the do-it-yourself guide to

ioc integration with splunk
who the ____ is this guy?

about me

- I am a digital security person with a diverse technology background and extensive experience building and supporting internet systems. I enjoy drawing on my varied skill set to understand emerging technologies, attack types, and other complex problems in the security space. My recent focus has been on continuous monitoring, forensics, incident response, and the creation of data-driven tools and solutions for these problems.

  - [github.com/adamjnichols](http://github.com/adamjnichols)
  - adamjnichols@gmail.com
  - twitter.com/adamjnichols

about the ministry

- The { Ministry of Promise } is dedicated to the evangelism of code-driven, data-centric solutions to the challenges posed by the modern global threat landscape. We believe that Orwellian data collection coupled with custom solutions based on open standards and technologies are the only way organizations stand a chance of defending themselves in the modern era.

  - [github.com/ministryofpromise](http://github.com/ministryofpromise)
  - info@ministryofpromise.co.uk
  - ministryofpromise.co.uk
i have a lot of opinions

- those expressed are my own, and do not reflect those of my employer
- i’ll probably keep many of them to myself
- if you’d like to hear more about them or share yours, please find me later!
what are we talking about again?

agenda

- threat intel
- what we did
- how we did it
- why we didn’t just buy a thing
- q&a
threat intel
disclaimer #2

- means many different things to many people
- remember what i said about opinions?
- i am not an expert
- many here are
- please seek them out accordingly
"threat intelligence is evidence-based knowledge, including context, mechanisms, indicators, implications and actionable advice, about an existing or emerging menace or hazard to assets that can be used to inform decisions regarding the subject's response to that menace or hazard."

- gartner
are iocs ‘intel’..?

- some would say no
- fintel > iocs, ipso facto...
- ...but
- even scant knowledge is power
- any information that improves your posture can be intel
- even if they aren’t intel, using iocs has become compulsory
still more opinions

- I don't like 'sharing formats'
- I don't think they work
- We won't use them here
- Allow the data to describe itself
- You should control of the way you use your data
- <insert obligatory xkcd here>
what we did
problem statement

WHY CAN'T I HOLD

ALL THESE THREAT INTELS?
no, but really...

- 2011-lish
- complete overhaul of dfir function; proactive focus
- lots of threat intel data coming online
- no ‘good’ way to use it
- we’ll make a database or something..?
- maybe one day it will be useful
ok, we need a thing...
ok, we need a thing...
what

ok, we need a thing...

SPLUNK

DATABASE THING
ok, we need a thing...
what kind of database thing?

- a thing to combine external (threat) data with audit (log) data for the purpose of detecting badness
- must require minimal care and feeding
- must be scalable
- must be modular
- should be highly usable and useful
could splunk work?

▸ maybe
▸ lots of complexity
▸ not a lot of flexibility
▸ cumbersome interface
▸ yo dawg, we heard you like to pay for intel...
why you might use splunk for threat data

▸ you’re still trying to figure out what you want to do
▸ you lack the resources to create your own solutions
▸ off-the-shelf integrations appeal to you
▸ you enjoy spending money on things
▸ you find the next few slides snore-worthy
why splunk for threat data is a bad idea

- index relationships in splunk kinda suck
- finding collisions via sub searches is slooooonoow
- finding multiple collisions via multiple sub searches is even mooooorrrrrreeee  sloooooowwwwerrrr
- complex queries are hard to write (well)
- only works for static, local data
- you enjoy usability
why splunk for threat data is a bad idea - a use case

- i want to find bad c2 activity on my firewall
- i'll start by looking at drops to external ip addresses...
- ...then try to find collisions with threat data
- improve fidelity of those collisions by using domain names associated with those ips to search for dns resolutions
- observed dns resolutions of known bad domains = bad
why splunk for threat data is a bad idea
why splunk for threat data is a bad idea
why splunk for threat data is a bad idea
why splunk for threat data is a bad idea
why splunk for threat data is a bad idea
why splunk for threat data is a bad idea - the worst bits

- how long would that query take to run?
- ...and over what timescale?
- how much data are you working with?
- what if your indexed threat data is incomplete, or out-of-date?
- multiply these problems x sources, logic streams, concurrency...
- ingestion, normalization, retention, etc.
- this is to say nothing of the narrow usefulness of these results
- live apis, anyone?
why splunk for threat data is a bad idea - some rebuttals

▸ “you just suck at splunk”
  ▸ probably

▸ “why not use a lookup table?”
  ▸ spreadsheets? for real?

