Sprint review

Week 1 - Team Jupiter

- Bui Xuan Phuoc
- Dang Chi Cong
- Pham Nguyen Thanh An
- Vo Le Tung

Sprint plan

Develop task 1 and task 2 simultaneously as two seperated modules.

Expected outcome:

- Reading/Parsing data that is needed to raise alerts.
- Show alerts using log messages.
- Ability to programmatically change alert threshold.

Sprint backlog

Subtasks	Time allocated	Personnel
How to setup Apache Server	4 hours	Cong
Research how to read Apache log (raw event)	6 hours	Phuoc
Research on how to use Pcap4J (raw event)	7 hours	An
Classify Apache log messages for ConsecutiveFailedPasswordEvent	10.5 hours	Cong

Sprint backlog

Subtasks	Time allocated	Personnel
~~Classify Apache log messages for UnauthorizedAccessEvent~~	~~10.5 hours~~	
Classify Apache log messages for FileTooLargeEvent	10.5 hours	Phuoc
Detect HorizontalPortScanEvent	10.5 hours	An
Detect BlockPortScanEvent	10.5 hours	Tung
Detect VerticalPortScanEvent	10.5 hours	Tung

Setup a web service using Apache server

Setup a simple web server that requires *BasicAuth* to access its root.

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory "/var/www/html">
        AuthType Basic
        AuthName "Restricted Content"
        AuthUserFile /etc/apache2/.htpasswd
        Require valid-user
    </Directory>
</VirtualHost>
```

Reading Apache's log messages

Default log directory: `/var/log/apache2/access.log`.

Example access log entry.

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700]
"GET /apache_pb.gif HTTP/1.0"
200 2326 "http://www.example.com/start.html"
"Mozilla/4.08 [en] (Win98; I ;Nav)"
```

Reading Apache's log messages

We wrote our own parser to parse to log entry into the following data structure

```java
public class httpLogEvent {
    private String IPAddress;
    private String identd;
    private String userID;
    private String time;
    private String timeZone;
    private String protocol;
    private String statusCode;
    private String returnObjSize;
    private String referer;
    private String clientBrowser;
}
```

Web service alert: Events hierarchy

```
                                      httpLogEvent
                                          /\
                                         /  \
                                        v    v
TargetOneAccount <-- httpFailedLoginEvent    httpFileTooLargeEvent --> OneSameFile
                              /\                       /\
                             /  \                     /  \
                            v    v                   v    v
                       SameIP    Consecutive    SameIP    Consecutive
```

Web service alert

- `httpFailedLoginEvent` is raised when a client is responded with status code 401
- `httpFileTooLarge` is raised when
  - A client's post request is responded with status code 413, or
  - The size of the request exceeds a certain threshold.

Web service alert

- Alerts relating to authentication are raised when the number of `httpFailedLoginEvent` reaches a certain threshold within a fixed time window
- Alerts relating to post request's file size are raised when the number of `httpFileTooLarge` reaches a certain threshold withn a fixed time window

Capture network packets

Instruct Pcap4J to filter for packets of TCP protocol.

```
PcapHandle handle = nif.openLive(65536, PcapNetworkInterface.PromiscuousMode.PROMISCUOUS, 100);
handle.setFilter("tcp", BpfProgram.BpfCompileMode.OPTIMIZE);
```

Raw event for TCP packets.

```
public class TcpPacket {
    private Long timestamp;
    private TcpPacket.TcpHeader tcpHeader;
    private IpPacket.IpHeader ipHeader;
}
```

Port scan alerts: Events hierarchy

```
                                 |--> VerticalScanAlert
                                 |
TcpPacket --> TcpToClosedPort -->|--> HorizontalScanAlert
                                 |
                                 |--> ClosedPortsPerAddr --> BlockScanAlert
```

Port scan alerts: TcpToClosedPort

Netwoking patterns of a TCP packet that is sent to a closed port.

```
|                 |
|  SYN=1, ACK=0   |
|---------------->|
|                 |
|  RST=1, ACK=1   |
|<----------------|
|                 |
v                 v
```

Port scan alerts

The scan alerts are raised when the number of TCP connection to closed ports reaches a threshold within a fixed time windows.

- Vertical scan alert counts the number of closed ports that were accessed per host address.
- Horizontal scan alert counts the number of host addresses that were accessed per closed port.
- Block scan alert counts the number of host addresses whose many closed ports were accessed.

Plans for next week

- Tackle task 4.
    - Fix existing modules (if needed) for integration.
    - Starts prioritizing the alerts.
- Might start working on task 3, when we come up with some interesting ideas.