What is a Bro log?
What is a Bro log?

• A stream of high level entries that correspond to lower level events.
  – An HTTP request/reply pair.
  – An email sent using SMTP.
  – A login over SSH.
Not log, but logs

• Bro does not have a single “alert” type log. Instead, each stream has a dedicated type with its own set of fields.

• More than one file?
  – The SMTP log has ‘from’ and ‘subject’ fields.
  – The HTTP log has ‘method’ and ‘uri’ fields.
  – The ‘from’ field would not make sense for HTTP and the ‘method’ field would not make sense for SMTP.

...........................................
How many log files are there?

• By default, Bro will output about two dozen log files, depending on what types of traffic it can see.
  – A few: conn, dhcp, dns, dpd, files, http, intel, known_certs, known_hosts, known_services, modbus, notice, radius, smtp, snmp, socks, software, ssh, ssl, syslog, traceroute, weird, x509, and quite a few others.

• More in every release!
Signal to Noise ratio

• The main way that log files can be categorized is by their size and signal to noise ratio. Some logs files are large and will contain entries that can be either benign or malicious. Other files are smaller and contain more actionable information.

  – **24K** known services.log
  – **28K** software.log
  – **68K** notice.log
  – **311M** dns.log
  – **856M** conn.log
High signal log files

• Inventory related log files
  – known_hosts
  – known_services
  – known_certs
  – software

• Other high signal log files
  – notice: When Bro detects something that it thinks is exceptional it raises a notice.
  – intel: Traffic that matches a list of known bad indicators is logged here.

...
Increasing signal to noise!

• Bro makes it easy to take a large log file and filter a subset of the entries to a smaller file with a higher signal to noise ratio.
  – Filtering the http.log to http_exe.log
  – Filtering the http.log to http_wget.log
  – Filtering the http.log to http_java.log
  – Filtering the conn.log to conn_cn.log
  – Filtering the ssh.log to ssh_non_us.log
function is_it_exe(rec: HTTP::Info): bool
{
    if ( rec?$resp_mime_types )
        for ( i in rec$resp_mime_types )
        {
            if ( "application/x-dosexec" == rec$resp_mime_types[i] )
                return T;
        }
    return F;
}

event bro_init()
{
    local filt = Log::Filter($name="http_exe",
        $path="http_exe",
        $pred = function(rec: HTTP::Info): bool
        {
            return is_it_exe(rec);
        });
    Log::add_filter(HTTP::LOG, filt);
}
```java
function is_it_cn(rec: HTTP::Info): bool {
    local orig_loc = lookup_location(rec$id$orig_h);
    local resp_loc = lookup_location(rec$id$resp_h);
    return (orig_loc?$country_code && orig_loc$country_code == "CN") ||
            (resp_loc?$country_code && resp_loc$country_code == "CN") );
}

event bro_init() {
    local filt = Log::Filter($name="http_cn",
                             $path="http_cn",
                             $pred(rec: HTTP::Info): bool = {
                             return is_it_cn(rec);
                             });
    Log::add_filter(HTTP::LOG, filt);
}```
What does the stream of events look like?

• A key-value mapping. Like a CSV file or a relational database table.

  – We can create some log files by starting Bro and running the following command line:

    • curl www.google.com

  – This will request the google home page, but not any of the associated javascript or image files.

  – Bro will write an entry in the http.log describing this event. The http.log contains 27 columns which can be a bit daunting. We can transpose the columns into rows to make this single line from http.log easier to understand.
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ts</td>
<td>time</td>
<td>1408828734.304076</td>
</tr>
<tr>
<td>uid</td>
<td>string</td>
<td>CZceY8wvnES5foJp4</td>
</tr>
<tr>
<td>id.orig_h</td>
<td>addr</td>
<td>192.168.43.222</td>
</tr>
<tr>
<td>id.orig_p</td>
<td>port</td>
<td>65032</td>
</tr>
<tr>
<td>id.resp_h</td>
<td>addr</td>
<td>74.125.226.50</td>
</tr>
<tr>
<td>id.resp_p</td>
<td>port</td>
<td>80</td>
</tr>
<tr>
<td>trans_depth</td>
<td>count</td>
<td>1</td>
</tr>
<tr>
<td>method</td>
<td>string</td>
<td>GET</td>
</tr>
<tr>
<td>host</td>
<td>string</td>
<td><a href="http://www.google.com">www.google.com</a></td>
</tr>
<tr>
<td>uri</td>
<td>string</td>
<td>/</td>
</tr>
<tr>
<td>referrer</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>----------------</td>
</tr>
<tr>
<td>user_agent</td>
<td>string</td>
<td>curl/7.30.0</td>
</tr>
<tr>
<td>request_body_len</td>
<td>count</td>
<td>0</td>
</tr>
<tr>
<td>response_body_len</td>
<td>count</td>
<td>21232</td>
</tr>
<tr>
<td>status_code</td>
<td>count</td>
<td>200</td>
</tr>
<tr>
<td>status_msg</td>
<td>string</td>
<td>OK</td>
</tr>
<tr>
<td>info_code</td>
<td>count</td>
<td>-</td>
</tr>
<tr>
<td>info_msg</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>filename</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>tags</td>
<td>set[enum]</td>
<td>(empty)</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>proxied</td>
<td>set[string]</td>
<td>-</td>
</tr>
<tr>
<td>orig_fluids</td>
<td>vector[string]</td>
<td>-</td>
</tr>
<tr>
<td>orig_mime_types</td>
<td>vector[string]</td>
<td>-</td>
</tr>
<tr>
<td>resp_fluids</td>
<td>vector[string]</td>
<td>FvwPGj436gbcfXpCGf</td>
</tr>
<tr>
<td>resp_mime_types</td>
<td>vector[string]</td>
<td>text/html</td>
</tr>
</tbody>
</table>
Not just HTTP!

• This one HTTP download caused Bro to write entries to 6 log files:
  – `http.log` has the above entry
  – `dns.log` has an entry from the dns query for www.google.com
  – `files.log` has an entry from the html file that was downloaded
  – `conn.log` has an entry for both the dns an http connections
  – `known_hosts.log` has an entry for 192.168.43.222
  – `software.log` has an entry for an HTTP::BROWSER of curl/7.30.0 seen on 192.168.43.222
known_hosts

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ts</td>
<td>time</td>
<td>1408828734.303825</td>
</tr>
<tr>
<td>host</td>
<td>addr</td>
<td>192.168.43.222</td>
</tr>
</tbody>
</table>

This means 192.168.43.222 was seen for the first time at 1408828734.303825
This means curl/7.30.0 was seen for the first time on 192.168.43.222 at 1408828734.304076