

A glowing owl with large, feathered wings is centered in the image. The owl's body is a bright, golden-yellow color, and its eyes are large and white. Its wings are dark and textured, with a glowing purple and blue aura around them. The background is a deep space scene filled with numerous stars of various colors (blue, white, yellow) and nebulae in shades of purple and pink. The overall composition is symmetrical and visually striking.

**WISE**

With Intelligence See Everything

Andy Wick

---

# What is it

- **Moloch SPI data enhancer**
  - Can match on host/domains, md5, url, ip, ja3, email or almost any field
  - Can set almost any field in SPI data
  - Can add menu options (called right clicks still)
- **Supported data sources**
  - Simple Files
  - Commercial Services: OpenDNS, Emerging Threats Pro, Threatstream, ...
  - Elasticsearch/Redis
  - Splunk
- **Multilayer caching**
  - Capture
  - WISE Memory
  - Redis



---

# Sessions Example #1 - Subnets Database

Moloch uses WISE to query every single IP in a subnets database. Any matches creates new fields in sessions

## Subnets

Backplane ▾	1-GCI
Description ▾	Raptor Cluster sr#1-1735290491 Vlan257 TOOLS
Label ▾	PBY GCI
Security Zone ▾	CORP
Vlan ▾	205 257
Site ▾	corp-nyc1 corp-bf1 ▾



---

# Sessions Example #2 - Threatstream

Moloch uses WISE to query every single IP, domain, md5 in Threatstream. Matches add fields to sessions

## Threatstream

Severity ▾	very-high
Confidence ▾	24
Id ▾	466,860,800
Type ▾	mal_domain
Malware Type ▾	<a href="http://www.fireeye.com/blog/threat-research/2016/06/latest-android-overlay-malware-spreading-in-europe.html">http://www.fireeye.com/blog/threat-research/2016/06/latest-android-overlay-malware-spreading-in-europe.html</a>
Source ▾	Anomali Labs OSINT





# SPI View Example #3 - Threatstream

SPI View allows you to see all the unique values for each field with counts

**threatstream** Unload All Load All —

Search for fields in this category Confidence Import Id Malware Type Severity Source Type Confidence Cnt Id

⇩

**Confidence** 20<sup>(677)</sup> 24<sup>(578)</sup> 46<sup>(74)</sup> 40<sup>(71)</sup> 70<sup>(13)</sup> 73<sup>(12)</sup> 19<sup>(8)</sup> 26<sup>(8)</sup> 28<sup>(8)</sup> 90<sup>(8)</sup> 44<sup>(7)</sup> 85<sup>(4)</sup> 89<sup>(3)</sup> 75<sup>(2)</sup> 83<sup>(2)</sup> 88<sup>(2)</sup> 35<sup>(1)</sup> 48<sup>(1)</sup> 50<sup>(1)</sup> 72<sup>(1)</sup> 80<sup>(1)</sup> 81<sup>(1)</sup> 94<sup>(1)</sup> 100<sup>(1)</sup>

**Import Id** 256,569<sup>(16)</sup> 258,970<sup>(8)</sup> 260,164<sup>(8)</sup> 257,602<sup>(1)</sup>

**Malware Type** malware-fox-stealer<sup>(669)</sup> <http://www.fireeye.com/blog/threat-research/2016/06/latest-android-overlay-malware-spreading-in-europe.html><sup>(578)</sup>  
source:circi<sup>(83)</sup> fb-tx-id-1699985690012726<sup>(67)</sup> coinhive<sup>(16)</sup> alienvault<sup>(15)</sup> csit-17171<sup>(12)</sup> coin-hive<sup>(8)</sup> get-/lib/coinhive.min.js<sup>(8)</sup> dionaea<sup>(4)</sup> blocklist-brute-force-ips<sup>(3)</sup>  
fb-tx-id-1486882751358538<sup>(2)</sup> fb-tx-id-1525277244160318<sup>(2)</sup> 2e547e00c9b00c10127799f91323a9eb853fab6b<sup>(1)</sup> crisp-17-1218<sup>(1)</sup> running<sup>(1)</sup> sofacy<sup>(1)</sup>

