アンラボ・セキュリティレター

Press Ahn

2021.4 Vol.88

クラウド環境を狙う７つの脅威

クラウドにおけるセキュリティ脅威動向

クラウド環境を狙う 7 つの脅威

攻撃者は、金銭や情報を狙い、多くのユーザーが集まる環境に着目する。そのため、クラウド人気が高まることで、攻撃者の視線も自然とクラウドに向けられることになる。

ユーザーは大抵の場合、クラウドがもたらす効率性や生産性に注目する。しかし、不十分なセキュリティによりクラウド環境が攻撃を受けた場合、様々な価値が無用の長物になるかもしれない。あなたがもし、クラウドを基盤にしたビジネス未来戦略を描いているなら、最優先に考慮すべき点の一つがセキュリティだ。

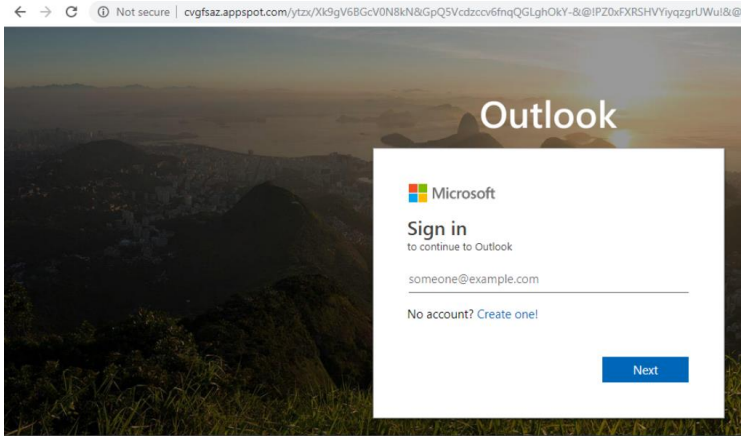
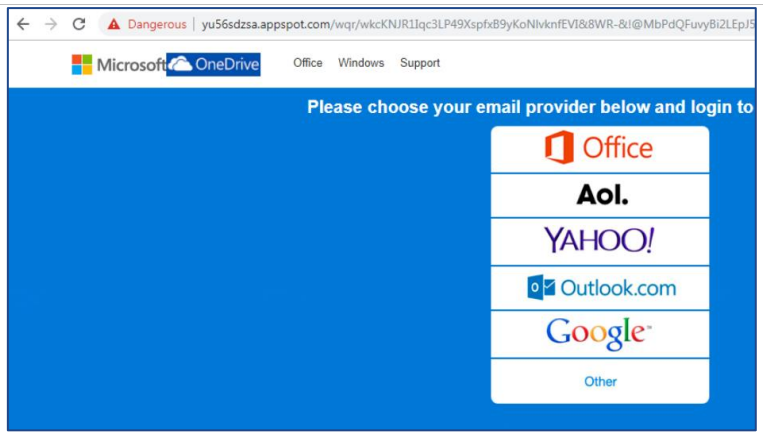
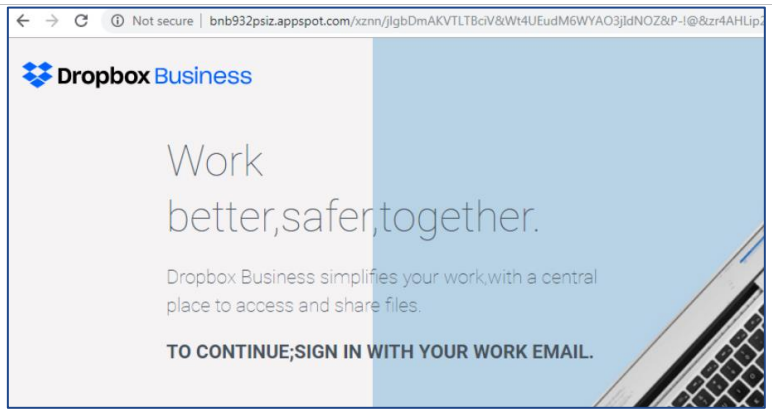
今回の記事では、クラウドセキュリティに関する7つの代表的な脅威を紹介する。

1. cloud ドメインを利用した攻撃

パブリッククラウドサービスの利用時に提供されるドメインを、フィッシングサイト、マルウェア配布、攻撃者サーバー（C&C）通信などに悪用する攻撃手法がある。また、多数の CSP (Cloud Solution Provider、クラウドソリューションプロバイダー）から、Web アプリケーションを手軽に開発・運営できる、PaaS (Platform as a Service) 形態の開発環境プラットフォームが提供されている。こうした開発環境プラットフォームに、Web アプリケーションのソースコードをアップロードすると、Web ページにアクセス可能なクラウド独自のドメインが提供される。

ここで実際の攻撃例を見てみよう。Google Cloud Platform は、サーバーレス状態で Web アプリケーションの即時配布が可能な、「App Engine」という PaaS サービスを提供している。このサービスを介して Web ソースコードをアップロードすると、「*.appspot.com」というドメインが自動で支給され、開発社はアプリケーションを簡単にテストし、配布することができる。

攻撃者は、この App Engine サービスを悪用し、下記のようなフィッシングサイトを立ち上げた。

フィッシングサイトドメイン	フィッシングサイト画面
cvgfsaz.appspot.com	 Outlook フィッシングページ
yu56sdzsa.appspot.com	 OneDrive フィッシングページ
bnb932psiz.appspot.com	 Dropbox フィッシングページ

[表1] フィッシングサイト画面

一般ユーザー側は、上位ドメイン（'*.appspot.com'）の確認だけで正常なサイトと錯覚し、個人情報を入力してしまうケースが後を絶たない。そのため、下記のように数多くのフィッシングサイトが次々と制作され、流布されているのが実状だ。

