CROND Cyber Security Training System v1.0
インストールガイド

小松　源

2018 年 12 月 18 日
IS-TM-2018-003

12/18/2018

CROND Cyber Security Training System v1.0

インストールガイド

北陸先端科学技術大学院大学　知念研究室
博士前期課程 2 年　小松　源

1. 本ドキュメントの位置付け

本ドキュメントでは CROND Cyber Security Training System のインストールのみに止まらず、Proxy サーバの作成方法、CROND Cyber Security Training System の使い方のごく一部（より具体的には、サイバーレンジ上のマシンに対するセットアップ方法、演習の作り方等）にも言及している。ただし、本ドキュメントで示す Proxy サーバの設定方法は必要最低限なものであり、セキュリティ機構として十分だと保障するものではないということに注意されたい。また、できるだけ（UNIX コマンドに不慣れ、ssh で作業を行ったことがない、など。）初心者にもわかりやすいように配慮している。

対象バージョンは v1.0 である。

1.1. 本ドキュメントで作る構成

本ドキュメントでは 2 台の物理マシン（以下、ホストマシン）を用いる。まず、1 台のホストマシン上で CROND Cyber Security Training System が動作する構成をとる。次に、セキュリティ演習という特性上、サイバーレンジ内でさまざまなトラフィックが流れたりマルウェア等が動作することが考えられる。よって、ホストマシンとインターネットの間にもう 1 台のホストマシン（以下、GW）を設置することにより、外部との隔離を行う。本ドキュメントで構築できる構成図を図 1 に示す。

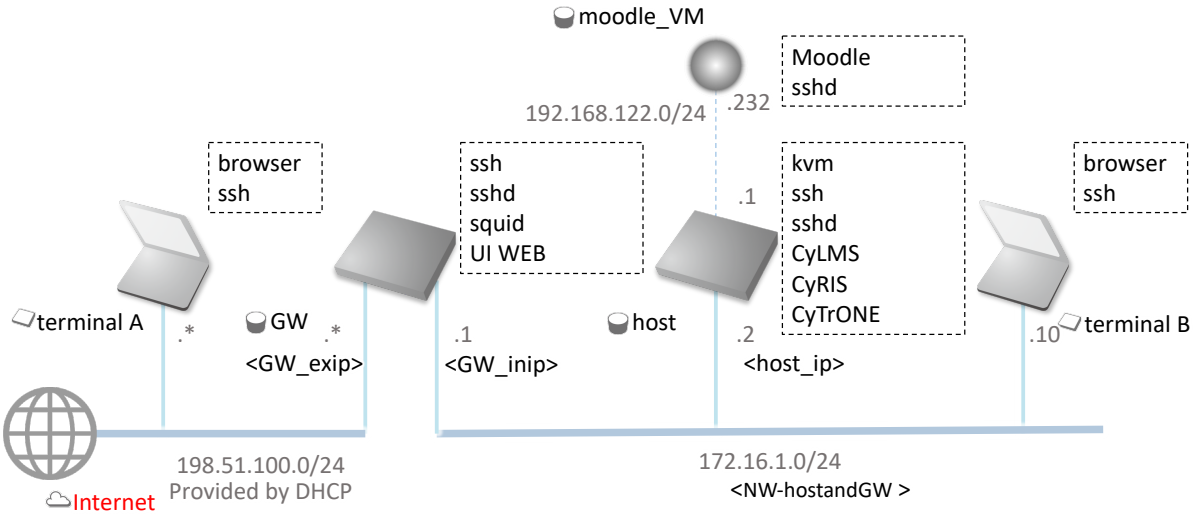

図 1：構成図

1.2. 本ドキュメントの読み方

以下に示す形式で書かれているものを実行すればよい。

```
% ホストマシン上で実行するコマンド
user% ユーザ端末上で実行するコマンド
GW% GW 上で実行するコマンド
moodle_VM% moodle_VM 上で実行するコマンド

;読んで指示通り実行する

