SPECIAL REPORT

CLOUD SECURITY THREATS

클라우드 보안 위협 동향

클라우드 환경을 노리는 7가지 위협

공격자들은 금전과 정보를 노리며, 많은 사용자들이 모이는 환경에 주목한다. 그리고 클라우드가 말 그대로 성황을 이루면서 공격자들의 시선도 자연스럽게 클라우드를 향하고 있다. 사용자들은 대개 클라우드가 가져다 주는 효율성과 생산성에 주목한다. 하지만 미흡한 보안으로 인해 클라우드 환경에서 공격을 당한다면 여러 긍정적인 가치들도 무용지물이 될 수 있다. 클라우드를 기반으로 미래 비즈니스 전략을 그리고 있다면 보안은 최우선적으로 고려해야할 요소 중 하나다.

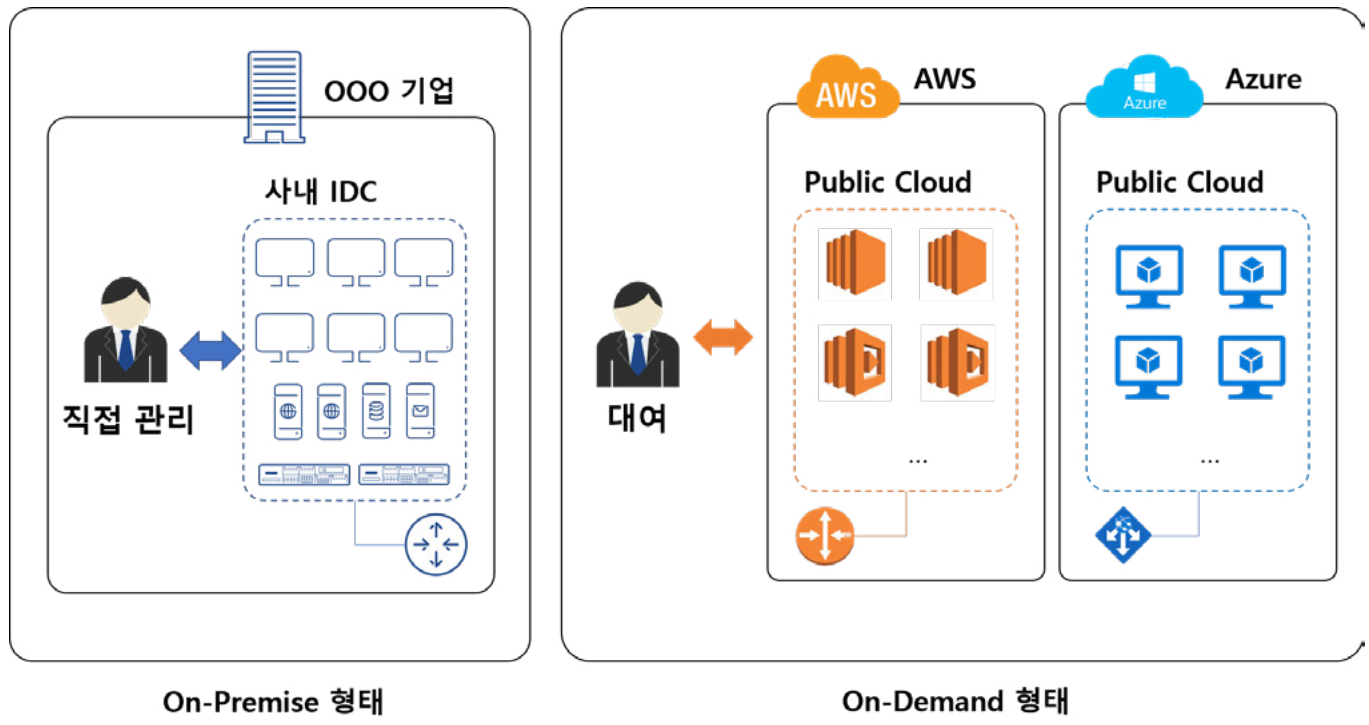
이번 글에서는 클라우드 보안에 대한 간단한 개념과 7가지 대표적인 위협에 대해 알아본다.

기존 IT 기업들은 사내 IDC(Internet Data Center)에서 IT 인프라 관련 장비를 직접 구매하여 구축하고 운영해왔다. IDC를 안전하게 운용하기 위해 항온항습기, 무정전전원 공급장치(UPS) 등을 사서 직접 관리하고 있지만, 인프라 장비 구입, 관리 비용, 인건비 등에 비용과 시간을 지속적으로 투입해야 한다.

이처럼 사내 IDC 내 인프라 환경을 구축하는 형태를 '온프레미스(On-Premise)'라고 한다. 최근에 기업들은 온프레미스 환경에서 발생하는 비용과 시간을 줄이기 위해 '온디맨드(On-Demand)' 환경인 클라우드를 도입하고 있다.

다양한 클라우드 서비스, 그리고 책임공유모델

클라우드는 기업 자체적으로 회사 내부 IDC에 서버를 설치하여 운영하는 형태가 아닌 클라우드 서비스 제공업체(Cloud Service Provider: CSP)로부터 컴퓨팅 자원을 빌리고 사용한 시간만큼만 비용을 지불하면 되기 때문에 온프레미스에 비해 구축 시간과 운영 비용을 절감할 수 있다. 또한 '스케일 업(Scale-Up)', '스케일 아웃(Scale-Out)' 등 갑작스러운 환경 변화에도 신속하고 탄력적으로 대응할 수 있기 때문에 많은 기업들이 클라우드를 도입하고 있는 상황이다.



[그림 1] 온프레미스와 온디맨드 비교

일반적으로 CSP에서 제공하는 서비스는 'IaaS(Infrastructure as a Service)', 'PaaS(Platform as a Service)', 'SaaS(Software as a Service)'로 나뉜다. 오라클(Oracle)과 KPMG의 클라우드 보안 위협 보고서에 따르면, 90%에 달하는 기업들이 SaaS를, 76%의 기업들이 IaaS를 도입해 사용하고 있고, 절반 이상의 기업들은 향후 2년 안에 모든 데이터를 클라우드로 이전할 계획인 것으로 나타났다. 이처럼 기업들은 인프라스트럭처 뿐만 아니라 플랫폼과 소프트웨어 형태의 클라우드 서비스도 활발하게 이용하고 있다.

사용자들이 클라우드 서비스를 사용하면서 오해하는 것 중 하나는 바로 'CSP에서 모든 보안을 알아서 제공할 것이다'라는 것이다. 하지만 CSP와 사용자 간 책임져야 할 영역은 명확히 구분된다. 그러므로 클라우드를 도입하려는 사용자는 책임져야할 부분을 명확히 구분 짓고 책임을 공유한다는 개념인 '책임공유모델(Shared Responsibility)'에 대해 인지하고 있어야 한다. 아래 [그림 2]는 일반적으로 클라우드 서비스 업체에서 제공하고 있는 책임공유모델을 나타낸 것이다.

항목	Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
Data	고객 책임	고객 책임	고객 책임
Application	고객 책임	고객 책임	클라우드 서비스 제공업체 책임
Operation System	고객 책임	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임
Servers	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임
Storage	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임
Network	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임
Physical	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임	클라우드 서비스 제공업체 책임

[그림 2] 클라우드 서비스 형태에 따른 책임공유모델 (출처: AWS)

기업 입장에서는 자신들의 핵심 사업에 집중하고자 하는 상황에서 클라우드의 각 영역별로 책임져야할 부분을 인지하고 전문적으로 또 지속적으로 관리하는 것이 어려울 수 있다. 이에, 클라우드 보안 중개 서비스 'CASB(Cloud Access Security Broker)', 클라우드 워크로드 보안 플랫폼 'CWPP(Cloud Workload Protection Platform)', 클라우드 정책 모니터링 및 관리 서비스 'CSPM(Cloud Security Posture Management)' 등 다양한 보안 서비스가 등장하게 되었다.

