Security Engineer @Yandex Cloud
- Author of quite a few articles on macOS malware
- Former macOS malware analyst @Kaspersky
- /in/mogilin/

Agenda

- What are AWS Lambdas?
- Why go serverless?
- How do Lambdas work?
- Security and risk assessment
- Case 1: Abusing environment secrets
- Case 2: Abusing request forgery + demo
- Case 3: Avoiding fork bombs
- Questions and feedback

What are AWS Lambdas?

AWS Lambda is a compute service that runs your code in response to events and automatically manages the compute resources, making it the fastest way to turn an idea into a modern, production, serverless applications.

Why go serverless?

AWS Database Blog

Introducing Amazon Aurora DSQL

by Raluca Constantin and Arun Sankaranarayanan | on 03 DEC 2024 | in Amazon Aurora, Announcements, DSQL, Featured, Foundational (100) | Permalink | Comments | Share

Why go serverless?

AWS Database Blog

Introducing Amazon Aurora DSQL

Serverless >

- Cloud Functions
  Running your code as a function
- API Gateway
  Integration with Yandex Cloud services
- Data Streams
  Data streams management
- Yandex Cloud Postbox
  A transactional email service
- Managed Service for YDB
  Distributed fault-tolerant SQL DBMS
- Object Storage
  Scalable data storage
- Serverless Containers
  Running containers without Kubernetes®
- Yandex Query
  Serverless S3 analytics and streaming queries
- Message Queue
  Queues for messaging between applications
- IoT Core
  Solutions for Internet of Things
- Cloud Apps Preview
  Ready-to-use cloud apps
- Serverless Integrations Preview
  Configure and manage Serverless-based service integrations.

Why go serverless?

AWS Database Blog

Introducing Amazon Aurora DSQL

Serverless >

Cloud Functi
Running your c

API Gateway
Integration wit

Data Stream
Data streams r

Yandex Clou
A transactiona

DEV339

Supercharge Lambda functions with Powertools for AWS Lambda

Raphael Manke
(he/him)
Senior IT Consultant, Cloud
codecentric AG

Message Queue
Queues for messaging between applications

IoT Core
Solutions for Internet of Things

Cloud Apps Preview
Ready-to-use cloud apps

Serverless Integrations Preview
Configure and manage Serverless-based service integrations.

Why go serverless?

🎤 Topic: "Doing serverless on AWS with Terraform for real"

AWS Database Blog

Introducing Amazon Aurora DSQL

Serverless >

Cloud Functi
Running your c

API Gateway
Integration wit

Data Stream
Data streams r

Yandex Clou
A transactiona

DEV339

Supercharge Lambda functions with Powertools for AWS Lambda

Raphael Manke
(he/him)
Senior IT Consultant, Cloud
codecentric AG

Message Queue
Queues for messaging between applications

IoT Core
Solutions for Internet of Things

Cloud Apps Preview
Ready-to-use cloud apps

Serverless Integrations Preview
Configure and manage Serverless-based service integrations.

Why go serverless?

- **Cost-effective**. Pay as you go

Why go serverless?

- **Cost-effective**. Pay as you go
- **No ops**. No need to provision additional resources, k8s-clusters, schedulers

Why go serverless?

- **Cost-effective**. Pay as you go
- **No ops**. No need to provision additional resources, k8s-clusters, schedulers
- **Speed**. Lambda functions provide cached runtime

Why go serverless?

- **Cost-effective**. Pay as you go
- **No ops**. No need to provision additional resources, k8s-clusters, schedulers
- **Speed**. Lambda functions provide cached runtime
- **Scalability**. Let your CSP handle the scaling

How do Lambdas work?

```python
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):

    # Get the length and width parameters from the event object. The
    # runtime converts the event object to a Python dictionary
    length = event['length']
    width = event['width']

    area = calculate_area(length, width)
    print(f"The area is {area}")

    logger.info(f"CloudWatch logs group: {context.log_group_name}")

    # return the calculated area as a JSON string
    data = {"area": area}
    return json.dumps(data)

def calculate_area(length, width):
    return length*width
```

Your_code.zip

How do Lambdas work?

