only Public IP addresses. We will then track the first occurrence of each users' login and sort the login time in ascending order:

```
_sourceCategory=Labs/AWS/CloudTrail
| json "userIdentity.userName","sourceIPAddress" as
user, ip nodrop
| where user matches "{{actor}}"
| where isPublicIP(ip)
| min(_messagetime) AS login_time BY user, ip
| sort BY user, +login_time
```

4. In this section, we are converting the IP address to a decimal and moving through the users' IP addresses and getting both the last login time and the previous login time:

   ```
   | ipv4ToNumber(ip) AS ip_decimal
   | backshift ip_decimal BY user
   | backshift login_time AS previous_login
   | where !(isNull(_backshift))
   ```

5. This section of code will convert the decimal IP address back to a IPv4 address:

   ```
    | toInt(floor(_backshift/pow(256,3))) AS octet1 |
   toInt(floor((_backshift-
   octet1*pow(256,3))/pow(256,2))) AS octet2 |
   toInt(floor((_backshift-
   (octet1*pow(256,3)+octet2*pow(256,2)))/256)) AS
   octet3 | toInt(_backshift-
   (octet1*pow(256,3)+octet2*pow(256,2)+octet3*256)) AS
   octet4 |
   concat(octet1,".",octet2,".",octet3,".",octet4) AS
   previous_ip
   ```

6. This section of the query is going to lookup the country information for each IP address.   We will filter out any NULL values.

```
| lookup latitude AS lat1, longitude AS long1,
country_name AS country_name1 FROM geo://location ON
ip
| lookup latitude AS lat2, longitude AS long2,
country_name AS country_name2 FROM geo://location ON
ip=previous_ip
| where !(isNull(lat1))
| where !(isNull(long1))
| where !(isNull(lat2))
| where !(isNull(long2))
```

7. In this section of code we will be using the Haversine function to calculate the Kilometers in distance between the two locations.

```
| haversine(lat1, long1, lat2, long2) AS distance_kms
```

8. Next we will calculate the time difference between the latest and previous logins and then calculate based on the distance, how fast that user must have been travelling in order to have personally logged in at both locations and times:

```
| (login_time - previous_login)/3600000 AS
login_time_delta_hrs
| distance_kms/login_time_delta_hrs AS
apparent_velocity_kph
| where apparent_velocity_kph > 0
```

9. Add the speed threshold by which the calculated speed above would be considered "suspicious" or "unrealistic":

```
| 500 AS suspicious_speed
| where apparent_velocity_kph > suspicious_speed
```

10. This last section of code will clean up the results and format for better presentation on our dashboard:

```
| concat(ip,", ",previous_ip) AS ip_addresses
| if(country_name1 <>
country_name2,concat(country_name1,",",country_name
2),country_name1) AS countries
| fields user, ip_addresses, countries,
distance_kms,login_time_delta_hrs,apparent_velocity
_kph
| where !isNull(user)
| where apparent_velocity_kph != "Infinity"
| sort by apparent_velocity_kph
```

11. Under chart type, select **Table**
12. Rename this panel **Landspeed Violation**
13. Click the **Add to Dashboard** button

Lab Activity 8 - Using Sumo Logic Threat Intelligence

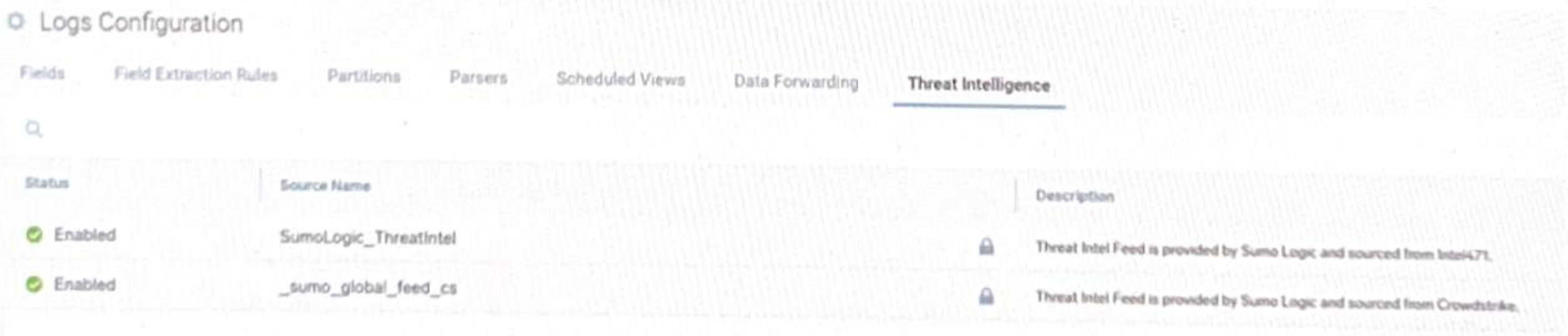
Threat intelligence is information that helps you prevent or mitigate cyber attacks. *Threat intelligence indicators* are individual data points about threats that are gathered from external sources about various entities such as host names, file hashes, IP addresses, and other known possible sources of attack and compromise.

