

We found potential security vulnerabilities in your dependencies.

You can see this message because you have been granted [access to Dependabot alerts for this repository](#).

See Dependabot alerts

master ▾

...

devops / terraform / environments / dsva-vagov-dev / eks.tf

mleclerc00 Test redirecting traffic to eks vets-api instance based on http head... ... History

9 contributors

564 lines (507 sloc) | 16.8 KB

...

```
1  locals {
2    eks_cluster_name = "${var.base_tags.group}-${var.base_tags.project}-${var.base_tags["environment
3  }
4
5  module "dev_eks_cluster" {
6    source           = "git::https://github.com/cloudposse/terraform-aws-eks-cluster.git?re
7    region           = "us-gov-west-1"
8    vpc_id           = aws_vpc.vpc.id
9    subnet_ids       = [aws_subnet.subnet_1a.id, aws_subnet.subnet_1b.id, aws_subnet.subnet
10   kubernetes_version = "1.19"
11   name               = local.eks_cluster_name
12   oidc_provider_enabled = true
13   enabled_cluster_log_types = ["api", "audit", "authenticator", "controllerManager", "scheduler"]
14   endpoint_private_access = true
15   endpoint_public_access  = true
16   # Temporarily disabled until a solution for changes via tf outside Ops team is determined - stev
17   apply_config_map_aws_auth = false
18   map_additional_iam_users = [
19     {
20       userarn = "arn:aws-us-gov:iam::008577686731:user/Jeremy.Britt",
21       username = "Jeremy.Britt"
```

```
22     groups    = ["system:masters"]
23 },
24 {
25     userarn    = "arn:aws-us-gov:iam::008577686731:user/Demian.Ginther",
26     username   = "Demian.Ginther"
27     groups    = ["system:masters"]
28 },
29 {
30     userarn    = "arn:aws-us-gov:iam::008577686731:user/Srikanth.Valluru",
31     username   = "Srikanth.Valluru"
32     groups    = ["system:masters"]
33 },
34 {
35     userarn    = "arn:aws-us-gov:iam::008577686731:user/Matt.Leclerc",
36     username   = "Matt.Leclerc"
37     groups    = ["system:masters"]
38 },
39 {
40     userarn    = "arn:aws-us-gov:iam::008577686731:user/Eric.Oliver",
41     username   = "Eric.Oliver"
42     groups    = ["system:masters"]
43 },
44 {
45     userarn    = "arn:aws-us-gov:iam::008577686731:user/Elijah.Lynn",
46     username   = "Elijah.Lynn"
47     groups    = ["system:masters"]
48 },
49 {
50     userarn    = "arn:aws-us-gov:iam::008577686731:user/Nathan.Douglas",
51     username   = "Nathan.Douglas"
52     groups    = ["system:masters"]
53 },
54 {
55     userarn    = "arn:aws-us-gov:iam::008577686731:user/Cameron.Eagans",
56     username   = "Cameron.Eagans"
57     groups    = ["system:masters"]
58 },
59 {
60     userarn    = "arn:aws-us-gov:iam::008577686731:user/Neil.Hastings",
61     username   = "Neil.Hastings"
62     groups    = ["system:masters"]
63 },
64 {
65     userarn    = "arn:aws-us-gov:iam::008577686731:user/John.Stevens-Garmon",
66     username   = "John.Stevens-Garmon"
67     groups    = ["system:masters"]
68 },
69 {
70     userarn    = "arn:aws-us-gov:iam::008577686731:user/Ryan.Beckwith",
```

```

71     username = "Ryan.Beckwith"
72     groups   = ["system:masters"]
73 },
74 {
75     userarn = "arn:aws-us-gov:iam::008577686731:user/Oseas.Moran",
76     username = "Oseas.Moran"
77     groups   = ["system:masters"]
78 },
79 {
80     userarn = "arn:aws-us-gov:iam::008577686731:user/Rebecca.Tolmach",
81     username = "Rebecca.Tolmach"
82     groups   = ["system:masters"]
83 },
84 {
85     userarn = "arn:aws-us-gov:iam::008577686731:user/service_account/dsva-eks-service-account",
86     username = "dsva-eks-service-account"
87     groups   = ["system:masters"]
88 },
89 ]
90 map_additional_iam_roles = [
91     {
92         rolearn = "arn:aws-us-gov:iam::008577686731:role/dsva-vagov-utility-cluster-argocd"
93         username = "argocd-application-controller"
94         groups   = []
95     },
96     {
97         rolearn = "arn:aws-us-gov:iam::008577686731:role/dsva-vagov-utility-cluster-argocd-server"
98         username = "argocd-server"
99         groups   = []
100     },
101 ]
102 }
103
104 module "eks_node_asg_group" {
105     source          = "github.com/department-of-veterans-affairs/terraform-aws-vsp
106     region          = "us-gov-west-1"
107     vpc_id          = aws_vpc.vpc.id
108     desired_capacity = 5
109     min_size        = 3
110     max_size        = 20
111     eks_cluster_security_group_id = module.dev_eks_cluster.security_group_id
112     cluster_name      = module.dev_eks_cluster.eks_cluster_id
113     eks_alb_ingress_security_group_id = module.internal-tools-alb.alb-sg-id
114     name              = "dsva-vagov-dev-eks-node-group-main"
115     subnet_ids        = [aws_subnet.subnet_1a.id, aws_subnet.subnet_1b.id, aws_subne
116     instance_types     = ["c5.4xlarge", "c5a.4xlarge", "c5n.4xlarge", "c5d.4xlarge"]
117     ## After initial creation AMI ID is controlled by GHA https://github.com/department-of-veterans-
118     ami_id             = "ami-0ce45355d016dc86b"
119     bootstrap_args     = "--use-max-pods false"

