
# Custom Logs for Elastic SIEM with ECS

February 5, 2020

**Dain Perkins**
**Security Solution Architect**
**ECS Contributor**

risk & information security stuff
husband, father, soccer & bass player, beer & bbq snob

## Webinar Agenda:

- Elastic
- Elastic Common Schema
- Elastic SIEM
- Integrating Custom Logs
  - Part 1: ASA Logins
  - Part 2: Meraki 802.1x
- Q/A

# Elastic

# Elastic Solutions

| Search | Observe | Protect |
| --- | --- | --- |
| • App Search<br>• Site Search<br>• Enterprise Search | • Logs<br>• Metrics<br>• APM<br>• Uptime | • SIEM<br>• Endpoint Security |

## Elastic Stack


canvas


machine learning


maps


reporting


dashboards

*and so much more*

elastic

# Elastic Solutions

| Search | Observe | **Protect** |
| --- | --- | --- |
| • App Search<br>• Site Search<br>• Enterprise Search | • Logs<br>• Metrics<br>• APM<br>• Uptime | • **SIEM**<br>• Endpoint Security |

## Elastic Stack


canvas


machine learning


maps


reporting


dashboards

*and so much more*

elastic

# Elastic Common Schema

# ECS in a Nutshell


elastic

# ECS Benefits - Visualizations

## ASA / PANW / IPTables Dashboards


ECS

**Unified Firewall Reporting**

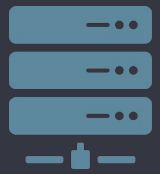
**Filter by vendor, type, ip, etc.**

elastic

# ECS Primer - agent, host, & observer

## Host

Laptop - Desktop
Server - Appliance
VM / Container



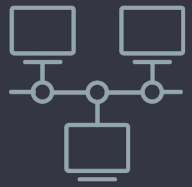
Events happening on the device (logins, sessions, services, application logs, &c)

Metrics and states of local resources (cpu, memory, configs, &c)

## Agent

Software that relays host / observer data

## Observer



Switch - Router
Firewall - WAF
Load Balancer
Network Sensor

External events witnessed by or acted on by the device (connections, traffic, user sessions, etc.)

Metrics and measurements about monitored or processed traffic

elastic

# ECS Primer - Field Levels

- **ECS-Core:** A fully defined set of field names that exist under a defined set of ECS top-level objects
  - e.g.: source.ip, host.name, client.mac, etc.
- **ECS-Extended:** A partially defined set of field names that exist under the same set of ECS top-level objects
  - e.g.: source.nat.ip, client.geo.name, as.number, etc.
- **Custom:** An undefined set of fields that exists under a user-supplied set of Non-ECS top-level objects
  - e.g.: vendor.platform.field.sub-field (cisco.wlc.ssid, meraki.vpn.status)

# ECS Primer - Custom Fields

**Rules for Custom Fields:**

- Do not use any field name that conflicts with any ECS namespace objects
- All fields must be lower case
- No special characters except underscore (_) and dot (.)
- Words are combined with underscore
- Do not use abbreviations
- Use present tense unless field describes historical information
- Use singular and plural names properly to reflect the field content

# ECS Primer - Where do I start?

Start with **REQUIRED** fields

- @timestamp “Feb 03 2020 11:27:05 -5”
- message “raw message goes here”
- ecs.version 1.4.0

Then move on to the core & extended fields that best **describe** the event, or the fields **necessary** for the use case:

| | | | | |
|---|---|---|---|---|
| event.* | host.* | source.* | destination.* | network.* |
| user.* | observer.* | client.* | server.* | etc. |