클라우드 환경을 노리는 7가지 위협

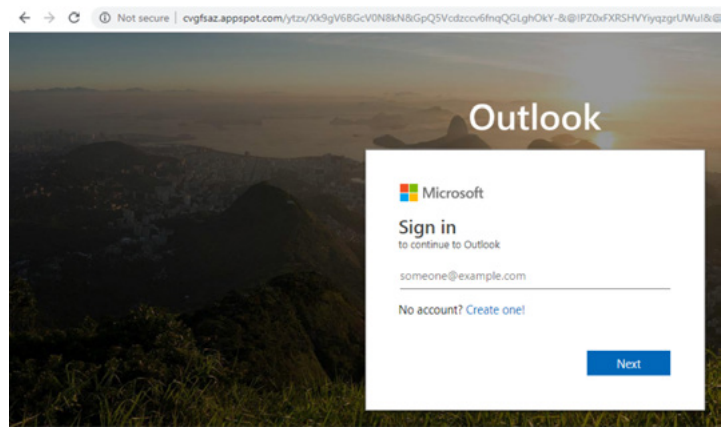
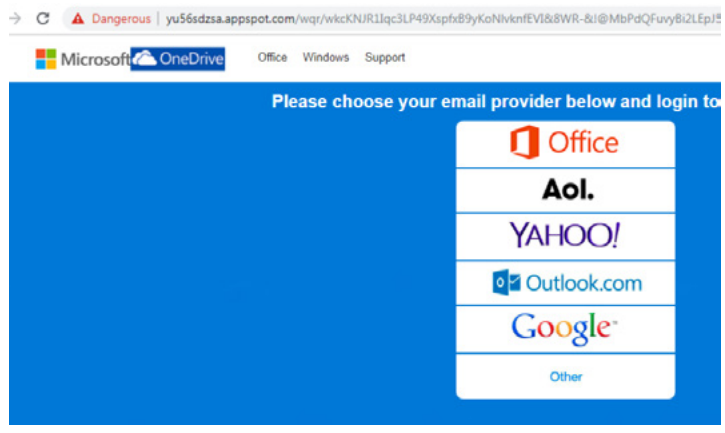
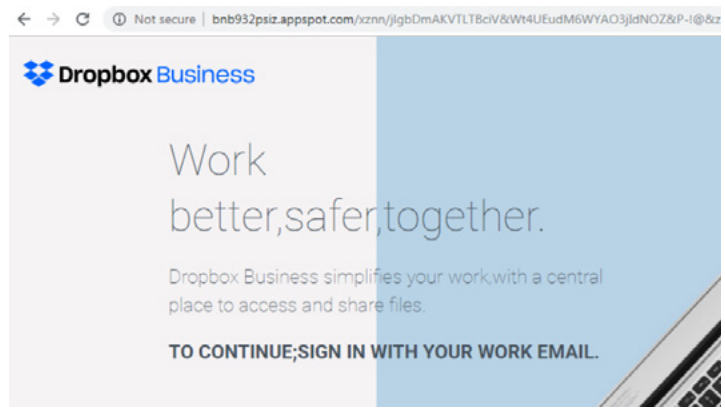
다양한 클라우드 보안 서비스 등장의 이면에는, 클라우드를 대상으로 한 공격 증가가 있다. 클라우드 도입이 급속도로 확산되고, 금전과 정보를 탈취하기 위해 사용자가 모이는 곳을 노리는 공격자들의 오랜 습성을 생각해보면 그리 놀라운 현상은 아니다. 실제로도 다양한 형태의 클라우드 보안 위협 사례가 매년 나타나고 있다. 이에, 최근 발생했던 보안 이슈들 중 빈번하게 일어나는 7가지를 다음과 같이 정리했다.

1. 클라우드 도메인을 이용한 공격

퍼블릭 클라우드 서비스 이용 시 제공받는 도메인을 피싱 사이트, 악성코드 배포, 공격자 서버(C&C) 통신 등으로 악용하는 공격이다. 많은 CSP들이 웹 애플리케이션을 쉽게 개발하고 운용할 수 있는 PaaS 형태의 개발 환경 플랫폼을 제공한다. 이러한 개발 환경 플랫폼에 웹 애플리케이션 소스코드를 업로드하면 웹 페이지에 접근할 수 있는 클라우드 자체 도메인이 제공된다.

공격 예시를 살펴보도록 하자. 구글 클라우드 플랫폼(Google Cloud Platform)은 서버리스(Serverless) 상태에서 웹 애플리케이션을 바로 배포할 수 있는 '앱 엔진(App Engine)'이라는 PaaS 서비스를 제공한다. 해당 서비스를 통해 웹 소스코드를 업로드하면 '*.appspot.com'이라는 도메인이 자동으로 지급되며, 개발자는 손쉽게 애플리케이션을 테스트하고 배포할 수 있다.

공격자들은 앱 엔진 서비스를 악용하여 아래 [표 1]과 같은 피싱 사이트를 생성했다.

피싱 사이트 도메인	피싱 사이트 화면
cvgfsaz.appspot.com	 Outlook 피싱 페이지
yu56sdzsa.appspot.com	 OneDrive 피싱 페이지
bnb932psiz.appspot.com	 Dropbox 피싱 페이지

[표 1] 피싱 사이트 화면

일반 사용자들은 상위 도메인('*.appspot.com')만 보고 해당 사이트가 정상 사이트라고 착각하여 개인정보를 입력하는 경우가 많다. 하여, [표 2]와 같이 수많은 피싱 사이트가 지속적으로 제작되어 유포되는 실정이다.

피싱 사이트 도메인	
uy67dass[.]appspot[.]com	ja8fspxzosaa[.]appspot[.]com
gjf9pxzosa[.]appspot[.]com	egoew023pzas[.]appspot[.]com
vhkad03pas[.]appspot[.]com	kda8gazxa[.]appspot[.]com
adgkao93pz[.]appspot[.]com	l9rwpodsxcs[.]appspot[.]com
cvgfsaz[.]appspot[.]com	jga9spzas[.]appspot[.]com
jjad9gdpxzsa[.]appspot[.]com	vadgka932oa[.]appspot[.]com
ls9ixosdsasa[.]appspot[.]com	qwsa92oozxa[.]appspot[.]com
adlg402ooz[.]appspot[.]com	bnb932psiz[.]appspot[.]com
vfhgj3sz[.]appspot[.]com	eyq246ddpoas[.]appspot[.]com
h45dsagga[.]appspot[.]com	yt76uyhxzz[.]appspot[.]com

