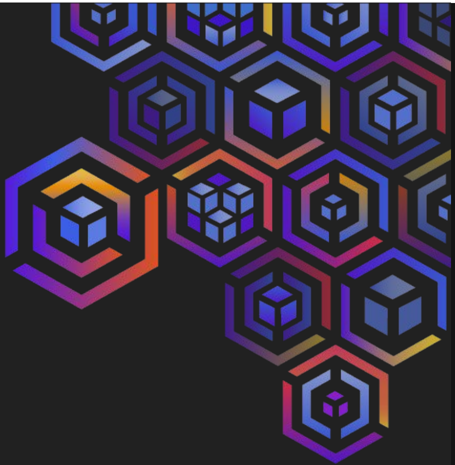


OWASP Serverless Top 10

Tal Melamed
Sr. Director, Cloud Native Security Research
Tal.Melamed@ContrastSecurity.com

```
user@host:~ █
```

4ppsec

@ 4ppsec

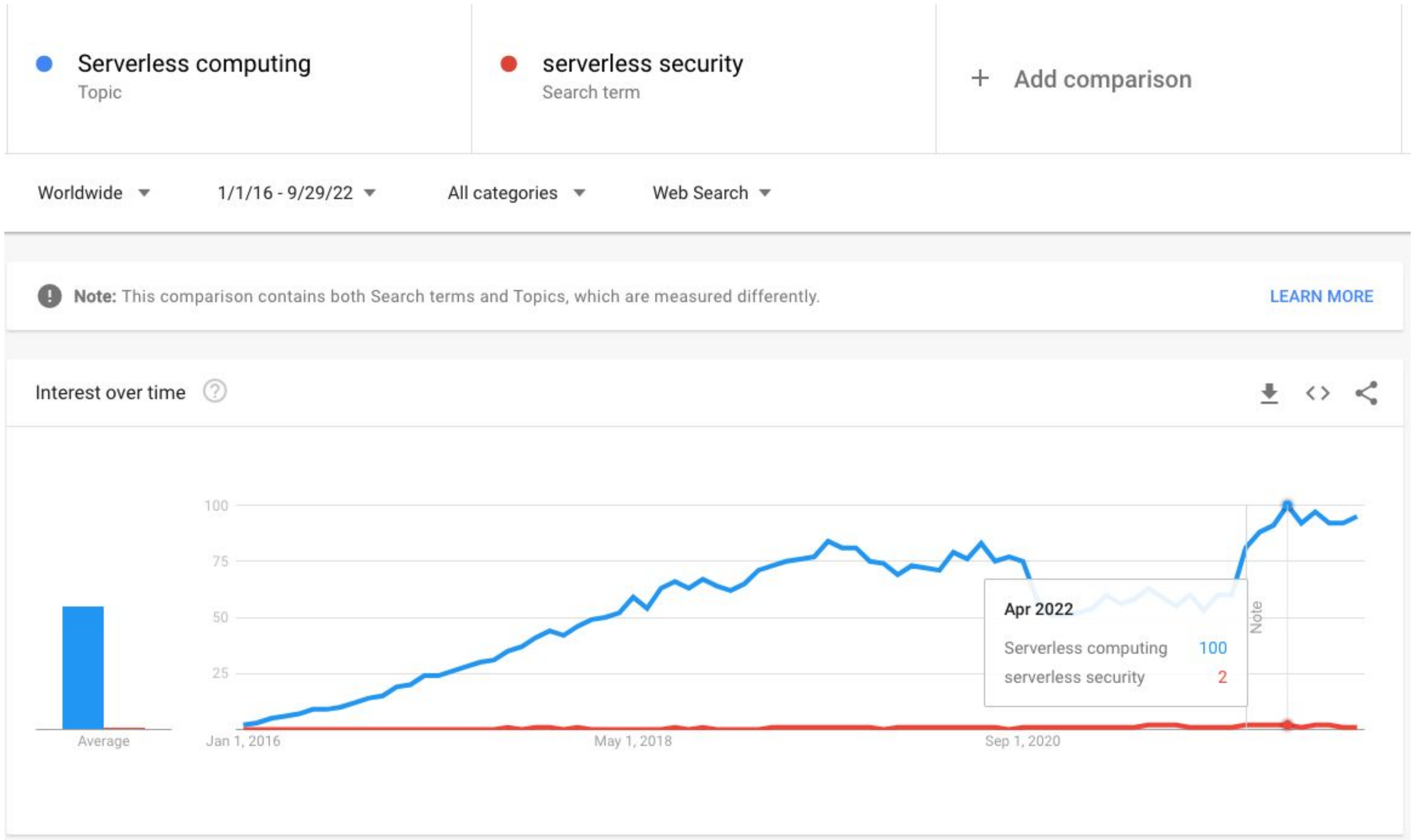
talmelamed

tal.melamed@qu.edu

tal@appsec.it

cloudessence — Co-Founder & CTO
Acquired by Contrast Security, 2020

