Vol.2015-IOT-28 No.3
2015/3/5

OpenFlow とハニーポットを用いた SSH ログイン攻撃防止手法の提案と実装

薛 微微[1]　石橋 勇人[1]

概要：
携帯電話網に代表されるように，近年インターネットへのアクセス手段が多様化しており，それにともなって外出先からのリモートアクセスのニーズが高まっている．このような場合によく使用される手段の 1 つに SSH があるが，パスワード管理の甘さを利用しようとする SSH への攻撃は後を絶たない．そこで，SSH サーバへの攻撃を防止するとともに，攻撃者の活動を観察可能とするため，OpenFlow と SSH ハニーポットを利用した攻撃防止手法を提案する．本手法は，SSH サーバに対するアクセスパターンを調べることによって攻撃を検出し，攻撃者からのアクセスを SSH ハニーポットへと誘導する．

Design and Implementation of an SSH Server Protection System using OpenFlow and Honeypots

Xue, Weiwei[1]　Ishibashi, Hayato[1]

Abstract:
Variety of Internet access methods, typically via mobile networks, are available these days. This makes it easy to access remote servers reside in user's internal networks from outside. Although SSH is one of the most popular access method in such a situation, SSH servers are always under attack by numerous brute force cracking attempts. This paper describes a novel technique to protect SSH servers using OpenFlow and SSH honeypots. The system examines the access pattern of a user and when it concludes the access as cracking, the traffic is redirected to an SSH honeypot to protect the SSH server.

1. はじめに

近年，インターネットへのアクセス手段が多様化し，携帯電話のデータ通信，WiMAX あるいは公衆無線 LAN などを用いることによって，外出先でもインターネットを手軽に利用できるようになっている．このような手段を利用すると，外出先からでもリモートアクセス機能を用いて容易に組織内のコンピュータの操作を行うことができる．

このような場合に利用されるアクセス方法の 1 つが Secure Shell(SSH) であり，暗号化と認証によって安全なリモートアクセスを実現している．

しかし，近年，SSH サーバに対するブルートフォース攻撃や辞書攻撃の増加が問題となっており，たとえば，2014 年に有限責任中間法人 JPCERT コーディネーションセンターから発表された 2014 年第 4 四半期の「インターネット定点観測レポート」[1] では，SSH サービスに対する攻撃が前四半期に引き続いて上位 3 位以内に含まれている (**表 1**)．

表 1　(2014 年 10 月〜12 月) 宛先ポート番号別パケット観測数のトップ 5 ([1] より引用)

宛先ポート番号	本四半期順位	前四半期順位
23/TCP(telnet)	1	1
22/TCP(ssh)	2	3
445/TCP(microsoft-ds)	3	2
0/ICMP	4	4
1433/TCP(ms-sql-s)	5	5

そこで，本稿では，OpenFlow[2] を用いて SSH サーバ

[1] 大阪市立大学大学院創造都市研究科
Graduate School for Creative Cities, Osaka City University

への攻撃トラフィックをリダイレクトし，SSH サーバに対する不正ログインを防止する手法を提案する．

OpenFlow は Software Defined Network (SDN) と呼ばれるネットワーク技術の 1 つであり，ネットワーク上のパケットの制御と転送を切り離し，制御を集中化することによって柔軟に行えるようにした点が特徴である．

提案手法では，SSH サーバのアクセスログを監視することによって不正アクセスを検知し，攻撃者と判断した相手ホストからのトラフィックを OpenFlow を用いて SSH ハニーポットに誘導する．これによって攻撃者から SSH サーバへのアクセスを遮断すると同時に，SSH ハニーポットによる攻撃パターンの観察が可能となる [*1]．

本手法を用いると，ハニーポット用に用意された IP アドレスではなく，実際に存在する SSH サーバに対する不正アクセスをハニーポットへと誘導できる．また，使用されていない IP アドレスが分かっていれば，OpenFlow スイッチを通過するトラフィックのうち，使用されていない IP アドレスに対するアクセスをハニーポットへと誘導することも可能である．