# Elastic SIEM

elastic

# SIEM / ECS Key Fields

**Host Overview**

Fields:
- host.name
- host.os.*

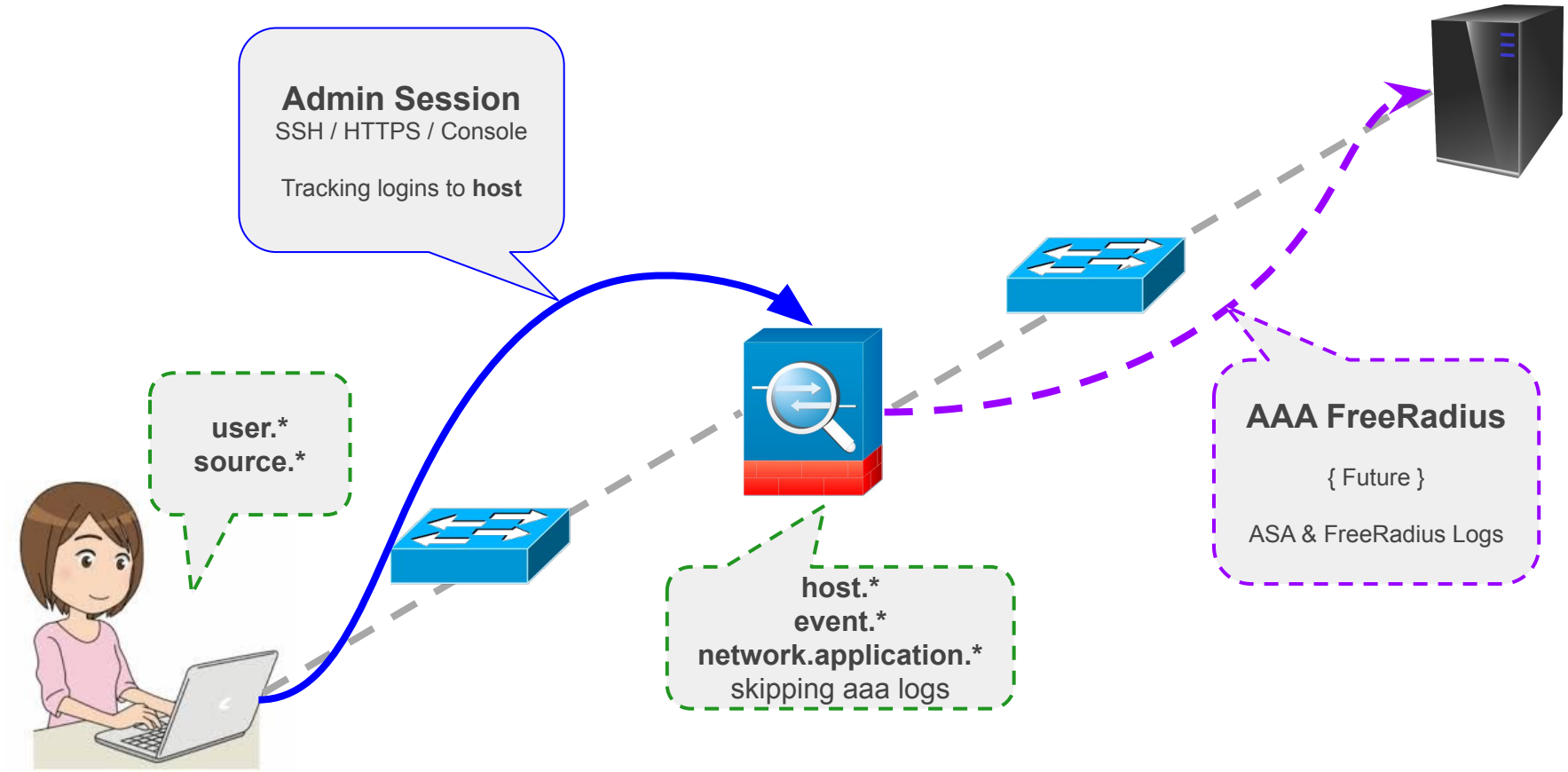
**Authentications**

Fields:
- user.name
- event.type =
  - authentication_failure
  - authentication_success
- event.category =
  - authentication

# Integrating Custom Logs
# Part 1: ASA Logins

elastic

# ASA Logins - Overview



# ASA Logins - Step 1: Analyze what's coming in

## Failure

605004

**Error Message** `%ASA-6-605004: Login denied from source-address/source-port to interface:destination/service for user "username "`

**Explanation** The following form of the message appears when the user attempts to log in to the console:

```
Login denied from serial to console for user "username"
```