```

```

120     target_group_arns = [
121         aws_lb_target_group.traefik_eks_dev.arn,
122         aws_lb_target_group.traefik_eks_dev_https.arn,
123     ]
124 }
125
126 resource "aws_security_group_rule" "revproxy-to-eks-worker" {
127     type           = "ingress"
128     from_port      = 0
129     to_port        = 0
130     protocol       = "-1"
131     security_group_id = module.eks_node_asg_group.asg_sg_id
132     source_security_group_id = module.revproxy.revproxy-alb-security-group
133     description     = "revproxy ALB"
134 }
135
136 ## External Secrets Service Role
137 data "aws_iam_policy_document" "external_secrets_controller_dev" {
138     statement {
139         actions = ["sts:AssumeRoleWithWebIdentity"]
140         effect   = "Allow"
141         condition {
142             test      = "StringEquals"
143             variable = "${replace(module.dev_eks_cluster.eks_cluster_identity_oidc_issuer, "https://", "
144             values = [
145                 "system:serviceaccount:external-secrets:external-secrets-controller",
146             ] // IMPORTANT! This must match the serviceaccount in k8s 'system:serviceaccount:<namespace>
147         }
148         principals {
149             identifiers = [module.dev_eks_cluster.eks_cluster_identity_oidc_issuer_arn]
150             type        = "Federated"
151         }
152     }
153 }
154
155 data "aws_iam_policy_document" "external_secrets_controller_policy_dev" {
156     statement {
157         effect = "Allow"
158         actions = [
159             "ssm:DescribeParameters",
160             "ssm:GetParameter",
161             "ssm:GetParametersByPath"
162         ]
163         resources = ["*"]
164     }
165 }
166
167 resource "aws_iam_role" "external_secrets_controller_dev" {
168     assume_role_policy = data.aws_iam_policy_document.external_secrets_controller_dev.json