▸ “my ‘security’ cloud offering/mssp/intel repo/feed/guy has a t.a. for that”
  ▸ if you like it, use it!
  ▸ we haven’t liked anything we’ve seen :(

▸ “isn’t building your own toolchain expensive?”
  ▸ as compared to what, exactly?
how we did it
a... database thing... to the rescue?!

- provide static and live data to Splunk at search time
- allow fine-grained control over what is requested/returned
- permit compound queries
- empower use of the pipeline
- do all this without impacting performance
dbthing is...

- commodity lamp-ish stack
- https server w/tls client auth serving a restful api
- some static html/js/css too
- mysql/redis for data
- python for data ingestion/etl
- good, fast, and cheap!
dbthing high-level architecture

- DATA TIER
  - DB
  - DB
  - DB
  - CACHE
  - CACHE
  - CACHE
  - ...

- HTTP/PHP TIER
  - DATA BROKER
  - ABUSE.CH
  - TOR EXITS
  - EMERGING THREATS
  - FOWLER PREP
  - PAID PROVIDER
  - PAID PROVIDER
  - ...

- UI COMPONENTS
  - /html
  - /js
  - /css
  - /img

- TO INTERPIPES

https://dbthing.com/<resource>/<plugin>
dbthing plug-in detail

how

API FRAMEWORK

https://dbthing.com/<resource>/<plugin>

ABUSE.CH

TOR EXITS

ET

FOWLER IPREP

CACHE

DATA TIER

HTTP/PHP TIER

DATA BROKER

- receive request
- determine type (regex)
- handle sanity

- manage response object
- provide response

if(query == $IP){
  if(data != null){
    return data;
  }
  else {
    return "no results";
  }
}

if(query == $IP){
  if(data != null){
    return data;
  }
  else {
    return "no results";
  }
}

if(query == $DOMAIN){
  if(data != null){
    return data;
  }
  else {
    return "no results";
  }
}

if(query == $IP){
  if(data != null){
    return data;
  }
  else {
    return "no results";
  }
}
**dbthing plug-in modularity**

**DATA BROKER**
- receive request
- determine type (regex)
- handle sanity

**API FRAMEWORK**

**DATA TIER**
- ABUSE.CH DB
- TOR EXITS DB
- ET DB

**HTTP/PHP TIER**

**FOWLER IPREP**
- manage response object
- provide response

**FOWLER IPREP CACHE**

**FUNCTIONALITY**
- if(query == $IP){
  if(data != null){
    return data;
  }
  else {
    return “no results”;
  }
  if(query == $DOMAIN){
    if(data != null){
      return data;
    }
    else {
      return “no results”;
    }
    if(query == $IP){
      if(data != null){
        return data;
      }
      else {
        return “no results”;
      }
    }
  }
}

**URL**
https://dbthing.com/<resource>/<plugin>
**dbthing plug-in modularity**

- **DATA TIER**
  - ABUSE.CH DB
  - TOR EXITS DB
  - ET DB

- **API FRAMEWORK**

- **HTTP/PHP TIER**
  - DATA BROKER
    - receive request
    - determine type (regex)
    - handle sanity
  - EMERGING Threats
    - manage response object
    - provide response

- **https://dbthing.com/<resource>/<plugin>**
**dbthing plug-in modularity**

How dbthing works:

1. **HTTP/PHP Tier**
   - Data Broker
     - Receive request
     - Determine type (regex)
     - Handle sanity

2. **Data Tier**
   - Abuse.CH DB
   - Tor Exits DB
   - ET DB

3. **API Framework**
   - Manage response object
   - Provide response

4. **Moon Domainrep Cache**
   - Moon Domainrep

Example code snippet:

```php
if(query == $IP){
    if(data != null){
        return data;
    } else {
        return "no results";
    }
} else {
    return "no results";
}
```

API URL:

https://dbthing.com/<resource>/<plugin>
dbthing modularity

- seamlessly board new data
- easily jettison poor (viking grade) data
- minimal changes required
- sometimes as quick as a few minutes
the splunk side of dbthing

- leverages splunk custom commands
- essentially a python script that imports a module (splunk.Intersplunk)
- can interact directly with search results in the pipeline
- employs threaded parallelism to decrease latency
- very lightweight; no app, no t.a.
- augments log events with additional fields in-line
user starts with a query that yields a result set of interest
how

the splunk side of dbthing - how it works

```
index=proxy_index sourcetype=proxy_sourcetype auth_group=executives
|   stats count by dst_ip
|   search dst_ip!=192.168.0.0/16
|   dbthing dst_ip
|   <do other cool splunk stuff here>
```