An incorrect login attempt or a failed login to the ASA occurred. For all logins, three attempts are allowed per session, and the session is terminated after three incorrect attempts. For SSH and Telnet logins, this message is generated after the third failed attempt or if the TCP session is terminated after one or more failed attempts. For other types of management sessions, this m
appears when valid or the **no logging hide username** con

- source-address– Source address of the login attempt
- source-port– Source port of the login attempt
- interface– Destination management interface
- destination– Destination IP address
- service– Destination service
- username – Destination management interface

**Recommended Action** If this message appears infrequen
the user to verify the username and password.

## Success

605005

**Error Message** `%ASA-6-605005: Login permitted from source-address /source-port to interface:destination /service for user "username "`

The following form of the message appears when the user logs in to the console:

```
Login permitted from serial to console for user "username"
```

**Explanation** A user was authenticated successfully, and a management session started.

- source-address– Source address of the login attempt
- source-port– Source port of the login attempt
- interface– Destination management interface
- destination– Destination IP address
- service– Destination service
- username– Destination management interface

**Recommended Action** None required.

https://www.cisco.com/c/en/us/td/docs/security/asa/syslog/b_syslog/syslogs6.html#con_6732707

# ASA Logins - Step 2: Verify what's coming in

```
### Logstash ASA INPUT
input {
  udp {
    port  => 5140
    type => "asa"
    }
}

### Logstash ASA Output
output {
  if "60500" in [message] {
    ##DEBUG
    file {
        path => "/asa-auth.log"
        codec => "rubydebug"
        }
    }
}
```

**Success:**

```
<166>Feb 03 2020 11:27:05 5508x-1_9.12(3) : %ASA-6-113004: AAA user authentication Successful :
server =  192.168.1.50 : user = dain

<166>Feb 03 2020 11:27:05 5508x-1_9.12(3) : %ASA-6-113008: AAA transaction status ACCEPT : user
= dain

<166>Feb 03 2020 11:27:05 5508x-1_9.12(3) : %ASA-6-611101: User authentication succeeded: IP
address: 192.168.1.250, Uname: dain

<166>Feb 03 2020 11:27:05 5508x-1_9.12(3) : %ASA-6-605005: Login permitted from 192.168.1.250/59277
to management:192.168.1.9/ssh for user "dain"
```

**Failure:**

```
<166>2020-02-03T10:12:18-05:00 5508x-1_9.12(3) : %ASA-6-113005: AAA user authentication Rejected
: reason = AAA failure : server = 192.168.1.50 : user = dain : user IP = 192.168.1.250

<166>2020-02-03T10:12:18-05:00 5508x-1_9.12(3) : %ASA-6-611102: User authentication failed: IP
address: 192.168.1.250, Uname: dain

<166>2020-02-03T10:12:18-05:00 5508x-1_9.12(3) : %ASA-6-611102: User authentication failed: IP
address: 192.168.1.250, Uname: dain\n",

<166>2020-02-03T10:12:18-05:00 5508x-1_9.12(3) : %ASA-6-605004: Login denied from
192.168.1.250/58691 to management:192.168.1.9/ssh for user "dain"
```

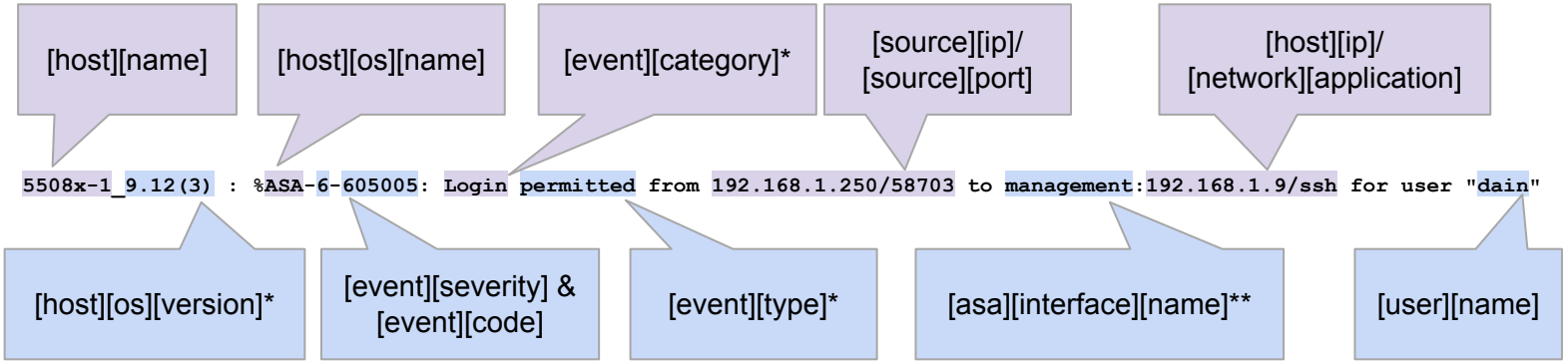
# ASA Logins - Step 3: Check configs

```
5508x-1# sh run logging
logging enable
logging timestamp ~~rfc~~5424
no logging hide username
logging trap informational
logging host management 192.168.1.50 17/5140

logging device-id string 5508x-1_9.12(3)
```