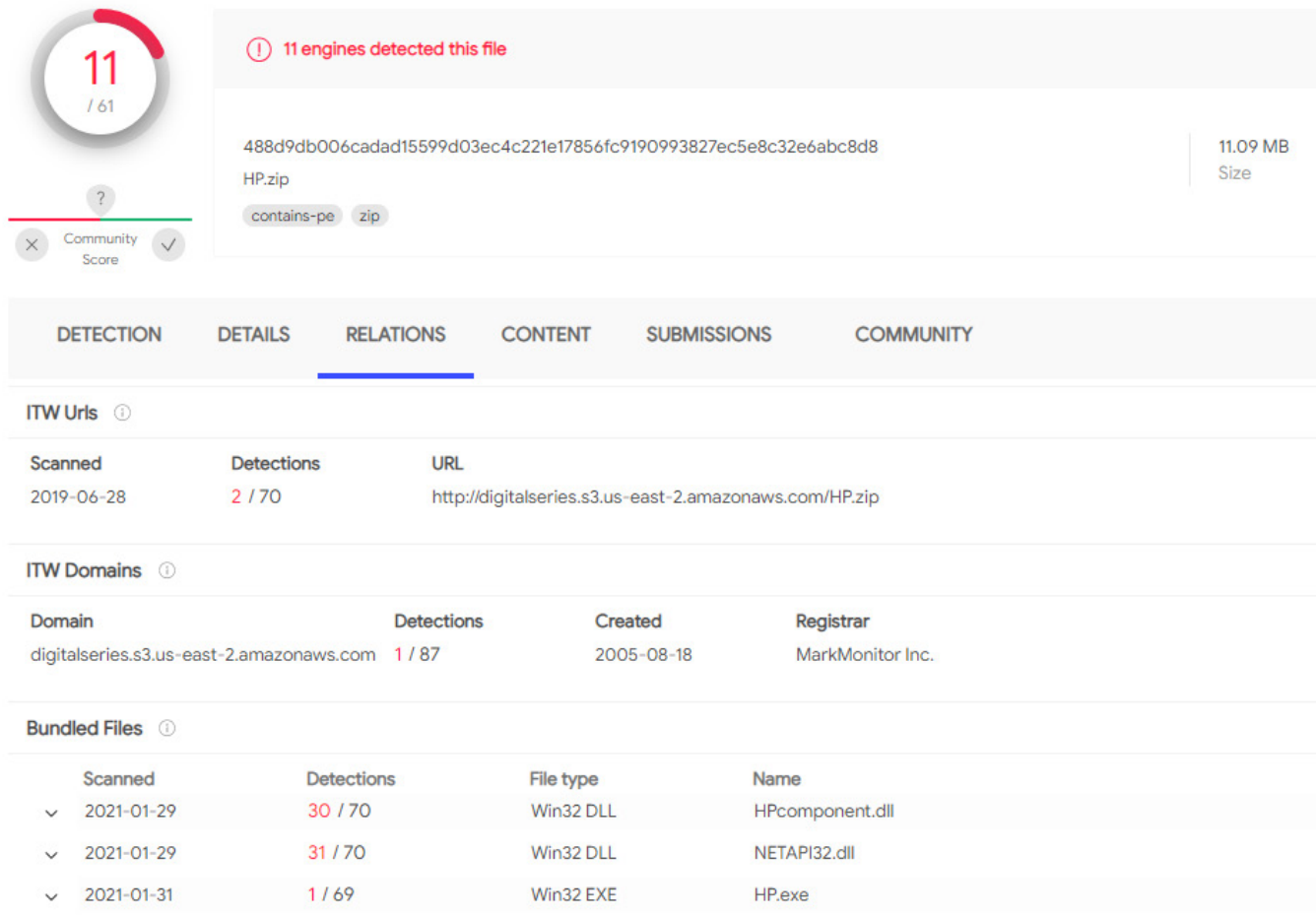
[표 2] 앱 엔진으로 제작된 피싱 사이트 도메인의 일부

개발 환경 플랫폼 서비스 뿐만 아니라 스토리지 서비스도 악성코드 배포에 활용되고 있다. AWS의 S3와 네이버의 MYBOX 서비스를 비롯한 스토리지 서비스들은 파일 다운로드 서비스를 제공할 때 대부분 [표 3]과 같이 공식 도메인의 하위 도메인을 제공한다.

스토리지 서비스	URL 형식
AWS – S3	https://[버킷 이름].s3-[AWS 리전].amazonaws.com/[파일명]
	https://s3.[AWS 리전].amazonaws.com/[버킷 이름]/[파일명]
NAVER – MYBOX	https://file.scloud.naver.com/.fileLink/랜덤값

[표 3] 스토리지 서비스에 따른 URL 형식

실제로 유입되는 많은 악성코드들을 조사해보면 클라우드 스토리지 서비스에서 제공되는 도메인을 통해 들어오는 경우가 많다. 이는 도메인으로만 판단했을 때 공식 도메인으로 착각하여 위협을 놓치기 때문이다



[그림 3] 클라우드 스토리지 서비스를 통해 유입된 악성코드 – 바이러스토탈(Virustotal) 화면

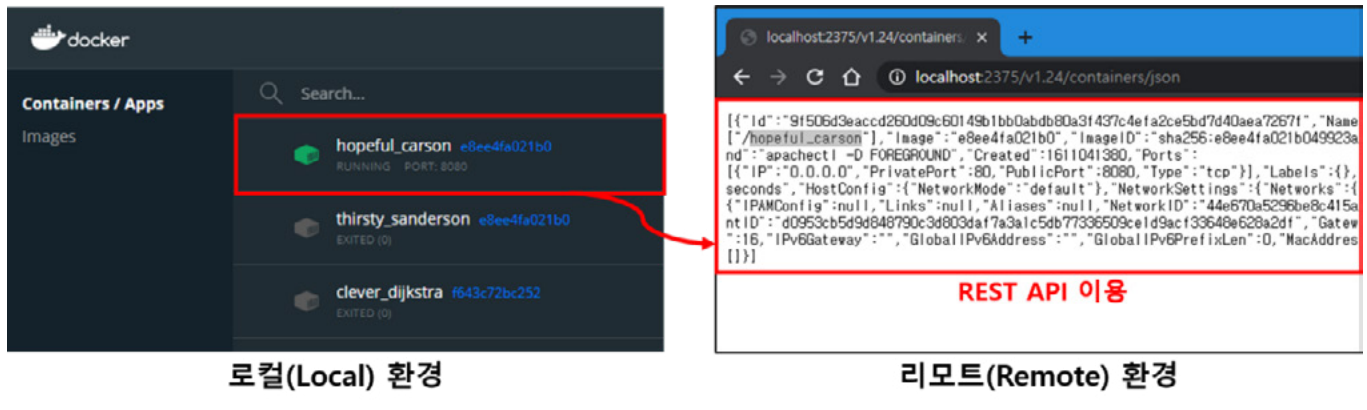
이처럼 다양한 클라우드 서비스들이 공격에 활용되고 있기 때문에 방어자는 사전에 위협들을 인지하고 클라우드 서비스로 나올 수 있는 도메인들을 사전에 파악해 면밀히 살펴봐야 한다.

2. 취약한 도커 & 쿠버네티스 API 서버 공격

클라우드 도입이 확대되면서 많은 IT 인프라 환경이 '도커(Docker)'로 구축되고 있다. 도커란 애플리케이션과 라이브러리들을 '컨테이너(Container)'로 묶어 서비스 구동을 위한 격리 환경을 만들어 주는 오픈소스 플랫폼을 말한다. 도커는 개발 및 서버 환경을 쉽게 구축하고 관리할 수 있기 때문에 많은 IT 인프라에서 사용되고 있다. 도커는 '도커 이미지(Docker Image)'와 '도

커 컨테이터(Docker Container)'들을 관리할 수 있는 'CLI(Command Line Interface)'를 제공하지만, 기본적으로 제공된 CLI는 로컬 환경에서만 사용할 수 있다.