フィッシングサイトドメイン	
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

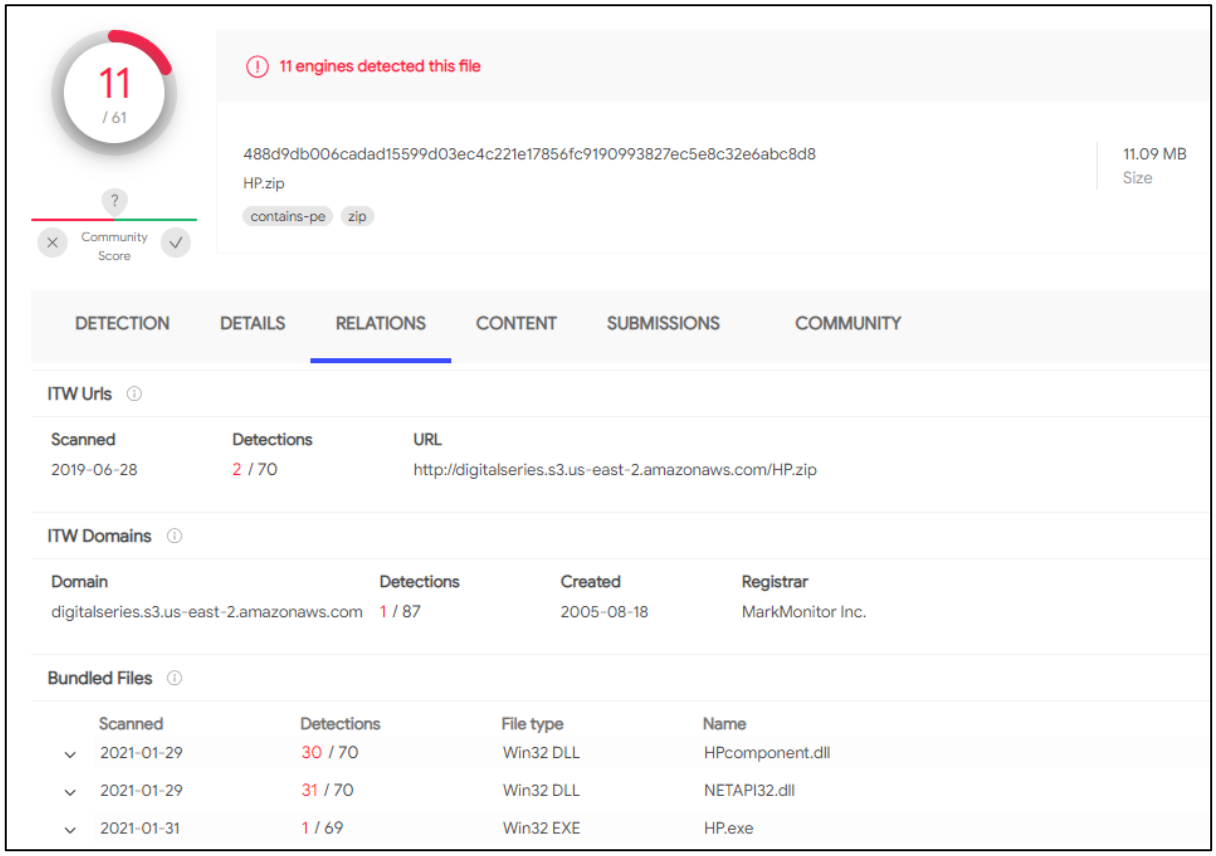
[表2] App Engine で製作されたフィッシングサイトドメインの一部

また、開発環境のプラットフォームサービスだけでなく、ストレージサービスも同様にマルウェアの配布に悪用されている。AWS (Amazon Web Service) の S3 (Amazon Simple Storage Service) や、NAVER の MYBOX サービスをはじめとするストレージサービスは、ファイルのダウンロードサービスを提供する際、ほとんどの場合で下記のような公式の下位ドメインを提供している。

ストレージサービス	URL 形式
AWS - S3	https://[バケット名].s3-[AWS リージョン].amazonaws.com/[ファイル名]
	https://s3.[AWS リージョン].amazonaws.com/[バケット名]/[ファイル名]
NAVER - MYBOX	https://file.scloud.naver.com/.fileLink/ランダム値

[表3] ストレージサービス別のURL形式

実際に流入してくる多くのマルウェアを調べてみると、クラウドストレージサービスの提供ドメインから流れ込むケースが多い。これは、ドメイン名のみで判断したユーザーが公式ドメインだと錯覚し、脅威の存在を見逃してしまうからだ。



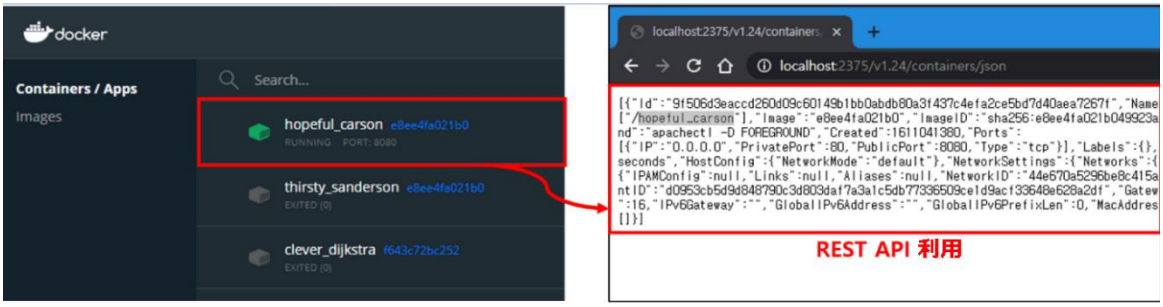
[図3] クラウドストレージサービスから流入するマルウェア - ウイルストータル (Virustotal) 画面

このように、現在様々なクラウドサービスが攻撃者によって悪用されている。そのため、防御側は事前に脅威を認知し、クラウドサービスから提供されるドメインをあらかじめ把握した上で、徹底的に調べる必要がある。

2. 脆弱な Docker & Kubernetes API へのサーバー攻撃

クラウド導入が拡大するにつれ、現在多くの IT インフラ環境が Docker (ドッカー) で構築されている。Docker とは、アプリケーションやライブラリを「コンテナ (Container)」にまとめ、サービス稼働のための隔離環境を作り出すオープンソースプラットフォームのことだ。Docker は、開発およびサーバー環境を手軽に構築・管理することができるため、多くの IT インフラに使用されている。さらに Docker は、「Docker イメージ (Image)」や「Docker コンテナ (Container)」などの管理が行える「CLI (Command Line Interface、コマンドラインインターフェイス)」も提供している。この CLI は基本的に、ローカル環境でのみ使用することが可能だ。