- Enable logging
  - Log timestamps (no rfc5424!)
- Show usernames in logs
- Log severity 1-6
- Log to filebeat default port from ASA management interface
- Custom host string includes Name & software version for ingest w/out enrichment

# ASA Logins - Step 4: What's in the log?




What I need to add or modify:

| | | |
|---|---|---|
| ecs.version | = | 1.4 |
| *event.category | = | Login -> authentication |
| *event.type | = | permitted / denied -> authentication_success OR authentication_failure |

**Custom field for ASA interface used as the destination (asa.interface.name), could add enrichment for source.ip

# ASA Logins - Step 5: Grok Check


```
{
  "[event][category]": "Login",
  "[host][os][name]": "ASA",
  "[event][code]": "605004",
  "[event][type]": "denied",
  "[source][port]": "60551",
  "[host][name]": "5508x-1",
  "[host][ip]": "192.168.1.9",
  "[network][application]": "ssh",
  "[syslog][facility]": "166",
  "[syslog][severity]": "6",
  "[host][os][version]": "9.12(3)",
  "[source][ip]": "192.168.1.250",
  "@timestamp": "Feb 03 2020 16:31:37",
  "[user][name]": "\"dain\"",
  "[asa][interface][name]": "management"
}
```


Grok:

```
<%{NUMBER:[syslog][facility]}>%{CISCOTIMESTAMP:@timestamp}
%{SYSLOGHOST:[host][name]}_%{NOTSPACE:[host][os][version]} :
%%{WORD:[host][os][name]}-%{INT:[syslog][severity]}-%{NOTSPACE:[ev
ent][code]}: %{WORD:[event][category]} %{WORD:[event][type]} from
%{IPORHOST:[source][ip]}/%{NUMBER:[source][port]} to
%{WORD:[asa][interface][name]}:%{IPORHOST:[host][ip]}/%{WORD:[net
work][application]} for user %{QS:[user][name]}
```

**To Dos:**

| | |
|---|---|
| event.category | => authentication |
| event.type | => authentication_success/failure |
| **event.outcome** | **=> success/failure (future SIEM)** |
| user.name | => strip quotes |
| ecs.version | => add |
| user.name | => enrichment? |
| source.ip | => enrichment? |

# ASA Logins - Final Logstash Additions

```
mutate {
        remove_field => [ "host" ]
        add_field => { "[ecs][version]" => "1.4" }
    }

    ## GROK WENT HERE

    if [event][category] == "Login" {
        mutate { update => { "[event][category]" => "authentication" }}
    }

    if [event][type] == "permitted" {
        mutate { update => { "[event][type]" => "authentication_success" }}
    }

    if [event][type] == "denied" {
        mutate { update => { "[event][type]" => "authentication_failure" }}
    }

    mutate {
        gsub => [ "[user][name]","\"","" ]
    }
```

- removing default host, to use host objects in Grok
- Add ecs.version

- Set event.category  (per SIEM Query)