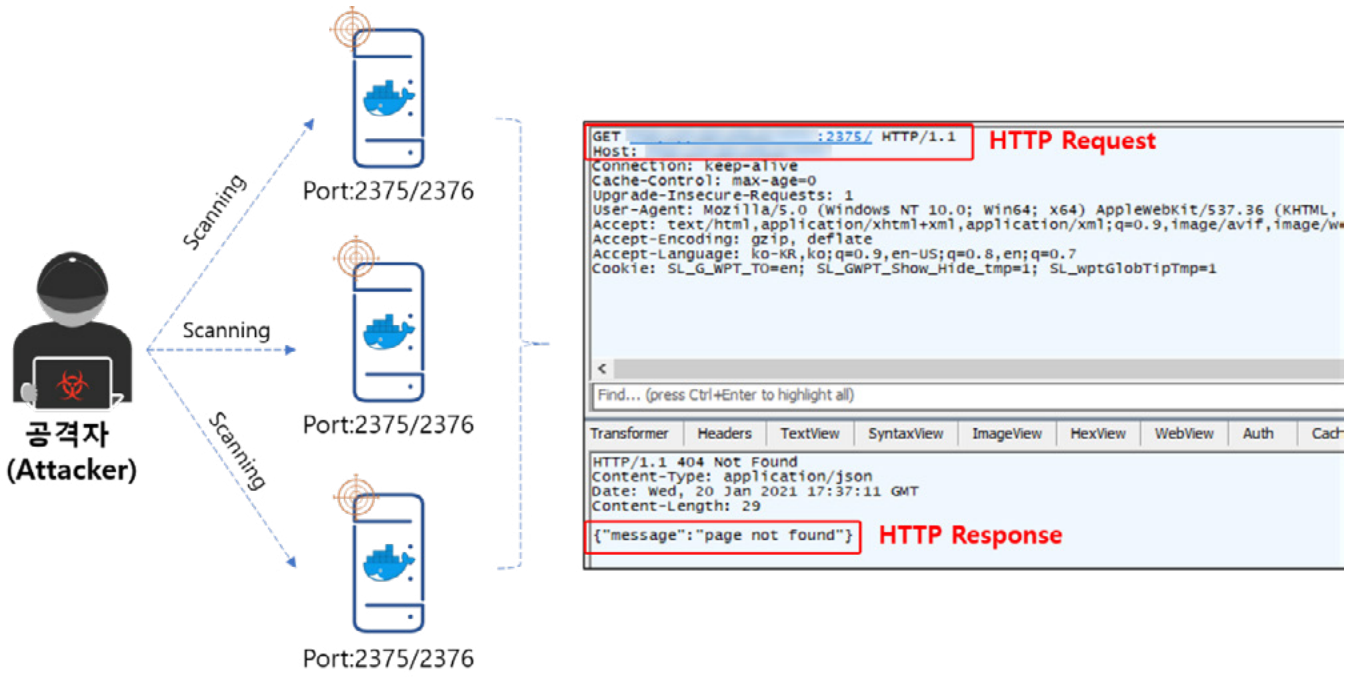
하지만 도커의 추가 설정을 통해 리모트(Remote) 환경에서도 'CLI' 사용이 가능한 'REST API'를 활용할 수 있다. 'REST API'를 이용하면 아래와 같이 리모트 환경에서도 동일하게 도커들을 관리할 수 있다.



[그림 4] 리모트 환경에서 REST API를 이용해 도커 조회

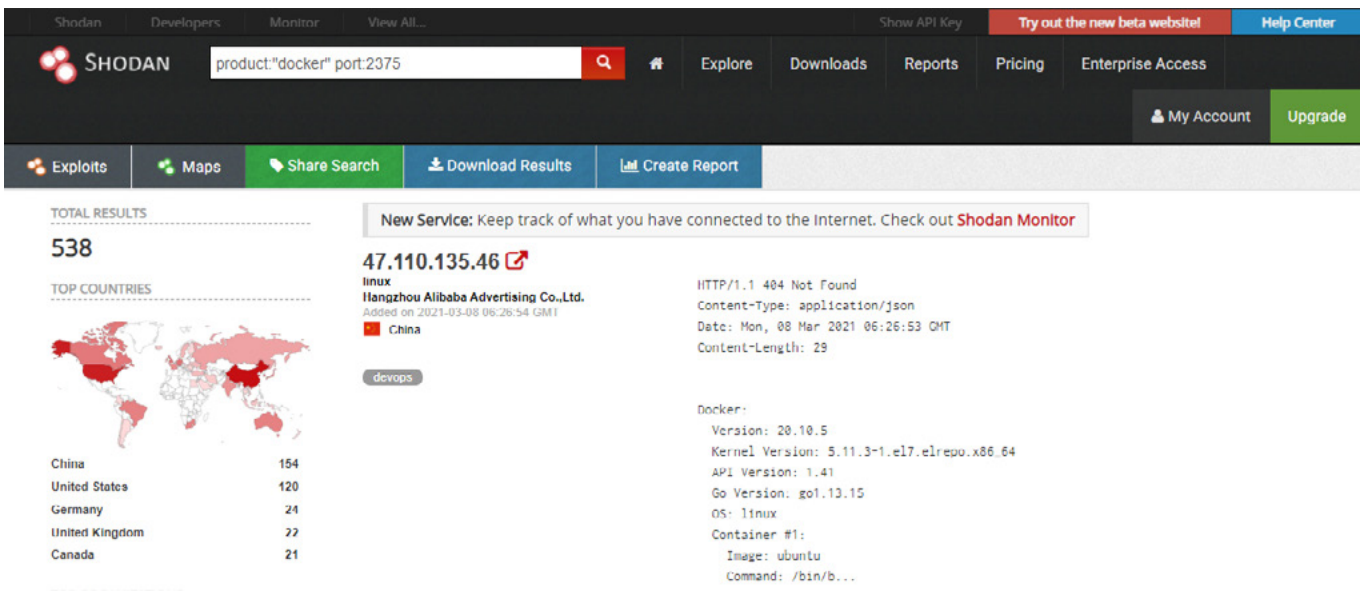
이 때, 도커 REST API 서버의 IP와 포트가 외부에 노출될 경우, 권한이 없는 제3자도 REST API를 통해 도커 이미지와 도커 컨테이너들을 관리할 수 있다. 도커 REST API 서버 접속 시 '{"message":"page not found"}'라는 메시지 값이 나오는데, 이를 통해 누구나 도커 REST API 서버인지 여부를 쉽게 판별할 수 있다.

공격자들은 이러한 방법으로 도커 REST API 서버를 스캐닝한 뒤 REST API를 악용하여 마이너 도커(Miner Docker), 디도스 도커(DDoS Docker) 등 악성 도커를 다운로드 받아 실행시키거나 혹은 시스템을 방해하기 위해 기존 도커들을 임의로 삭제한다.



[그림 5] 공격자의 도커 REST API 스캐닝

도커 뿐만 아니라 도커들을 관리하고 운영할 수 있는 오케스트레이션(Orchestration) 도구인 '쿠버네티스(Kubernetes)'에도 REST API가 존재하기 때문에 공격자의 위협에 노출되고 있다. 특히, OSINT(Open Source INTelligent) 도구 중 하나인 '쇼단(Shodan)'에 많은 도커 및 쿠버네티스 REST API 서버들이 노출되어 있어 언제든지 공격에 악용될 수 있다. 방어자 입장에서는 해당 보안 위협을 사전에 인지하고 추가 인증 절차와 특정 IP 허용과 같은 정책들을 설정해야 한다.