Sumo Logic provides global feeds of threat intelligence indicators that can help security analysts leverage a large body of information to surface potential threats. Sumo users can also upload their own threat intelligence indicator feeds to add additional information to security log searches.

In this lab we'll walk through some of the basic Threat Intelligence features provided by the Sumo Logic platform.

To access the Threat Intelligence view, click **Data Management > Threat Intelligence** in the left nav menu (under the **Logs** header).

Any global feeds (from third party vendors such as Crowdstrike or Intel 471) will be listed here, along with any custom feeds. Global feeds cannot be edited or disabled.



Status	Source Name	Description
● Enabled	SumoLogic_ThreatIntel	Threat Intel Feed is provided by Sumo Logic and sourced from Intel471.
● Enabled	_sumo_global_feed_cs	Threat Intel Feed is provided by Sumo Logic and sourced from Crowdstrike.

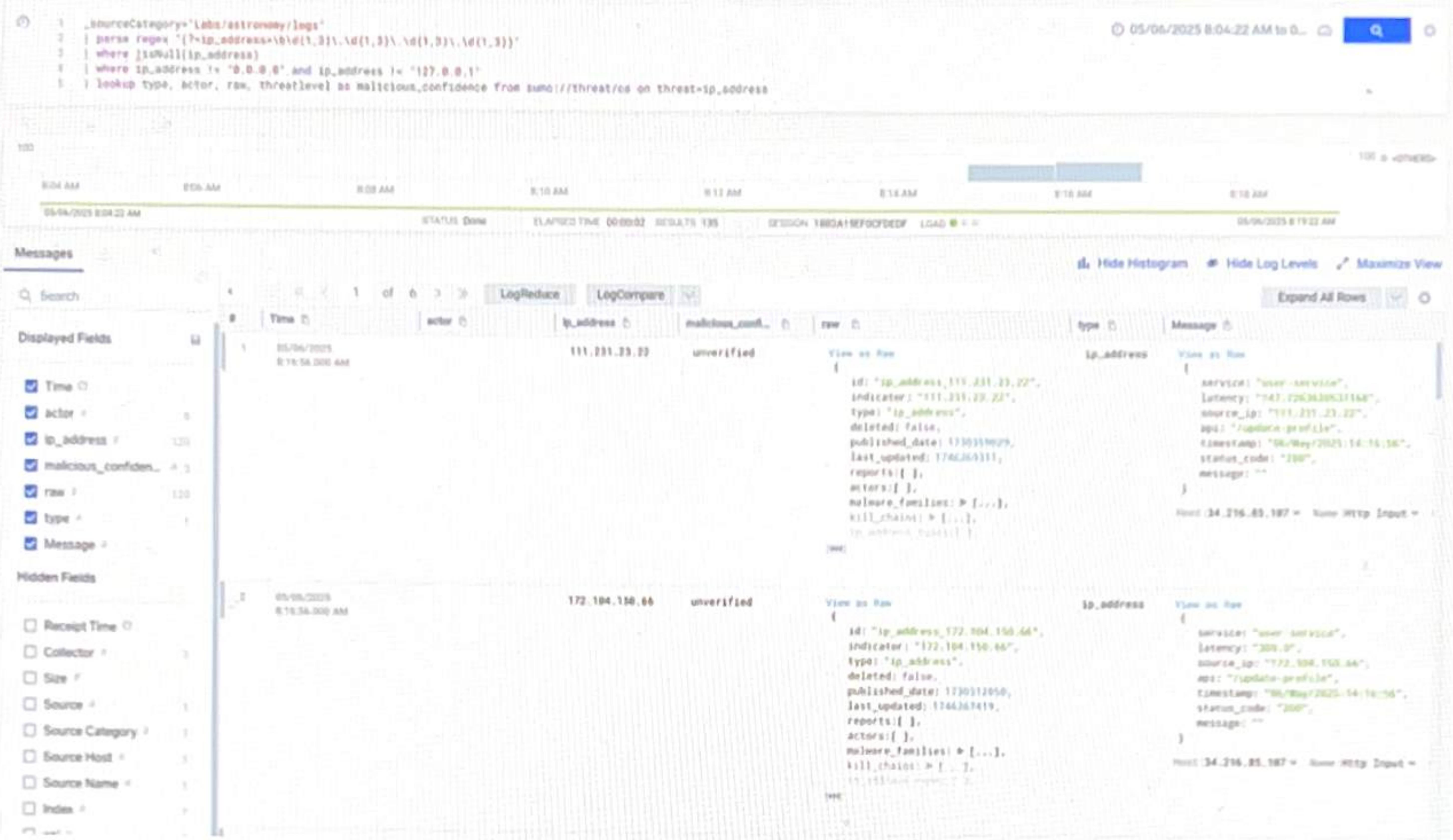
Threat indicators from global feeds can be used in search query lookups to cross-reference log entities with known threats from outside sources.