Init section ↑

Function-handler of the event ↓

```
import json
import logging


logger = logging.getLogger()
logger.setLevel(logging.INFO)


def lambda_handler(event, context):
  # main code goes here…
```

Your_code.zip

How do Lambdas work?

```
import json
import logging


logger = logging.getLogger()
logger.setLevel(logging.INFO)


def lambda_handler(event, context):
  # main code goes here…
```

Your_code.zip

Init section ↑

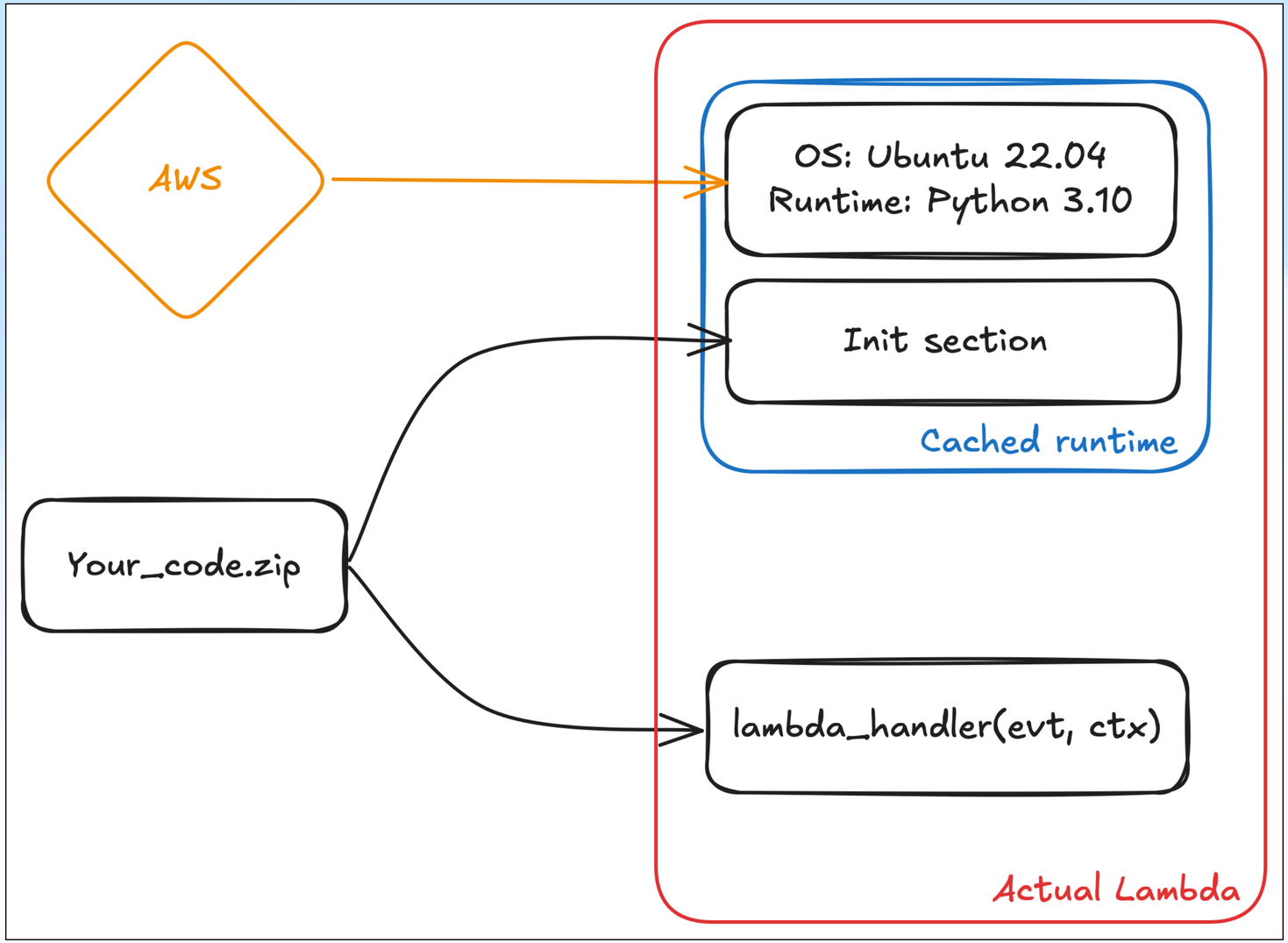
Function-handler of the event ↓

`context` includes:

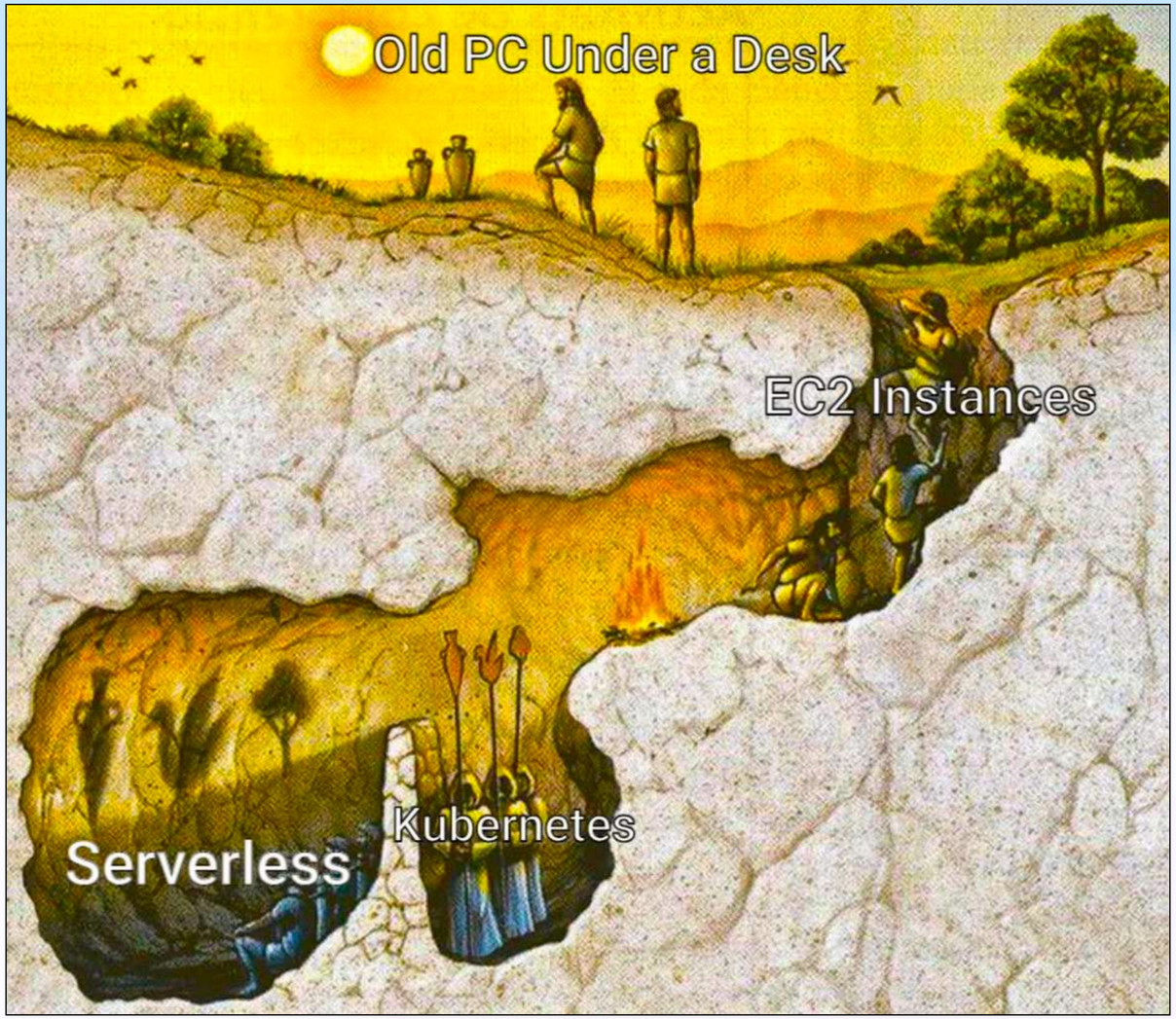
- function ARN
- CloudWatch log group name
- Lambda request ID