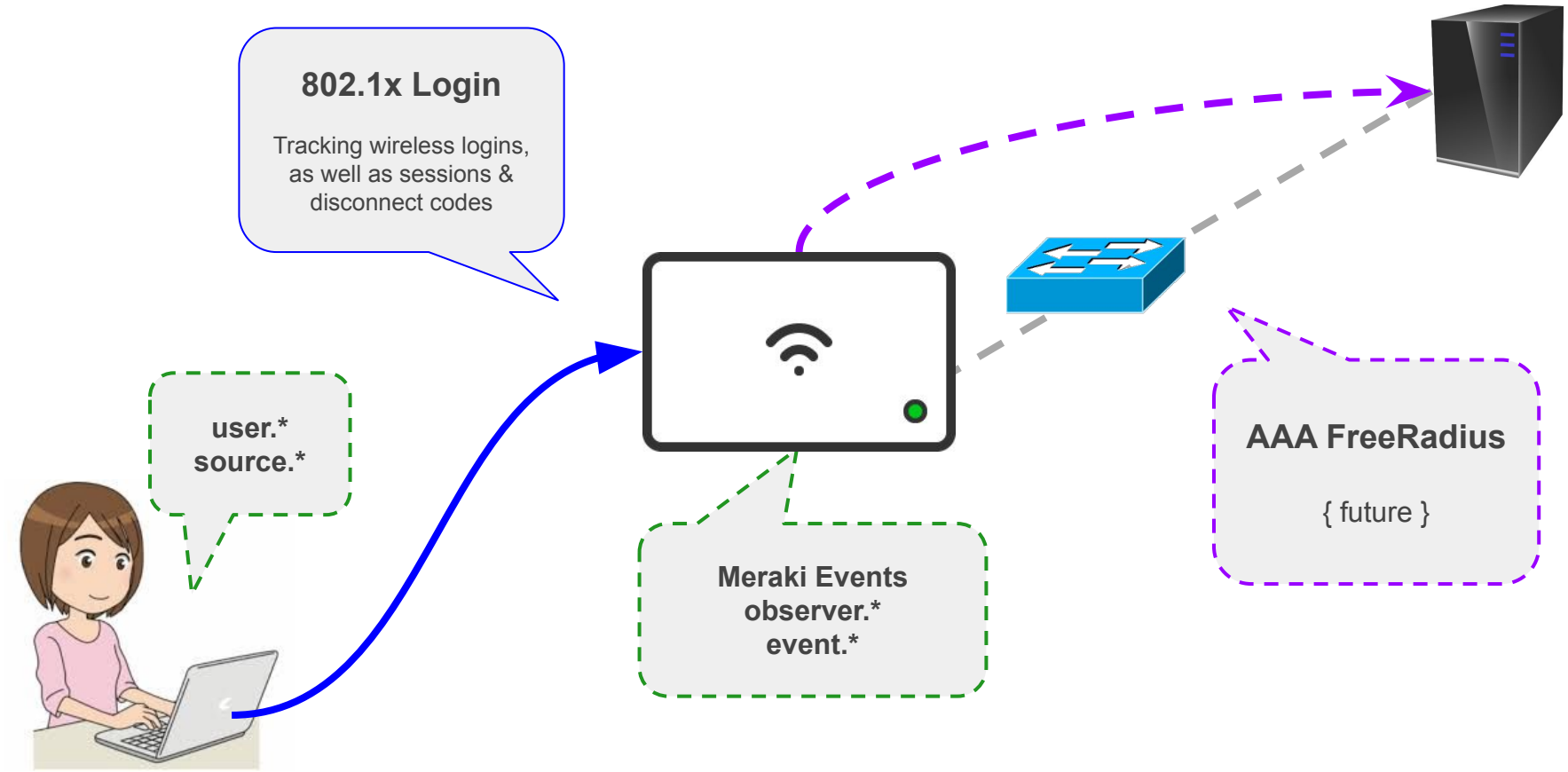
- Conditionals for event.type (per SIEM Query)