また，本手法は，該当するサービスのハニーポットが用意できれば (遮断せずに観察する場合)，SSH ログインの防御だけでなく他のサービスに対しても適用することができる．提案手法の対象となるネットワークは，すべてが OpenFlow 対応機器で構成されている必要はなく，後述する条件が満たされるならば，既存のネットワーク機器が混在する環境に対しても適用が可能である．

2. OpenFlow

OpenFlow では，ネットワークの経路制御機能（コントロールプレーン）とデータ転送機能（データプレーン）を分離している．経路制御機能は OpenFlow コントローラが行い，データ転送を OpenFlow スイッチが行う集中制御型のアーキテクチャとなっている (**図 1**)．OpenFlow コントローラとスイッチの間の通信に用いられるのが OpenFlow プロトコルである．

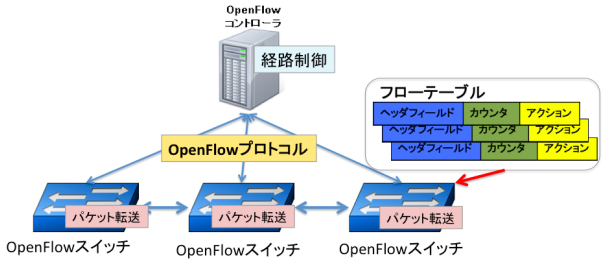

図 1 OpenFlow のネットワーク構成

[*1] 観察が不要であれば，SSH ハニーポットに誘導することなくパケットを破棄するようにしても良い．

OpenFlow スイッチは，接続された機器からのパケットを受け取ると，フローテーブルを確認する．受け取ったパケットにマッチするフローエントリが存在する場合は，指定されたアクションを実行する．マッチするフローエントリが存在しない場合は，OpenFlow 1.2 以前であれば Packet-In メッセージをコントローラに送信する．OpenFlow 1.3 の場合には，通常同様の働きをもつエントリを明示的に用意することになるが，いずれにせよ，マッチするフローエントリがなければ Packet-In メッセージを発行してコントローラに処理を委ねることになる．そこで，以降では，OpenFlow のバージョンによる差異は明示しないことにする．

あるパケットにマッチするフローエントリが作成されると，次からはそのフローエントリに従って自動的に (コントローラに依存せずに) パケットが処理される．

2.1 フローエントリ

OpenFlow スイッチは各々フローテーブルを持っており，フローテーブルには，複数のフローエントリが登録できる．フローエントリは，ヘッダフィールド，カウンタ，アクションの 3 つの要素から構成される．

- ヘッダフィールド
  ヘッダフィールドには，送信元 MAC アドレス，送信元 IP アドレス，宛先 MAC アドレスと宛先 IP アドレスなどの情報が含まれている．
- カウンタ
  フローテーブルのカウンタは，フローテーブルに登録されているフローエントリ数やパケットのマッチング処理回数などの統計情報を管理している．
- アクション
  フローエントリにマッチしたパケットに対して，実行すべき動作を定義する．アクションには，Forward，Drop，Enqueue，Modify-Field の 4 種類の処理が指定できる．

2.2 OpenFlow プロトコル

OpenFlow プロトコルには，次のようなメッセージが定義されている．

- Packet-In メッセージ
  OpenFlow スイッチは，受信したパケットにマッチするフローエントリが存在しない場合，コントローラに Packet-In メッセージを送信して処理を依頼する．OpenFlow コントローラは，このメッセージを受信すると，その契機となったパケットを解析し，フローエントリを作成するなど，必要な処理を行う．
- Flow-Mod メッセージ
  OpenFlow コントローラが OpenFlow スイッチに対して送信する，フローエントリの処理に関するメッセー

ジである．このメッセージによって，フローエントリの追加や削除などを行うことができる．

- Packet-Out メッセージ
  Packet-In によってコントローラに処理が依頼されたパケットを，スイッチの特定のポートから出力するよう指示するメッセージである．

3. 関連研究

3.1 アクセス制御ファイルの動的変更による SSH 総当り攻撃への対策

大隅らの研究では，SSHD(SSH サーバプログラム) のアクセスログを監視することにより不正な攻撃を検知し，アクセス制御ファイルを動的に更新して，自動拒否する方式を提案している [3]．従来の SSH 総当り攻撃や辞書攻撃に対する対策は，固定的な設定のため，変化する状況に対して動的に適用することができないという問題がある．この研究は，不正な攻撃を検知すると，他のサーバやネットワーク機器にその情報を伝達して不正な攻撃から防御するという特徴を持つ．

