ElasticIntel

Scalable Threat Intel Aggregation in AWS
Obligatory Who I Am slide.

- Builder/Automator
- I put things in clouds
- Open Source Advocate
- <Insert credential alphabet soup here if it makes you feel better>
The ideas and opinions expressed in this talk....

Are not those of my employer, my imaginary friend, or anyone else
Warning: There will be Memes
The slide before the other slides...

Backstory and why I started this project

Current status of the project

Goals

Future plans
Why Build Elastic Intel?

- Researched products and services that could provide “Threat Intelligence”
- The findings were really bleak
- REALLY expensive
- The automation possibilities for most solutions were BAD
Value of threat intelligence
(Stolen from Scott Roberts)

1. Your own incidents
2. Vendor Reports
3. Honeypots
4. Peers/Sharing Communities
5. 3rd Party Paid Intelligence
Most vendor’s “Threat Intelligence” was just IOCs

Most IOCs were just recycled open-source feeds

Roughly 5-10% of IOCs produced by vendors were original material
“Intelligence”

- IOCs. JUST IOCs
- Occasionally we get WHOIS info
- Sometimes even LOSE what little context may have been available
“Intelligence”

YOU KEEP USING THAT WORD

I DO NOT THINK IT MEANS WHAT YOU THINK IT MEANS
Goodbye, context

- BotScout SSH brute force → “Malicious Activity”
- Phishing URL targeting O365 → “Credential Harvesting”
Targeted Intelligence!

Turns out to be a string search for some keywords

- Limited to 5 or 10 keywords
- Searches only across certain types of IOCs
- Google search results....
- Marketing campaigns...
What if we charged for EVERYTHING!
Entry cost

Ranged between $100k - $400k

Gets you...

A (Sometimes) pretty web GUI
Ability to make basic queries through web UI
Limited # of users (usually 1-5)
Pay To Play Model

- Want API access? $$
- Want more users? $$
- Want to add a feed? $$
- Want more contextual searches? $$$
- Want BULK api access? That’s your first born + $$$
- Want a chat bot? LOLOLOLOLOL
Automation & Integrations

We have an API but....

- You can only query it for single indicators
- You can only query it once a minute
- You can only query it 20 times a day (seriously)
- If you want more than 1 result in a list, you need to query it for every result
- If you want more than 50 queries, you can buy packs of 50/day for $10k
Well...
There’s Got to be a Better Way!
ElasticIntel

• Primary component is Elasticsearch
• Aggregates Threat Feeds (IOC feeds)
• Provides API for automating searches
• Slack bot for quick search
• Low Barrier to Entry (stand up with one command)
GOALS

- Gather roughly the same data (minus the 5-10% of custom data)
- Make it cost effective (I run a personal version)
- Make it performant (1k queries/min or better).
Continued...

- Make it usable
- Make it valuable regardless of company size
- Zero Maintenance (Or as close as possible)
- API as a first class citizen
“Serverless”

There is no cloud
it’s just someone else’s computer
● Don’t worry about patching
● Do worry about credentials and sensitive information
AWS Services

- Elasticsearch service
- Lambda
- SNS
- API Gateway
- S3
- Cloudwatch
- KMS
- IAM
Elasticsearch service

Managed Elasticsearch Cluster
Lambda

Serverless compute: You give it code, it runs it for you
Simple Notification Service – Pass Message and signals between resources
API Gateway

Acts as a RESTful proxy to Lambda
KMS
Key Management Service - Store access keys securely

IAM
Identity and Access Management - Permissions
S3

Storage

Cloudwatch

Logging
EVERYTHING is written in Python3

Please stop writing things in Python2

No really, stop it.
Infrastructure as Code

- Infrastructure is code too
- Terraform controls everything
- Python wrappers around Terraform abstract all actions
Feed Scheduler Lambda

- Runs once an hour
- Checks the schedule of all defined feeds
- If a feed is scheduled to be retrieved, it publishes a message to the SNS topic
Ingest Lambda Function

- Subscribed to SNS Topic
- Triggers on SNS Topic message publish
- Retrieves the data from the specified feed URL
- Parses the feed based on its type (txt, csv, etc.)
- Uploads to Elasticsearch Service
Lambda Performance

- Lambda’s are time bound (5m max)
- You only pay for used time
- Therefore....
Lambda optimization

- Whatever you’re trying to do, it better finish within 5 minutes
- The less time it takes your lambda to run, the cheaper it will be
Lambdas and Threading

- Threading is still a thing
- Its just...slightly harder
- Process Pools and Thread Pools not available
- Named pipes still work just fine though!

IO bound tasks can still be Parallelized in Lambda

- You just have to try a little harder
- (PS, WHOIS lookups are pretty good candidates for this 😊 )
Let’s talk performance...

- What’s the difference between:
- Making 100 queries
- Making 10 queries for 10 documents each
TCP/IP overhead is a real thing

- It doesn’t matter much when making a few requests
- When you make 5-10k in a few seconds...
Whois Lambda

- Runs every 3 minutes, across 15 different regions
- Grabs all IP addresses without WHOIS information
- Retrieves the whois information for them
- Updates the IOC in Elasticsearch with the whois data
Feeds and Feeds.d/

- All feeds are described as json blobs
- The json files representing feeds are all located in the feeds.d/ directory of the project
- Adding a feed is as simple as adding to an existing json file, or creating one in the directory
Example Feed

```json
{
  "feed_name": "Bambenek C2 IP Master High Confidence",
  "feed_type": "csv",
  "indicator_type": "ip_address",
  "field_mapping": {
    "separator": ",",
    "source": ["ip_address", "description", "created", "source_url"],
    "destination": ["ip_address", "description", "created", "source_url"],
    "has_headers": false
  },
  "check_interval": [8]
}
End Result

- 37 feeds = 250k IOCs/day
- TCO = (roughly) $1,600/yr.
- (Personal = $40-$50/month)
- Unlimited API access
- 10k+ queries/min
- Full-text search
Contributing (in order of difficulty)

Get your feet wet

- Add more feeds
- Build some Dashboards in Kibana
- Submit Feature Requests for ChatBot

Go for a brisk swim

- Work on API-based feed handlers
- Write unit tests
- Add a new integration like Shodan or Pastebin!
Don’t see anything up there that appeals to you or that you feel comfortable with? No problem!

Feature requests are very welcome. Just tag with [feature request] in the issue title on GitHub.

Questions? Hit me up on twitter or drop me an email
Links

Repo: https://github.com/securityclippy/elasticintel
Blog: https://blog.securelyinsecure.com/
Twitter: @PansyMcCoward
Email: SecurityClippy@securelyinsecure.com