

【ベータ版】 ネットワークセキュリティ ポリシーツール



本文の内容は、【ベータ版】ネットワークセキュリティポリシーツールのドキュメント (https://docs.sysdig.com/en/-beta--network-security-policy-tool.html) 2020年11月20日時点を元に日本 語に翻訳・再構成した内容となっております。

【ベータ版】ネットワークセキュリティポリシーツール	4
ベネフィット	4
ネットワークセキュリティポリシーツールの使用	5
前提条件	5
スコープの設定	5
IngressとEgressの管理	6
トポロジーの可視化を使用する	7
生成されたポリシーのレビューとダウンロード	7
データの処理方法を理解する	8
使用例	9
ケース1:指定されたIngress/Egress通信のみを許可する	9
ケース2: プロキシ静的IPへのアクセスを許可する	9
ケース3:ネームスペース内でのみ通信を許可する	10
ケース4: 指定されたネームスペースへのアクセスを許可し、Egressのみを許可する	10
ケース 5: デプロイメントがリラベルされた場合のアクセス許可	10
トラブルシューティング	11
エラーメッセージ: Namespaces without labels(ラベルのないネームスペース)	11
エラーメッセージ:Cluster subnet is incomplete(クラスターサブネットが不完全です)	12





【ベータ版】ネットワークセキュリティポリシー ツール

デフォルトでは、Kubernetes クラスター内のすべてのポッドは、制限なく相互に通信できます。 Kubernetes ネットワークポリシーを使用すると、マイクロサービスアプリケーションを互いに分離し て、ブラスト半径を制限し、全体的なセキュリティ姿勢を向上させることができます。

Network Security Policyツールを使用すると、Sysdig Secure内でKubernetesネットワークポリシーを作成し、微調整することができます。これを使用して、観察されたネットワークトラフィックと追加の ユーザー評価の両方を組み込んだ、ワークロードを保護するための「最小特権」ポリシーを生成しま す。ファイアウォールやインライン接続プロキシを導入することはなく、代わりにKubernetesに既に 存在する機能を活用します。

ベネフィット

このツールは、マイクロサービス通信を深く洞察し、観測されたトラフィックに基づいてネットワーク・ポリシーを自動的に記述することで時間と労力を節約し、適切なポリシーをオーサリングするためのガイドを提供します。

具体的には、次のような機能を提供します:

- アプリケーションとサービス間のネットワーク・トラフィックを即座に可視化し、通信を識別するのに役立つ視覚的なトポロジー・マップを提供します。
- ベースライン・ネットワーク・ポリシーは、必要な宣言状態に合わせて直接調整・修正することができます。
- ネットワーク通信のベースラインに基づいた KNP の自動生成とユーザー定義の調整。
- 最小特権:KNPは許可のみのモデルに従うため、明示的に許可されていない通信は禁止されま す。
- Kubernetesのコントロールプレーンに委任された強制力により、追加の計測やホストのネット ワーク設定の直接の改ざんを回避することができます。





ネットワークセキュリティポリシーツールの使用

ネットワーク セキュリティ ポリシー ツールを使用するには、以下の基本的な手順に従います。

- 1. 環境が前提条件を満たしていることを確認します。
- 2. スコープを設定します(どの期間にどのエンティティを分析するか)。
- 3. Ingress/Egress テーブルを確認し、検出された通信を必要に応じて編集します。
- 4. トポロジーマップですべてを視覚的に確認します。
- 5. 生成されたポリシーをクリックして、結果のファイルをダウンロードします。



Sysdigエージェントバージョン10.7以上

サポートされているOrchestratorディストリビューションとCNIプラグイン:

- Calicoを使用したバニラKubernetes (kops, kube-admin)
- OVSを使用したOpenShift 4.x
- Calicoを使用したGoogle GKE
- Calicoを使用したAmazon EKS
- Calicoを使用したRancher Kubernetes

Sysdig Secure アカウントでこの機能を有効にするには、Sysdig サポートにお問い合わせください。

スコープの設定

まず、通信を集約するKubernetesエンティティと時間枠を定義します。

- 1. Sysdig Secure UI で、左メニューから Policies >Network Security Policies を選択します。
- 2. ドロップダウンメニューから [Cluster and Namespace] を選択します。
- 3. ポリシーを作成する Kubernetes エンティティの種類を選択します。
 - Service
 - Deployment