Protego — Head of Security Research
Acquired by CheckPoint, 2019



● Serverless computing
Topic

● serverless security
Search term

+ Add comparison

Worldwide ▾ 1/1/16 - 9/29/22 ▾ All categories ▾ Web Search ▾

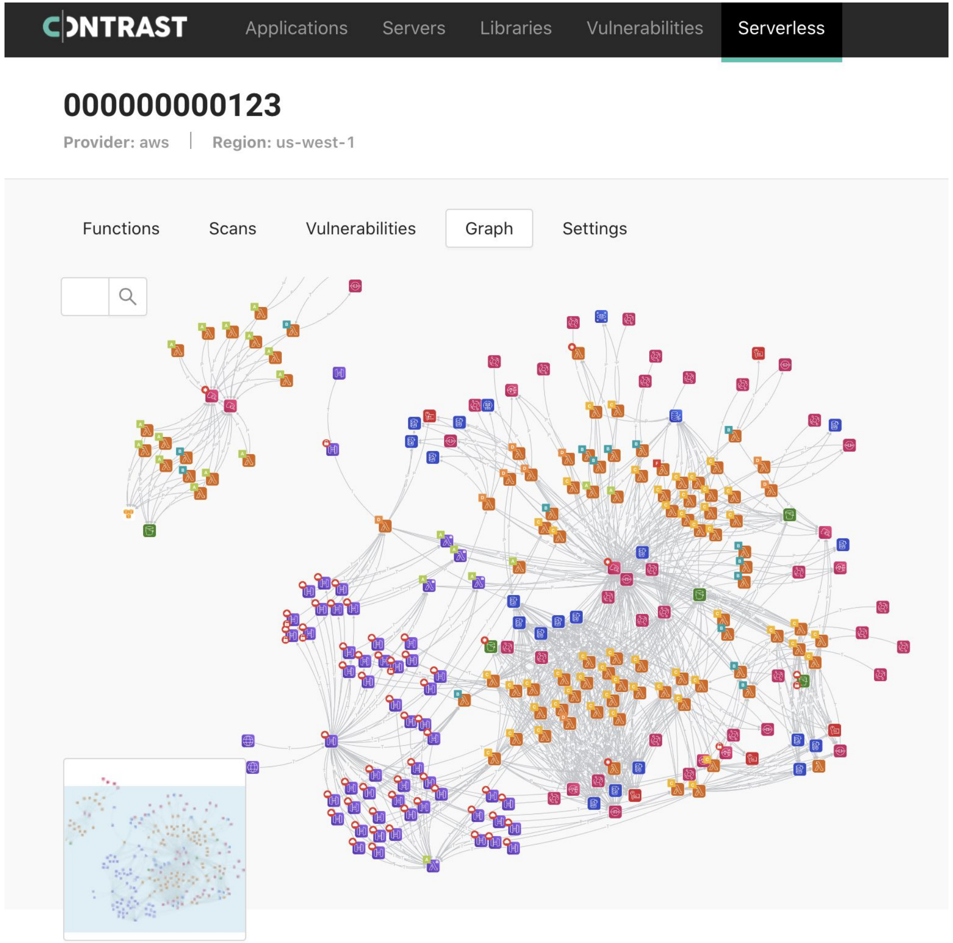
Note: This comparison contains both Search terms and Topics, which are measured differently. LEARN MORE