`event` holds the data of request

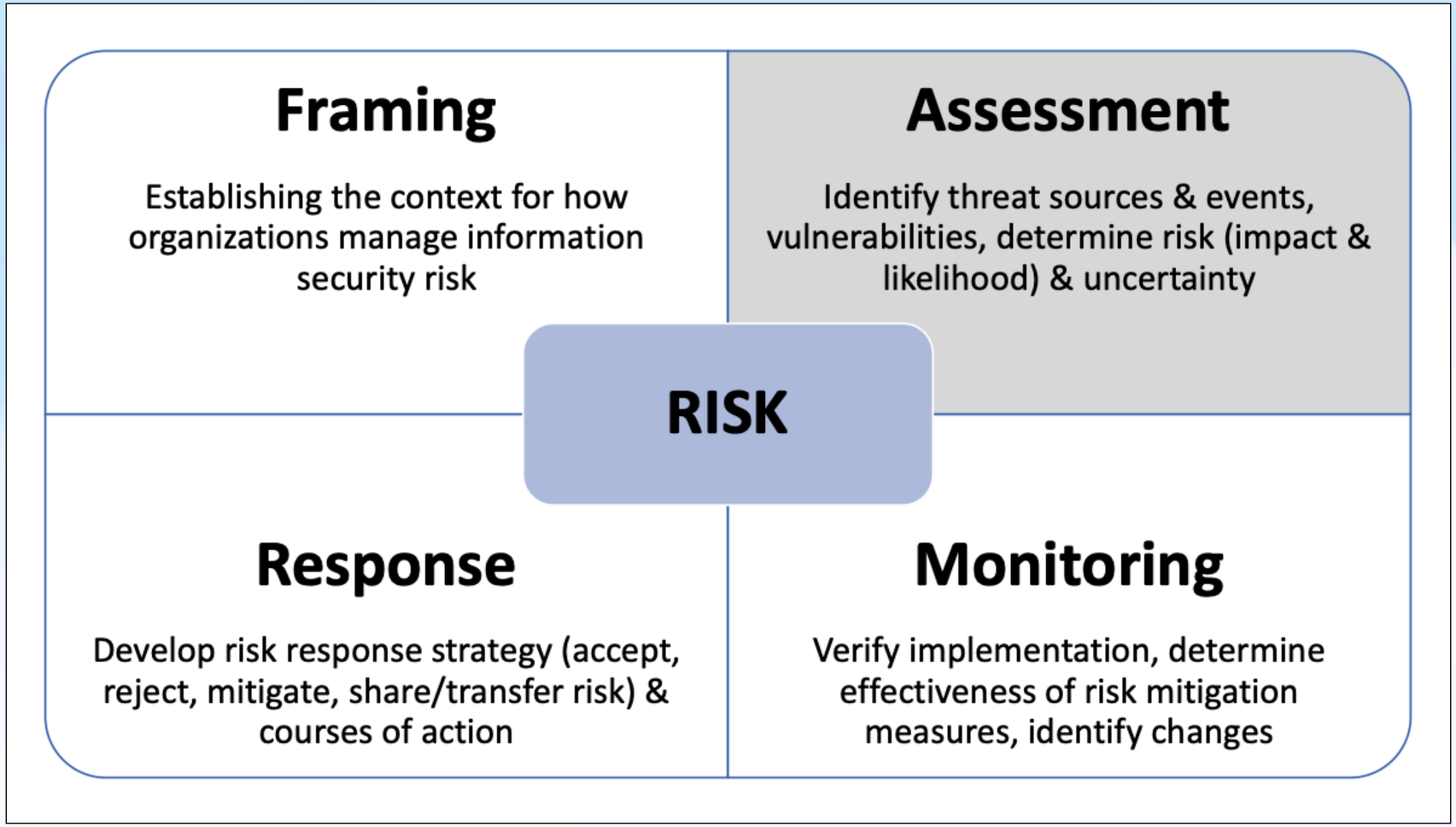
How do Lambdas work?



How do Lambdas work?



Security and risks assessment



Original NIST 800-30 presentation

Key Lambda risks

1. backdoor Lambda 👉 leak subsequent events via RCE*
2. retrieve the source via RCE*
3. retrieve environment variables, given a file read vulnerability or SSRF**
4. given permission to invoke the function, view its logs
5. generate a fork bomb

*RCE = Remote code execution
**SSRF = Server Side Request Forgery

Rami McCarthy on Lamda risks

What?!