```

```
169     name                = "${module.dev_eks_cluster.eks_cluster_id}-ext-secrets-controller-dev"
170   }
171
172   resource "aws_iam_role_policy" "external_secrets_controller_dev" {
173     name     = "${module.dev_eks_cluster.eks_cluster_id}-External-Secrets-Controller-Policy-dev"
174     role     = aws_iam_role.external_secrets_controller_dev.id
175     policy   = data.aws_iam_policy_document.external_secrets_controller_policy_dev.json
176   }
177
178   ## Traefik target group and listener rules
179   resource "aws_lb_target_group" "traefik_eks_dev" {
180     name        = "dsva-vagov-dev-traefik-tg"
181     port        = 32112
182     protocol    = "HTTP"
183     vpc_id      = aws_vpc.vpc.id
184     target_type = "instance"
185
186     deregistration_delay = 10
187
188     health_check {
189       path      = "/ping"
190       port      = 32112
191       protocol  = "HTTP"
192     }
193   }
194
195   resource "aws_lb_target_group" "traefik_eks_dev_https" {
196     # a temporary target group used for testing revproxy in EKS
197     # since a target group can't be assigned to multiple listeners
198     name        = "dsva-vagov-dev-https-traefik-tg"
199     port        = 32494
200     protocol    = "HTTPS"
201     vpc_id      = aws_vpc.vpc.id
202     target_type = "instance"
203
204     deregistration_delay = 10
205
206     health_check {
207       path      = "/ping"
208       port      = 32494
209       protocol  = "HTTPS"
210     }
211   }
212
213   ### VETS-API ALB to EKS ###
214   module "vets-api-elb-logs-bucket" {
215     source      = "github.com/department-of-veterans-affairs/terraform-aws-vsp-s3-bucket"
216     name        = "dsva-vagov-dev-vets-api-alb-logs"
217     environment = "dev"
```

```
218     access_logs_policy = true
219 }
220
221 resource "aws_security_group_rule" "alb-to-worker" {
222     type                = "ingress"
223     description         = "ALB to EKS worker" # allow new ALB to hit the EKS worker
224     from_port           = 0
225     to_port             = 0
226     protocol            = "-1"
227     source_security_group_id = module.vets-api.elb-security-group
228     security_group_id      = module.eks_node_asg_group.asg_sg_id
229 }
230
231 #####
232 ## Temporary rule to test routing traffic based on http header to eks vets-api
233 resource "aws_lb_listener_rule" "vets_api_eks" {
234     listener_arn = module.internal-tools-alb.listener-http-arn
235     priority     = 49001
236
237     action {
238         type          = "forward"
239         target_group_arn = aws_lb_target_group.traefik_eks_dev.arn
240     }
241
242     condition {
243         host_header {
244             values = ["dev-api.va.gov"]
245         }
246     }
247 }
248
249 resource "aws_lb_listener_rule" "traefik_http" {
250     listener_arn = module.internal-tools-alb.listener-http-arn
251     priority     = 49000
252
253     action {
254         type          = "forward"
255         target_group_arn = aws_lb_target_group.traefik_eks_dev.arn
256     }
257
258     condition {
259         host_header {
260             values = ["*.vfs.va.gov"]
261         }
262     }
263 }
264
265 resource "aws_lb_listener_rule" "traefik_https" {
266     listener_arn = module.internal-tools-alb.listener-https-arn
```

```
267     priority      = 50000
268
269     action {
270         type      = "forward"
271         target_group_arn = aws_lb_target_group.traefik_eks_dev_https.arn
272     }
273
274     condition {
275         host_header {
276             values = ["*.vfs.va.gov"]
277         }
278     }
279 }
280
281 ## External DNS Service Role
282 data "aws_iam_policy_document" "external_dns_controller_dev" {
283     statement {
284         actions = ["sts:AssumeRoleWithWebIdentity"]
285         effect  = "Allow"
286         condition {
287             test      = "StringEquals"
288             variable = "${replace(module.dev_eks_cluster.eks_cluster_identity_oidc_issuer, "https://", "
289             values = [
290                 "system:serviceaccount:external-dns:external-dns",
291             ] // IMPORTANT! This must match the serviceaccount in k8s 'system:serviceaccount:<namespace>
292         }
293         principals {
294             identifiers = [module.dev_eks_cluster.eks_cluster_identity_oidc_issuer_arn]
295             type        = "Federated"
296         }