~~~;コマンドを実行した結果、対話的な処理が発生している

<適宜要変更箇所>

<machine> % vi ファイル名
;新規作成されたファイルの中身 or ファイルの最後に行を挿入



<machine> % vi ファイル名
<行番号>. 行を挿入

<machine> % vi ファイル名
<行番号>. ファイル変更前の中身
↓
<行番号>. ファイル変更後の中身
```

<>で囲まれたものはそれぞれの環境によって違う（例えば、ホストマシンの IP アドレスなど。変数のようだと思えば良い)。適宜読み替えてほしい。以下に主なものを挙げる。読者の環境ではどんな値になるかを確認・メモしてから読み進めると良い。

本ドキュメント内での表記	意味	各ユーザ での値
<GW_inip>	GW のホストマシンに接続しているインターフェースの IPv4 アドレス	
<GW_exip>	GW の外部のネットワークに接続しているインターフェースの IPv4 アドレス	
<host_ip>	ホストマシンの GW に接続しているインターフェースの IPv4 アドレス	
<NW-hostandGW>	GW とホストマシンの間の NW（172.20.1.0/24 のように CIDR 表記で書くこと。）	

その他、枠で囲まれていないところは説明等が書いてある。

2. 事前準備

2.1. 前提（筆者の環境）

ホストマシン、GW の OS：Ubuntu 16.04 LTS

server 版の OS インストール時に OpenSSH を選択し、インストールされていることを前提としている（desktop 版の場合、ターミナルで「sudo apt install openssh-server」を実行する必要がある）。ホストマシン、GW 以外に、ユーザの手元で動作している PC（以下、ユーザ端末）がある。ユーザ端末上で Terminal 等のツールを使って、ホストマシンに ssh を行なって作業を行うものとする。

2.2. GW のセッティング

本ドキュメントでは Squid というソフトウェアを用いて Proxy サーバを作成する。

CROND Cyber Security Training System 用のユーザとして「cyuser」を作成し、パスワードを設定する。

```
GW% sudo adduser --shell /bin/bash cyuser
```

~~~;任意のパスワードを設定する。Full Name 等は何も入力せず Enter で良い。

cyuser がパスワードなしで sudo できるようにする。

```
GW% sudo visudo
;最終行に
;cyuser ALL=NOPASSWD: ALL
;を追記し、保存して終了。
```

以後、cyuser として作業を行う。

```
GW% su cyuser
~~~; <設定した cyuser のパスワード>
GW% cd ~/
```

GW から GW 自身へ鍵認証で（パスワードなしで）ssh できるようにする。

```
GW% ssh-keygen
~~~;エンターキーを三回入力（全てデフォルトのままで良い。）
GW% ssh-copy-id localhost
~~~;「yes」を入力し、<設定した cyuser のパスワード>を入力
```

Squid をインストールする。

```
GW% sudo apt update
GW% sudo apt install -y squid
```

Squid の設定を行う。

```
GW% sudo vim /etc/squid/squid.conf
976. acl cyber_range src <NW-hostandGW>
1188. http_access allow cyber_range
; <NW-hostandGW>は、172.16.1.0/24 のように、CIDR 形式で書くこと。
```

Squid の再起動を行う。

```
GW% sudo /etc/init.d/squid restart
```
```

unzip をインストールする。

```
GW% sudo apt install -y unzip
```

2.3. ホストマシンのセッティング

CROND Cyber Security Training System 用のユーザとして「cyuser」を作成し、パスワードを設定する。

```
% sudo adduser --shell /bin/bash cyuser
~~~;任意のパスワードを設定する。Full Name 等は何も入力せず Enter で良い。
```

cyuser がパスワードなしで sudo できるようにする。

```
% sudo visudo
;最終行に
;cyuser ALL=NOPASSWD: ALL
;を追記し、保存して終了。
```

以後、cyuser として作業を行う。

```
% su cyuser
~~~; <設定した cyuser のパスワード>
% cd ~/
```

ホストマシンからホストマシン自身へ鍵認証で（パスワードなしで）ssh できるようにする。

```
% ssh-keygen
~~~;エンターキーを三回入力（全てデフォルトのままで良い。）
% ssh-copy-id localhost
~~~;<設定した cyuser のパスワード>
;以下の自分の IP に対しても ssh-copy-id をしておく。（ssh-copy-id に「する必要なかった」と言われるが、それでもやる。）
% ssh-copy-id 127.0.0.1
~~~;「yes」を入力
% ssh-copy-id <host_ip>
~~~;「yes」を入力
```