しかし、Docker の追加設定により、リモート環境でも「CLI」を活用することができる。それが「REST API」だ。下記のように「REST API」を利用すると、リモート環境下でも同様に Docker の管理が可能となる。

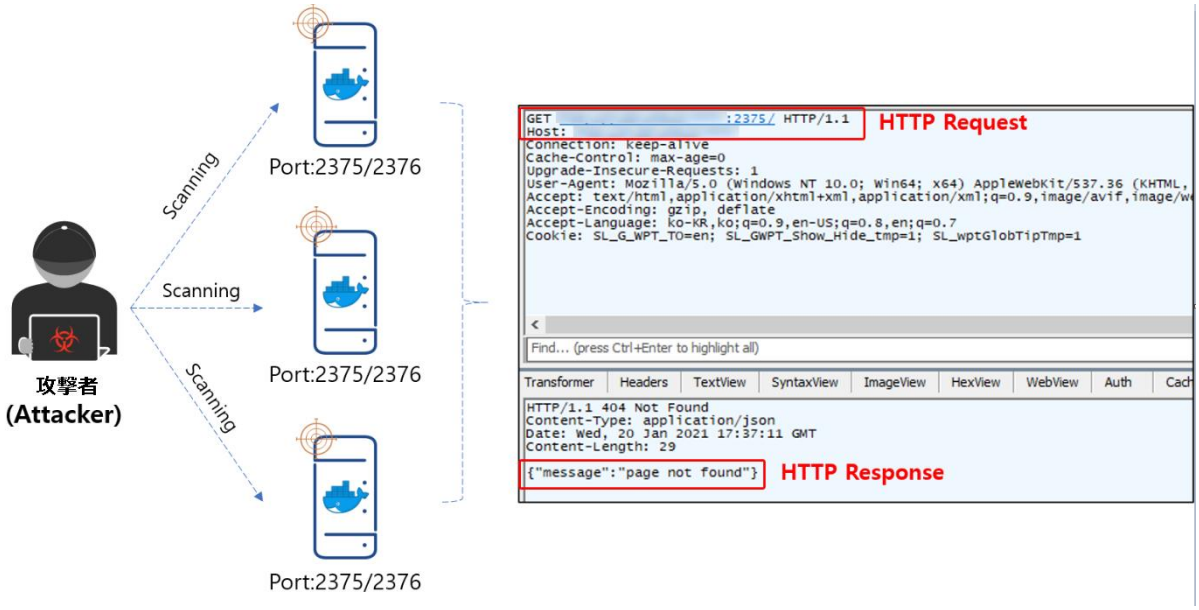


ローカル(Local)環境　　リモート(Remote)環境

[図4] REST API を利用したリモート環境下での Docker 照会

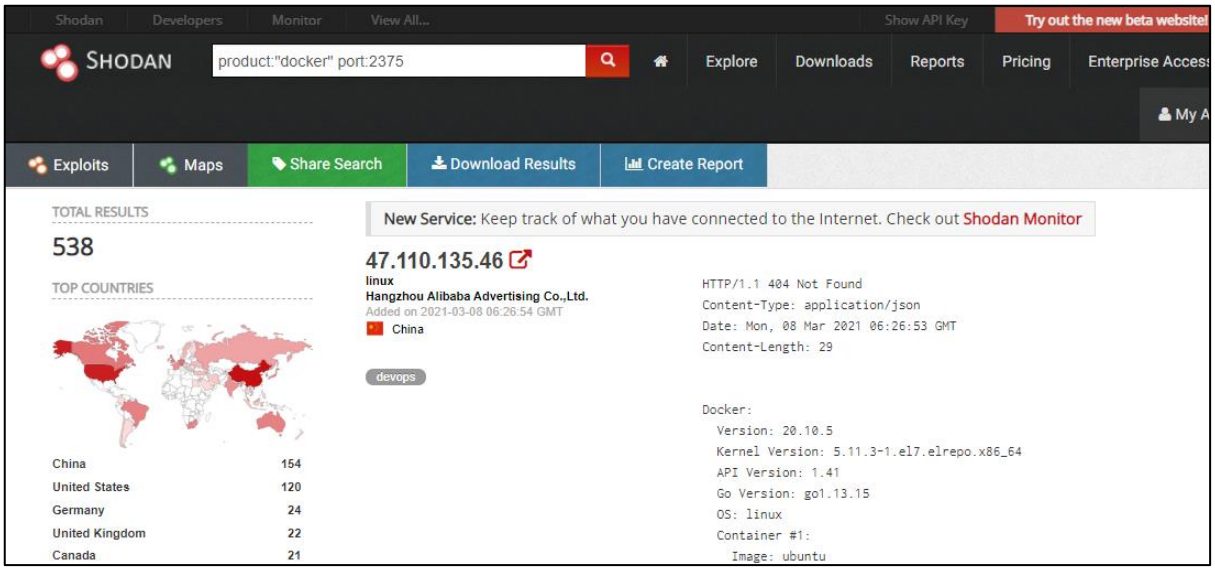
この際、Docker REST API サーバーの IP やポートが外部に露出すると、権限のない第三者も、REST API を介し Docker イメージや Docker コンテナの管理が可能になる。また、Docker REST API サーバーへのアクセス時、{“message”:”page not found”} というメッセージコードが表示されるが、この情報から、Docker REST API サーバーであるかどうか、誰でも簡単に判別することが可能だ。

攻撃者は、こうした方法で Docker REST API サーバーをスキャンした後、REST API を悪用し、Miner Docker (マイナードッカー)や DDoS Docker (ディードスドッカー) など、不正 Docker のダウンロードを行う。そして実行、もしくはシステム妨害を狙い、元々ある Docker を悪意のもと削除する。