Abusing environment secrets

```
def lambda_handler(event, context):

  command = f"aws s3 ls s3://{os.environ['IMAGE_BUCKET']}/{event['prefix']}"
  files = os.popen(command).read()
  return(files)
```

Do you see the vulnerability?👀

Abusing environment secrets

```
def lambda_handler(event, context):

  command = f"aws s3 ls s3://{os.environ['IMAGE_BUCKET']}/{event['prefix']}"
  files = os.popen(command).read()
  return(files)
```

try `event[‘prefix’]=“; env”`

`event[‘prefix’]` is unsanitized

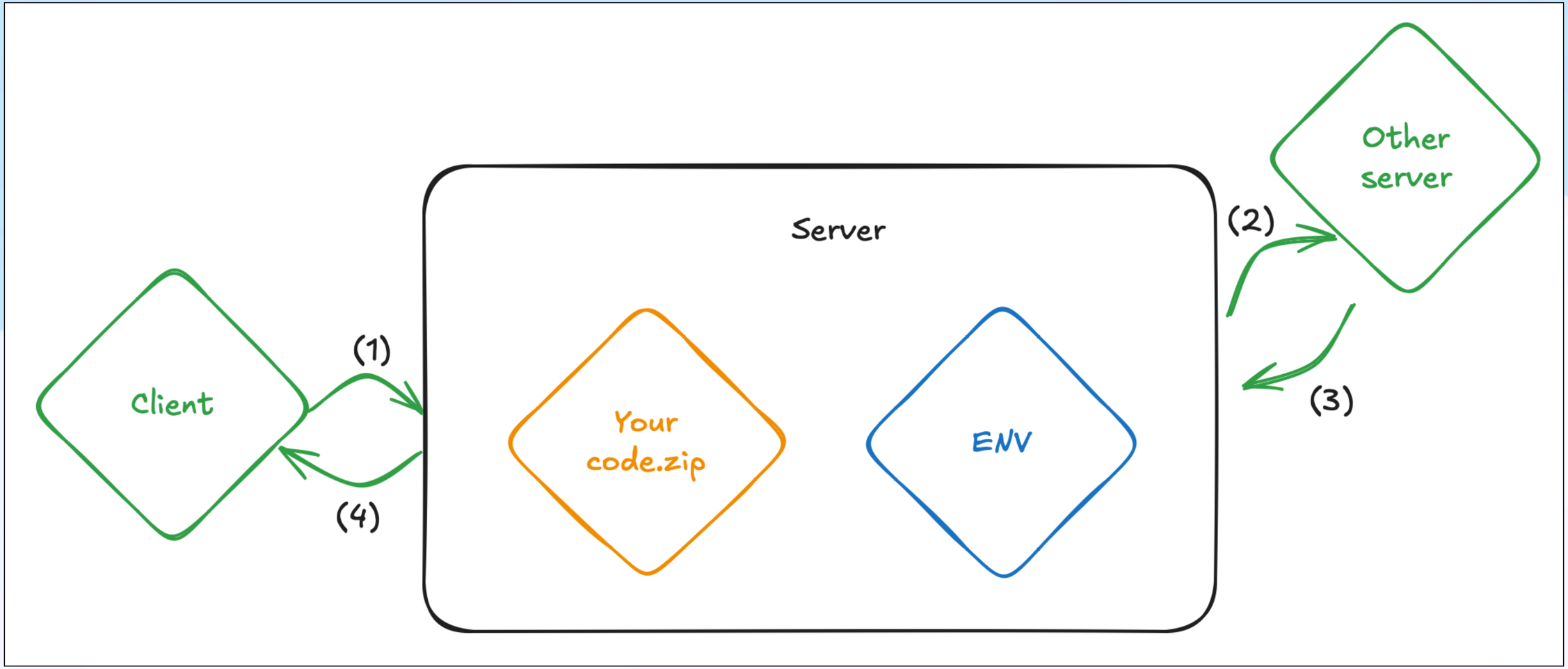
and passed directly into bash

Abusing environment secrets

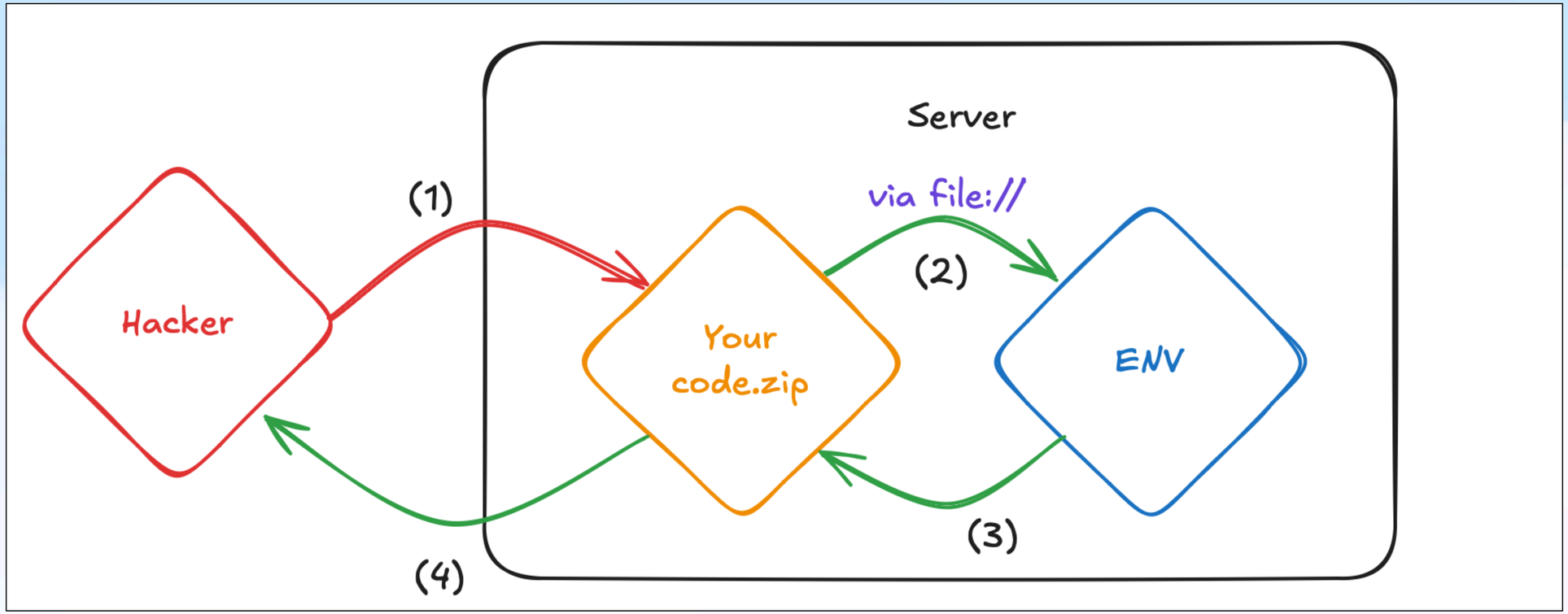
Lessons learned

- Sanitize your input
- No `eval()` or `os.Popen()` or any other direct executions
- `AWS_SESSION_TOKEN`, `AWS_SECRET_ACCESS_KEY`, `AWS_ACCESS_KEY_ID` should be kept in secret!!

Abusing request forgery



Abusing request forgery



Demo

Abusing request forgery

Lessons learned

- Validate your input — no internal IPs, no excess URL schemes
- One Lambda per one task 👉🏻 least privilege

Avoiding fork bombs

Y **Hacker News** new | past | comments | ask | show | jobs | submit login

▲ Ask HN: Experience with AWS goodwill in case of self-inflicted high bills

21 points by henriklippke 11 months ago | hide | past | favorite | 7 comments

Do you have experiences with high AWS bills due to your own mistakes?

For example, I have generated a 4000$ bill within 24 hours by an AWS Lambda function reading a row from a DynamoDB table and then called itself again using an EventBridge event ;(And that with a concurrency of 300 and so I quickly reached the 4000$ within a few hours.

I immediately opened a support ticket, but it's been 4 weeks now and it seems nobody can or wants to help me.

What are your experiences?

Avoiding fork bombs

▲ akira2501 11 months ago | parent | next [–]

Well.. I created a fork bomb on AWS Lambda. It was supposed to check a condition before self executing with a new payload, but the check was borked, so it just always self executed.

It turns out, if you do this _directly_, and not through a second service like Event Bridge, AWS will notice this fact, and will just suspend your lambda for a few minutes until all the executions die out.

It was only a $20 mistake in the end.

The real nightmare with Event Bridge is the default "retry" threshold is 185 times. It's a nuclear level nightmare. I just use the cron part now.

Avoiding fork bombs

Lessons learned

- Avoid recursion — 1 Lambda per 1 task
- Configure logging inside Lambda
- Create a billing alarm per service
- Limit concurrent executions (if possible)

Thank you!

Questions?

Feedback form