Interest over time

Apr 2022

Serverless computing	100
serverless security	2

@ 4ppsec

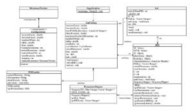


000000000123

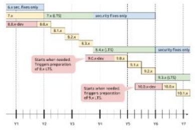
Provider: aws | Region: us-west-1

Serverless Architecture

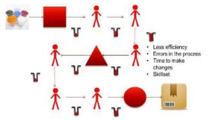
Architecture



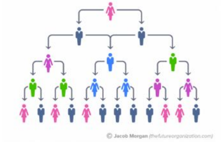
Cycles

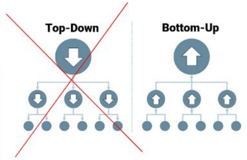


Process



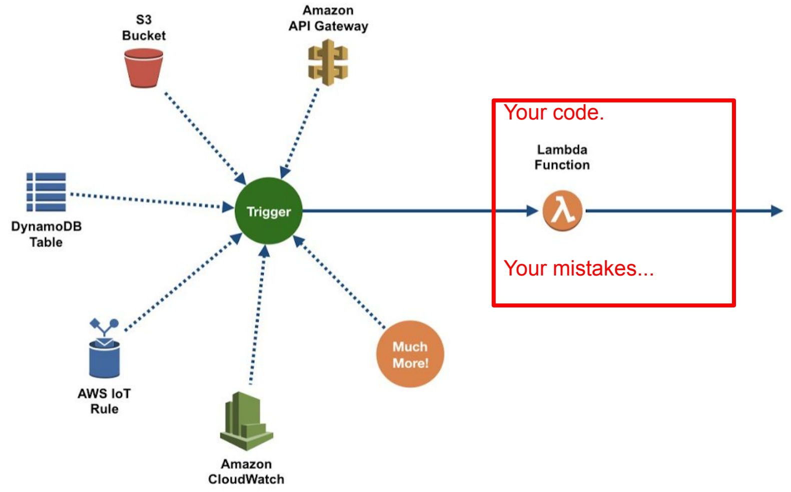
Decision

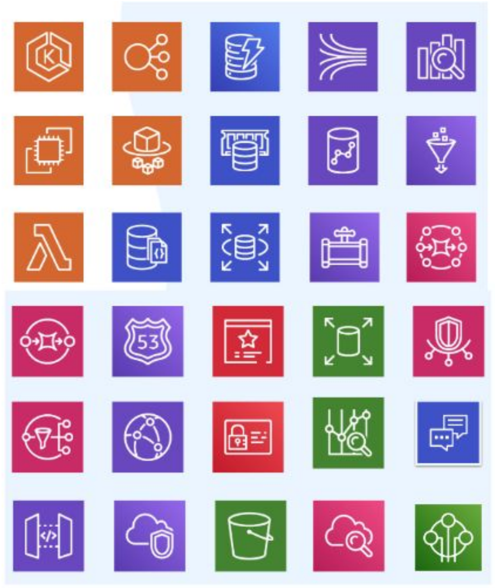




@ 4ppsec

Event-driven architecture

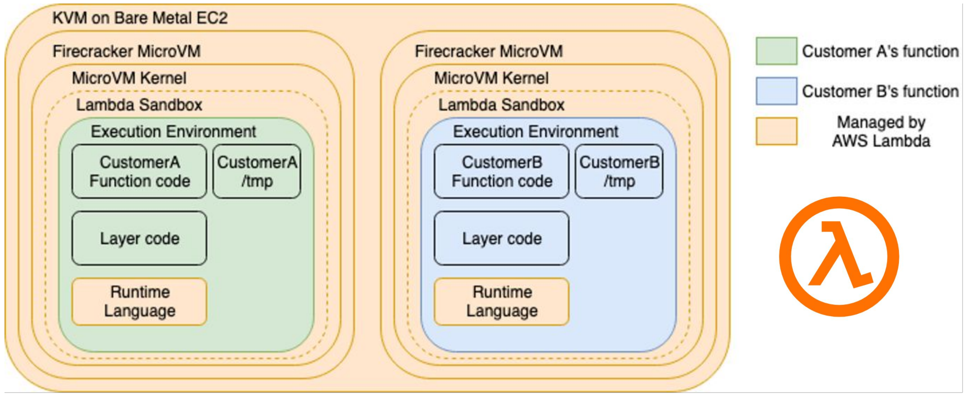




- Triggered via events
- Container spins up when required
- Terminates when code execution