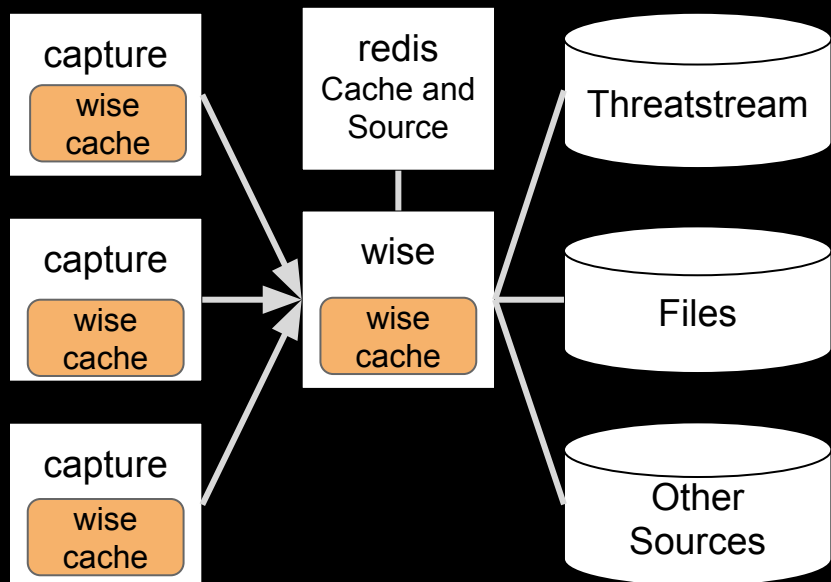
**Severity** very-high<sup>(1,367)</sup> low<sup>(101)</sup> medium<sup>(4)</sup>

**Source** CrowdStrike<sup>(681)</sup> Anomali Labs OSINT<sup>(578)</sup> verizon.com<sup>(83)</sup> Facebook ThreatExchange<sup>(71)</sup> Analyst<sup>(25)</sup> Crimeware Extractor<sup>(24)</sup>  
Alien Vault OTX - Malware C2 IP's<sup>(13)</sup> Malware-Traffic-Analysis.net<sup>(8)</sup> Blocklist Brute Force<sup>(3)</sup> CI Army<sup>(3)</sup> Alien Vault OTX Malicious IPs<sup>(2)</sup> Anomali Labs MHN<sup>(2)</sup>  
Anomali Labs MHN Tagged<sup>(2)</sup> Inactive - Anomali Labs MHN Community Malicious MD5s<sup>(2)</sup> Anomali Labs TOR Nodes<sup>(1)</sup> Emerging Threats - Compromised<sup>(1)</sup>  
Inactive - Anomali Labs Linux Malware<sup>(1)</sup> Maxmind Proxy List<sup>(1)</sup> Rulez.sk Brute Force IP<sup>(1)</sup> TOR Exit Nodes<sup>(1)</sup>

**Type** mal\_md5<sup>(745)</sup> mal\_domain<sup>(593)</sup> mal\_ip<sup>(114)</sup> adware\_domain<sup>(8)</sup> comm\_proxy\_ip<sup>(8)</sup> apt\_domain<sup>(5)</sup> bot\_ip<sup>(4)</sup> brute\_ip<sup>(4)</sup> scan\_ip<sup>(4)</sup> mal\_url<sup>(3)</sup> proxy\_ip<sup>(1)</sup> tor\_ip<sup>(1)</sup>