Proxy サーバを使うための設定を行う。

```
% sudo vi /etc/apt/apt.conf
Acquire::http::Proxy "http://<GWin_ip>:3128";
Acquire::https::Proxy "http://<GWin_ip>:3128";
% sudo vi /etc/wgetrc
http_proxy=http://<GWin_ip>:3128/
https_proxy=http://<GWin_ip>:3128/
```

zip, unzip をインストールする。

```
% sudo apt update
% sudo apt install -y zip unzip
```

最後に、GW からホストマシンへ鍵認証で（パスワードなしで）ssh をできるようにしておく。

```
GW% ssh-copy-id <host_ip>
```

3. CyRIS のインストール

github より CyRIS をダウンロードする。

```
% cd ~/
% wget https://github.com/crond-jaist/cyris/archive/1.0.zip
% unzip 1.0.zip
% mv cyris-1.0 cyris
% mv 1.0.zip cyris-1.0.zip
```

CyRIS の実行に必要なパッケージをインストールし、cyuser に関する設定を行う。

```
% ./cyris/HOST-PREPARE.sh
~~~;様々なパッケージをインストールするかを聞かれるので、「Y」を押し続ける。wireshark
をインストールする際に出るウィザードは「No」で良い。
% sudo usermod -aG libvirtd cyuser
```

再起動を行う。

```
% sudo reboot
```

```
;強制ログアウトされるので、再度ログインを行う。
```

CROND の GitHub ページで配布されているサイバーレンジ上に作成する VM イメージをダウンロードする。

```
% mkdir images
% cd ~/images
% wget https://github.com/crond-
jaist/cyris/releases/download/1.0/basevm.tgz
% md5sum basevm.tgz
;上記コマンドの結果が
; be104ac392423412a9cfe4b7649d89b9  basevm.tgz
;であることを確認する。
% tar -zxvf basevm.tgz
```

4. CyLMS のインストール

github より CyLMS をダウンロードする。

```
% cd ~/
% wget https://github.com/crond-jaist/cylms/archive/1.0.zip
% unzip 1.0.zip
% mv cylms-1.0 cylms
% mv 1.0.zip cylms-1.0.zip
```

Moodle を用意する。CROND より Moodle がインストールされている VM（以下、moodle_VM）が提供されているため、それをダウンロードする。

```
% cd images
% wget https://github.com/crond-
jaist/cylms/releases/download/1.0/moodle.tgz
% md5sum moodle.tgz
;上記コマンドの結果が
; 0819fbab1bdab488143d6921248e9273  moodle.tgz
;であることを確認する。
```

```
% tar -zxvf moodle.tgz
```

ダウンロードした VM を立ち上げる。

```
% virsh define moodle.xml
% virsh start moodle
```

moodle_VM にも鍵認証で（パスワードなしで）ssh できるようにする。

```
% ssh-copy-id root@192.168.122.232
~~~;「yes」を入力、パスワード：theroot
```

CyLMS が使用する SCORM テンプレートを作成する。

```
% cd ~/
% wget https://github.com/crond-
jaist/cylms/releases/download/1.0/create_scorm_template.sh
% chmod +x create_scorm_template.sh
% ./create_scorm_template.sh /home/cyuser/cylms/
```

5. Moodle への接続性

実際にインストールができたかの確認として、CyTrONE で演習を作成し解いてみたいという要求があるだろう。CyTrONE で作成した演習を解くためには Moodle から演習にアクセスする必要があるが、現在は moodle_VM にはホストマシンの外からの接続性がない。接続性を持たせるためにはいくつかの方法が考えられる（ホストマシン上でリバースプロキシを動作させる、moodle が接続している仮想 NIC にホストマシンの物理 NIC を接続する、など）が、本ドキュメントでは ssh のポートフォワーディング 機能を使う方法を説明する。