AWS Lambda - Security Aspects

- Read-only environment, except for */tmp*
- Not wired to the internet*
- Data is temporary**
- Code reside in environment
- Keys are available as environment variables



OWASP Serverless Top 10

- Current project state:
  - Interpretation of Top 10
  - Open Data Call: https://appsec.it/serverless-call
- Goal: Serverless-tailored Top 10

https://owasp.org/www-project-serverless-top-10/

Event Injection

- Multiple, uncontrolled entry points
- Traditional injections (cmdi, no/sqli, etc).
- Per-language Code Injection
- New Injections (MQTT, Email, Pub/Sub)
- Impact depends on the function permissions

Injection Entry Points

REST APIs

3rd-Party Application

Cloud Storage (S3)

Authentication Services

Logs and Events

IoT

Voice (Alexa)

Email

SNS

Code pipelines

Event Injection - Best Practices

- Never trust, pass or make any assumptions regarding input and its validity from any resource
- Use positive or “allowlist” input validation when possible
    - Api Gateway allow configuring json model for requests
- Consider all event types and entry points into the system
- Run functions with the least privileges required to perform the task to reduce attack surface

Broken Authentication

- Functions are Stateless
- Multiple entry points, services, events and triggers
- No continuous flow

Broken Authentication - Best Practices

Apply the zero-trust principle to your code

- Use authentication services whenever possible
- Access tokens (e.g., JWT) can include signed custom data
- If necessary, store a “state” OOB
- Perform input validation and run with “Least Privileges”

Sensitive Data Exposure

- Same as any other cloud-based data
- Common serverless scenarios:
  - Data under /tmp
  - Sensitive data in environment variable
  - Sensitive data in an open bucket
  - Source code is also in the environment

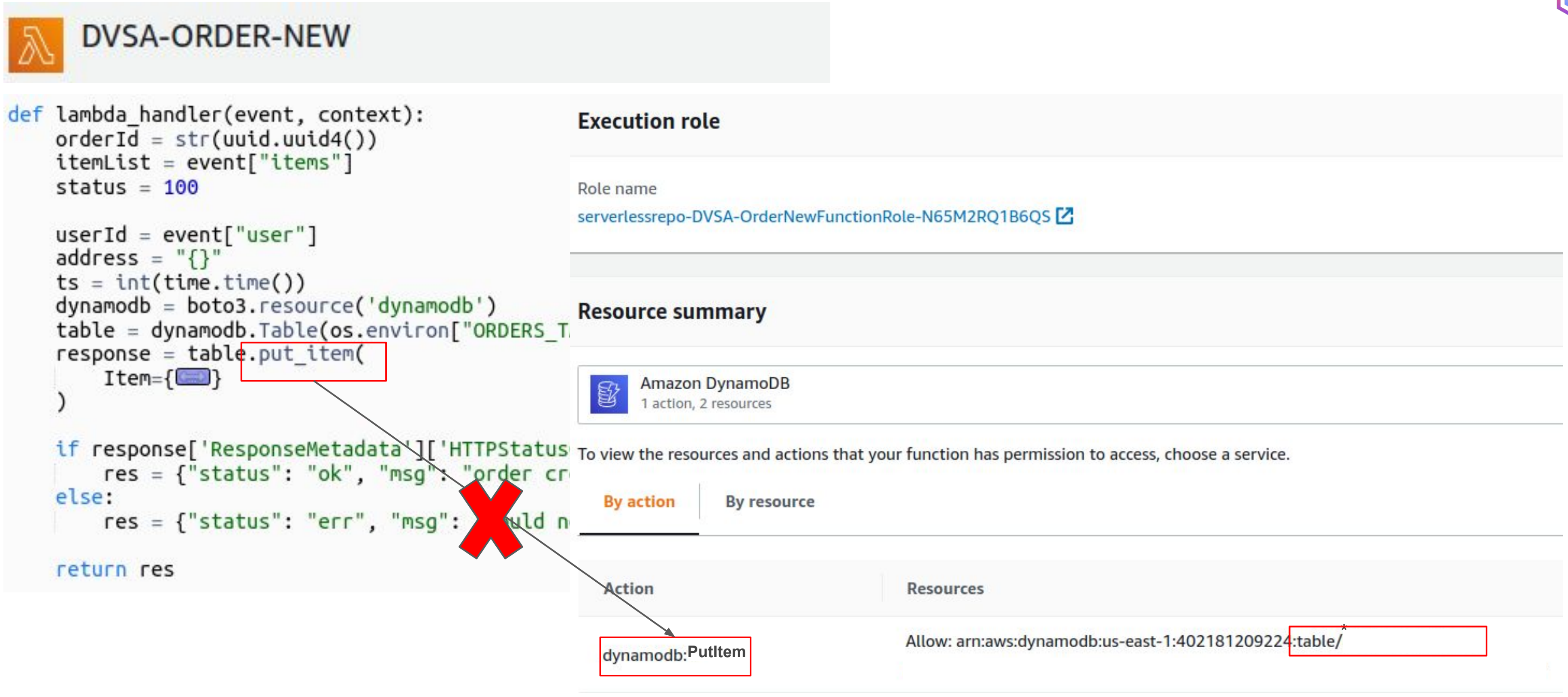
Sensitive Data Exposure - Best Practices