- Daemonset
- Stateful Set
- Job
- 4. タイムスパン(タイムスパン)を選択します。インターフェースは、そのKubernetesエンティティの Ingress / Egressテーブルとタイムフレームを表示します。

IngressとEgressの管理

ingress/egress テーブルは、選択したエンティティ(ポッドの所有者)と時間帯の観測された通信の詳細を示しています。

粒度の高い割り当てとグローバルな割り当て:ポリシーに含める/除外する行を細かく選択したり、ドロップダウンのグローバルルールオプションを使用して一般的なルールを設定したりすることができます。

Network Security Policie	ES BETA					
Cluster is cluster_large ~	+ Names	pace is sysdig	~			
svc Service v	Ingres	s Egress	Generated Policy	Topology		
Q Search	(i) Belov	w you see the netw	rork connections we have d	etected for sysdigcloud-anchore-core organized by i L	earn more	
redis	-∋ IN-	CLUSTER ENTITIE	S General ingress rule	Select ingress rule		
svedigeloud-anchore-core	Allow	CLIENT SIDE				SERVER SIDE
sysulgeodd anenore core		Namespace 🔺	Namespace labels	Controlled by	Pod controller labels	Listening process and port
sysdigcloud-api		sysdig	istio-injection=disabled sysdigcloud-app=true	Deployment: sysdigcloud-anchore-worker		twistd:8083
sysdigcloud-elasticsearch		sysdig	istio-injection=disabled sysdigcloud-app=true	Deployment: sysdigoloud-scanning-alertmgr		twistd:8083
sysdigcloud-events-api-grpc		sysdig	istio-injection=disabled sysdigcloud-app=true	Deployment: sysdigcloud-scanning-analysiscollector		twistd:8083
sysdigcloud-events-dispatc		sysdig	sysdigcloud-app=true istio-injection=disabled	Deployment: sysdigcloud-scanning-api		twistd:8083
			() 8:39:41 am - 8:39	:40 pm Last 12 hours 3H 12H 1D 3D 7	D	

検出された通信を確認して編集するには、Ingress または Egress を選択します。





- 1. 上記のようにスコープを選択します。
- 2. 必要に応じて、許可された通信を編集します。
 - a. 許可された通信の行の選択/選択解除、または
 - b. 一般的なIngress/Egressルールの選択: すべてをブロックする、ネームスペース内のすべて を許可する、またはすべてを許可する。
- 3. 他のテーブルで繰り返し、トポロジーの確認および/またはポリシーの生成に進みます。

トポロジーの可視化を使用する

トポロジービューを使用して、これが希望のポリシーなのか、何か変更すべきことがあるのかを視覚 的に検証します。トポロジービューは、ポッドの所有者、リスニングポート、サービス、ラベルなど の高レベルのKubernetesメタデータを表示します。

このポリシーを適用すると決めた場合に許可されない通信は赤で色分けされています。



生成されたポリシーのレビューとダウンロード

ルールとコミュニケーションラインに満足したら、Generated Policyタブをクリックするだけで、瞬時 に生成されたファイルを取得できます。





生成されたYAMLファイルを確認して、ブラウザにダウンロードします。

cluster_large	 + Namespace is sysdig 	
Service V	Ingress Egress Generated Policy Topology	
X Search	Network Policy Name: generated-network-policy	🚯 Download Policy
redis	1 apiVersion: networking.k8s.io/v1 2 kind: NetworkPolicy	
sysdigcloud-anchore-core	3 metadata: 4 name: generated-network-policy	
sysdigcloud-api	5 namespace: sysdig 6 spec:	
sysdigcloud-cassandra	7 ingress: 8 - from:	
sysdigcloud-elasticsearch	9 - namespaceSelector: 10 matchLabels:	
sysdigcloud-events-api-grpc	11 istio-injection: disabled 12 sysdigcloud-app: 'true'	

データの処理方法を理解する

アグリゲーション:通信はKubernetesメタデータを使用して集約され、ポリシー作成に関係のないエントリが追加されないようにします。たとえば、デプロイメントAのポッドAがデプロイメントBのポッドBと複数回通信した場合、インターフェースには1つのエントリしか表示されません。