[図5] 攻撃者による Docker REST API スキャン

また、Docker だけではなく、「Kubernetes (クバネティス)」にも REST API が存在するため、攻撃者の脅威にさらされている。Kubernetes とは、Docker の管理・運営が可能なオーケストレーション (Orchestration) ツールだ。特に、OSINT (Open Source INTelligent、オープン・ソース・インテリジェンス) ツールの一つである「Shodan (ショーダン)」では、多くの Docker および Kubernetes REST API サーバーが露出している。そのため、攻撃に悪用される危険性と常に隣りあわせだ。防御側としては、こうしたセキュリティ脅威を事前に認知し、追加の認証手続きや特定 IP の接続許可など、別途ポリシー設定を行う必要がある。

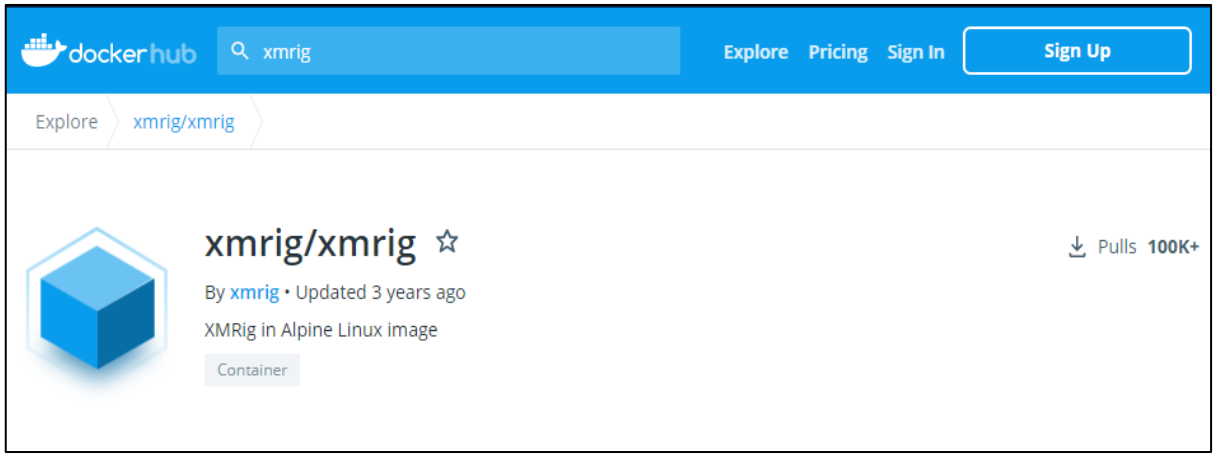


[図6] Shodan 内に露出している Docker REST API サーバー

3. Docker Hub 内の Docker イメージを悪用した脅威

近年、「Docker Hub (ドッカーハブ)」に不正 Docker イメージが流布され、多くのユーザーが被害を受けている。Docker Hub とは、ユーザーが Docker イメージを保存・配布することで、下記のように Docker イメージの形成や管理を行える、パブリックリポジトリ (Public Repository) のことだ。

攻撃者は、DDoS またはマイニングアプリケーションが含まれた不正 Docker イメージを、悪意のもと Docker Hub にアップロードする。さらにアップロードする際も、セキュリティ教育やテストなどの正当な目的、あるいは正常な Docker イメージに偽装した上でアップロードを行う。



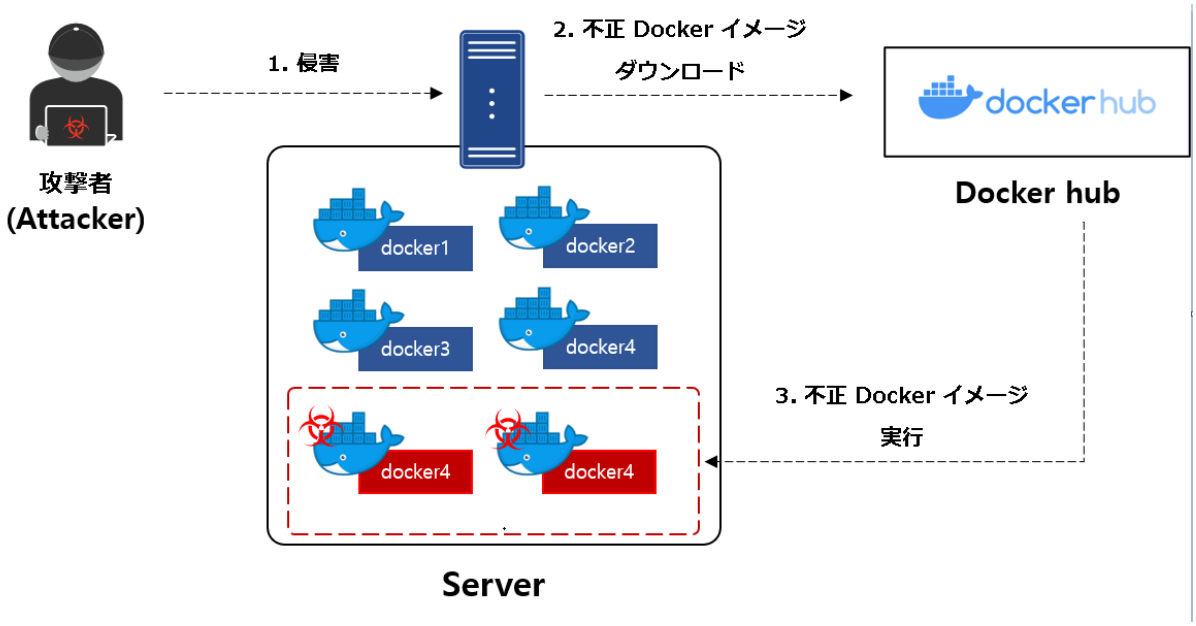
[図7] 正当な目的でアップロードされた Docker Hub 内の Docker イメージ

正当な目的でアップロードされた不正 Docker イメージを調査してみると、その用途の大部分は DDoS、もしくは仮想通貨マイニングである。

用途	正当な目的でアップロードされた不正 Docker イメージ
DDoS	douglasslow/slowhttptest:latest
	foxleon/udpflood:latest
	nxqsmfxx2/stupid:latest
	nxqsmfxx2/stupid:slowhttptest
仮想通貨マイニング	bitnn/alpine-xmrig:latest
	patsissons/xmrig:latest
	hildeteamtnt/xmrigminer:latest
	kannix/monero-miner:latest