ホストマシンの 8081 へアクセスすると、moodle の 443 へ接続させる。

```
% ssh -fgL 0.0.0.0:8081:192.168.122.232:443 localhost -N
```

なお、このトンネリングを行うプロセスはバックグラウンド実行されているため、止めるためには、

```
% lsof -i:8081
```

などとして、PID を確認し、

```
% sudo kill <ssh_pid>
```

を実行する。

Moodle の web ページ上のリンクが例えば「https://example.com/~~」になっていると、それをクリックした際にブラウザが example.com:443 に接続しに行ってしまう。そこで、Moodle が表示する URL の先頭が常に「https://<host_ip>:8081」になるように設定を行う。

```
% ssh root@192.168.122.232
moodle_VM% vi /var/www/html/moodle/config.php
21. $CFG->wwwroot = 'https://localhost';
↓
21. $CFG->wwwroot = 'https://<host_ip>:8081';
moodle_VM% service httpd restart
```

ssh を抜けておく。

```
moodle_VM% exit
```

6. CyTrONE のインストール

github より CyTrONE をダウンロードする。

```
% cd ~/
% wget https://github.com/crond-jaist/cytrone/archive/1.0.zip
% unzip 1.0.zip
% mv cytrone-1.0 cytrone
% mv 1.0.zip cytrone-1.0.zip
```

CyTrONE が使う Python のライブラリをインストールする。

```
% pip install pyyaml passlib --proxy=http://<GW_inip>:3128/
```

CyTrONE を利用するユーザに関する設定を行う。

```
% cd ~/cytrone/database/
% vi users.yml
9.    host_mgmt_addr: 172.16.1.7
```

```
↓
9.    host_mgmt_addr: <host_ip>
```

GW にも CyTrONE をダウンロードする。（より正確には、CyTrONE を管理するためのスクリプトを取得するために、CyTrONE をダウンロードする。）

```
GW% cd ~/
GW% wget https://github.com/crond-jaist/cytrone/archive/1.0.zip
GW% unzip 1.0.zip
GW% mv cytrone-1.0 cytrone
GW% mv 1.0.zip cytrone-1.0.zip
```

CyTrONE が動作するサーバの情報を設定する。

```
GW% cd ~/cytrone/scripts
GW% vi start_cytrone.sh
10. TRAINING_HOST = 172.16.1.7
12. INSTANTIATION_HOST = 172.16.1.7
14. CONTENT_HOST = 172.16.1.7
↓
10. TRAINING_HOST = <host_ip>
12. INSTANTIATION_HOST = <host_ip>
14. CONTENT_HOST = <host_ip>

GW% vi stop_cytrone.sh
5. TRAINING_HOST=172.16.1.7
7. INSTANTIATION_HOST=172.16.1.7
8. CONTENT_HOST=172.16.1.7
↓
5. TRAINING_HOST=<host_ip>
7. INSTANTIATION_HOST=<host_ip>
8. CONTENT_HOST=<host_ip>

GW% vi get_sessions.sh
```

```
16. TRAINING_SERVER=cytrone_host_name_or_ip
↓
16. TRAINING_SERVER=<GW_inip>

GW% vi create_training.sh
17. TRAINING_SERVER=cytrone_host_name_or_ip
↓
17. TRAINING_SERVER=<GW_inip>

GW% vi ./end_training.sh
19. TRAINING_SERVER=cytrone_host_name_or_ip
↓
19. TRAINING_SERVER=<GW_inip>
```

CyTrONE を管理するためのスクリプトを動作させるためにパッケージを入れる。

```
GW% sudo apt install -y python-pip
GW% pip install pyyaml
```

7. CyTrONE UI WEB のインストール

GW に github より CyTrONE UI WEB をダウンロードする。

```
GW% cd ~/
GW% wget https://github.com/crond-jaist/cytrone-ui-web/archive/v0.3.zip
GW% unzip v0.3.zip
GW% mv cytrone-ui-web-0.3 cytrone-ui-web
GW% mv v0.3.zip cytrone-ui-web-0.3.zip
```

