CLOUDFLARE CONNECT 2026

Prevent cloud misconfigurations and data leakage before they happen

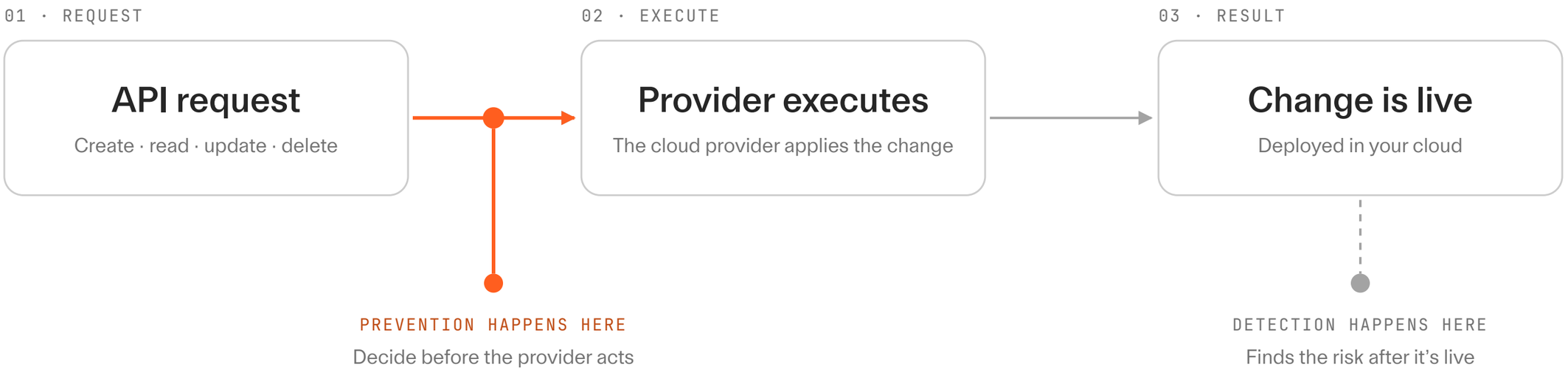
Introducing Cloud Workload Security

Neil Brown
Senior Cloudflare One GTM Specialist

Vernon Jefferson
Principal Systems Engineer

THE CONTROL POINT

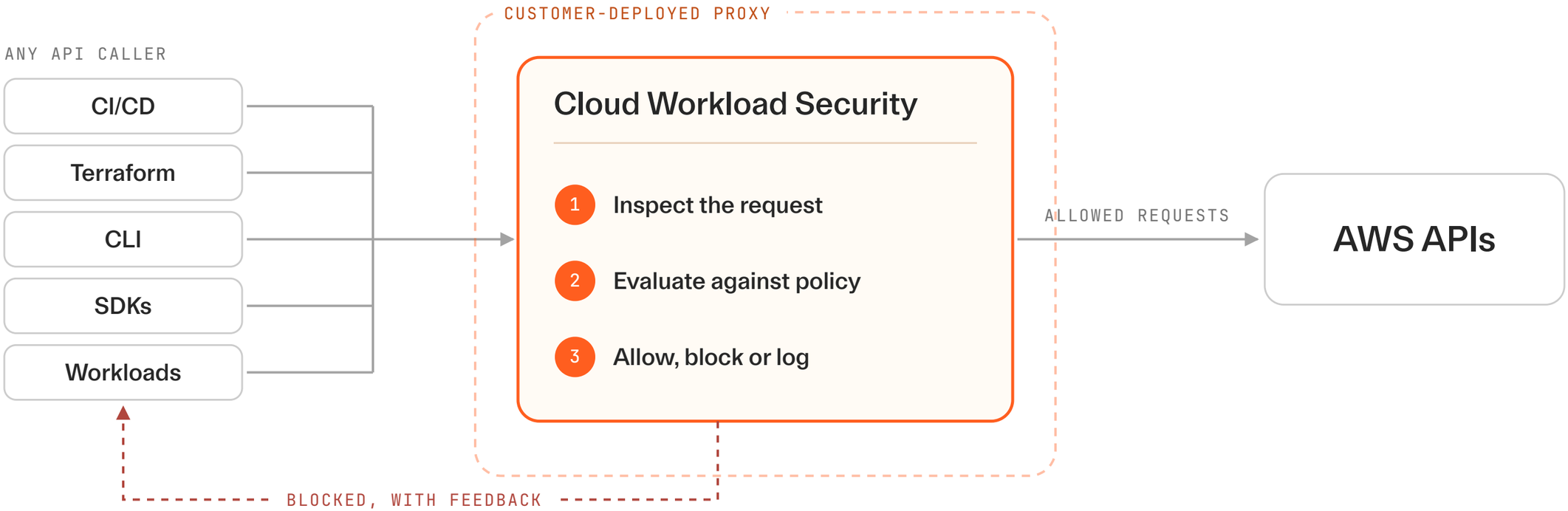
Every cloud change is an API call



Detection finds the risk after it's live. Evaluate the request first, and a risky change never goes live.