- Whenever possible, delete /tmp after use
- Use KMS to encrypt environment variable/sensitive data
- AWS Secret manager (or Parameter Store)
- Make sure your Buckets and other resources are set with secure configuration
- Use designated tools (e.g., AWS Macie) to identify sensitive data
- Run as “Least Privilege” to reduce access to sensitive data

Over-privileged Functions

- Over privileged functions
- More than 90% are misconfigured
- Impact of other vulnerabilities depends on the permission given to the function
    - In extreme cases - full cloud account takeover

Resource-Based IAM



DVSA-ORDER-NEW

```
def lambda_handler(event, context):
    orderId = str(uuid.uuid4())
    itemList = event["items"]
    status = 100

    userId = event["user"]
    address = "{}"
    ts = int(time.time())
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table(os.environ["ORDERS_T
    response = table.put_item(
        Item={}
    )

    if response['ResponseMetadata']['HTTPStatus
        res = {"status": "ok", "msg": "order cr
    else:
        res = {"status": "err", "msg": [illegible]uld n

    return res
```

Execution role

Role name

serverlessrepo-DVSA-OrderNewFunctionRole-N65M2RQ1B6QS

Resource summary

Amazon DynamoDB
1 action, 2 resources

To view the resources and actions that your function has permission to access, choose a service.

Action	Resources
dynamodb:PutItem	Allow: arn:aws:dynamodb:us-east-1:402181209224:table/*

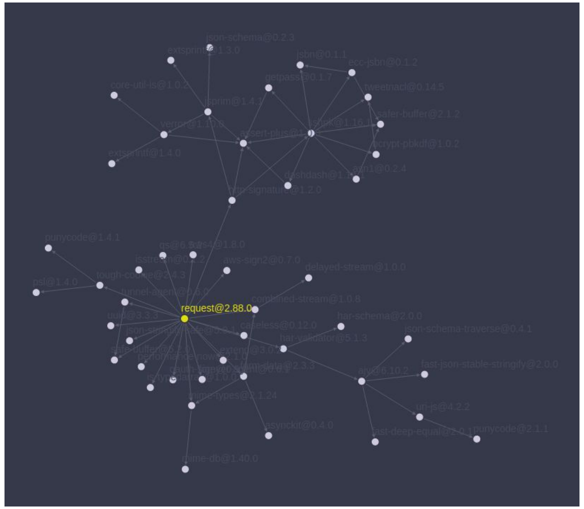
@ 4ppsec

Over-privileged Functions - Best Practices

- **Review each resource and apply least privileges**
- Automate!