未解決のIP:一部の通信では、エンドポイントの1つをKubernetesメタデータで解決できない場合があ ります。たとえば、マイクロサービスが外部のWebサーバーと通信している場合、その外部IPはクラ スター内のKubernetesメタデータに関連付けられていません。それでもUIはこれらのエンティティを "unresolved IPs"として表示します。未解決IPはデフォルトではKubernetesネットワークポリシーから 除外されますが、ingress/egressインターフェースを介して手動で追加することができます。

クラスター CIDR::未解決 IP はリスト化され、"internal"(クラスター内)、"external"(クラスター 外)、または "unknown"(サブネット情報が不完全)に分類されます。未解決の場合、Sysdigはエラー メッセージを表示して解決を促します。エラーメッセージを参照してください:クラスターのサブ ネットが不完全です。未解決のIPでUIが乱雑になるのを避けるために、ツールは内部、外部、未知のIP をそれぞれ最大5個まで保持します。





使用例

いずれの場合も、エージェントが情報を収集できるようにするために、少なくとも12時間はアプリ ケーションを稼働させたままにすることから始めます。

ケース1: 指定されたIngress/Egress通信のみを許可する

開発者として、サービス/デプロイメントが明示的に許可したIngressとEgressのネットワーク通信の確 立のみを許可するKubernetesネットワークポリシーを作成したいとします。

アプリケーションのクラスターネームスペースとデプロイメントを選択します。

事前に計算されたIngressとEgressのテーブルが表示されているはずです。このアプリケーション は、たIngressまたはEgressのために外部IPと通信しないことがわかっているため、未解決のIPは表 示されません。トポロジーマップにも同じ情報が表示されます。

- ルールの変更:アプリケーションが通信しているサービスの1つが廃止されたと判断しました。
 Egressテーブルのその行のチェックを外します。
- トポロジーマップを確認:通信はまだ存在していますが、現在の Kubernetes ネットワーク ポリシー (KNP)を使用して禁止されていることを意味する赤で描画されていることがわかります。
- **生成されたポリシーコードを確認**:プランに沿っていることを確認します。
 - raw IPではingress/egressしない
 - 明示的に除外したサービスのエントリがない
- **生成されたポリシーをダウンロード**し、Kubernetes環境にアップロードします。
- アプリケーションがトポロジーで黒くマークされ、テーブルでチェックされたサービスとしか通信できないことを確認します。その後、ポリシーを生成してダウンロードして適用します。

ケース2: プロキシ静的IPへのアクセスを許可する

開発者として、アプリケーションが静的 IP のプロキシを使用していることを知っていて、アプリケー ションがプロキシにアクセスできるようにするためのポリシーを設定したいと思います。

- インターフェイスの egress セクションのプロキシ IP を参照してください。
- Allow Egress to IP mask を使用して、特にそれらの IP を許可するための手動ルールを作成します。





- Ingress/Egressテーブルの他のすべてのエントリーの選択を解除します。
- トポロジーマップを見て、これらの外部IPへの通信のみが黒でマークされ、他のサービス/デプロ イメントとの通信は赤でマークされていることを確認します。
- 生成されたKubernetesネットワークポリシーをダウンロードして適用します。

ケース3:ネームスペース内でのみ通信を許可する アプリケーションは、ネームスペース内でのみ通信する必要があります。

- 一般的なルールを使用して、ネームスペース内へのIngressを許可します。
- 一般的なルールを使用して、ネームスペース内のEgressを許可します。
- ポリシーを生成し、特定のサービス/デプロイメントを指定せずに、ネームスペース内のすべてが 許可されていることを確認してから、ポリシーを適用します。

ケース4: 指定されたネームスペースへのアクセスを許可し、Egressのみを許可する

アプリケーションデプロイメントAは、別のネームスペースにあるデプロイメントBのアプリケー ションとのみ通信します。通信に必要なのはIngressトラフィックだけで、その通信に必要なIngressト ラフィックはありません。

- Ingress テーブルが、Kubernetes エンティティと生の IP の両方で空であることを確認します。
- Egress テーブルに記載されている通信がデプロイメントBとの通信のみであることを確認します。
- 自動生成されたポリシーをダウンロードして適用し、検証してください。
- アプリケーションは、Aのネームスペース内の他のエンティティと通信できません。
- アプリケーションは、他のエンティティを解決するためにクラスタDNSサーバに連絡することができます。