CyTrONE UI WEB が使用するパッケージをインストールする。

```
GW% cd ~/cytrone-ui-web
GW% sudo apt install -y libssl-dev
GW% sudo cpan install Net::WebSocket::Server
~~~;できるだけ自動でやりますか？と聞かれるので、「yes」と答える。（cpan を実行したことがあるなら、表示されない。）
GW% sudo cpan install Digest::MD5
```

```
GW% sudo cpan install URI::Escape
GW% sudo cpan install LWP::UserAgent
GW% sudo cpan install JSON
GW% sudo cpan install IO::Socket::SSL
GW% sudo apt install -f -y libyaml-tiny-perl libdata-dump-perl

;次にインストールするモジュールは安定して動作しないようなので、システムにインストールはしないようにしている。(make install をしない。)
GW% wget http://search.cpan.org/CPAN/authors/id/M/MS/MSCHILLI/LWP-Protocol-https-6.06.tar.gz
GW% tar xvfz LWP-Protocol-https-6.06.tar.gz
GW% cd LWP-Protocol-https-6.06
GW% perl Makefile.PL
GW% make
```

設定ファイルを作成する。

```
GW% cd ~/cytrone-ui-web
GW% vi door.conf
httpd_addr <GW_exip>
httpd_port 1180

sweep_HTMLcontfiles .
set_maincontfile np.html
httpd_userpasswd john_doe john_passwd

wsd_addr <GW_exip>
wsd_port 9999

trngsrv_proto https
trngsrv_host <host_ip>
trngsrv_port 8082
trngsrv_lang en
```

環境変数を設定する。

```
GW% export PERL_LWP_SSL_VERIFY_HOSTNAME=0
GW% vi ~/.bashrc
PERL_LWP_SSL_VERIFY_HOSTNAME=0
export PERL_LWP_SSL_VERIFY_HOSTNAME;
```

8. サイバーレンジ上の VM とインターネットの接続性

サイバーレンジ上の VM に、様々なソフトウェアをインストールするという要求があるだろう。ここで注意が必要なのは、サイバーレンジ上の VM とインターネットとの間にはホストマシンと GW が挟まっている点である。ホストマシンの内外の通信に関しては、サイバーレンジを作成する最中のみ可能である（サイバーレンジ構築後は外部との通信を行えないような設定を CyRIS が行う）。しかし、GW を超える通信を行うためには別途設定が必要である。この設定にも、いくつかの選択肢がある（GW に DNS キャッシュサーバの役割も持たせ透過型 Proxy にする、basevm というイメージ自体に Proxy 設定をしておく）。本ドキュメントでは、Proxy 設定を行うシェルスクリプトを作成し、サイバーレンジ作成時に VM 上でそのシェルスクリプトを実行させる方法を説明する。

ホストマシン上で、Proxy 設定を行うシェルスクリプトを作成する。

```
% cd ~/
% mkdir vm_setting
% cd vm_setting
% vi set_proxy.sh
#!/bin/bash
sudo sh -c "echo 'http_proxy=http://<GW_inip>:3128' >> /etc/environment"
sudo sh -c "echo 'https_proxy=http://<GW_inip>:3128' >> /etc/environment"
```

サイバーレンジを作成する際に、VM で上記スクリプトを実行させるようにする。

```
% cd ~/cytrone/database
% vi NIST-level1-range.yml
```

```
18.    - copy_content:
19.      - src: /home/cyuser/vm_setting/set_proxy.sh
20.        dst: /tmp/
21.    - execute_program:
22.      - program: .
23.        args: /tmp/set_proxy.sh
24.        interpreter: bash
```

;上記ファイルのインデントがきちんとなっていないと CyTrONE は正常に動作しない。18 行目は「-」の左にスペース 4 つ、19 行目は「-」の左にスペース 6 つ、20 行目の dst は 19 行目の src と揃っていること。

9. テストラン

CROND Cyber Security Training System が正常に動作しているかを確認する。

➢ ステップ 1

GW より CyTrONE を起動する。

CyTrONE はログを標準出力する。よって、ssh で接続しているなら、ユーザ端末上でもう一つ新しく terminal を開いて、ホストマシンへ ssh し本ステップのみをその terminal で実行することをおすすめする。