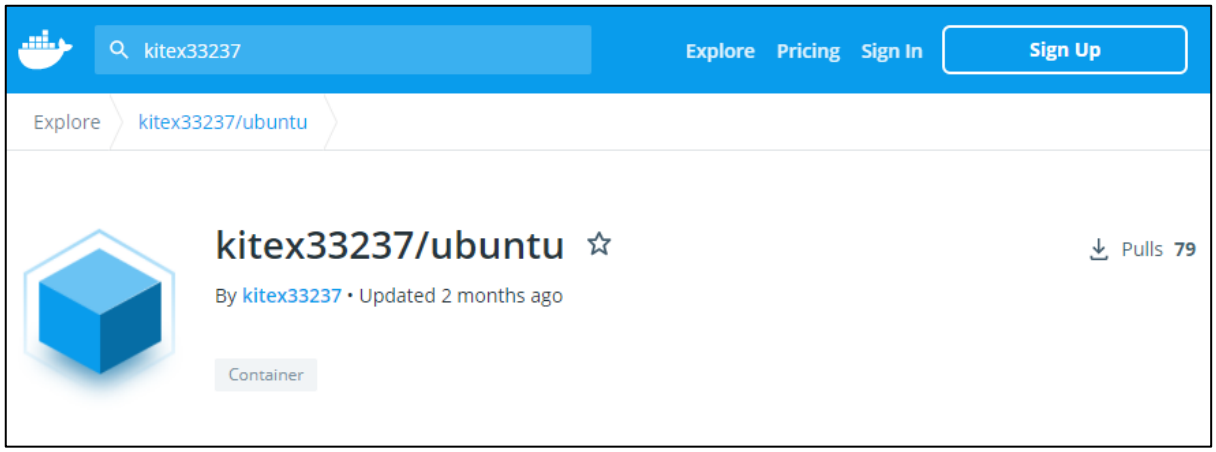
[表4] 合法的な用途でアップロードされた不正 Docker イメージリスト

本来ならば、アップロードされたイメージは正当な目的で使用される。しかし、攻撃者はほとんどのケースで「Docker サーバー」を侵害後、アップロードされたイメージをダウンロードし、DDos およびマイニングを目的として使用している。



[図8] 侵害されたサーバー内における不正 Docker イメージのダウンロードおよび実行

また、正常な Docker イメージに偽装し、アップロードを行う攻撃手法もある。



[図9] 正常な Docker イメージに偽装した不正 Docker イメージ

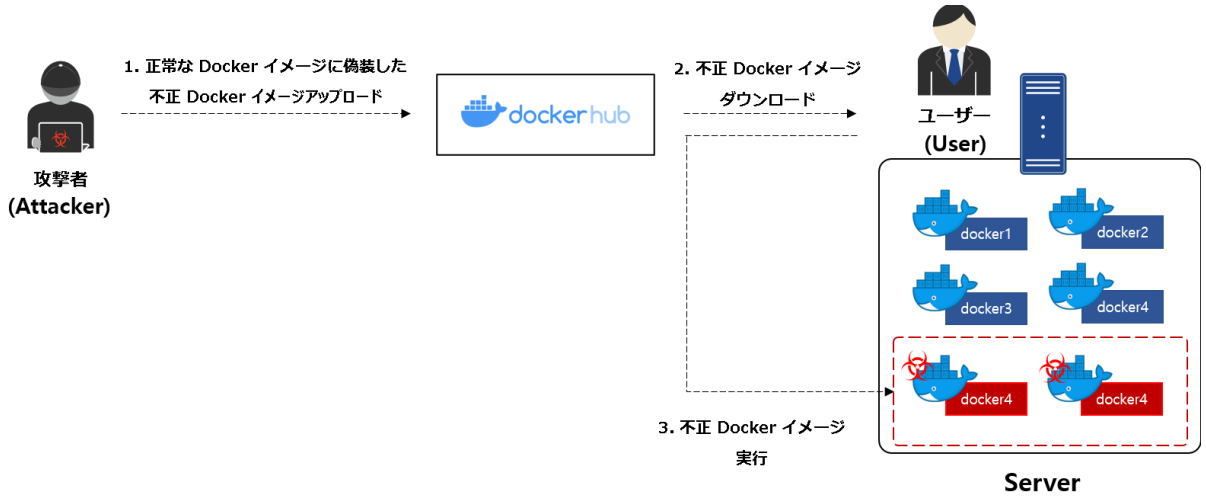
さらに下記のように、多くの不正 Docker イメージが正常な Docker イメージに偽装された状態で配布されている。

用途	正常な Docker イメージに偽装した Docker イメージ
DDoS	userubuntu1/zores:latest
仮想通貨マイニング	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

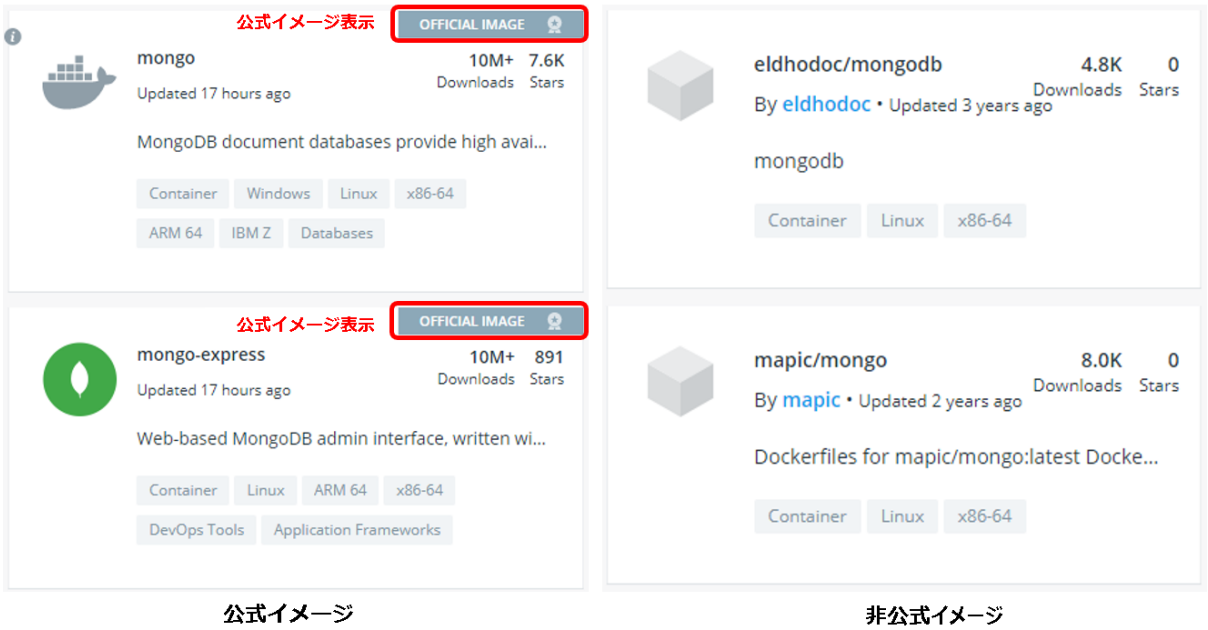
[表5] 正常な Docker イメージに偽装した不正 Docker イメージリスト