# Architecture

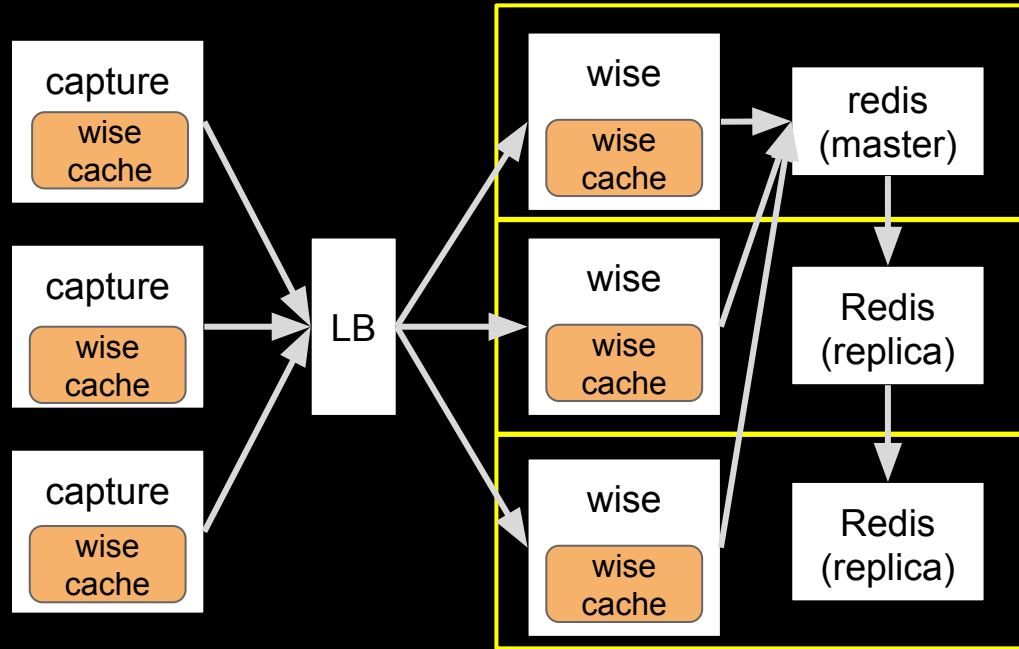


**For performance reasons lookups are cached at multiple layers.**

- 1) Check wise cache in capture (ALWAYS)
- 2) Check wiseService cache (for some sources)
- 3) Check redis cache (if configured)
- 4) Query the data source for information



# Architecture - Caching with multiple wise



---

# Capture & Viewer Configuration

# Set in [default] and/or for each capture node  
**wiseHost=wisehost.example.com**

# Semicolon ';' separated list of viewer plugins to load and the order to load in  
**viewerPlugins=wise.js**

# Semicolon ';' separated list of capture plugins to load and the order to load in  
**plugins=wise.so**





---

# Data source configuration

- **Like capture/viewer, everything in an ini file**
- **Each data source has its own section**
  - Some sections are unique like [threatstream]
  - Some sections have prefixes like [file:filename] and [url:urlName]
- **Most feeds just require simple configuration with defaults being good enough**
- **All WISE sources implement some common options**
  - cacheAgeMin - For those that cache
  - excludeDomains, excludeEmails, excludeURLs - don't lookup matching items
  - excludeEmails, excludeURLs - support wildcards
  - excludeIPs - CIDR
- **See [WISE settings page](#) for configuration options**



---

# Sample WISE Configuration

```
# wiseService contains global settings and global excludes
[wiseService]
excludeDomains=*.zen.spamhaus.org;*.in-addr.arpa;*.dnsbl.sorbs.net;*.ip6.arpa

[reversedns]
ips=192.168.0.0/16;10.0.0.0/8;172.20.0.0/21
field=asset

[file:badbadbad.ip]
file=/data/moloch/wisefiles/badbadbad.ip
tags=badbadbad
type=ip
format=tagger
```



---

# Tagger Format - badbadbad.ip

#field:whatever.str;kind:lotermfield;count:true;friendly:A  
String;db:whatever.str-term;help:Help for String;shortcut:0

#field:tags;shortcut:1

10.0.0.1;0=this is really bad;1=reallyBadTag

10.0.0.2;tags=anotherRealBadTag

10.0.0.3



# IPAM Example

## IPAM

Name ▾ Public space - Unused (was legacy DAHA) Dulles Campus Wireless  
Datacenter ▾ office none  
Security Zone ▾ office none

## ipam

Search for fields in this category

DataCenter ▾

Name ▾

Security Zone ▾

Security Zone Cnt ▾

DataCenter ▾ none<sup>(195,917)</sup> office<sup>(195,917)</sup>

Name ▾ Dulles Campus Wireless<sup>(195,917)</sup> Public space - Unused (was legacy DAHA)<sup>(195,917)</sup>

Security Zone ▾ none<sup>(195,917)</sup> office<sup>(195,917)</sup>

Security Zone Cnt ▾ 2<sup>(196,026)</sup>



---

# JSON Format - IPAM

[url:ipam]

type = ip

format = json

url = https://exempl.com/getipam.json

reload = 60

keyColumn = CIDR

fields=field:ipam.datacenter;kind:termfield;count:false;friendly:DataCenter;db:ipam.dc-term;help:DataCenter;shortcut:DataCenter\nfield:ipam.zone;kind:termfield;count:true;friendly:Security Zone;db:ipam.zone-term;help:Security Zone;shortcut:SecurityZone