- Strip quotes

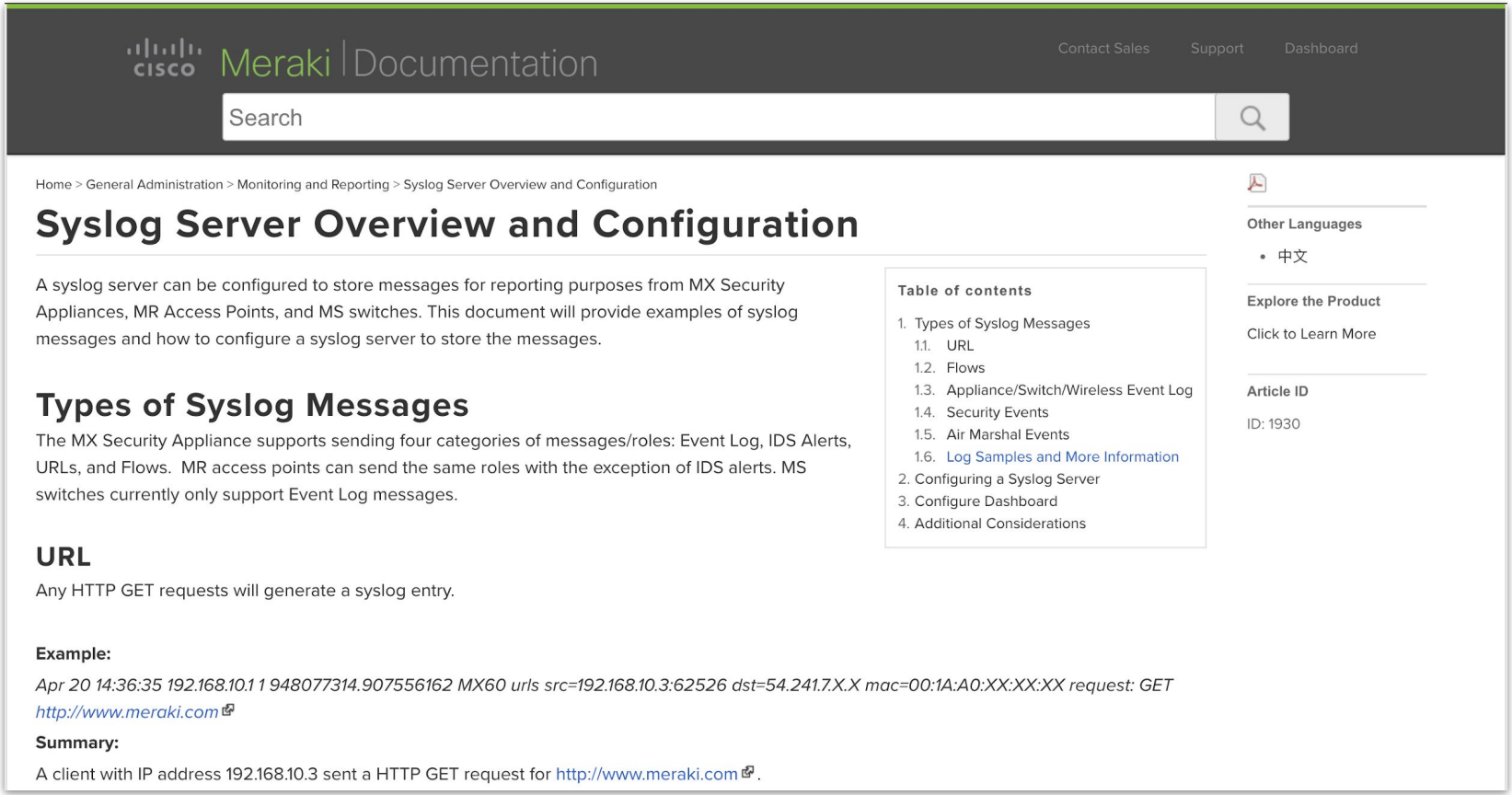
# Meraki 802.1x Logs

elastic

# Meraki 802.1x - Overview



# Meraki 802.1x - Step 1: Analyze what's coming in



cisco Meraki | Documentation

Contact Sales Support Dashboard

Search

Home > General Administration > Monitoring and Reporting > Syslog Server Overview and Configuration

## Syslog Server Overview and Configuration

A syslog server can be configured to store messages for reporting purposes from MX Security Appliances, MR Access Points, and MS switches. This document will provide examples of syslog messages and how to configure a syslog server to store the messages.

