Intercept：VM を利用した高インタラクションハニーポットの提案

寺村理[†1]　宮本大輔[†2]　中山雅哉[†2]

概要
ハニーポットとは、攻撃をあえて受けることで攻撃者の行動に関する情報を収集するシステムである。より多くの情報を得るためには、攻撃を誘致し、またハニーポットであることを疑われないようなシステムが求められる。それらの実現を目的として KVM の Live migration 機能を利用した高インタラクションのハニーポットを提案する。

Intercept：A proposal of VM-based High-interaction honeypot

Teramura Satoru[†1]　Daisuke Miyamoto[†2]　Masaya Nakayama[†2]

Abstract
A honeypot is a system that collects information about attackers' behavior by intentionally being attacked. To obtain more information, the system needs to be able to attract attackers and should not be suspected that the system is a honeypot. Based on these requirements, we propose a High-interaction honeypot using KVM Live migration technique

1. はじめに

ハニーポットとは、本物のシステムを装った、脆弱性を持つ偽のシステムであり、インターネットを介して行われる攻撃をあえて受けることで攻撃者の情報を収集し、攻撃内容を詳細に解析する目的で使用される。 本物の OS やアプリケーションを採用するハニーポットは高インタラクションのハニーポットと呼ばれる。本稿では、KVM のライブマイグレーション機能を利用した高インタラクションのハニーポットを提案する。

2. 提案手法

より詳細な攻撃に関する情報を収集するために、VMを用いた高インタラクションのハニーポットを提案する。

ハニーポットはなんらかの方法で攻撃者の不正なアクセスを誘導する必要がある。ハニーポットというものが存在することは攻撃者にも認識されているため、攻撃を受けるためには、それが通常のシステムであると見せかける必要がある。これが偽装能力である。また攻撃を誘致するためには、システムが攻撃者にとって価値のあるものであると思わせること、攻撃が成功しそうな脆弱性があることを見せることが必要である。また高インタラクションのハニーポットにおいては、侵入された際に、重要なリソースにアクセスできないようにネットワークを切り分けることや、ハニーポット自身を悪用されて、他のホストに対して攻撃が行われることを防ぐ防御能力も必要である。これは containment（封じ込め）と呼ばれるものである。

これらの能力に着目し、Fig.1 のようなシステム:*intercept* を提案する。通常稼働している何らかのサービスを行っているサーバ VM に対し攻撃者のアクセスが検知されたとき、Live migration 機能を用いてその VM のコピーを瞬時に作成し、攻撃者のみをそちらに誘導する。このとき新たに作られた VM のコピーがハニーポットとなる。攻撃者にとって、アクセスした段階では通常稼働している本物のサーバにアクセスしているので偽装を疑う余地はない。コピーに誘導することで通常のユーザへのサービスは停止することなく行える。Live migration を用いる利点はメモリの情報もコピーすることである。これによりサービスが通常通り継続しているように見せかけることができる。一度ログインしたはずがコピーに誘導された後に再びログインし直すことを求められることなどで攻撃者にハニーポットの存在を疑われるリスクを軽減する。

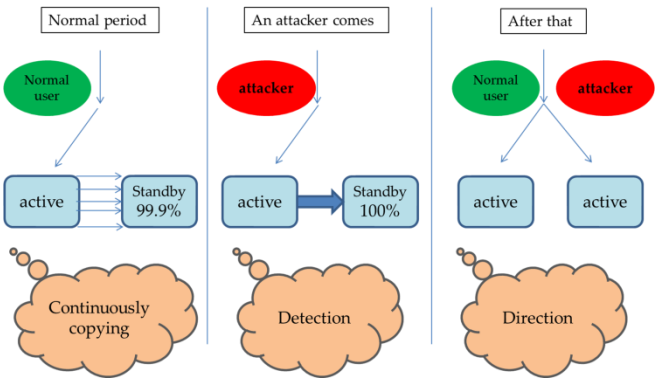

Fig.1 *Intercept*

†1 東京大学大学院工学系研究科
Graduate school of Engineering, The University of Tokyo
†2 東京大学情報基盤センター
Information Technology center, The University of Tokyo

3. システムの設計

2.で述べたシステム:*intercept* においては、悪意のあるアクセスを検知した際に可能なかぎり素早くハニーポットとなるセカンダリ VM を起動させ、プライマリ VM とセカンダリ VM の２つの VM を独立に稼働させることが必要である。今回我々は KVM の Live migration 機能、qemu のスナップショット機能[1]、NFS、Aufs、kemari[2]を用いてこれを実現した。