ケース 5: デプロイメントがリラベルされた場合のアクセス許可





開発者として、サービス/デプロイメントが明示的に許可したIngress/Egressネットワーク通信の確立の みを許可するポリシーを作成したいと考えており、変更を行う必要があります。

アプリケーションを数時間実行した後、このポリシーに関係するすべてのネームスペースにタグ
 を付けていないことに気がつきました。

ビューの上部に「you need to assign labels to this namespace」というメッセージが表示されます。

- 異なるビューで状況を確認します:
 - 生成されたポリシーには、その通信のためのエントリがないはずです。
 - トポロジーマップは、赤線で接続していることを示す必要があります。
- 抜けていたネームスペースにラベルを貼り付けます。しばらくすると、更新された情報が行に表示されます。
- 接続を適切にホワイトリスト化します。
- ポリシーを生成してダウンロードし、適用します。

トラブルシューティング

よくあるエラーメッセージを解決するためのヒント:

エラーメッセージ: Namespaces without labels(ラベルのないネームスペー ス)

問題点: Ingress/Egressルールを定義するためのKNPでは、ネームスペースにラベルを付ける必要があります。対象の通信でラベル付けされていないネームスペースが検出されると、UIに「Namespaces without labels」というエラーメッセージが表示されます:





KUBERNETES Network Security Po	licies 🖭	A				*	
Cluster is default	✓ + N	amespace is artifactory	~				
Svc Service	~ <u>In</u>	i gress Egress G	enerated Policy Topology				
Q Search	Communications to/from namespaces without labels cannot be added to the network policy. Please label those namespaces.						
artifactory-oss-artifactory IN-CLUSTER ENTITIES General ingress rules Select ingress rule 							
	All	ow CLIENT SIDE				SERVER SIDE	
		Namespace 🔺	Namespace labels	Controlled by	Pod controller labels	Listening process and port	
		artifactory		CronJob; foobar	release=artifactory-oss component=nginx app=artifactory	jf-router:8082	
	Ð	UNRESOLVED IPs	ALLOW ingress from IPs/mask	e.g. 8.8.8/32			
				o data was found wi uster, namespace ar	th the selected nd pod owner		
			Nov 17, 6:09:41 pm - Nov 18,	6:09:41 pm Last 1 day 3	H 12H 1D 3D 7D		

解決方法:該当するネームスペースにラベルを割り当て、システムの自動検出が追いつくまで数分待 ちます。

エラーメッセージ: Cluster subnet is incomplete(クラスターサブネットが不完 全です)

問題点:未解決のIPをクラスター内かクラスター外かに分類するために、エージェントはどのCIDR範囲 がクラスターに属するかを知る必要があります。デフォルトでは、エージェントは、kube-apiserverプ ロセスとkube-controller-managerプロセスのコマンドライン引数を調べることで範囲を発見しようと します。

クラスターサブネットを自動検出できない場合は、「cluster subnet is incomplete」というエラーメッ セージがUIに表示されます。





Je v	Network Security Policies						
SECURE	Cluster is demo-kube-gke 🗸	+ Namespace is jenkins 🗸					
©verview	Service V	Ingress Egress Generated Policy Topolog	JУ				
	Q Search	Below you see the network connections we have detected for	or Jenkins organized by i Learn more				
Image Scanning	ienkins	General egress rules Set	ect egress rule				
D	,						
Compliance		(i)	No data was found with the selected				
Policies			cluster, normespace and pod owner				
Ē		UNRESOLVED IPs ALLOW egress to IPs/mask					
Events		A For some communications, cluster subnet list is incomplete.	IPs not mapping to known subnets are marked as unknown				
کی Activity Audit	*	Client process name	Destination A	Address and port			
0		java	unknown	10.15.240.1:443			
Captures		java	unknown	52.202.51.185:443			
Get Started							

解決方法:エージェントのconfigmapに以下を追加して、範囲を明示的に指定してください:

network_topology:

cluster_cidr: <A.B.C.D/MASK>

service_cidr: <E.F.G.H/MASK>

まれに、デフォルトのkube-apiserver、kube-controller-managerプロセス以外のプロセスでCIDR範囲 を探すようにエージェントを設定する必要があるかもしれません。その場合は、エージェントの configmapに以下のvvetoを追加してください。

network_topology:

```
pod_prefix_for_cidr_retrieval:
```

[<PROCESS_NAME>, <PROCESS_NAME>]