---

# JSON Sample Data

```
[  
  {"DataCenter": "none",  
   "SecurityZone": "none",  
   "CIDR": "10.0.0.0/8"},  
  
  {"DataCenter": "none",  
   "SecurityZone": "office",  
   "CIDR": "10.66.0.0/16"}  
]
```





---

# Elasticsearch Source - Get username from panos

```
[elasticsearch:user]
type=ip
onlyIPs=10.10.0.0/16
elasticsearch=https://elk.example.com:9200
esIndex=panos-*
esTimestampField=@timestamp
esQueryField=sourceIP
esMaxTimeMS=86400000
esResultField=sourceUserName
fields=field:user;shortcut:sourceUserName
```

```
{"sourceIP" : "10.10.10.10",
"sourceUserName" : "andywick",
"@timestamp" : "2014-11-13T00:13:32.000Z", ...}
```

= Our VPN space

= index to search against

= what field has the timestamps

= field to check against

= range of data to search around

= what json field must exist in results

= what SPI data fields to set



---

# Example Users Display

Protocols ▾	tls tcp
IP Protocol ▾	tcp
Users ▾	vladp



---

# Splunk - Table Query

```
type = ip
format = json
host = splunk.host.example.com
port=5500
username={{wise.splunk.user}}
password={{wise.splunk.password}}
periodic=60
query=search index="vpnlog" sourcetype="vpn" assigned earliest=-24h | rex
"User <(?!<user>[^\>]+)>.*IPv4 Address <(?!<vpn_ip>[^\>]+)>" | dedup vpn_ip |
table user, vpn_ip
keyColumn=vpn_ip
fields=field:user;shortcut:user
```



---

# Right clicks

[right-click]

VTIP=url:https://www.virustotal.com/en/ip-address/%TEXT%/information/;name:Virus Total  
IP;category:ip

VTHOST=url:https://www.virustotal.com/en/domain/%HOST%/information/;name:Virus Total  
Host;category:host

VTURL=url:https://www.virustotal.com/latest-scan/%URL%;name:Virus Total URL;category:url

PTHOST=url:https://passivetotal.org/search/%TEXT%;name:Passivetotal Host;category:host

PTIP=url:https://passivetotal.org/search/%TEXT%;name:Passivetotal IP;category:ip

PTEMAIL=url:https://passivetotal.org/search/%TEXT%;name:Passivetotal User;category:user

(should be renamed “Field Actions”)



---

# Creating Views - Old way

```
this.api.addView("threatstream",
    "if (session.threatstream)\n" +
    "  div.sessionDetailMeta.bold Threatstream\n" +
    "  dl.sessionDetailMeta\n" +
    "    +arrayList(session.threatstream, 'severity-term', 'Severity',
'threatstream.severity')\n" +
    "    +arrayList(session.threatstream, 'confidence', 'Confidence',
'threatstream.confidence')\n" +
    "    +arrayList(session.threatstream, 'id', 'Id', 'threatstream.id')\n" +
    "    +arrayList(session.threatstream, 'importId', 'Import Id',
'threatstream.importId')\n" +
    "    +arrayList(session.threatstream, 'type-term', 'Type', 'threatstream.type')\n" +
    "    +arrayList(session.threatstream, 'maltype-term', 'Malware Type',
'threatstream.maltype')\n" +
    "    +arrayList(session.threatstream, 'source-term', 'Source', 'threatstream.source')\n"
)
```



---

# Creating Views - New way

```
require:threatstream;title:Threatstream;fields:threatstream
.severity,threatstream.confidence,threatstream.id,threatstr
eam.importId,threatstream.type,threatstream.maltype,threats
tream.source
```





---

# Wise Types

You can now add fields to already created wise types, or create new wise types

This examples add a new “mac” type and adds to the md5 type a new field “blahblah.md5”

```
[wise-types]  
mac=db:srcMac;mac.dst  
md5=db:http.md5;db:email.md5;db:blahblah.md5
```



---

# Todo

- Make creating new sources easier
- Add UI to see wise state and configuration
- Support multiple WISE servers on one machine better
- Include more examples with the release





**QUESTIONS?**