Vulnerable Dependencies

- Using dependencies which are insecure
- Very common
- Functions may have 100 lines of code, but they bring everything with them



Vulnerable Dependencies - Best Practices

- Scan your dependencies before deploying into production
- Open-source, 3r-party
- Use secure versions, replace library or apply patch

Insufficient Logging & Monitoring

- More difficult than traditional web applications
- We don’t own the infrastructure - where to deploy?
- Logs exist, but we need to know how and what to extract.
- Even if we do:
  - with 1M invocations - how can we learn anything?

Sump-up

~~Event Injection~~

~~Broken Authentication~~

~~Sensitive Data Exposure~~

~~Over-Privileged Functions~~

~~Vulnerable Dependencies~~

~~Insufficient Logging & Monitoring~~

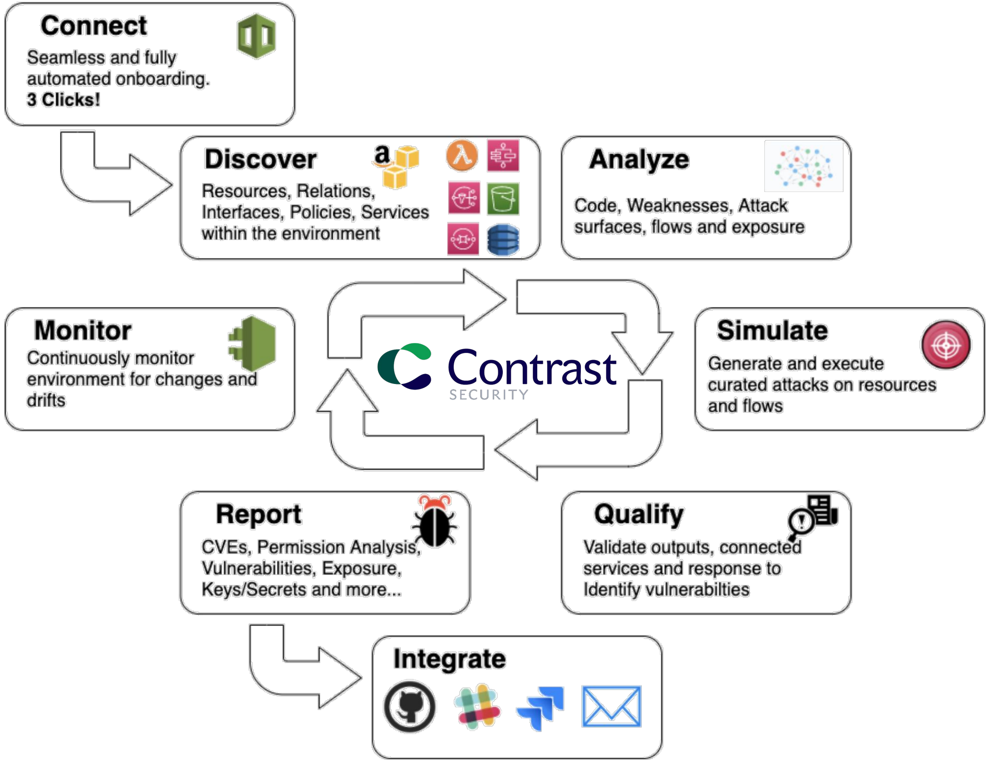
Open Resources

DoW / DoS

Insecure Shared Space

Insecure Secret Management

Contrast Serverless


Connect
Seamless and fully automated onboarding.
3 Clicks!
Discover
Resources, Relations, Interfaces, Policies, Services within the environment
Analyze
Code, Weaknesses, Attack surfaces, flows and exposure
Monitor
Continuously monitor environment for changes and drifts
Contrast SECURITY
Simulate
Generate and execute curated attacks on resources and flows
Report
CVEs, Permission Analysis, Vulnerabilities, Exposure, Keys/Secrets and more...
Qualify
Validate outputs, connected services and response to Identify vulnerabilties
Integrate