Live migration とは、あるホスト上の VM を稼働したままの状態で別のホスト上に移動する技術である。イメージファイルをあらかじめ NFS で共有しておき、メモリのコピーを行う。

qemu では VM のスナップショットを作成することができる。これは元となった VM イメージ（ベースイメージ）との差分及び参照するベースイメージの場所のみが書き込まれたファイルである。このスナップショットをイメージファイルに指定して VM を立ち上げるとベースイメージと差分が合算された VM が立ち上がる。この状態において VM 上でディスク書き込みなどを行ってもベースイメージは一切変更されない。差分であるスナップショットにのみ変更が加えられる。

NFS は、ネットワークを介してローカルのストレージを他のマシンに提供するシステムである。同一のディスク領域を複数のマシンで共有できる。

Aufs(Another unionfs)は同一マシン内の異なるディスク領域を透過的に重ねるシステムである。マージされたそれぞれのブランチに対して読み込みの優先度や、読み込みや書き込みの可否を指定できる。

kemari は qemu に変更が加えられたオープンソースである。VM を稼働してサービスを行っているホストに何らかの障害が発生した際に、サービスを停止せずに別のホストへ VM を以降することを目的として作られた。平常時、常にマイグレーションを行いプライマリ VM とセカンダリ VM を同期させておく。故障を検知した段階でスタンバイ状態になっていたセカンダリ VM を起動し、プライマリ VM シャットダウンするように実装されている。

これらを用いて以下のように提案手法を実現した。まず、中身が同一の VM のベースイメージをプライマリのホストとセカンダリのホストがそれぞれローカルに保持する。このとき、/root など、同一の名前のディレクトリに保持する必要がある。この理由は、次に述べるスナップショットにはベースイメージの参照場所が書かれているからである。プライマリのホストはこの VM のスナップショットを作成し NFS サーバに置く（ここでは例として /nfs をエクスポートポイントとする）。この/nfs に対してセカンダリのホストにおける/ronfs（ディレクトリ名は任意）を読み取り専用でマウントする。即ち、セカンダリからはプライマリ VM の最新の状態を読み込むことができるが、このディスクに変更を加えることはできない。次にセカンダリにおいて/ronfs を読み取り専用で、/tmp/unionfs を読み書き可能で、/nfs にマウントする。

VM の稼働

スナップショットイメージを指定してプライマリ VM を立ち上げる。セカンダリ VM も同様にして -incoming（マイグレーション待ち受け）モードで立ち上げる。プライマリ VM を、*kemari* オプションをつけてセカンダリへマイグレーションする。セカンダリ VM がスタートするまで、メモリのコピーを常にし続ける。ディスクは NFS で共有しているため、2 つの VM は常に同期されている。

攻撃検知後

悪意のあるアクセスを検知すると、マイグレーションを終了し、停止状態になっていたセカンダリ VM が起動される。以後の読み書きは/tmp/unionfs/にあるスナップショットに対して行われる。kemari のコード書き換えにより、マイグレーションを終了してもプライマリ VM が停止状態にならず稼働を続ける。これにより瞬時に 2 台の VM が稼働する状況を作り出すことが出来る。セカンダリ VM は稼働後、プライマリ VM が使用するスナップショットとは別のものを使用して読み書きを行うため、ディスクの齟齬が発生することもない。

4. おわりに

本稿では、KVM のライブマイグレーション機能を利用した高インタラクションのハニーポットを提案した。提案手法の実現のために既存の技術をいくつか組み合わせたが、それに加え *kemari* のソースコードに変更を加えた。通常のマイグレーション処理では移行元の VM は停止するが、これを停止しないようにソースコードに変更を加えた。また、移行先の VM で外部からシグナルを受信した場合に、移行が完了した VM を起動するように改良を行った。

今後の研究については、攻撃検知機能を導入しシグナル送信と結びつけること及び攻撃者と通常のユーザの振り分けを実装していく予定である。

5. 参考文献

[1]http://www.linux-kvm.com/content/how-you-can-use-qemukvm-base-images-be-more-productive-part-1

[2] Kemari: Fault Tolerant VM Synchronization based on KVM, Yoshi Tamura, Aug 2010 http://www.linux-kvm.org/wiki/images/0/0d/0.5.kemari-kvm-forum-2010.pdf