297     }
298 }
299
300 data "aws_iam_policy_document" "external_dns_controller_policy_dev" {
301     statement {
302         effect = "Allow"
303         actions = [
304             "route53:ListHostedZones",
305             "route53:ListResourceRecordSets"
306         ]
307         resources = ["*"]
308     }
309
310     statement {
311         effect = "Allow"
312         actions = [
313             "route53:ChangeResourceRecordSets"
314         ]
315         resources = ["arn:aws-us-gov:route53:::hostedzone/Z10020332CIPNJYY8UL70"]
316     }
317 }
```

```
316     }
317   }
318
319   resource "aws_iam_role" "external_dns_controller_dev" {
320     assume_role_policy = data.aws_iam_policy_document.external_dns_controller_dev.json
321     name               = "${module.dev_eks_cluster.eks_cluster_id}-ext-dns-controller-dev"
322   }
323
324   resource "aws_iam_role_policy" "external_dns_controller_dev" {
325     name = "${module.dev_eks_cluster.eks_cluster_id}-External-DNS-Controller-Policy-dev"
326     role = aws_iam_role.external_dns_controller_dev.id
327     policy = data.aws_iam_policy_document.external_dns_controller_policy_dev.json
328   }
329
330   ## Loki Service Role
331   data "aws_iam_policy_document" "loki" {
332     statement {
333       actions = ["sts:AssumeRoleWithWebIdentity"]
334       effect  = "Allow"
335
336       condition {
337         test      = "StringEquals"
338         variable = "${replace(module.dev_eks_cluster.eks_cluster_identity_oidc_issuer, "https://", "
339         values = [
340           "system:serviceaccount:observability:loki",
341         ] // IMPORTANT! This must match the serviceaccount in k8s 'system:serviceaccount:<namespace>
342       }
343
344       principals {
345         identifiers = [module.dev_eks_cluster.eks_cluster_identity_oidc_issuer_arn]
346         type        = "Federated"
347       }
348     }
349   }
350
351   data "aws_iam_policy_document" "loki_policy" {
352     statement {
353       sid = "LokiAccessToS3"
354       effect = "Allow"
355       actions = [
356         "s3:PutObject",
357         "s3:GetObject",
358         "s3:ListBucket",
359         "s3:DeleteObject"
360       ]
361       resources = [
362         aws_s3_bucket.loki-dev-s3.arn,
363         "${aws_s3_bucket.loki-dev-s3.arn}/*"
364       ]
365     }
366   }
```



```
365     }
366   }
367
368   resource "aws_iam_role" "loki_role" {
369     assume_role_policy = data.aws_iam_policy_document.loki.json
370     name                = "${module.dev_eks_cluster.eks_cluster_id}-loki"
371   }
372
373   resource "aws_iam_role_policy" "loki_policy" {
374     name   = "${module.dev_eks_cluster.eks_cluster_id}-Loki-Policy"
375     role   = aws_iam_role.loki_role.id
376     policy = data.aws_iam_policy_document.loki_policy.json
377   }
378
379   ## Vets-api Service Role
380   data "aws_iam_policy_document" "vets_api_dev" {
381     statement {
382       actions = ["sts:AssumeRoleWithWebIdentity"]
383       effect  = "Allow"
384       condition {
385         test      = "StringEquals"
386         variable  = "${replace(module.dev_eks_cluster.eks_cluster_identity_oidc_issuer, "https://", "
387         values = [
388           "system:serviceaccount:vets-api:vets-api",
389         ] // IMPORTANT! This must match the serviceaccount in k8s 'system:serviceaccount:<namespace>
390       }
391       principals {
392         identifiers = [module.dev_eks_cluster.eks_cluster_identity_oidc_issuer_arn]
393         type        = "Federated"
394       }
395     }
396   }
397
398   data "aws_iam_policy_document" "vets_api_policy_dev" {
399     statement {
400       effect = "Allow"
401       actions = [
402         "s3:ListAllMyBuckets",
403         "s3:GetBucketLocation"
404       ]
405       resources = ["*"]
406     }
407
408     statement {
409       effect = "Allow"
410       actions = [
411         "s3:ListBucket"
412       ]
413       resources = ["arn:aws-us-gov:s3:::dsva-vagov-dev-config-bucket"]
414     }
415   }
```

```
414     }
415
416     statement {
417         effect = "Allow"
418         actions = [
419             "s3:GetObject"
420         ]
421         resources = ["arn:aws-us-gov:s3:::dsva-vagov-dev-config-bucket/vets-api-server/*"]
422     }
423
424     statement {
425         effect = "Allow"
426         actions = [
427             "ecr:StartImageScan",