```
GW% cd ~/cytrone/scripts
GW% ./start_cytrone.sh
```

➢ ステップ 2

現在のセッション数（作ったトレーニング数）を確認する。

```
GW% cd ~/cytrone/scripts
GW% ./get_sessions.sh
```

まだ作っていないので、SESSION INFO は 0 のはずである。

➢ ステップ 3

サンプルとして提供されているトレーニング（NIST Level 1）を生成してみる。

```
GW% ./create_training.sh 1
```

実行後、VM を立ち上げたりするため、しばらく時間がかかる。

（筆者の場合は 2 分程度であった。）

実行後、-で区切られた「Dear user,」から始まるメッセージが表示される。これは entry point と呼ばれる CR に接続するための接続点に関する情報で、ssh 用の IP アドレス、ポート番号、ユーザーネーム、パスワードが記載されている。なおこの情報は、

```
% less ~/cyris/cyber_range/<range_id>/range_notification-cr<range_id>.txt
```

で参照できる。

➢ ステップ 4

現在作成されているトレーニングセッションに関する情報を確認する。

```
GW% ./get_sessions.sh
```

SESSION INFO が 1 に変わっていて、先ほど作成したトレーニングセッションに関する情報が表示されているはず。

SESSION : の中の id を確認しておく。（トレーニングを終了させる際に必要）

➢ ステップ 5

実際に演習を解いてみる。

Moodle へのアクセス。

```
;ユーザ端末上で Web ブラウザを起動し、
;https://<host_ip>:8081
;へアクセス。
;Moodle の画面上の「CyTrONE Training」というリンクをクリックし、ログインする。
; アカウント以下の二つが用意されている。
;管理者用
;   Username : admin
;   Password : The.admin01
;生徒用
;   Username : trainee01
;   Password : The.trainee01
;ログイン後、「Training Session #1」というリンクをクリックし、「Enter」をクリックする
```

```
と、問題が表示される。
;演習環境へのアクセスはステップ 3 で表示された情報を用いる。
```

好きなように演習環境を触り、Moodle の問題に答えて、「Submit Answers」を押すと入力した解答が採点される。

- ステップ 6

トレーニングを終了する。

```
GW% ./end_training.sh <ステップ 4 で確認した id>
```

- ステップ 7

トレーニングが終了していることを確認する。

```
GW% ./get_sessions.sh
```

SESSION INFO が 0 になっているはず。

- ステップ 8

CyTrONE UI WEB を起動する。

```
GW% cd ~/cytrone-ui-web/
GW% perl -I LWP-Protocol-https-6.06/lib door.pl -f door.conf -m
```

- ステップ 9

CyTrONE UI WEB へアクセスする。