攻撃者はこれにより、仮想通貨マイニング、または DDos に使用されるコンピューティングリソースを、より素早くかつ手軽に得ることが可能となる。



[図10] 偽装された不正 Docker イメージのダウンロードおよび実行

不正 Docker イメージによる攻撃増加に伴い、今までいくつもの対応方法が提示されてきた。しかし、攻撃者は毎年様々な方法で防御網を迂回し、不正 Docker イメージをアップロードし続けている。また、不正 Docker イメージ以外にも、古いオープンソースパッケージやライブラリを使用することで、正常な状態であるにも関わらず、脆弱性をついた攻撃にさらされることもある。したがって、Docker イメージを使用する際は、信頼度の高い公式イメージをダウンロードし、イメージ内で使用されているソフトウェアの安全性に関し、使用前に必ず検査しなければならない。



[図11] 公式イメージと非公式イメージの区分

4. 設定に脆弱性のあるクラウドサービスを対象とした攻撃

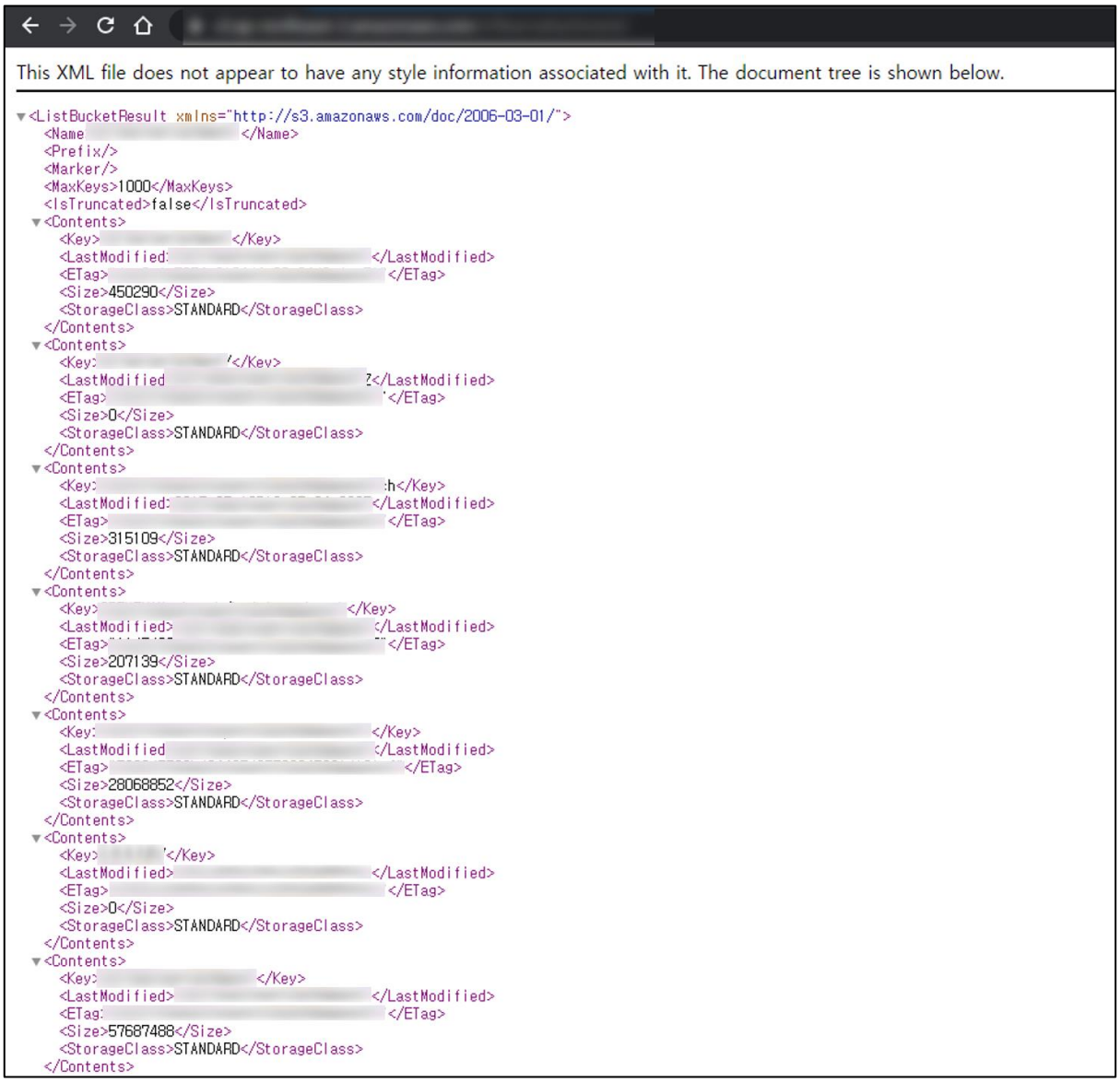
市場調査機関である「ガートナー (Gartner)」の調査結果によると、2023年までに発生するクラウドセキュリティ上の失敗事例のうち、顧客側の過失によるものがその99%以上を占める。また、2021年までのクラウド内で起こる重要情報の流出事例のうち、95%以上が管理者側の過失によるものと予測されている。実際に発生した情報流出の被害事例を見てみると、設定上に脆弱性のあるクラウドサービスにて、重要情報が度々流出していることが判明した。