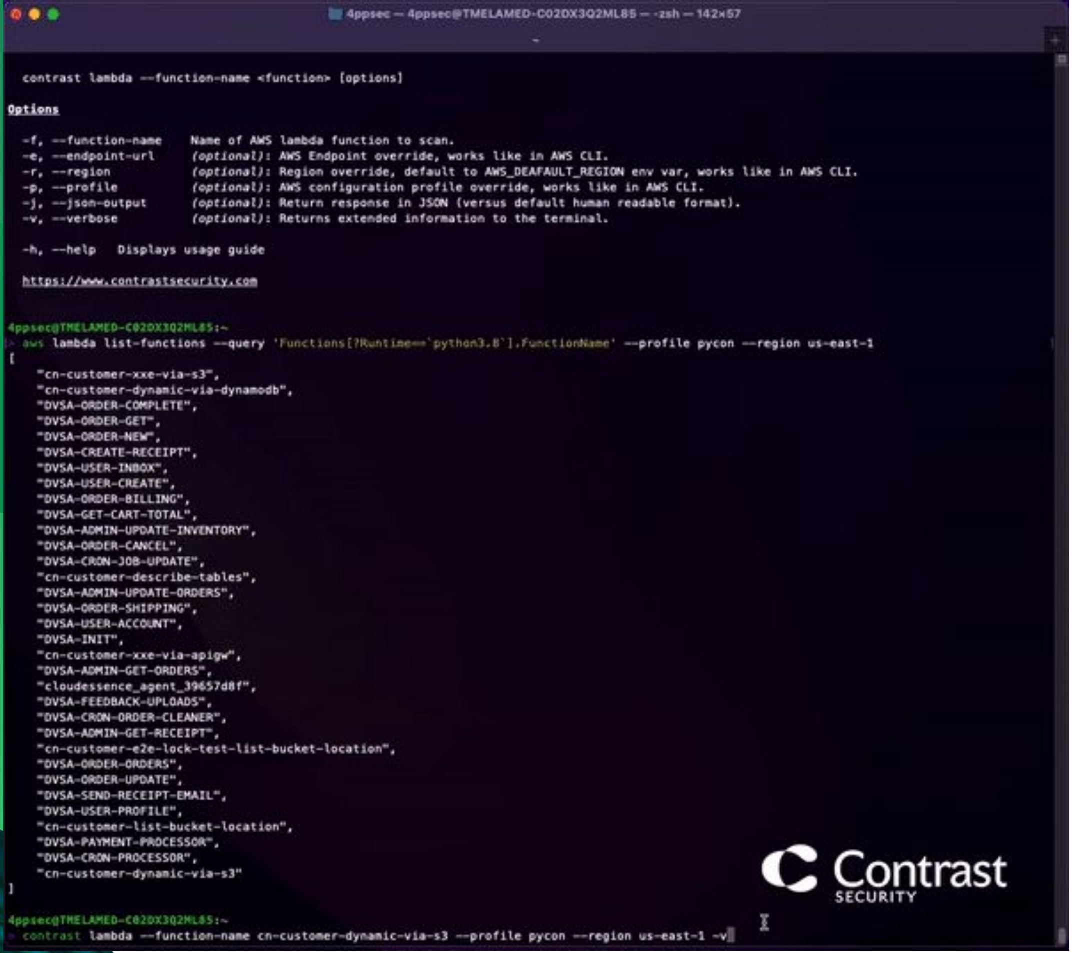
@ 4ppsec

CodeSec by Contrast

10X FASTER. 100% MORE ACCURATE, ALWAYS DEVELOPER FOCUSED.

Run scans in any application without ever leaving your hub by using CodeSec. Get results in less than 5 minutes for free.