HOW IT WORKS

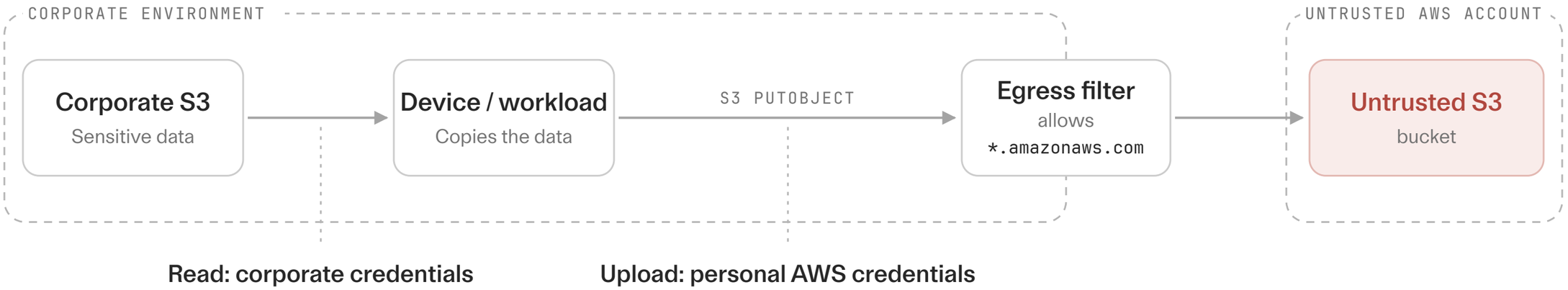
One checkpoint for cloud API requests



Policy is applied to the API request, not to a particular deployment tool.

RISK 1 · DATA LEAKAGE

Data can leave through AWS APIs your environment already trusts



The API endpoint is trusted. The destination account may not be.

LIVE DEMO 1 · DATA LEAKAGE

Stop uploads to untrusted accounts

WHAT YOU'LL SEE

1 THE POLICY

2 UPLOAD TO CORPORATE BUCKET

3 UPLOAD TO UNTRUSTED BUCKET

4 THE LOG

WHAT JUST HAPPENED

Keep corporate data inside your data perimeter

CWS enforces the destination, not just the trusted source.

CLOUD WORKLOAD SECURITY

IDENTITY

All

ACTION

s3:PutObject

POLICY

Ensure destination is within the corporate data perimeter

S3 PUTOBJECT	Corporate account	ALLOWED
S3 PUTOBJECT	Untrusted account	BLOCKED

Same AWS endpoint. Different destination account. Different decision.

RISK 2 · MISCONFIGURATION

Different tools. Same risky API call.

EXAMPLE · SSH OPEN TO THE INTERNET

TERRAFORM

```
resource "aws_vpc_security_group_ingress_rule" "ssh" {
  security_group_id = aws_security_group.app.id
  ip_protocol       = "tcp"
  from_port         = 22
  to_port           = 22
  cidr_ipv4         = "0.0.0.0/0"
}
```

PYTHON SDK (BOTO3)

```
boto3.client("ec2").authorize_security_group_ingress(
    GroupId="sg-0abc1234",
    IpPermissions=[{
        "IpProtocol": "tcp",
        "FromPort": 22, "ToPort": 22,
        "IpRanges": [{"CidrIp": "0.0.0.0/0"}],
    }],
)
```

SAME API REQUEST

`ec2:AuthorizeSecurityGroupIngress`
`TCP 22 · CidrIp=0.0.0.0/0`

IAM decides **who** can change **which** groups.

It can't see the **port or source** in the rule.

CWS checks the whole request.

LIVE DEMO 2 · MISCONFIGURATION

Block SSH open to the internet

WHAT YOU'LL SEE

1 THE POLICY

2 CLI · OPEN TO THE INTERNET

3 CLI · INTERNAL RANGE

4 TERRAFORM

5 THE LOG

WHAT JUST HAPPENED

One policy, whatever the tool

CLOUD WORKLOAD SECURITY

IDENTITY

All

ACTION

ec2:AuthorizeSecurityGroupIngress

POLICY

source ≠ 0.0.0.0/0

AWS CLI	22 ← 0.0.0.0/0	BLOCKED
AWS CLI	22 ← 10.0.0.0/8	ALLOWED
TERRAFORM	22 ← 0.0.0.0/0	BLOCKED

Same pattern: inspect the request, evaluate it against policy, decide before AWS acts.

WHY IT WORKS

What made both blocks possible

1

Identifies the caller

Scope any policy to who's calling

2

Validates the whole request

Every parameter, not just the action

3

Guides the developer

Explains why the policy blocked the request

The risk is in the API request. CWS stops it before AWS acts.

AVAILABLE IN BETA

Get started

Quick wins for your security posture

01 **Stop uploads to untrusted accounts**

02 **Block critical misconfigurations**

03 **Turn a recurring finding into a prevention**

developers.cloudflare.com/cloudflare-one/cloud-workload-security

CONTACT nbrown@cloudflare.com

Your feedback is important

Please complete the session survey in the Cloudflare Events app: tap the bell, then “Survey”.