```
;ユーザ端末上で Web ブラウザを起動し、
;http://<GW_exip>:1180
;へアクセス。
;ユーザ名 : john_doe
;パスワード : john_passwd
```

図 2 のような画面が出るはず。

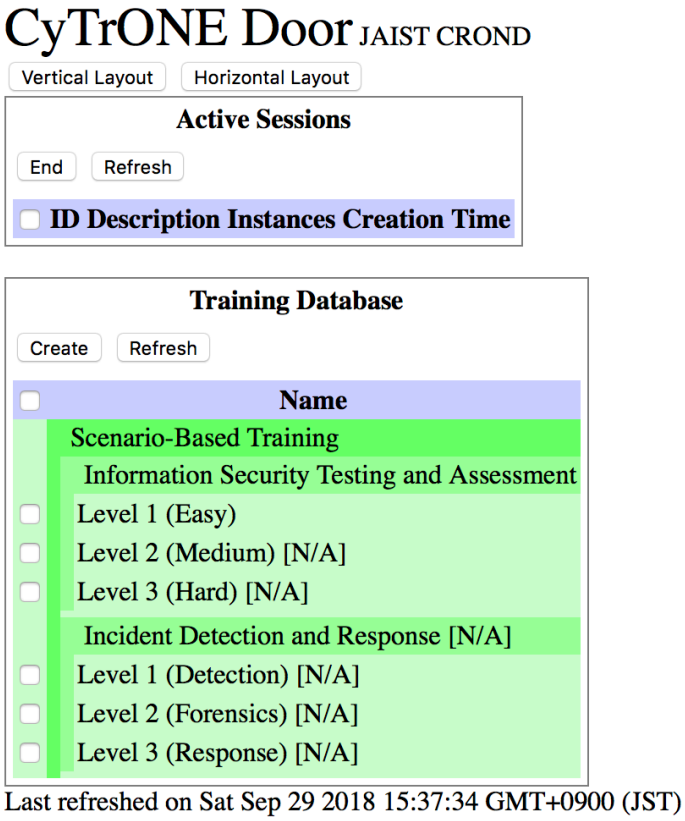

図 2 : CyTrONE UI WEB の画面

➢ ステップ 10

CyTrONE UI WEB を使ってトレーニングを生成する。

;「Training Database」の中の「Level 1(Easy)」のチェックボックスにチェックを入れ、「Create」のボタンをクリックする。
;ポップアップが出てくるので、「OK」を押す。

「Active Sessions」に「~ just starting...」と出てくるので、しばし待つ。

「~ just starting...」が Level 1 (Easy)に変わると生成完了。「Active Sessions」の一番左の ID が <range_id>に対応する。(なかなか (Level1 なら 5 分以上)「~ just starting...」から変わらない場合、ブラウザの再読み込み機能を使ってみる。)

➢ ステップ 11

実際に演習を解いてみる。(ほとんどステップ 5 と同じ。)

entry point に関する情報は

```
% less ~/cyris/cyber_range/<range_id>/range_notification-cr<range_id>.txt
```

で参照のこと。

- ステップ 12

CyTrONE UI WEB を使ってトレーニングを終了する。

```
;先ほど作成した「Active Sessions」の中にある演習のチェックボックスにチェックを入れ、
「End」をクリックする。
```

（なかなか（Level1 なら 1 分以上）「~ just starting...」から変わらない場合、「Active Sessions」の中の「Refresh」をクリックしてみる。）

- ステップ 13

CyTrONE を終了する。

```
% ./stop_cytrone.sh
```

以上でテストランは終わり。

以上でインストールならびにテストランは終わり。
実際に CyTrONE を使って演習を作る際は、それぞれのプログラムの Users Guide を参照されたい。

10. 補足

本ドキュメントでは、「cyuser」というアカウントを作成しており、このアカウントに「パスワードなしで root 権限でコマンド実行可能」という非常に高い権限を与えている。このアカウントのパスワードは読者によっては（アカウント名と同じ「cyuser」のような）非常に簡単なものに設定されることが予想される。それに加えて、GW が外部ネットワークに繋がれる可能性があり、その OS 上では sshd が動作していることから、不正ログインの危険性が非常に高まる。よって、以下のようにしてパスワード認証で ssh 出来ないようにすることを強くおすすめする。なお、ユーザ端末から ssh で作業している場合、先にユーザ端末から鍵認証によって ssh できるように設定する必要がある（この方法は後述するので、次の作業を後回しにして、先に後述を参照されたい)。

以下の作業を GW 上で行う（ホストマシンも外部ネットワークに直接接続する場合、ホストマシンにも同様の作業を行う)。

```
GW% vi /etc/ssh/sshd_config
52. #PasswordAuthentication yes
↓
52.PasswordAuthentication no
GW% sudo service sshd restart
```

ユーザ端末から GW に鍵認証で ssh するように設定する方法は、上述の作業を行う前に以下の作業を行う。

```
user% cd ~/
user% ssh-keygen
user% ssh-copy-id <GW_exip>
user% vi ~/.ssh/config
Host GW
    HostName <GW_exip>
    User cyuser
    IdentityFile ~/.ssh/id_rsa
;以後、ユーザ端末から ssh する際は、以下のようにすれば良い。
user% ssh GW
```

以上。