Systems Manager セッションマネージャを使ったら SSH 管理不要になった

AWS SSH devops session-manager SystemsManager

はじめに

こんにちは、しょいみんです。
今日は、AWS Summit Tokyo 2019 のセッション 「AWS Systems Manager 徹底活用 ～エンタープライズのユースケースから～」
の講演を聞いて前から導入してみたかった「セッションマネージャ」を使った所感と導入方法を記述していきます。

結論

小規模なシステムや、ちょっとしたAWS上の**開発環境**などわざわざ Active Directory を通したユーザ管理をする必要もないような場合だと簡単に導入できる上便利。

ユーザのログイン履歴と作業コマンドなどすべてログに取らないといけないような規約がある会社でも**ログの収集・分析**や**通知の連携**、**レポートの作成**などを CloudWatch や SNS QuickSight などと連携させて作り込めば使えそう。

一方、大規模で**複雑なユーザ管理**は **IAM ユーザ**と **IAM ポリシー**、**EC2サーバのタグ**管理が肝。

Systems Manager とは

Systems Manager は AWS 公式ページでは下記のように紹介されています。

> AWS Systems Manager は、AWS でご利用のインフラストラクチャを可視化し、制御するためのサービスです。Systems Manager を使用すると、統合ユーザーインターフェイスで AWS のさまざまなサービスの運用データを確認でき、AWS リソース全体に関わる運用タスクを自動化できます。Systems Manager では、Amazon EC2 インスタンス、Amazon S3 バケット、Amazon RDS インスタンスなど

> のリソースをアプリケーションごとにグループ化し、運用データを表示できます。これにより、さまざまなリソースグループのモニタリングやトラブルシューティングを迅速に行うことができます。また、リソースとアプリケーションの管理を簡素化することも可能です。運用上の問題の検出と解決に要する時間が短縮され、大規模なインフラストラクチャでも安全に運用、管理できます。

AWS公式ページ： https://aws.amazon.com/jp/systems-manager/

簡単に言うと Systems Manager はサーバの運用周りの自動化に特化したサービスで下記のような機能が利用できます。

・**オートメーション**
　インスタンスのパッチ適応、AMIの自動作成
・**メンテナンスウィンドウ**
　スケジュール化されたタスクを cron 形式で実行、監視、記録
・**コンプライアンス**
　オンプレまたは EC2 サーバに対して社内コンプライアンス

データ(ミドルウェアの禁止バージョンなど)を作成しコンプライアンスに非準拠のサーバを特定

・インベントリ

　インストール済みアプリケーション、ネットワーク設定その他のシステムプロパティに関するメタデータを収集、監視、追跡

・パラメータストア

　オンプレ、EC2 サーバや Lambda などのAWSサービスの環境変数を一括管理、暗号化

・ランコマンド

　グルーピングされたオンプレ、EC2サーバに対し SSH 接続無しでコマンドを実行

などなど、他にも十数種類の運用自動化を補助するような機能が備わっていますのでぜひ気になった方は他の機能も調べてみてください。

AWS SystemsManager公式：https://aws.amazon.com/jp/systems-manager/

AWS Summit Tokyo 2019 AWS 公式セッションで企業のユースケースについて講演されてたのでその内容をクラスメソッドさんが上げてるのでそちらもぜひ。

クラスメソッド：https://dev.classmethod.jp/cloud/aws/aws-summit-2019-report-a1-04/

今回は簡単に導入できるセッションマネージャについてお話します。他のサービスもまた別の記事で投稿するかも。

セッションマネージャの機能

セッションマネージャは SSM エージェントを通してサーバのアクセスを制御する機能です。SSMエージェントを通したアクセスは、SSHポートの開放も SSH ユーザの作成もいらずオンプレ、EC2 サーバ問わず AWS マネジメントコンソール上からアクセスすることができます。

セッションマネージャを利用するメリット

SSH 管理不要のサーバアクセス

SSH を利用しないので Active Directory のユーザ登録、サーバ内の SSH ユーザ作成をする必要がありません！！
鍵の共有や鍵の受け渡しも必要なくなるのメンテナンスフリーでサーバ環境が運用できます。
また、SSM エージェントを通してコマンド通信を行っているため SSH ポート開放もいらなくなります。

一元管理されたアクセス制御

セッションマネージャでは SSH を使わない代わりにユーザのアクセス制御を IAM ユーザで行います。
IAM ユーザに紐付いている IAM ポリシーでアクセスできるインスタンスを制御することができます。

タグによる EC2 インスタンスアクセス制御

EC2 インスタンスに付与されているタグによってアクセスを制御できます。

Condition に複数条件を指定することもできるので複雑な制御もできそうです。

下記 yaml は EC2 タグに **「Key: Step」 「value: develop」** がついたインスタンスに対してのみセッションが開始できるポリシーです。

IAMpolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/Step": [
                        "develop"
                    ]
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
    ]
}
```

インスタンスIDによる EC2 インスタンスアクセス制御

IAM ポリシーにアクセスを許可したい EC2 インスタンスの ID を列挙することもできます。

IAMPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1:123456789012:instance/i
                "arn:aws:ec2:us-east-1:123456789012:instance/i
                "arn:aws:ec2:us-east-1:123456789012:instance/i
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
    ]
}
```