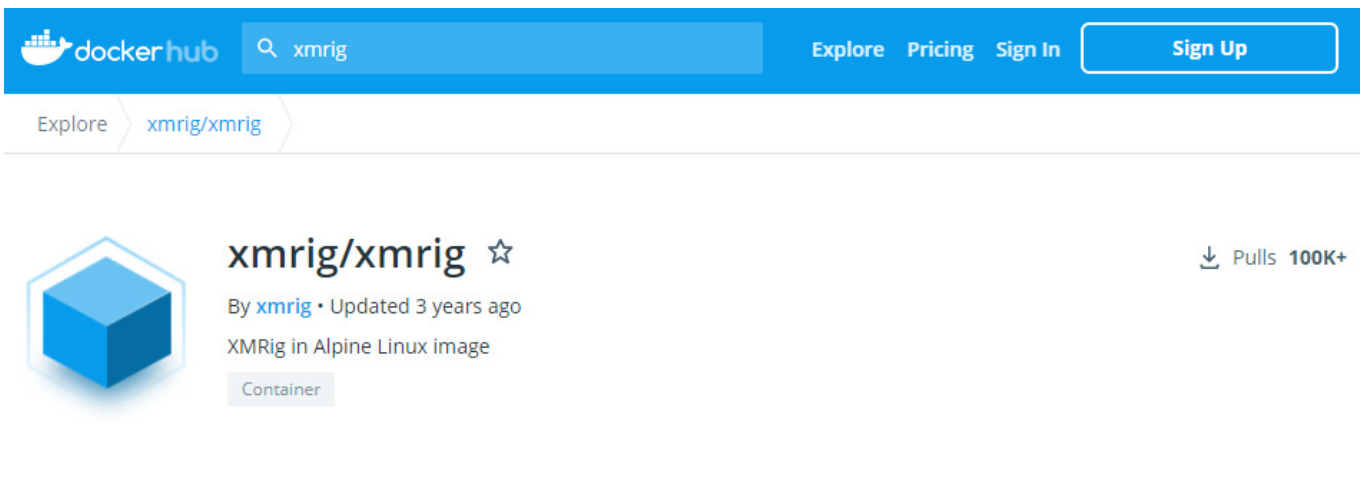


[그림 6] 쇼단에 노출된 도커 REST API 서버

3. 도커 허브 내 도커 이미지 위협

최근 들어 '도커 허브(Docker Hub)에 악성 도커 이미지가 유포되어 많은 사용자들이 위협 받고 있다. 도커 허브란 사용자들이 도커 이미지들을 저장 및 배포함으로써 아래와 같이 도커 이미지들을 형상 관리할 수 있는 공개 저장소(Public Repository)를 말한다.

공격자들은 도커 허브에 악의적으로 디도스 또는 채굴 애플리케이션이 포함된 악성 도커 이미지를 업로드한다. 업로드 시 보안 교육, 테스트 등 정당한 목적으로 업로드하거나 정상 도커 이미지로 위장해 업로드한다.



[그림 7] 도커 허브에 정당한 목적으로 업로드된 도커 이미지

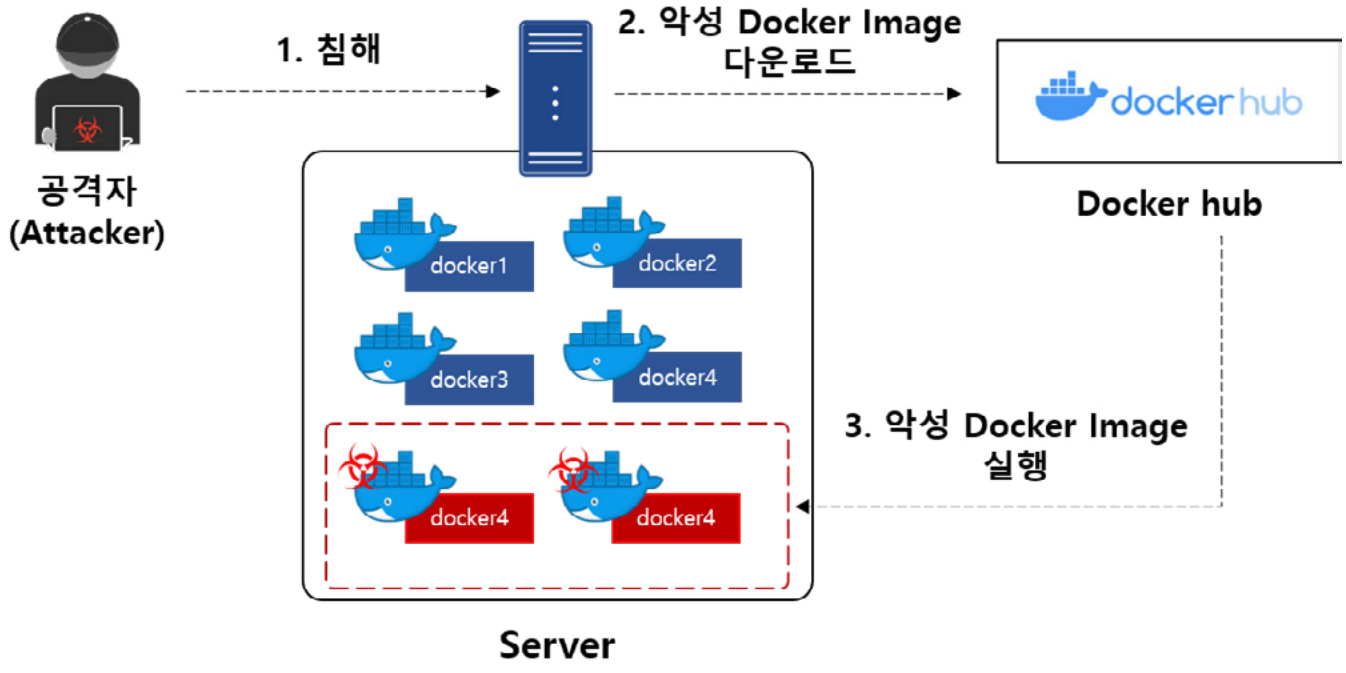
정당한 목적으로 업로드된 악성 도커 이미지를 살펴보면, 용도가 대부분 디도스 또는 암호화폐 채굴이다.

용도	합법적 용도로 업로드된 악성 도커 이미지
디도스	douglasslow/slowhttptest:latest
	foxleon/udpflood:latest
	nxqsmfxx2/stupid:latest

디도스	nxqsmfxx2/stupid:slowhttptest
암호화폐 채굴	bitnn/alpine-xmrig:latest
	patsissons/xmrig:latest
	hildeteamtnt/xmrigminer:latest
	kannix/monero-miner:latest

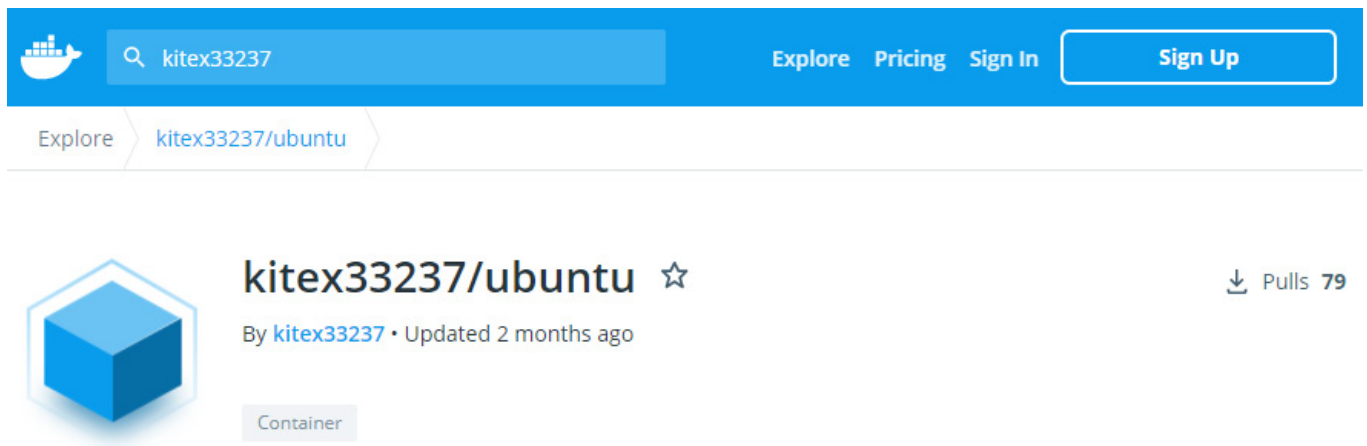
[표 4] 합법적 용도로 업로드된 악성 도커 이미지 리스트

업로드된 이미지들은 정당한 목적으로 사용될 수 있지만, 공격자들은 대부분의 경우 '도커 서버'를 침해한 후 업로드된 이미지들을 다운로드하여 디도스 및 채굴 용도로 사용한다.