Table of contents

1. Types of Syslog Messages
   - 1.1. URL
   - 1.2. Flows
   - 1.3. Appliance/Switch/Wireless Event Log
   - 1.4. Security Events
   - 1.5. Air Marshal Events
   - 1.6. Log Samples and More Information
2. Configuring a Syslog Server
3. Configure Dashboard
4. Additional Considerations

### Types of Syslog Messages

The MX Security Appliance supports sending four categories of messages/roles: Event Log, IDS Alerts, URLs, and Flows. MR access points can send the same roles with the exception of IDS alerts. MS switches currently only support Event Log messages.

### URL

Any HTTP GET requests will generate a syslog entry.

**Example:**

*Apr 20 14:36:35 192.168.10.1 1 948077314.907556162 MX60 urls src=192.168.10.3:62526 dst=54.241.7.X.X mac=00:1A:A0:XX:XX:XX request: GET http://www.meraki.com*

**Summary:**

A client with IP address 192.168.10.3 sent a HTTP GET request for http://www.meraki.com.

Other Languages

- 中文

Explore the Product

Click to Learn More

Article ID

ID: 1930

elastic

https://documentation.meraki.com/zGeneral_Administration/Monitoring_and_Reporting/Syslog_Server_Overview_and_Configuration#URL

# Meraki 802.1x - Step 2: Verify what's coming in

```
### Logstash Meraki INPUT
input {
  udp {
    port  => 5141
    type => "meraki-dot1x"
    }
}

### Logstash Meraki Output
output {
  if "8021x" in [message] {
    ##DEBUG
    file {
        path => "/meraki.log"
        codec => "rubydebug"
        }
    }
}
```

**Successful Login**

```
<134>1 1580551704.928047208 1flr_1_ap events type=8021x_eap_success radio='1' vap='2'
client_mac='6C:96:CF:F0:BC:92' client_ip='192.168.2.109' identity='dain.perkins@elastic.co'
aid='1687088497'
```

**Failure:**

```
<134>1 1580666566.648676097 2flr_1_ap events type=8021x_eap_failure radio='0' vap='3'
client_mac='38:F9:D3:74:E2:71' client_ip='0.0.0.0' identity='dain.perkins@elastic.co'
aid='1442808319'
```

If we strip out the facility <134>1 @timestamp (in nanoseconds since epoch!), ap name and "events" we are left with key value pairs that are easy to manage!

I found Grok'ing the whole thing ended up being easier

# Meraki 802.1x - Step 3: What's in the log?

[host][name]

```
1flr_1_ap events type=8021x_eap_success radio='1' vap='2' client_mac='6C:96:CF:F0:BC:92' client_ip='192.168.2.109'
identity='dain.perkins@elastic.co' aid='1687088497'
```

Since the end of the log is KV pairs it was easy to dump into a spreadsheet to analyze

| Keys: | ECS Fields | Description |
|---|---|---|
| Aid | => [wireless][association.id] | Wireless Session ID |
| Channel | => [wireless][channel] | Radio Channel |
| Client_ip | => [source][ip] | IP Address of client ( **source** of auth request) |
| Client_mac | => [source][mac] | Mac Address of client ( **source** of auth request) |
| Identity | => [user][name] | Used USEREMAIL to parse |
| Radio | => [wireless][ap][radio_id] | AP Radio ID |
| Type | => 8021x_eap_(success/failure) | evaluate for event.category & event.type |
| Vap | => [wireless][ssid][id] | SSID |

To do's:

Add Host information (Meraki AP 25.12, etc.), fix Meraki host names (should be "." not "_")

Enrichment of host data, user data (normalize to common naming)?

# Meraki 802.1x - Step 4: What's in the log?


```
{
  "nanos": "928047208",
  "[host][name]": "1flr_1_ap",
  "[source][mac]": "6C:96:CF:F0:BC:92",
  "[syslog][facility]": "134",
  "[source][ip]": "192.168.2.109",
  "[user][name]": "dain.perkins@elastic.co",
  "millis": "1580551704",
  "eap_outcome": "8021x_eap_success",
  "[wireless][radio][id]": "1",
  "[wireless][ssid][id]": "2"
  "[wireless][association][id]": "1687088497"
}
```