・タグや、インスタンスID によってアクセスできる EC2 を制御する IAM ポリシーを作成。

・本番環境インスタンスはインフラ管理者のみ、開発環境は

開発者全員アクセスできるようなグループを作成。
みたいな感じでポリシーのグループを作ればあとは適切なタグやポリシーを更新していくことでユーザアクセス制御ができそうですね。

セッション履歴とログ記録

セッションマネージャでは、どのサーバに誰がアクセスしたかのセッションの履歴はデフォルトでコンソール画面に記録されます。
また、オプションでセッションのアクセスログと操作ログを **CloudWatch**、 **CloudWatch Events**、**S3**へ自動で流すこともできます。

・ S3 に操作ログを保管
・ CloudWatch で操作ログを分析し、禁止されている操作に対して alart を通知
・ CloudWatch Events で特定操作に対してイベントを発行し lambda を起動。セッションを強制切断

みたいなこともできます。

指定された端末からログインしなければならない場合

企業によっては、保護されていない不特定なクライアントPCからアクセスしてはいけないといった規約がある所も多いと思います。
会社の社内ネットワークから隔離されたアクセス専用端末がある場合インターネットにも繋げずブラウザ上で AWS 管理コンソールが開けないでしょう。
その場合は AWS CLI を利用して対象サーバにアクセスすることもできます。

```
$ aws ssm start-session --target i-XXXXXXXXXXXXX

Starting session with SessionId: xxxxx-XXXXXXXXXXXXX
sh-4.2$
```

ターゲットには instance ID を指定します。オートスケールを利用している場合だと public IP が都度変わるので instance ID でアクセスできるのは非常に便利です。

導入方法

導入方法はとても簡単です。

Step1：AmazonLinux2 を立てる
Step2：セッションマネージャのポリシーを持ったロールを EC2 に付与
Step3：セッションマネージャ管理画面へアクセス！！！

なんと EC2インスタンス にセッションマネージャからのアクセス許可を持った IAM ポリシーをもつ IAM ロールを付与するだけです!!!
これだけでセッションマネージャ画面にアクセスできるインスタンス一覧が表示されるようになります。

付与するポリシーは下記のマネージドポリシーです。

~~AmazonEC2RoleforSSM~~

AmzonEc2RoleforSSM のポリシーは権限が広すぎるため非推奨になりました。今後このポリシーは廃止される可能性があります。
代わりに下記のポリシーを利用することが推奨されています。

AmazonSSMManagedInstanceCore

クラスメソッドさんの記事に何が変わったかを記事にされていました。

https://dev.classmethod.jp/cloud/aws/not-recommended-amazonec2roleforssm/

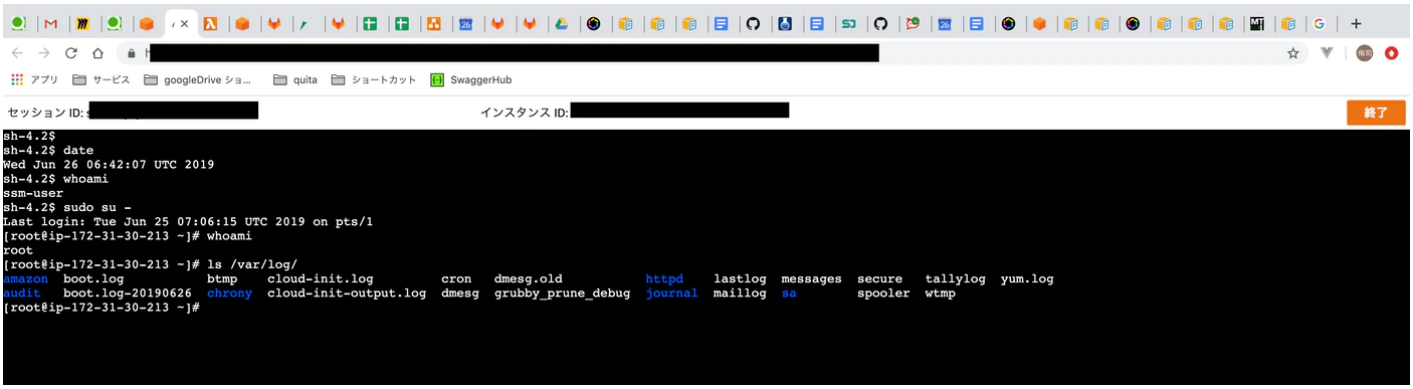
AmazonLinux2 では SSM エージェントが最初からインストールされ、自動で起動しています。もしほかの OS でもセッションマネージャで管理したい場合は SSM エージェントをインストールしましょう。

CloudFormation などで EC2 インスタンスを作成する場合はこのポリシーを必ず付与するか IAM Role のテンプレートを作成しておいてもいいかもしれません。

操作画面

ブラウザでサーバのコンソールを操作します。
コマンド送信に遅延などもなくサクサク快適にコマンドを叩くことができます。

コンソール画面



できることとできないこと

できること

Linux のコマンドはだいたい実行することができます。サーバのログや top、ps、sar、netstat などのコマンドでサーバの現在の状況を把握したいって時には使えそうです。

できないこと

ファイルのアップロード・ダウンロードができる機能はついていません。
サーバにファイルをアップロードするには wget を使えばできなくもないですが、ダウンロードはどうやってもできなさそうです。
サーバ内にファイルを貯めるようなサーバの場合はおとなしく S3 を利用する設計にしましょう。