日付	企業	内容
2020.03.17.	Advantage Capital Funding	AWS S3 バケット内の425GBのデータ流出
2020.02.12	JailCore	AWS S3 バケット内の36000件のデータ流出
2020.01.22.	THSuite	AWS S3 バケット内の30,000名のデータ流出
…	…	…
2019.04.03.	Facebook	AWS S3 バケット内の5億5千個の顧客データ流出
2018.02.15.	FedEx	AWS S3 バケット内の119,000個のデータ流出
2017.07.19.	Dow Jones	AWS S3 バケット内の220万名のデータ流出

[表6] クラウド内の情報流出に関する被害事例

設定に脆弱性が起こりやすいクラウドサービスには、AWS S3、Azure Blob Storage、AWS Lambda、Azure AD などが挙げられる。ストレージサービスとして多く利用されている AWS S3 の場合、「buckets.grayhatwarfare.com」のような OSINT を介し、設定に脆弱性のあるバケットを容易に見つることが可能だ。

AWS S3 バケットは、基本的に非公開モードに設定されている。しかし、管理者の過失により設定に脆弱性がある場合、AWS S3 バケットへのアクセス時、下記のようにディレクトリリスティングされ、テキストファイルや写真、動画などの「オブジェクト (Object)」に難なくアクセスし、誰でも情報を奪取することが可能だ。



[図12] 設定に脆弱性のある AWS S3 バケット

攻撃者はこのように、設定に脆弱性のあるクラウドサービスを見つけ、データ流出を狙った攻撃を仕掛けてくる。企業側は、特定のクラウドサービスを使用する前に、脆弱な設定により発生しうる脅威に対し、事前に認知しておかなければならない。

5. パブリックリポジトリ上に露出する資格証明コード

ここ最近、「CI (Continuous Integration、継続的インテグレーション)」や「CD (Continuous Delivery、継続的デリバリー)」を介した「Devops (デブオプス)」の開発が増加するにつれ、クラウドサーバーへのアクセスが必要になっている。クラウドサーバーへのアクセス時、開発コード内には必要な資格証明の一部が含まれているが、開発者がパブリックリポジトリ (Public Repository) に作業ファイルをアップロードする際、人為的な過失により資格証明ファイルをアップロードしてしまうことで、クラウドサーバーが危険にさらされるケースが発生している。

アメリカ・ノースカロライナ州立大学の研究によると、「Github (ギットハブ)」内のファイルをスキャンした結果、10万件のリポジトリ内に、暗号化キーおよび API トークンのような資格証明ファイルが露出していることが判明した。事実 Github では、下記のように資格証明ファイルが頻繁に露出している。

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

[図13] Github 内に露出する AWS 資格証明内容

市場調査機関の「Comparitech」によると、Github にハニーポット (HoneyPot) 目的で AWS 資格証明ファイルをアップロードした結果、使用を試みる記録が1分足らずで1000回に達したという。これは、開発者が Github にコードを誤ってアップロードし、その後削除したとしても、削除する前に攻撃者側に奪取される可能性を示唆している。

Github のリポジトリだけでなく、Docker Hub 上のリポジトリに Docker イメージをアップロードする際にも、資格証明ファイルを一緒にアップロードしてしまい、クラウドサーバーが脅威にさらされるケースもある。尚、Docker イメージ内に資格証明内容が含まれる場所は、大きく3つに分類できる。1つ目は「ソースコード」だ。Docker イメージを実行し、クラウドサーバーと連動させるため、サーバーに対する資格証明コードが下記のように含まれている場合がある。

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

const dbRoute = "[redacted]
// const dbRoute = "[redacted]";
//connection with database.

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

[図14] ソースコード内に含まれている資格証明コード

2つ目は「環境変数」だ。 Docker イメージをビルドする過程で、環境変数内にも資格証明コードが含まれることがある。

```
C:¥Users¥User>docker run [redacted] printenv
MYSQL_PASSWORD=[redacted]
HOSTNAME=[redacted]
MYSQL_DATABASE=[redacted]
MYSQL_ROOT_PASSWORD=[redacted]
PWD=/
HOME=/root
MYSQL_MAJOR=5.7
GOSU_VERSION=1.7
MYSQL_USER=[redacted]
MYSQL_VERSION=5.7.26-1debian9
SHLVL=0
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

[図15] 環境変数内に含まれる資格証明コード

最後は「資格証明ファイル」である。下記の「Dockerfile（ドッカーファイル」に関する内容を見てみると、資格証明ファイルを保存・使用するように設計されている。これは、Docker イメージ内に資格証明ファイルが含まれていることを意味する。このような Dockerfile を介し Docker イメージを製作することで、攻撃者の脅威にさらされる場合がある。