Grok:

```
<%{NUMBER:[syslog][facility]}>%{CISCOTIMESTAMP:@timestamp}
%{SYSLOGHOST:[host][name]}_%{NOTSPACE:[host][os][version]} :
%%{WORD:[host][os][name]}-%{INT:[syslog][severity]}-%{NOTSPACE:[ev
ent][code]}: %{WORD:[event][category]} %{WORD:[event][type]} from
%{IPORHOST:[source][ip]}/%{NUMBER:[source][port]} to
%{WORD:[asa][interface][name]}:%{IPORHOST:[host][ip]}/%{WORD:[net
work][application]} for user %{QS:[user][name]}
```

**To Dos:**

| | |
|---|---|
| event.category | => authentication |
| event.type | => authentication_success/failure |
| **event.outcome** | **=> success/failure (future SIEM)** |
| ecs.version | => add |
| host.* info | => Meraki, AP, v25.13 |
| user.name | => normalization? |
| source.ip | => enrichment? |

# Meraki 802.1x - Final Logstash Additions

```
mutate {
        rename => ["host", "[host][ip]" ]
        add_field => { "[ecs][version]" => "1.4" }
      }
       ##   grok went here

       ##  Set event.type based on eap authentication
      if [eap_outcome] == "8021x_eap_success" {
        mutate {
          add_field => { "[event][type]" => "authentication_success" }
        } }
      if [eap_outcome] == "8021x_eap_failure" {
        mutate {
          add_field => { "[event][type]" => "authentication_failure" }
        }}

       mutate {
         add_field => { "[event][category]" => "authentication" }
        add_field => { "[host][os][name]" => "Meraki" }
        add_field => { "[host][os][platform]" => "AP" }
        add_field => { "[host][os][version]" => "25.13" }
        add_field => { "[event][category]" => "authentication" }
        remove_field => [ "millis", "nanos" ]
```

- removing default host, to use host objects in Grok
- Add ecs.version

- Conditionals for event.type (per SIEM Query)

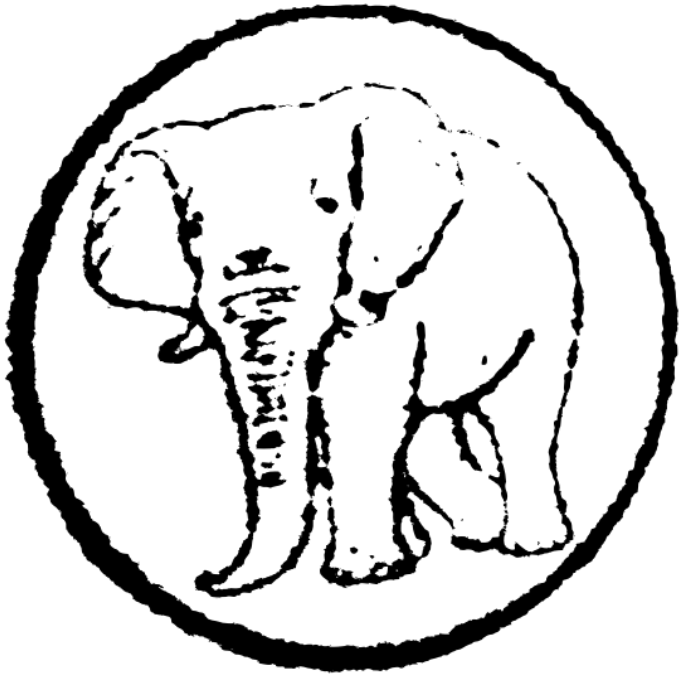
- Add host.* and event.category fields

# SIEM Check

# Wrap-up

# Things to remember

- Start with discrete use cases / events
- Check the SIEM app request/responses
- Work towards the visualization / filters
- Expand as necessary



# Links

| | |
|---|---|
| Elastic: | www.elastic.co |
| Products: | www.elastic.co/products |
| ECS Reference: | https://www.elastic.co/guide/en/ecs/current/index.html |
| ECS Github Project: | https://github.com/elastic/ecs |
| Discussion Group | https://discuss.elastic.co/tag/elastic-common-schema |
| Dain's Github*: | https://github.com/dainperkins |
| Community: | www.elastic.co/community/meetups |
| Twitter: | @elastic |


# Thank You