[그림 8] 침해된 서버에 악성 도커 이미지 다운로드 및 실행

또한, 공격자들은 정상 도커 이미지로 위장해 업로드하기도 한다.



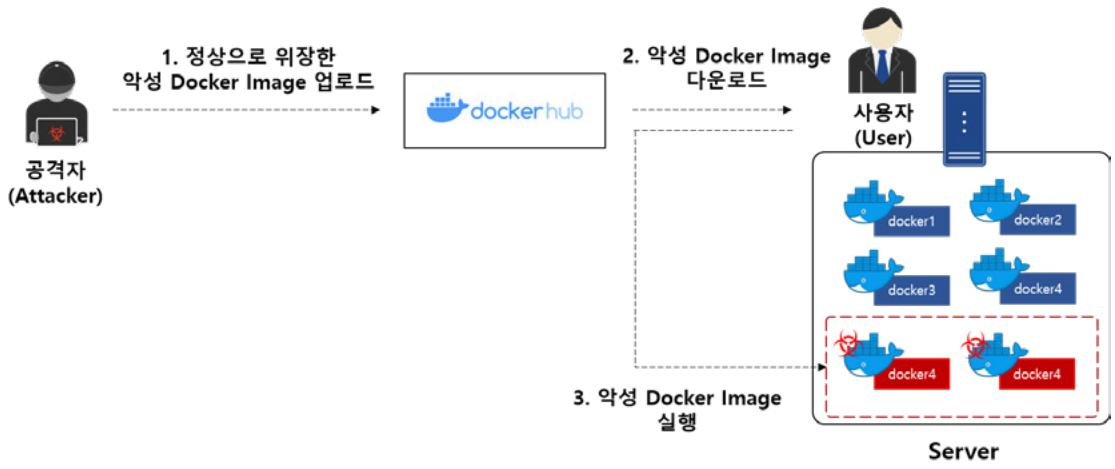
[그림 9] 정상 도커 이미지로 위장한 악성 도커 이미지

아래와 같이 많은 악성 도커 이미지들이 정상 도커 이미지로 위장해 배포되고 있다.

용도	정상 도커 이미지로 위장한 악성 도커 이미지
디도스	userubuntu1/zores:latest
암호화폐 채굴	kitex33237/ubuntu2:latest
	ubvntu/utnubu:latest
	felilca/ubuntu:latest
	greekgoods/kimura:1.0
	jzuluagau23/ubuntu:platzi
	tanchao2014/java1.6:latest
	vkhopade/nginx:v8.9
	pocosow/centos:7.6.1810
	shaylsholmes/myubuntu:3.0
	ubuntuz/jessy:latest

[표 5] 정상 도커 이미지로 위장한 악성 도커 이미지 리스트

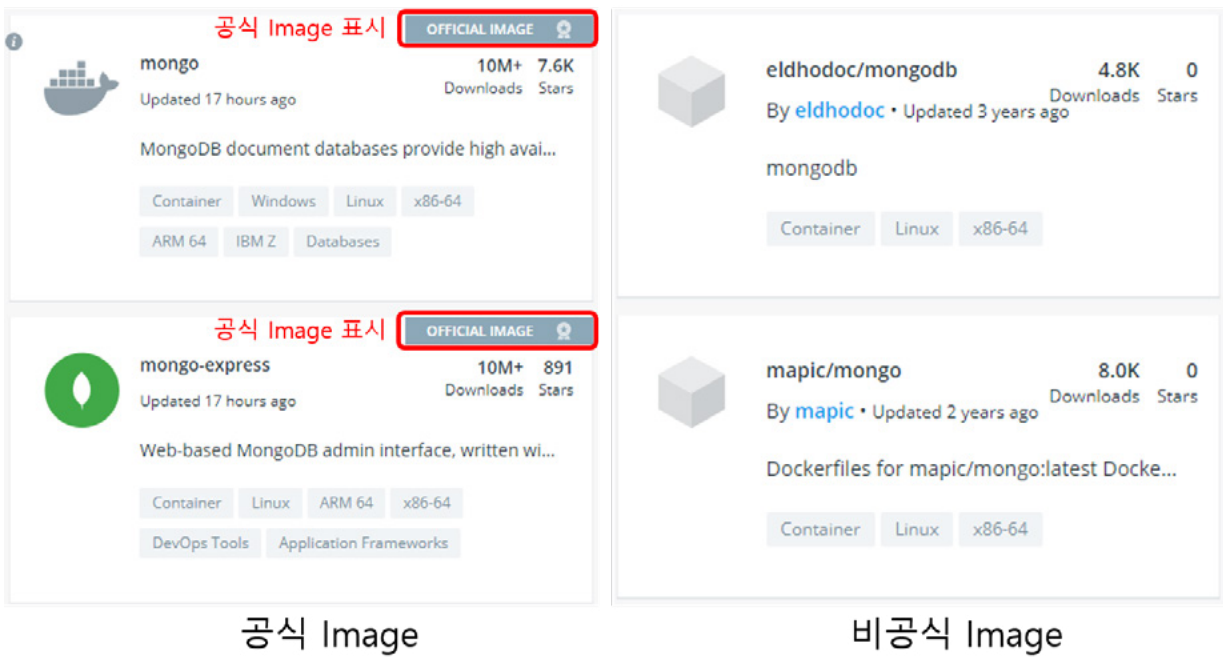
이를 통해 공격자들은 암호화폐 채굴 또는 디도스에 사용되는 컴퓨팅 자원을 보다 빠르고 쉽게 얻을 수 있다.



[그림 10] 위장 악성 도커 이미지 다운로드 및 실행

악성 도커 이미지 공격이 증가하면서 여러 해결책이 제시되고 있지만, 공격자들은 매년 다양한 방법으로 우회해 악성 도커 이미지를 업로드하고 있다. 악성 도커 이미지 외에도, 정상이지만 이미지 내에 오래된 오픈소스 패키지나 라이브러리가 사용되어 취약점 공격에 노출되기도 한다.

따라서 도커 이미지를 사용할 때, 신뢰도 높은 공식 이미지를 다운로드 받고 사용하기 전에 이미지 내에 사용된 소프트웨어가 안전한지 검사해야 한다.



