

Connect AWS with the instance role

The recommended way when the appliance runs in your AWS account, with the read policy and what each check reads.

Version 0.1.0

Generated 2026-10-16

Draft: steps are specified but not final.

Vesta Cyber Solutions. Made from the same source as the guide on the web, so the two always match.

Contents

How it works	3
Steps	3
The read policy	4
What each check reads	6
If it does not connect	6
What is stored	7

Draft, pending the appliance build. This way to connect is built, and was proven against a live AWS account on the Vesta test server on 2026-10-12. The steps may change before release.

Use this way when the appliance runs inside your own AWS account. It is the recommended way: no keys are stored, and the appliance uses the temporary credentials AWS gives the instance.

The appliance reads your account and never changes it. Every permission below is a read action, except `iam:GenerateCredentialReport`, which asks AWS to prepare the credential report that two of the checks read.

How it works

1. The appliance asks the instance metadata service for credentials. It uses IMDSv2 only.
2. If you give a role ARN, the appliance assumes that role with those credentials. This lets one appliance read another account. The role's trust policy names the instance role and requires an external ID the product makes for that one integration, so no other workspace on the same server can assume it.
3. Without a role ARN, the appliance reads only the account the instance runs in.

When you open Add AWS integration, the product offers the instance role first if the instance metadata service answers within one second.

Steps

1. In IAM, create a role for EC2 and attach it to the appliance instance as its instance profile.
2. Make sure the instance requires IMDSv2. The appliance runs the product in a container, so set the instance's metadata hop limit to 2 (`aws ec2 modify-instance-metadata-options --http-tokens required --http-put-response-hop-limit 2`); with the default of 1, the container cannot reach the metadata service.
3. Attach the read policy below to the instance role. If you will use a separate role in another account, attach the policy to that role instead.
4. If you use a separate role, add a trust policy to it that names the instance role as the principal allowed to assume it, with the integration's external ID. The product shows the exact trust policy, with the external ID, on the setup page after you add the account, and again under Policies on the account's card while it is not connected.
5. Allow the appliance to reach the instance metadata service. It stays closed by default, and opens only on EC2 and only after this switch exists. Sign in over SSH as `spadmin` and run `sudo install -o root -g root -m 0644 /dev/null /etc/securitposture/egress-imds`. Removing the file closes it again within a minute. Any process in the product's container can then obtain the instance role's credentials, so give the instance role the read only policy below and nothing more.
6. In SecurITPosture, open Integrations, then AWS, then Add AWS integration.
7. Choose This server's AWS role. Enter the account name, the account ID, the role ARN if you use one, the regions, and how often to scan. Leaving the role ARN empty reads the account the appliance runs in; that is offered only when the appliance holds one customer workspace.

8. Choose Add and validate. The product validates the connection before it marks the account connected.

The read policy

This is the minimum the product needs to validate the account and discover its instances:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeRegions",
        "ec2:DescribeInstances"
      ],
      "Resource": "*"
    }
  ]
}
```

The product adds what the evidence checks and the system's active Key Security Indicator checks read. With every AWS Key Security Indicator check active, the policy is:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeRegions",
        "ec2:DescribeInstances",
        "access-analyzer:ListAnalyzers",
        "access-analyzer:ListFindings",
        "acm:ListCertificates",
        "backup:ListBackupPlans",
        "backup:ListBackupVaults",
        "backup:ListRecoveryPointsByBackupVault",
        "cloudtrail:DescribeTrails",
        "cloudtrail:GetTrailStatus",
        "config:DescribeComplianceByConfigRule",
        "config:DescribeConfigurationRecorderStatus",
        "ec2:DescribeFlowLogs",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs",
        "ec2:GetEbsEncryptionByDefault",
        "elasticloadbalancing:DescribeListeners",
        "elasticloadbalancing:DescribeLoadBalancers",
        "guardduty:GetDetector",
        "guardduty:ListDetectors",
        "iam:GenerateCredentialReport",
        "iam:GetAccountSummary",
        "iam:GetCredentialReport",
        "kms:DescribeKey",
        "kms:GetKeyRotationStatus",
        "kms:ListKeys",
        "rds:DescribeDBInstances",
        "s3:GetAccountPublicAccessBlock",
        "secretsmanager:ListSecrets",
        "securityhub:DescribeHub"
      ],
      "Resource": "*"
    }
  ]
}
```

```
} ]
```

Always copy the policy from the product's setup page, because it lists exactly the checks active on your system.

What each check reads

- CloudTrail logs every region with log file validation: `cloudtrail:DescribeTrails`, `cloudtrail:GetTrailStatus`
- AWS Config records configuration changes: `config:DescribeConfigurationRecorderStatus`
- AWS Config rules are compliant: `config:DescribeComplianceByConfigRule`
- The root user has MFA and no access keys: `iam:GetAccountSummary`
- Every console user has MFA: `iam:GenerateCredentialReport`, `iam:GetCredentialReport`
- No access keys or passwords unused for 90 days: `iam:GenerateCredentialReport`, `iam:GetCredentialReport`
- IAM Access Analyzer is on with no active external access findings: `access-analyzer:ListAnalyzers`, `access-analyzer:ListFindings`
- No security group opens SSH or RDP to the internet: `ec2:DescribeSecurityGroups`
- Default security groups allow nothing: `ec2:DescribeSecurityGroups`
- EBS encryption by default is on: `ec2:GetEbsEncryptionByDefault`
- S3 public access is blocked for the account: `s3:GetAccountPublicAccessBlock`
- Customer managed KMS keys rotate: `kms:ListKeys`, `kms:DescribeKey`, `kms:GetKeyRotationStatus`
- Secrets Manager secrets rotate: `secretsmanager:ListSecrets`
- GuardDuty is enabled: `guardduty:ListDetectors`, `guardduty:GetDetector`
- Security Hub is enabled: `securityhub:DescribeHub`
- Backup plans exist and recovery points are recent: `backup:ListBackupPlans`, `backup:ListBackupVaults`, `backup:ListRecoveryPointsByBackupVault`
- RDS automated backups are on with retention set: `rds:DescribeDBInstances`
- Load balancer listeners use TLS: `elasticloadbalancing:DescribeLoadBalancers`, `elasticloadbalancing:DescribeListeners`
- Certificates in use are not expiring within 30 days: `acm:ListCertificates`
- Every VPC has flow logs: `ec2:DescribeFlowLogs`, `ec2:DescribeVpcs`
- Discovered instances match the inventory workbook: `ec2:DescribeInstances`, `ec2:DescribeRegions`

If it does not connect

- "This server has no AWS instance role. Choose another way to connect": the instance metadata service did not answer, or the instance allows only IMDSv1. Attach an instance role and require IMDSv2, or use IAM Roles Anywhere.
- A message naming the instance role policy: the role is missing a permission. Compare it with the policy on the setup page.
- With a role ARN, the account ID you entered must match the role's account, and the role's trust policy must carry the integration's external ID.
- "Reconnect this account with one of the three ways to connect": the account was added before the three ways existed. Choose Reconnect on its card; see Change how an account connects.

What is stored

Nothing secret for the server's own account. With a role to assume, the integration's external ID is stored encrypted. The product stores the account ID, the role ARN if any, the regions, and the scan frequency. It never stores Vesta credentials, and there is no Vesta identity in your account. Error logs hold AWS error codes and request IDs only.

See also [IAM Roles Anywhere](#), [Access key](#), and [Change how an account connects](#).