```
1 FROM ubuntu as intermediate
2
3 WORKDIR /app
4 COPY secret/key /tmp/
5 RUN scp -i /tmp/key build@acme/files .
6
7 FROM ubuntu
8 WORKDIR /app
9 COPY --from=intermediate /app .
```

[図16] 資格証明ファイルを使用した Dockerfile

Dockerfile で頻繁に使用される資格証明ファイルは、以下の通りである。

サービス	資格証明ファイル
AWS	.aws/credentials
	.aws/config
SSH	.ssh/config
Docker	.docker/config.json
Kubernetes	.kube/config

[表7] サービス別の資格証明ファイル

このように、パブリックリポジトリ内ではソースコード、また Docker イメージ内ではクラウドサーバーの資格証明ファイルが発見されるケースが起きている。したがって、パブリックリポジトリにアップロードする際は、資格証明に関するコードやファイルの有無を事前に検査する必要がある。

6. Linux プラットフォームにおけるマルウェア

クラウド技術の発展により、現在多くのサーバーが Docker 環境に変わりつつある。それにより、自然と Linux（リナックス）環境も活性化しており、Linux を狙ったマルウェアもまた増加傾向にある。攻撃者は、Linux へのマルウェアを新たに製作するのではなく、従来の Windows マルウェアを Linux 用にポーティング(porting) 後、配布していることがわかっている。下記の表は、「BIFROSE」という RAT（Remote Administration Tool）マルウェアに関する情報だ。BIFROSE は以前、Windows を狙ったマルウェアとの認識が強かったが、時代の変化に伴い、Linux を狙う BIFROSE も発見された。

ファイル名	説明
ファイルサイズ	596.07 KB (610376 bytes)
MD5	4a4ea45cb850e4decd5fdb23909ea728
SHA1	5d8b228e3014b4eb579e380b3a1113dd8c0d999a
主要機能および特徴	BIFROSE(ELF)
アンラボの診断名	Linux/Bifrose.610376

[表8] BIFROSE ファイル情報

Windows 基盤の BIFROSE と Linux 基盤の BIFROSE に関し、PC から収集した情報形式を実際に比較してみると、類似した形式で情報を収集していることが確認できる。このことから、攻撃者は Windows 用に製作された従来の BIFROSE ソースコードを、Linux 用にポーティング・製作していることが推察できる。

プラットフォーム種類	収集した情報形式
Windows 基盤 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|
Linux 基盤 BIFROSE	<victim IP>|unix|<hostname>|<username> |5.0.0.0|0|1|1|0|575|0|0|0|0|None||||

[表9] Windows および Linux 基盤の BIFROSE マルウェア比較

さらに BIFROSE だけにとどまらず、様々なマルウェアが下記のように Linux 用にポーティング後配布されている。

マルウェア名	変更前のプラットフォーム	変更後のプラットフォーム
GravityRAT	Windows	Linux
PLEAD	Windows	Linux
RansomExx	Windows	Linux

TrickBot Anchor	Windows	Linux

[表10] 多様なプラットフォームにポーティング・配布されたマルウェア (出典：intego、SECURELIST)

マルウェアがここまで簡単にポーティングされる背景として、「クロスプラットフォーム (Cross Platform)」に容易な言語を使用していることが理由の1つに挙げられる。クロスプラットフォームに容易な言語としては、Go 言語、.NET Core、PowerShell Core などがある。特に、1つのソースコードで Windows、Linux、MacOS など、様々なプラットフォームのプログラムを簡単に製作できる Go 言語は、クロスプラットフォームの代表的なサポート言語となっている。

近年、Linux マルウェアのうち、Go 言語で製作されたマルウェアの割合が増加してきている。以下は、実際に発見された Go 言語によるマルウェアリストの一部だ。互いに別のプラットフォームで動作するマルウェアであっても、ポーティングを経て、新たなプラットフォームをターゲットに再配布されることがある。そのため、発見されたマルウェアがクロスプラットフォームに容易な言語で製作されていた場合、徹底したモニタリングが必要となってくる。

Go 言語により製作されたマルウェアリスト	
GDRat	Glupteba
Snake Ransomware	GoBot2
RansomExx	HERCULRES
CHAOS	GoBrut
ARCANUS	hershell
EGESPLOIT	Mauri870 Ransomware
r2r2	RobbinHood Ransomware
Rakos	TrumpHead Ransomware
telegrab	gorsh

[表11] Go 言語により製作されたマルウェアリスト

7. Docker Escape

一般的に、Docker 環境下ではマルウェアに感染しても、Docker コンテナが互いに隔離された環境で動作するため、他の Docker コンテナ、またはホスト内まで侵犯することは不可能だ。

しかし、この「Docker Escape」に脆弱性が存在する場合、隔離された Docker 環境を抜け出し、他の Docker コンテナやホスト内まで侵入することができてしまう。下記のように、Docker Escape における脆弱性のほとんどは、コンテナ作成や実行のために設計された CLI ツールである「runC」から発生していた。

脆弱性コード	内容
CVE-2019-19921	‘runC’ を利用した ‘Docker Escape’ の脆弱性
CVE-2019-5736	‘runC’ を利用した ‘Docker Escape’ の脆弱性
CVE-2016-5195	‘runC’ を利用した ‘Docker Escape’ の脆弱性
CVE-2016-9962	‘runC’ を利用した ‘Docker Escape’ の脆弱性
CVE-2016-3697	‘runC’ を利用した ‘Docker Escape’ の脆弱性

[表12] runC を利用した Docker Escape の脆弱性

この「runC」以外にも、Docker が提供する機能を介し、Docker Escape の脆弱性は今もなお発生し続けている。実際、Docker 基盤のクラウドサービスを提供する Microsoft Azure（アジュール）の「Azure Functions（アジュール ファンクションズ）」や、Google Cloud の「Google Cloud Shell」のようなサービスで、Docker Escape の脆弱性が発見される事例も報告されている。よって、Docker ユーザーや Docker を利用したサービスを提供している企業側は、こうした脅威へのモニタリングが常に求められている。

http://jp.ahnlab.com/site/main.do
http://global.ahnlab.com/site/main.do
http://www.ahnlab.com/kr/site/main.do

アンラボとは

株式会社アンラボは、業界をリードする情報セキュリティソリューションの開発会社です。
1995年から弊社では情報セキュリティ分野におけるイノベーターとして最先端技術と高品質のサービスをご提供できるように努力を傾けてまいりました。今後もお客様のビジネス継続性をお守りし、安心できるIT環境づくりに貢献しながらセキュリティ業界の先駆者になれるよう邁進してまいります。
アンラボはデスクトップおよびサーバー、携帯電話、オンライントランザクション、ネットワークアプライアンスなど多岐にわたる総合セキュリティ製品のラインナップを揃えております。どの製品も世界トップクラスのセキュリティレベルを誇り、グローバル向けコンサルタントサービスを含む包括的なセキュリティサービスをお届け致します。

AhnLab

〒108-0014 東京都港区芝4丁目13- 2 田町フロントビル3階 | TEL: 03-6453-8315（代）
© 2021 AhnLab, Inc. All rights reserved.