この方式では，SSHD のアクセスログファイルから SSH 接続でパスワード認証に失敗したものを抽出する．また，抽出されたパスワード認証エラーによって，パスワード総当り攻撃かどうか判定する．その後，攻撃者と断定された IP アドレスの情報をアクセス制御ファイルに記入して，接続を拒否することができる．さらに，このアクセス制御ファイルを利用して動的に他のサーバやネットワーク機器にこの情報を伝達して，動的にアクセス制御ファイルを更新する．これにより，不正な攻撃を防止する．

大隅らの研究では，アクセス制御ファイルの動的変更によって SSH ログインへの攻撃を防止する．しかし，各ルータへの設定が必要となる．

3.2 筑波大学におけるハニーポットを用いた不適切な SSH アクセスの収集とその解析

佐藤らは，筑波大学において，利用されていない IP アドレス領域宛のパケットを利用して，不適切な SSH アクセス情報の収集を行った [4]．収集されているパケットを解析した結果より，不適切なアクセスの実態を把握し，SSH サーバをより安全に運用することができる．

この研究では，Honeyd と Kippo[5] から構成される収集システムを設置して不適切なアクセス情報の収集を行い，分析している．未使用の IP アドレス宛のパケットを全部収集する機能を持っている Honeyd を用いて別サーバ上の SSH ハニーポット (Kippo) に通信を中継する．中継された通信に対して SSH サービスの擬態を行い，入力されたユーザ名やパスワードを収集する．また，収集した情報をもとに解析を行った．

佐藤らのシステムでは，未使用の IP アドレスへのアクセスの情報の収集と解析ができるが，実存するサーバへのアクセスは分析できない．また，ルーティングによるパケットの収集はネットワーク管理者による設定を要する．

3.3 OpenFlow スイッチによる悪意のある通信の集約

山田らは，OpenFlow スイッチを用いて悪意のある通信を判別する方法を提案している [6]．公開されている統計結果から得られた攻撃元 IP アドレスブラックリストの先頭 100 件と宛先ポート番号 27 種類を組み合わせたポリシーをコントローラにもたせ，ポリシーにマッチした通信の経路をハニーポットへ振り向ける．

ここでのポリシーはあらかじめ定められたものであり，アクセス時の挙動による判定ではない点において提案手法とは異なっている．

4. 提案手法

4.1 提案手法の概要

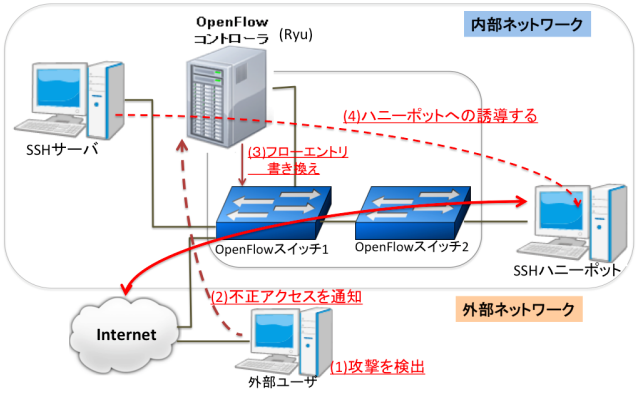

図 2 提案手法の概要

図 2 のネットワーク構成を例として，提案手法の概要を示す．

提案するシステムは，SSH サーバ，SSH ハニーポット，OpenFlow コントローラ，および 1 台以上の OpenFlow スイッチから構成される．外部から SSH サーバに対するアクセスは，OpenFlow スイッチを経由して SSH サーバおよび SSH ハニーポットに接続される．OpenFlow スイッチと SSH サーバの間に既存の (OpenFlow に対応しない) ルータや (L2) スイッチが存在しても構わないが，ルータが存在するか否かは後述する通知メッセージの選択に影響する．OpenFlow スイッチと SSH ハニーポットの間には既存のルータやスイッチが存在しても問題ない．