[그림 11] 공식 이미지와 비공식 이미지 구분

4. 설정이 취약한 클라우드 서비스 대상 공격

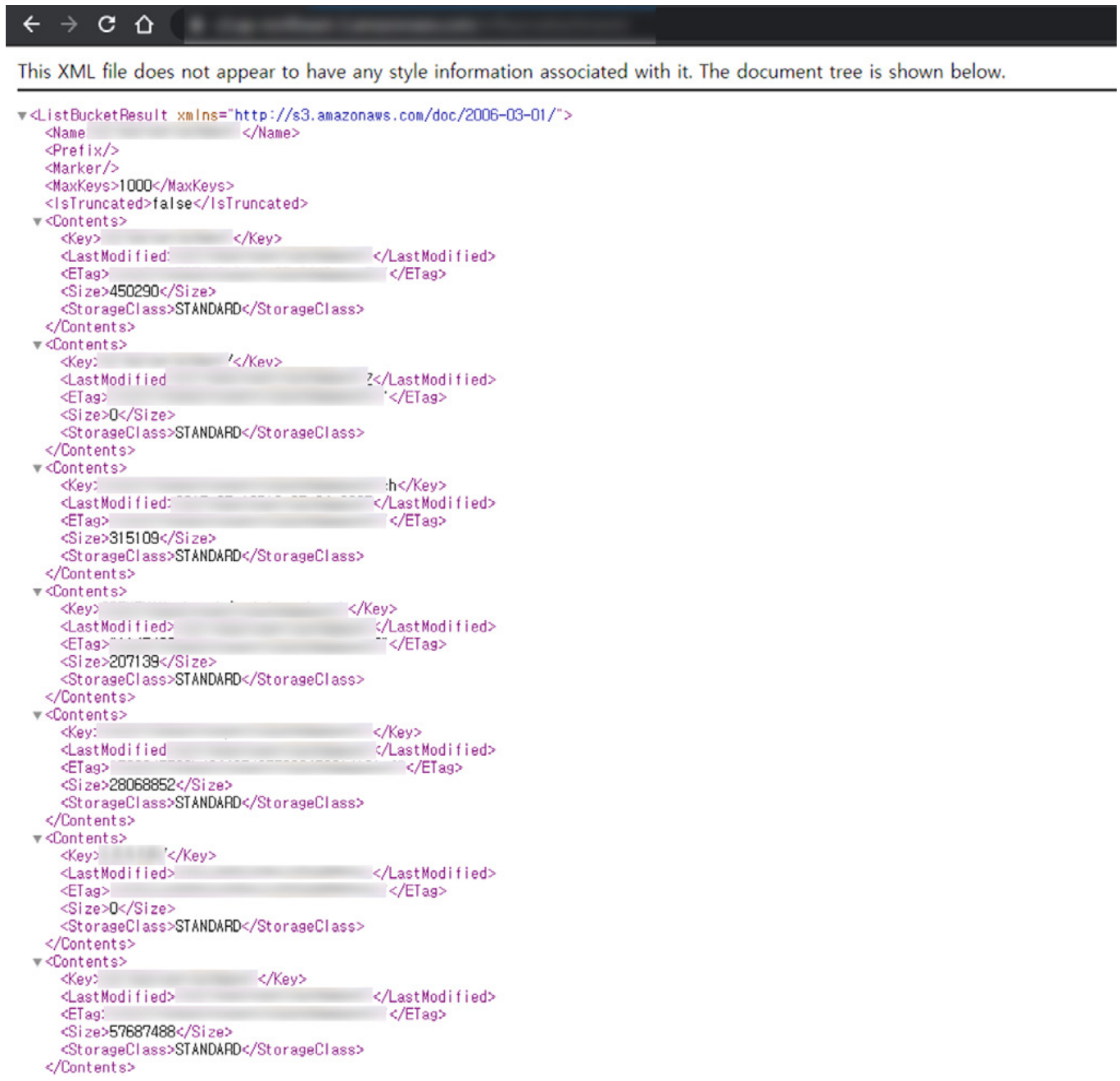
시장조사기관 가트너(Gartner)의 조사 결과에 따르면, 2023년까지 클라우드 보안 실패 사례 중 99% 이상이 고객의 잘못으로 인해 발생하고, 2021년까지 클라우드 내 중요 정보 유출의 95% 이상은 관리자 실수로 인해 발생할 것으로 전망됐다. 실제 일어났던 정보 유출 사례를 보면 설정이 취약한 클라우드 서비스에서 중요 정보가 자주 유출되었다.

날짜	업체	내용
2020.03.17.	Advantage Capital Funding	AWS S3 버킷내 425GB의 데이터 유출
2020.02.12	JailCore	AWS S3 버킷내 36000건의 데이터 유출
2020.01.22.	THSuite	AWS S3 버킷내 30,000명의 데이터 유출
…	…	…
2019.04.03.	Facebook	AWS S3 버킷내 5억5천개의 고객 데이터 유출
2018.02.15.	FedEx	AWS S3 버킷내 119,000개의 데이터 유출
2017.07.19.	Dow Jones	AWS S3 버킷내 220만명의 데이터 유출

[표 6] 클라우드 내 정보 유출 사례

설정이 취약할 수 있는 클라우드 서비스로는 AWS S3, Azure Blob Storage, AWS Lambda, Azure AD 등이 있다. 스토리지 서비스로 많이 쓰이는 AWS S3의 경우 'buckets.grayhatwarfare.com'과 같은 OSINT를 통해 취약한 설정을 가진 버킷들을 쉽게 발견할 수 있다.

AWS S3 버킷은 기본적으로 비공개 모드로 설정되어 있다. 하지만 관리자의 잘못으로 취약하게 설정되어 있다면, 아래와 같이 AWS S3 버킷 접근 시 디렉토리 리스팅이 되어 누구나 텍스트 파일, 사진 동영상 등 '객체(Object)'에 쉽게 접근하여 정보를 탈취할 수 있다.



[그림 12] 취약한 설정의 AWS S3 버킷

공격자는 이처럼 취약한 설정의 클라우드 서비스를 찾아 데이터를 유출시키는 공격을 시도하고 있다. 기업 입장에서는 특정 클라우드 서비스를 사용하기 전에 해당 서비스의 취약한 설정을 통해 발생될 수 있는 위협에 대해 사전에 인지해야 한다.

5. 공개 저장소에 노출된 자격 증명 값

최근 들어 'CI/CD(Continuous Integration/Continuous Delivery)'를 통한 '데브옵스(Devops)' 개발이 증가하면서, 클라우드 서버에 대한 접근이 필요해졌다. 클라우드 서버 접근 시 필요한 자격 증명이 개발 코드 내 일부 포함되면서, 개발자가 공개 저장소(Public

Repository)에 작업한 파일을 업로드 할 때, 실수로 자격 증명 파일까지 업로드해 클라우드 서버가 위험에 노출 경우가 발생하게 된다.

미국 노스캐롤라이나 주립 대학의 연구에 따르면 Github 내 파일을 스캔한 결과 100,000개의 저장소에서 암호화 키 및 API 토큰과 같은 자격 증명 파일이 노출되어 있는 것으로 나타났다. 실제로 [그림 13]과 같이 Github에서 자격 증명 파일이 빈번하게 노출된다.

```
73    "upload artifacts" :
74        - command : s3.put
75          params :
76            optional : true
77            aws_key : "
78            aws_secret : "
79            local_file :
80            remote_file :
```

[그림 13] Github에 노출된 AWS 자격 증명 내용

시장조사기관 '컴패리테크(Comparitech)'에 의하면 Github에 허니팟(HoneyPot) 목적으로 AWS 자격 증명 파일을 올려본 결과, 1분도 지나지 않아 1,000번의 사용 시도가 발견되었다. 이는 개발자가 코드를 Github에 실수로 업로드 후 삭제하더라도 공격자가 그전에 탈취할 가능성이 있다는 뜻이다.

Github 저장소 뿐만 아니라 도커 허브 저장소에 도커 이미지들을 업로드할 때에도, 자격 증명 파일을 같이 업로드하여 클라우드 서버가 위험에 노출될 수 있다. 도커 이미지 내에 자격 증명 내용들이 포함될 만한 곳은 크게 3가지로 나눌 수 있다. 첫 번째는 '소스코드' 부분이다. 도커 이미지를 실행해 클라우드 서버와 연동시키기 위해 아래와 같이 서버에 대한 자격 증명 값들이 포함되어 있는 경우가 있다.

```
const router = express.Router();
// Serve Static files from the React app
app.use(express.static(path.join(__dirname, 'client/build')));

const dbRoute = "
// const dbRoute = "
//connection with database.

mongoose.connect(
    dbRoute,
    {useNewUrlParser: true}
);
let db = mongoose.connection;
```

[그림 14] 소스코드 내 포함된 자격 증명 값

두 번째는 '환경 변수' 부분이다. 도커 이미지를 빌드하는 과정에서 환경 변수 내 자격 증명 값들이 포함되어 있는 경우가 있다

[그림 15] 환경 변수 내에 포함된 자격 증명 값

세 번째는 '자격 증명 파일'이다. [그림 16]에서 '도커파일(Dockerfile)'의 내용을 보면 자격 증명 파일을 저장하고 사용하도록 설계되어 있다. 이는 도커 이미지 내에 자격 증명 파일이 포함되어 있음을 의미하는데, 이러한 도커파일로 도커 이미지들을 제작하면 공격 위험에 노출될 수 있다.

```dockerfile
FROM ubuntu as intermediate

WORKDIR /app
COPY secret/key /tmp/
RUN scp -i /tmp/key build@acme/files .

FROM ubuntu
WORKDIR /app
COPY --from=intermediate /app .
```

[그림 16] 자격 증명 파일을 사용하는 도커파일

도커파일에서 사용되는 자주 사용되는 자격 증명 파일들은 [표 7]과 같다.