Use the `sumo://threat/<vendor>` syntax to reference specific vendor indicators within the global feeds. For instance, Crowdstrike data can be referenced using `sumo://threat/cs` while Intel 471 data can be referenced using `sumo://threat/i471`.

Let's run a sample log query to show how this can be done:

1. Select **Logs > Log Search** in the left nav menu.
2. In the log search query window, copy the following query:
```
_sourceCategory="Labs/astronomy/logs"
| parse regex
"(?<ip_address>\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| where !isNull(ip_address)
| where ip_address != "0.0.0.0" and ip_address !=
"127.0.0.1"
| lookup type, actor, raw, threatlevel as
malicious_confidence from sumo://threat/cs on
threat=ip_address
```

3. Hit **Enter** or click the magnifying glass button on the right side to run the query. You should see a number of log records with lookup information from the global feed added to the display.



```
_sourceCategory="Labs/astronomy/logs"
| parse regex "(?<ip_address>\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| where !isNull(ip_address)
| where ip_address != "0.0.0.0" and ip_address != "127.0.0.1"
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on threat=ip_address
```

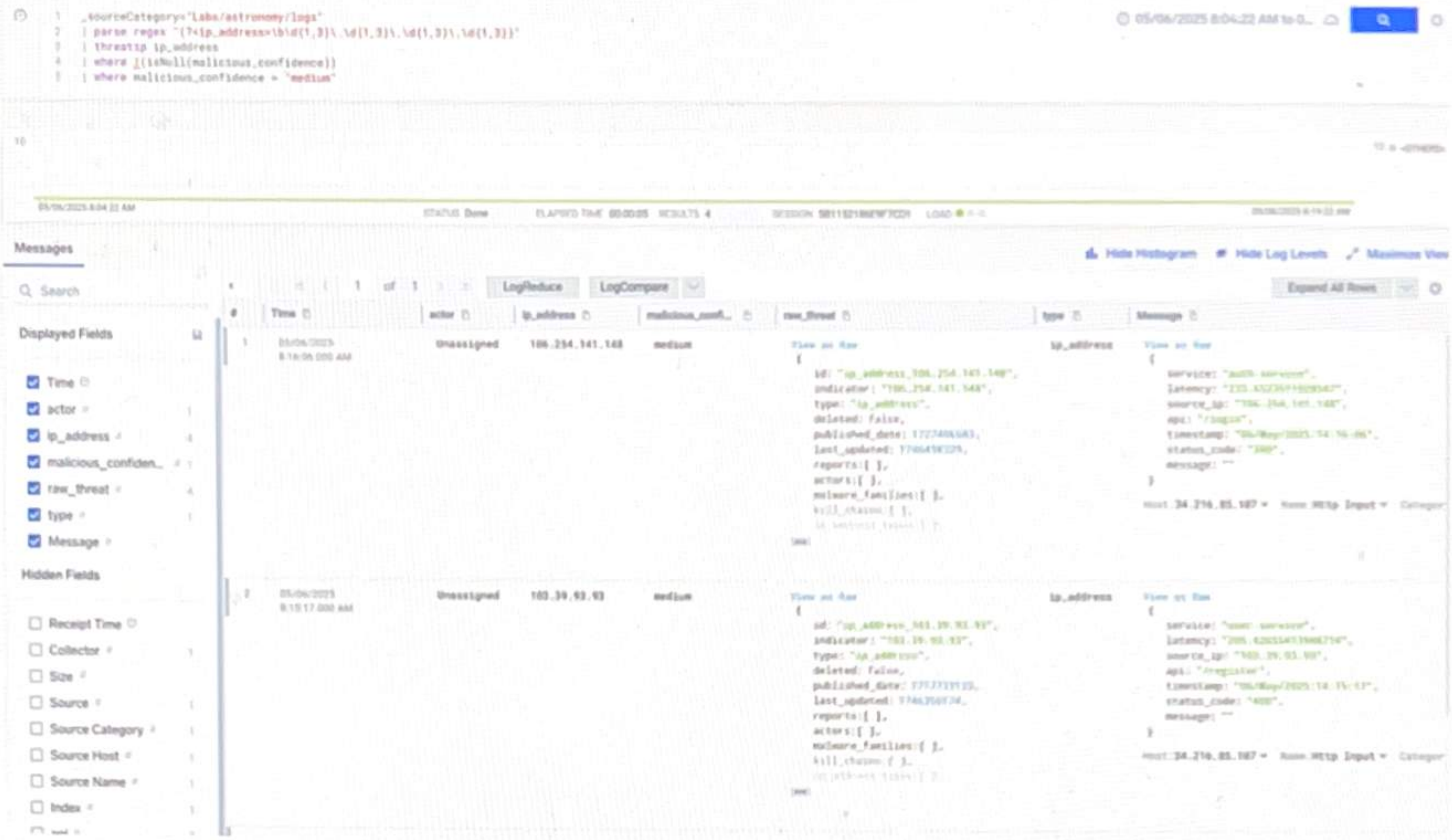
STATUS Done ELAPSED TIME 00:00:02 RESULTS 135 SESSION 180DA15EFOCFDEOF

#	Time	actor	ip_address	malicious_confi...	raw	type	Message
1	05/06/2025 8:19:56.000 AM		111.231.23.22	unverified	{ id: "ip_address_111.231.23.22", indicator: "111.231.23.22", type: "ip_address", deleted: false, published_date: [illegible], last_updated: [illegible], reports:[ ], actors:[ ], malware_families: ▶ [...], kill_chains: ▶ [...], ...	ip_address	{ service: "user-service", latency: [illegible], source_ip: "111.231.23.22", api: "/update-profile", timestamp: "06/May/2025:14:16:56", status_code: "200", message: "" }
2	05/06/2025 8:19:56.000 AM		172.104.150.66	unverified	{ id: "ip_address_172.104.150.66", indicator: "172.104.150.66", type: "ip_address", deleted: false, published_date: [illegible], last_updated: [illegible], reports:[ ], actors:[ ], malware_families: ▶ [...], kill_chains: ▶ [...], ...	ip_address	{ service: "user-service", latency: "309.0", source_ip: "172.104.150.66", api: "/update-profile", timestamp: "06/May/2025:14:16:56", status_code: "200", message: "" }

4. In the field browser on the left side, click on “malicious_confidence” to see the potential threat indicators according to the global feed, select a threat indicator value such as “medium” or “high” (if available) to look at just the records that have been tagged as higher risk according to the threat data in the global feed. (If needed, you can extend the time range for the query to encapsulate more data for inspection).