本提案手法では，SSH サーバのアクセスログを監視することにより，不正なアクセスを検出する (図 2(1))．攻撃であると判定された場合，SSH サーバは OpenFlow コントローラに不正アクセスを通知するために，通知メッセージを送出する (図 2(2))．OpenFlow コントローラ側では，送

られてきた通知メッセージから攻撃者の IP アドレスを抽出し，攻撃者からのトラフィックを SSH ハニーポットへと誘導するためのフローエントリを作成する (図 2(3))．この際，最優先に実行するために，フローエントリの優先度を高めに設定する．攻撃者が再度 SSH サーバにアクセスしようとした場合，OpenFlow スイッチに追加されたフローエントリのアクションにより，実際には SSH ハニーポットに接続することになる (図 2(4))．これによって，サーバへの SSH ログイン攻撃を防止できる．

4.2 不正アクセスの検出方法

ユーザが SSH サーバにログインしようとする場合，その情報がログファイルに記入される．このログファイルによると，パスワード認証のユーザ名，ユーザの IP アドレス及びアクセス時間が分かる．この情報に基いて，攻撃者であるかどうかを判断する．

攻撃者と判断したホストの IP アドレスはブラックリストに登録しておくが，正規の利用者がパスワードを忘れたなどの理由によって登録される可能性や，IP アドレスがその後別のユーザに割り当てられる可能性を考慮し，一定時間 (現在は 24 時間) が経過した後に削除する．

不正アクセスの検出方法を，以下に述べる．

4.2.1 IP アドレスやドメイン名などのブラックリスト

この方法では，ブラックリストファイル [7] を用いて，不正アクセスを検知する．ブラックリストファイルは，世界中で公開されている IP アドレスブラックリストもしくはネットワーク上から収集しているブラックリストを用いる．外部からのアクセスが来た場合，IP アドレスがこのファイルに登録されているかどうかを確認する．登録されている場合は，攻撃者と判断する．

4.2.2 同一ホストからの認証回数

同一ホストから連続して複数回のパスワード認証エラーが発生する場合は，攻撃されている可能性がある．しかし，単に正規ユーザによるパスワード入力ミスや，パスワード忘れなどの原因も考えられるので，攻撃されているかどうかの判定には注意する必要がある．例えば，次のような判定方法が考えられる．

(1) 繰り返し認証回数が一定値を超えた場合

これは，外部からのアクセス回数によって判断する．パスワード認証エラーのログ情報に基いて，認証失敗ユーザの情報を抽出し，前回の同じ IP アドレスからの認証エラーの情報と合わせて，一定時間内で，認証回数を集計する．集計された結果から，不正アクセスかどうかを判断する．

ここで，パスワード認証エラーには次のような原因が考えられる．

(a) 正規ユーザが，入力ミスやパスワード忘れなどの理由で，認証に失敗する

(b) 攻撃者が存在しているユーザ名を用いて侵入しようとし，パスワードを頻繁に変更してログインを試みたために認証エラーとなる

(a)，(b) を完全に区別することは困難である．論文 [3] では，同じ IP アドレスから 10 秒以内の時間差で，連続して 11 回のパスワード認証エラーが発生した時，攻撃者と判断しており，本論文でも同じ基準を採用する．ただし，以上の判定条件だけでは，不十分だと考えられる．なぜなら，10 秒間に 10 回以内のエラーの場合，実際は攻撃であっても検出できないからである．このため，1 時間内のアクセス回数あるいは 1 日のアクセス回数が一定の回数を超えるかどうかによって攻撃であるかどうかを判断するなど，複数の判断基準を導入することが望ましい．

(2) 頻繁にユーザ名を変更している場合

同じ IP アドレスから，正規ユーザが，短期間に何度もユーザ名を変更してアクセスすることは不自然と考えられる．この場合，異なるユーザ名によるアクセスの回数が一定数を超えた場合，攻撃と判断する．

4.2.3 異なるホストからの認証回数

登録されている同一のユーザ名に対して，一定時間内で複数の IP アドレスからのアクセスが来た場合，送信元 IP アドレスが同じ組織 (ネットワーク) であるかどうかをチェックする．同じ組織の場合，利用者が複数のパソコンを使って認証する可能性があると考えられる．しかし，違う組織から短期間に同じユーザ名を用いて認証することは，正常ではないと考えることができる．そこで，同一ユーザに対するアクセス元が一定数を超えた場合，攻撃と判断する．