서비스	자격 증명 파일
AWS	.aws/credentials
	.aws/config
SSH	.ssh/config
Docker	.docker/config.json
Kubernetes	.kube/config

[표 7] 서비스에 따른 자격 증명 파일

이처럼 공개 저장소에서 소스코드가 발견되거나, 도커 이미지 내에서 클라우드 서버의 자격 증명 파일이 발견되는 경우가 발생하고 있다. 따라서 공개 저장소에 업로드하기 전, 자격 증명 관련 값 또는 파일이 있는지 검사해야 한다.

6. 리눅스 플랫폼 악성코드

클라우드 기술이 발전하면서 많은 서버들이 도커 환경으로 바뀌고 있다. 자연스럽게 리눅스(Linux) 환경이 활성화되었고, 이를 노리는 리눅스 악성코드 또한 증가하고 있다. 공격자들은 리눅스 악성코드를 새로 제작하는 것이 아닌 기존 윈도우(Windows) 악성코드를 리눅스로 포팅(porting)하여 배포하는 것으로 파악되고 있다. [표 8]은 'BIFROSE'라는 RAT(Remote Administration Tool) 악성코드이다. BIFROSE는 예전부터 윈도우를 겨냥하는 것으로 알려졌지만, 시간이 지나면서 리눅스를 노리는 'BIFROSE'가 발견되었다.

파일 이름	설명
파일 크기	596.07 KB (610376 bytes)
MD5	4a4ea45cb850e4decd5fdb23909ea728
SHA1	5d8b228e3014b4eb579e380b3a1113dd8c0d999a
주요 기능 및 특징	BIFROSE(ELF)
안랩 진단명	Linux/Bifrose.610376

[표 8] BIFROSE 파일 정보

실제로 윈도우 기반 BIFROSE와 리눅스 기반 BIFROSE가 PC에서 수집한 정보 형식을 비교해 보면 비슷한 형식으로 정보를 수집한다는 것을 확인할 수 있다. 이를 통해 공격자가 리눅스 기반 BIFROSE를 기존 윈도우 기반으로 제작된 BIFROSE의 소스코드를 리눅스용으로 포팅하여 제작한 것으로 추정할 수 있다.

플랫폼 종류	수집된 정보 형식
윈도우 기반 BIFROSE	<victim IP>|default_zz|<hostname>|<username>|2.0.0a||1|-1|0|2600|1|1|0|0|982bc1da|C:\Documents and Settings\Administrator\Recent|C:\Documents and Settings\Administrator\Desktop|C:\Documents and Settings\Administrator\My Documents|US|00000409|
리눅스 기반 BIFROSE	<victim IP>|unix|<hostname>|<username>|5.0.0.0|0|1|1|0|575|0|0|0|0|None||||

[표 9] 윈도우와 리눅스 기반 BIFROSE 악성코드 비교

BIFROSE 뿐만 아니라 아래와 같이 여러 악성코드들이 리눅스로 포팅되어 배포되고 있다.

악성코드명	변경 전 플랫폼	변경 후 플랫폼
GravityRAT	윈도우	리눅스
PLEAD	윈도우	리눅스
RansomExx	윈도우	리눅스
TrickBot Anchor	윈도우	리눅스

[표 10] 다양한 플랫폼으로 포팅되어 배포된 악성코드 (출처: intego, SECURELIST)

악성코드가 이렇게 쉽게 포팅되는 이유 중 하나는 바로 '크로스 플랫폼(Cross Platform)'에 용이한 언어를 사용했기 때문이다. 크로스 플랫폼에 용이한 언어로는 Golang, .NET Core, PowerShell Core 등이 있다. 특히, Golang의 경우 하나의 소스코드로 윈도우, 리눅스, MacOS 등 다양한 플랫폼의 프로그램을 쉽게 제작할 수 있는 대표적인 크로스 플랫폼 지원 언어이다.

최근 들어, 리눅스 악성코드 중 Golang으로 제작된 악성코드의 비율이 증가하고 있다. 실제 발

견된 Golang 악성코드 목록의 일부는 [표 11]과 같다. 서로 다른 플랫폼에서 동작되는 악성코드라도 포팅되어 새로운 플랫폼을 타겟으로 재배포될 수 있으므로, 크로스 플랫폼에 용이한 언어로 제작된 악성코드를 포착하면 면밀히 모니터링해야 한다.

Golang으로 제작된 악성코드	
GDRat	Glupteba
Snake Ransomware	GoBot2
RansomExx	HERCULRES
CHAOS	GoBrut
ARCANUS	hershell
EGESPLOIT	Mauri870 Ransomware
r2r2	RobbinHood Ransomware
Rakos	TrumpHead Ransomware
telegrab	gorsh

[표 11] Golang으로 제작된 악성코드 리스트

7. Docker Escape

일반적으로 도커 환경에서는 악성코드에 감염되어도 도커 컨테이너들이 서로 격리된 환경에서 동작하기 때문에 다른 도커 컨테이너 또는 '호스트(Host)'에 침범할 수 없다.

하지만 'Docker Escape' 취약점이 존재한다면, 격리된 도커 환경을 벗어나 다른 도커 컨테이너나 호스트를 침범할 수 있다. Docker Escape 취약점들은 아래와 같이 대부분 컨테이너 생성과 실행을 위해 설계된 CLI 도구인 'runC'를 통해 발생되었다.

취약점 코드	내용
CVE-2019-19921	‘runc’를 이용한 ‘Docker Escape’ 취약점
CVE-2019-5736	‘runc’를 이용한 ‘Docker Escape’ 취약점
CVE-2016-5195	‘runc’를 이용한 ‘Docker Escape’ 취약점
CVE-2016-9962	‘runc’를 이용한 ‘Docker Escape’ 취약점
CVE-2016-3697	‘runc’를 이용한 ‘Docker Escape’ 취약점

[표 12] ‘runc’를 이용한 Docker Escape 취약점

‘runC’ 외에도 도커에서 제공하는 기능을 통해 Docker Escape 취약점이 계속 발생하고 있다. 실제 도커 기반으로 클라우드 서비스를 제공하는 마이크로소프트 애저(Microsoft Azure)의 ‘Azure Function’과 구글 클라우드의 ‘Google Cloud Shell’ 같은 서비스에서 Docker Escape 취약점이 발견된 사례들이 있다. 따라서, 도커 사용자나 도커를 이용하여 서비스를 제공하는 서비스 제공업체들은 이러한 위협을 지속적으로 모니터링할 필요가 있다.

결론

지금까지 클라우드에 대한 간단한 개념과 클라우드 환경에서 자주 일어나는 7가지 보안 위협에 대해 알아보았다. IT 기술이 발전함에 따라 클라우드 환경이 확대되고 있기 때문에 이러한 보안 위협을 인지하고 있는 것이 무엇보다 중요하다.

클라우드에서 일어날 수 있는 보안 위협은 본 문서에서 소개한 7가지 외에도 다양하게 존재한다. 클라우드 도입 시, ‘책임공유모델’을 기반으로 서비스 모델에 따라 클라우드에서 발생할 수 있는 다양한 보안 위협을 사전에 인지하고 보안 문제가 발생하지 않도록 지속적으로 관리해야 한다.