https://www.contrastsecurity.com/developer



```
4ppsec — 4ppsec@TMELAMED-C02DX3Q2ML85 — -zsh — 142x57

  contrast lambda --function-name <function> [options]

Options

  -f, --function-name    Name of AWS lambda function to scan.
  -e, --endpoint-url     (optional): AWS Endpoint override, works like in AWS CLI.
  -r, --region           (optional): Region override, default to AWS_DEAFAULT_REGION env var, works like in AWS CLI.
  -p, --profile          (optional): AWS configuration profile override, works like in AWS CLI.
  -j, --json-output      (optional): Return response in JSON (versus default human readable format).
  -v, --verbose          (optional): Returns extended information to the terminal.

  -h, --help   Displays usage guide

  https://www.contrastsecurity.com


4ppsec@TMELAMED-C02DX3Q2ML85:~
> aws lambda list-functions --query 'Functions[?Runtime==`python3.8`].FunctionName' --profile pycon --region us-east-1
[
    "cn-customer-xxe-via-s3",
    "cn-customer-dynamic-via-dynamodb",
    "DVSA-ORDER-COMPLETE",
    "DVSA-ORDER-GET",
    "DVSA-ORDER-NEW",
    "DVSA-CREATE-RECEIPT",
    "DVSA-USER-INBOX",
    "DVSA-USER-CREATE",
    "DVSA-ORDER-BILLING",
    "DVSA-GET-CART-TOTAL",
    "DVSA-ADMIN-UPDATE-INVENTORY",
    "DVSA-ORDER-CANCEL",
    "DVSA-CRON-JOB-UPDATE",
    "cn-customer-describe-tables",
    "DVSA-ADMIN-UPDATE-ORDERS",
    "DVSA-ORDER-SHIPPING",
    "DVSA-USER-ACCOUNT",
    "DVSA-INIT",
    "cn-customer-xxe-via-apigw",
    "DVSA-ADMIN-GET-ORDERS",
    "cloudessence_agent_39657d8f",
    "DVSA-FEEDBACK-UPLOADS",
    "DVSA-CRON-ORDER-CLEANER",
    "DVSA-ADMIN-GET-RECEIPT",
    "cn-customer-e2e-lock-test-list-bucket-location",
    "DVSA-ORDER-ORDERS",
    "DVSA-ORDER-UPDATE",
    "DVSA-SEND-RECEIPT-EMAIL",
    "DVSA-USER-PROFILE",
    "cn-customer-list-bucket-location",
    "DVSA-PAYMENT-PROCESSOR",
    "DVSA-CRON-PROCESSOR",
    "cn-customer-dynamic-via-s3"
]

4ppsec@TMELAMED-C02DX3Q2ML85:~
> contrast lambda --function-name cn-customer-dynamic-via-s3 --profile pycon --region us-east-1 -v
```

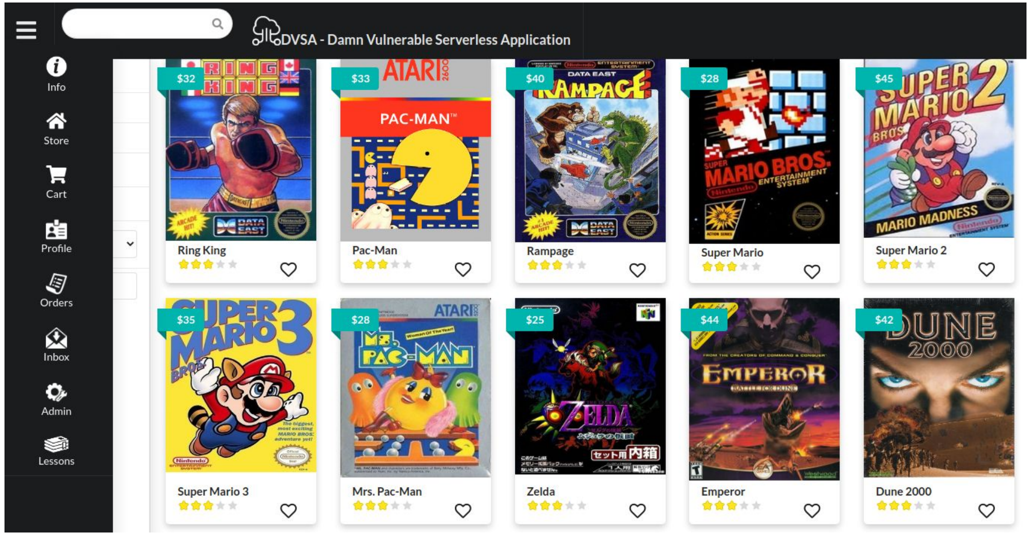
github.com/owasp/dvsa

@DVSAowasp

! NOT in PRODUCTION !

DVSA

DAMN VULNERABLE SERVERLESS APPLICATION



https://owasp.org/www-project-dvsa/

@ 4ppsec

Thank you

@ 4ppsec

Tal.Melamed@ContrastSecurity.com