4.2.4 異なる認証方式によるアクセス

公開鍵認証のみを使用するホストに対してパスワードによる認証の要求が来た場合は，不正アクセスが疑われる．しかし，正規ユーザが誤ってパスワードによる認証を要求することによって認証エラーが発生する可能性もあるため，同じ IP アドレスからの認証エラーが 2, 3 回程度である場合には，攻撃者とは判断しない．

4.2.5 使用していない IP アドレスへのアクセス

使用していない IP アドレスに対してアクセスがあった場合には，やはり不正アクセスが疑われる．そのようなあて先 IP アドレスを持つパケットが OpenFlow スイッチに到達すると，Packet-In メッセージによって OpenFlow コントローラに通知されるため，コントローラであて先アドレスを調べることによって検出できる．

なお，この方法のみ SSH サーバではなく OpenFlow コントローラによる判定となるところに留意が必要である．

4.3 通知メッセージ

SSH サーバがあるアクセスを攻撃であると判定した場

合，SSH サーバから OpenFlow コントローラに情報 (攻撃者の IP アドレス) を伝えるために，通知メッセージを送信する．

一般に，セキュリティの観点から，サーバやユーザ機器が接続されるネットワークと，OpenFlow コントローラおよびスイッチから構成される制御用ネットワークとは分離した形で構築される．このため，SSH サーバが直接コントローラに対して通信を行うことはできない．そこで，本システムに固有の，ある識別子を用意し，コントローラにおいてその識別子を持つパケットを検出することによってコントローラに対する通知を実現している．

通知メッセージのフォーマットとしては，任意のパケットフォーマットを使用することができるが，既存のプロトコルにないフォーマットを使用すると，IDS によって不正なパケットとして検出されるなどの問題も考えられるため，ここでは既存のパケットフォーマットを選択することにした．

既存のフォーマットとしては，ARP や UDP を利用することが考えられる．通知パケットは SSH サーバによって送出され，OpenFlow スイッチによって OpenFlow コントローラに転送されるため，SSH サーバから最初の OpenFlow スイッチまで到達できるプロトコルを使用する必要がある．したがって，SSH サーバと OpenFlow スイッチの間に OpenFlow 非対応のルータがある存在する場合は UDP のような L3 プロトコルを使用する必要があるし，そうでなければ ARP のような簡便なプロトコルで良い．

4.4 IP アドレスの書き換え

特定の送信元 IP アドレスを持つパケットを SSH ハニーポットの存在するポートへ出力するフローエントリを OpenFlow スイッチに追加することによって，攻撃者から SSH サーバへ向かうパケットを SSH ハニーポットへと誘導することはできるが，攻撃対象となった SSH サーバと SSH ハニーポットでは IP アドレスが異なるため，そのままではセッションを確立することができない．このため，攻撃者から SSH ハニーポットへ向かうパケットのあて先 IP アドレスを SSH ハニーポットの IP アドレスに書き換える必要がある．また，逆方向についても同様の書き換えが必要である．

5. 実装と実験

5.1 実装

提案手法の実装にあたって，OpenFlow コントローラには Ryu[8] を利用した．附属の L2 スイッチのプログラムをベースとし，Python 言語を用いて必要な機能を追加実装している．OpenFlow スイッチには Open vSwitch[9] を，SSH ハニーポットには Kippo を利用した．

SSH サーバ上には，不正アクセスを検出するプログラム (Python 言語) と通知用の ARP パケットを生成するプログラム (C 言語) を実装し，動作させている．

5.1.1 攻撃の判定

攻撃者と判断するための条件については 4.3 節で述べたが，そのうち次の各条件について実装を行った [*2]．

条件 1 ブラックリストファイルのチェック

条件 2 同じ IP アドレスからの繰り返し認証回数
10 秒以内の時間差で 11 回，5 分以内の時間差で 15 回，1 時間以内の時間差で 30 回

条件 3 同じ IP アドレスからのユーザ名変更の回数
1 分以内に時間差でユーザ名を 6 回変更

条件 4 ある正規ユーザに対する違う IP アドレスからの認証回数
10 分以内の時間差で異なる送信元 IP アドレス 6 箇所からのアクセス

5.1.2 通知メッセージのフォーマット

試作システムでは，通知メッセージには ARP パケットを利用している．テスト環境では，SSH サーバと OpenFlow スイッチの間に OpenFlow に対応しないルータが存在しないので，より簡潔な形式を持つ ARP パケットを利用することにした．通知メッセージであることを表す識別子にはあて先 MAC アドレスを使用し，値として 10:20:30:40:50:60 を用いている．