5. If you are specifically looking for threat indicators related to IP addresses, you can simplify your lookup queries by using the “threatip” operator instead. Copy/replace the current log query with the following:

```
_sourceCategory="Labs/astronomy/logs"
| parse regex
"(?<ip_address>\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| threatip ip_address
| where !(isNull(malicious_confidence))
```

6. Hit Enter or click the magnifying glass button to run the query.
7. In the field browser on the left side, click on “malicious_confidence” again and select a threat indicator value to inspect. (If needed, you can extend the time range for the query to encapsulate more data for inspection).



```
_sourceCategory="Labs/astronomy/logs"
| parse regex "(?<ip_address>\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| threatip ip_address
| where !(isNull(malicious_confidence))
| where malicious_confidence = "medium"
```

#	Time	actor	ip_address	malicious_confi...	raw_threat	type	Message

1	05/06/2025
	8:16:06.000 AM
		Unassigned	106.254.141.148	medium	View as Raw
					{
					id: "ip_address_106.254.141.148",
					indicator: "106.254.141.148",
					type: "ip_address",
					deleted: false,
					published_date: 1727406683,
					last_updated: 1746458229,
					reports:[ ],
					actors:[ ],
					malware_families:[ ],
					kill_chains:[ ],
					ip_address_types:[ ],
						ip_address	View as Raw
							{
							service: "auth-service",
							latency: "233.452359...",
							source_ip: "106.254.141.148",
							api: "/login",
							timestamp: "06/May/2025:14:16:06",
							status_code: "300",
							message: ""
							}
							Host 34.216.85.187 ▾ Name Http Input ▾ Category

2	05/06/2025
	8:15:17.000 AM
		Unassigned	103.39.93.93	medium	View as Raw
					{
					id: "ip_address_103.39.93.93",
					indicator: "103.39.93.93",
					type: "ip_address",
					deleted: false,
					published_date: 1717731125,
					last_updated: 1746350734,
					reports:[ ],
					actors:[ ],
					malware_families:[ ],
					kill_chains:[ ],
					ip_address_types:[ ],
						ip_address	View as Raw
							{
							service: "user-service",
							latency: "205.626554...",
							source_ip: "103.39.93.93",
							api: "/register",
							timestamp: "06/May/2025:14:15:17",
							status_code: "400",
							message: ""
							}
							Host 34.216.85.187 ▾ Name Http Input ▾ Category