428             "ecr:DescribeImageReplicationStatus",
429             "ecr:ListTagsForResource",
430             "ecr:UploadLayerPart",
431             "ecr:ListImages",
432             "ecr:CompleteLayerUpload",
433             "ecr:TagResource",
434             "ecr:DescribeRepositories",
435             "ecr:BatchCheckLayerAvailability",
436             "ecr:ReplicateImage",
437             "ecr:GetLifecyclePolicy",
438             "ecr:DescribeImageScanFindings",
439             "ecr:GetLifecyclePolicyPreview",
440             "ecr:PutImageScanningConfiguration",
441             "ecr:GetDownloadUrlForLayer",
442             "ecr:PutImage",
443             "ecr:BatchGetImage",
444             "ecr:DescribeImages",
445             "ecr:InitiateLayerUpload",
446             "ecr:GetRepositoryPolicy",
447         ]
448         resources = ["arn:aws-us-gov:ecr:us-gov-west-1:008577686731:repository/dsva/vets-api*"]
449     }
450 }
451
452 resource "aws_iam_role" "vets_api_dev" {
453     assume_role_policy = data.aws_iam_policy_document.vets_api_dev.json
454     name                = "${module.dev_eks_cluster.eks_cluster_id}-vets-api"
455 }
456
457 resource "aws_iam_role_policy" "vets_api_dev" {
458     name     = "${module.dev_eks_cluster.eks_cluster_id}-vets-api-policy"
459     role     = aws_iam_role.vets_api_dev.id
460     policy   = data.aws_iam_policy_document.vets_api_policy_dev.json
461 }
462
```

```
463 ## Grafana Service Role
464 data "aws_iam_policy_document" "grafana" {
465   statement {
466     actions = ["sts:AssumeRoleWithWebIdentity"]
467     effect  = "Allow"
468
469     condition {
470       test      = "StringEquals"
471       variable = "${replace(module.dev_eks_cluster.eks_cluster_identity_oidc_issuer, "https://", "
472       values = [
473         "system:serviceaccount:observability:grafana",
474       ] // IMPORTANT! This must match the serviceaccount in k8s 'system:serviceaccount:<namespace>
475     }
476
477     principals {
478       identifiers = [module.dev_eks_cluster.eks_cluster_identity_oidc_issuer_arn]
479       type        = "Federated"
480     }
481   }
482 }
483
484 data "aws_iam_policy_document" "grafana_policy" {
485   statement {
486     sid      = "AllowCloudWatchLogging"
487     effect   = "Allow"
488     resources = ["arn:aws-us-gov:logs:*:*:*"]
489
490     actions = [
491       "logs:CreateLogGroup",
492       "logs:CreateLogStream",
493       "logs:PutLogEvents",
494       "logs:DescribeLogStreams",
495     ]
496   }
497
498   statement {
499     sid      = "AllowLifecycleAction"
500     effect   = "Allow"
501     resources = ["*"]
502     actions  = ["autoscaling:CompleteLifecycleAction"]
503   }
504
505   statement {
506     sid      = "AllowLogsReadAccess"
507     effect   = "Allow"
508     resources = ["*"]
509
510     actions = [
511       "logs:Describe*",
```

```
512     "logs:Get*",
513     "logs:List*",
514     "logs:TestMetricFilter",
515     "logs:FilterLogEvents",
516   ]
517 }
518
519 statement {
520   sid      = "AllowEventsReadAccess"
521   effect    = "Allow"
522   resources = ["*"]
523
524   actions = [
525     "events:DescribeRule",
526     "events:ListRuleNamesByTarget",
527     "events:ListRules",
528     "events:ListTargetsByRule",
529     "events:TestEventPattern",
530     "events:DescribeEventBus",
531   ]
532 }
533
534 statement {
535   sid      = "AllowCloudwatchReadAccess"
536   effect    = "Allow"
537   resources = ["*"]
538
539   actions = [
540     "autoscaling:Describe*",
541     "cloudwatch:Describe*",
542     "cloudwatch:Get*",
543     "cloudwatch:List*",
544     "sns:Get*",
545     "sns:List*",
546   ]
547 }
548 }
549
550 resource "aws_iam_role" "grafana_role" {
551   assume_role_policy = data.aws_iam_policy_document.grafana.json
552   name                = "${module.dev_eks_cluster.eks_cluster_id}-grafana-amt"
553 }
554
555 resource "aws_iam_role_policy" "grafana_policy" {
556   name = "${module.dev_eks_cluster.eks_cluster_id}-Grafana-Policy"
557   role = aws_iam_role.grafana_role.id
558   policy = data.aws_iam_policy_document.grafana_policy.json
559 }
560
```

```
561 resource "aws_iam_role_policy_attachment" "grafana_cw_readonly_attach" {  
562     role      = aws_iam_role.grafana_role.name  
563     policy_arn = "arn:aws-us-gov:iam::aws:policy/CloudWatchReadOnlyAccess"  
564 }
```