5.2 実験

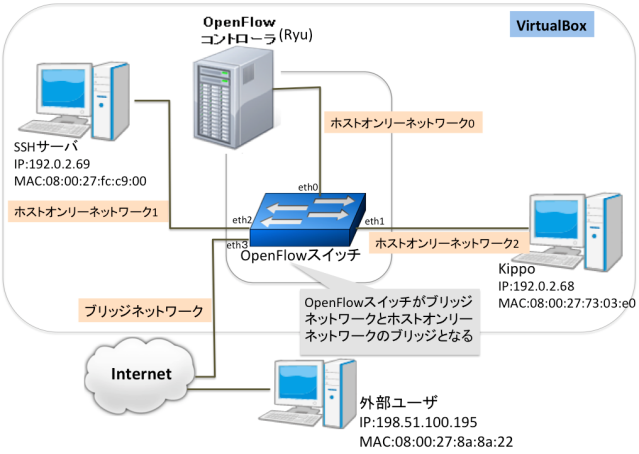

図 3 テスト環境のネットワーク構成

図 3 のようなネットワーク構成において試作システムの動作確認実験を行った (図中の IP アドレスおよび MAC アドレスは説明のためのものであり，実際に使用したアドレスとは異なっている)．

SSH サーバ，OpenFlow コントローラ，OpenFlow スイッチならびに Kippo はそれぞれ独立した仮想マシン上の

[*2] 一部不完全な部分が残っており，本稿執筆時点において修正中である．

Linux で動作している．OpenFlow スイッチは VirtualBox のブリッジネットワークを利用して外部 (インターネット) と接続されており，外部ユーザからのアクセスはこのスイッチを経由して SSH サーバに接続することになる．

5.2.1 通知メッセージの確認

実験において使用した SSH サーバのログを検査するプログラムに不具合が見つかったため，攻撃の検知が行われたという想定の下に通知メッセージ (ARP パケット) の送出プログラムを手動にて実行し，システムの動作を確認した．

```
pkt_arp----- [arp(dst_ip='攻撃者IPアドレス',dst_mac='10:20:30:40:50:60',hlen=6,hwt
ype=1,opcode=1,plen=4,proto=2048,src_ip='SSH SV IPアドレス',src_mac='08:00:27:fc:c9:
00')]
```

図 4 コントローラが受信した通知メッセージ

送出された通知メッセージを OpenFlow コントローラが受信した際に出力されたログを**図 4** に示す．図より，あて先 MAC アドレスが通知メッセージの識別子 (10:20:30:40:50:60) であること，および，あて先 IP アドレスが攻撃者の IP アドレスとなっていることがわかる．

5.2.2 フローテーブルの確認

```
 cookie=0x0, duration=2710.109s, table=0, n_packets=9, n_bytes=540, priority=1,i
n_port=2,dl_src=[redacted],dl_dst=[redacted] actions=output:2
 cookie=0x0, duration=2576.976s, table=0, n_packets=3, n_bytes=180, priority=1,i
n_port=1,dl_src=08:00:27:fc:c9:00,dl_dst=00:1d:71:99:93:80 actions=output:2
 cookie=0x0, duration=3081.255s, table=0, n_packets=752, n_bytes=119975, priorit
y=1,in_port=2,dl_src=[redacted],dl_dst=00:00:0c:07:ac:01 actions=output:2
 cookie=0x0, duration=149.765s, table=0, n_packets=0, n_bytes=0, priority=1,in_p
ort=2,dl_src=74:d0:2b:90:bc:a3,dl_dst=00:00:87:68:43:d3 actions=output:2
 cookie=0x0, duration=67.574s, table=0, n_packets=333, n_bytes=29197, priority=3
,ip,in_port=2,nw_src=攻撃者IPアドレス actions=set_field:08:00:27:73:03:e0->eth_dst
,set_field:SHP IPアドレス->ip_dst,output:3
 cookie=0x0, duration=67.574s, table=0, n_packets=185, n_bytes=11100, priority=3
,ip,in_port=3,nw_src=SHP IPアドレス actions=set_field:08:00:27:fc:c9:00->eth_src,
set_field:SSH SV IPアドレス->ip_src,output:2
```

図 5 フローテーブル

通知メッセージを受けてコントローラが OpenFlow スイッチ上に生成したフローエントリを**図 5** に示す．`cookie=` で始まる各行が 1 つのエントリを表しており，各エントリは複数のフィールドからなる．関連するフィールドの意味を次に示す．

priority フローエントリの優先度
in_port パケットを受信したポート
dl_dst あて先 MAC アドレス
output パケットを転送すべき物理ポート
set_field パケットヘッダの値を指定したものに書き換える

図 5 における最後の 2 行がハニーポットへ誘導するためにコントローラが追加したフローエントリである．下から 2 つめのエントリでは，攻撃者から送られてきたパケットについて，あて先 IP アドレスを SSH サーバの IP アドレスから SSH ハニーポット (SHP) の IP アドレスに書き換えるアクションを指定している．また，最後のエントリでは，逆に SSH ハニーポットから攻撃者へ送られるパケットについて，送信元 IP アドレスを SSH ハニーポットのものから SSH サーバ (SSH SV) のものへと書き換えるよう指定している．

フローエントリの追加前後にそれぞれ SSH によるサーバへのアクセスを行った結果，SSH サーバへのアクセスが SSH ハニーポットへと切り替わることが確認できた．

6. おわりに

本論文では，Openflow とハニーポットを用いた SSH ログイン攻撃防止手法について提案した．

今後の課題として，実装を完成させ，実環境におけるテストを通じて攻撃検知ルールのパラメータの妥当性を検討し，チューニングを行って精度を高めることや，攻撃者の行動に関する統計情報を収集することが挙げられる．

参考文献

[1] JPCERT/CC: インターネット定点観測レポート (2014 年 10 月〜12 月)，有限責任中間法人 JPCERT コーディネーションセンター（オンライン），入手先 ⟨https://www.jpcert.or.jp/tsubame/report/report201410-12.html⟩ （参照 2015 年 2 月 2 日）．
[2] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. and Turner, J.: OpenFlow: Enabling Innovation in Campus Networks, *ACM SIGCOMM Computer Communication Review*, Vol. 38, No. 2, pp. 69–74 (2008).
[3] 大隅淑弘，山井成良，井上一郎二：アクセス制御ファイルの動的変更による SSH 総当り攻撃の対策，学術情報処理研究，Vol. 11, pp. 68–73 (2007).
[4] 佐藤 聡，小川智也，新城 靖，吉田健一：筑波大学におけるハニーポットを用いた不適切な SSH アクセスの収集とその解析，情報処理学会研究報告，Vol. 2014-IOT-25, No. 17, pp. 1–6 (2014).
[5] desaster: Kippo – SSH Honeypot, Github (online), available from ⟨https://github.com/desaster/kippo⟩ (accessed 2015 年 2 月 2 日).
[6] 山田建史，戸部和洋，森 達哉，後藤滋樹：OpenFlow スイッチによる悪意のある通信の集約，コンピュータセキュリティシンポジウム 2011 論文集， Vol. 2011, No. 3, pp. 301–306 (2011).
[7] 鈴木 聡，湯浅富久子：ブラックリストを用いた PAM 遅延モジュールによる SSH への攻撃抑制，情報処理学会研究報告，Vol. 2006-DSM-40, pp. 1–5 (2006).
[8] Ryu project team: RYU SDN Framework, Ryu project team (online), available from ⟨http://osrg.github.io/ryu-book/ja/html⟩ (accessed 2015 年 1 月 20 日).
[9] Pettit, J., Gross, J., Pfaff, B., Casado, M. and Crosby, S.: Virtual Switching in an Era of Advanced Edges, *2nd Workshop on Data Center – Converged and Virtual Ethernet Switching (DC-CAVES)* (2010).