- user invokes dbthing, providing field of interest as arg
the splunk side of dbthing - how it works

- user invokes dbthing, providing field of interest as arg
- dbthing python queries dbthing rest api
- value of the arg field in each event is the resource that dbthing queries

```bash
index=proxy_index sourcetype=proxy_sourcetype auth_group=executives
| stats count by dst_ip
| search dst_ip!=192.168.0.0/16
| dbthing dst_ip
| <do other cool splunk stuff here>
```
the splunk side of dbthing – how it works

```
index=proxy_index sourcetype=proxy_sourcetype auth_group=executives
| stats count by dst_ip
| search dst_ip!=192.168.0.0/16
| dbthing dst_ip
| <do other cool splunk stuff here>
```

- user invokes dbthing, providing field of interest as arg
- dbthing python queries dbthing rest api
- value of the arg field in each event is the resource that dbthing queries
- all available data is added to json response from dbthing
- those json fields are added to splunk event as k=v pairs
- augmented data is returned to pipeline via Intersplunk
how

the splunk side of dbthing - how it works

**SPLUNK**

**SPLUNK EVENT**

src_ip=16.180.70.237 http_method=GET http_response=200
http_domain=bestcyclingreviews.com

**DBTHING CMD**

| dbthing src_ip |
| search src_ip_match=true |

**DBTHING PYTHON**

**SPLUNK EVENT**

src_ip=16.180.70.237 http_method=GET http_response=200
http_domain=bestcyclingreviews.com

**DBTHING API REQUEST**

GET https://dbthing.com/dbthing/16.180.70.237

**DBTHING API RESPONSE**

```json
{
  "match": true,
  "match_count": 2,
  "et_match": false,
  "fowler_match": false,
  "k_match": false,
  "match_count": 0,
  "match_type": "host,host",
  "paid_provider_detectedat": "2016-05-29T13:37:55.000-0700",
  "paid_provider_id": "1699279",
  "paid_provider_ip": "16.180.70.237",
  "paid_provider_malware": "1",
  "paid_provider_match": true,
  "paid_provider_matchtype": "host,host",
  "paid_provider_rank": "1",
  "paid_provider_score": "46",
  "paid_provider_spam": "",
  "siirt_block": "true",
  "siirt_blockloc": "pxy",
  "siirt_comments": "",
  "siirt_confidence": "high",
  "siirt_created_by": "bobama",
  "siirt_description": "Angler",
  "siirt_effective_date": "2016-05-31 00:00:00",
  "siirt_indicator": "16.180.70.237",
  "siirt_match": true,
  "siirt_source": "intel sharing",
  "success": true,
  "torexits_match": false
}
```
the splunk side of dbthing – how it works
the splunk side of dbthing - how it works
the splunk side of dbthing - how it works
the splunk side of dbthing - how it works

**SPLUNK**

**SPLUNK EVENT**

```
"GET /handle-bars HTTP/1.0" 200 2527 "http://
bestcyclingreviews.com/top_online_shops"
"Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/
20100101 Firefox/31.0"
```

src_ip=16.180.70.237  http_method=GET  http_response=200
http_domain=bestcyclingreviews.com

**DBTHING**

**DBTHING API REQUEST**

GET https://dbthing.com/dbthing/16.180.70.237

**DBTHING API RESPONSE**

```
{'et_match': 'false',
 'fowler_match': 'false',
 'k_match': 'false',
 'match': 'true',
 'match_count': 2,
 'paid_provider_detectedat': '2016-05-29t13:37:55.000-0700',
 'paid_provider_ip': '16.180.70.237',
 'paid_provider_id': '1699279',
 'paid_provider_malware': '1',
 'paid_provider_match': 'true',
 'paid_provider_matchtype': 'host,host',
 'paid_provider_rank': '2147483647',
 'paid_provider_score': '46',
 'paid_provider_spam': '',
 'siirt_block': 'true',
 'siirt_blockloc': 'pxy',
 'siirt_comments': '',
 'siirt_confidence': 'high',
 'siirt_created_by': 'bobama',
 'siirt_description': 'Angler ',
 'siirt_effective_date': '2016-05-31 00:00:00',
 'siirt_indicator': '16.180.70.237',
 'siirt_match': 'true',
 'siirt_source': 'intel sharing',
 'success': 'true',
 'torexits_match': 'false'}
```
the splunk side of dbthing - how it works

**SPLUNK**

**SPLUNK EVENT**
"GET /handle-bars HTTP/1.0" 200 2527 "http://bestcyclingreviews.com/top_online_shops"
"Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/20100101 Firefox/31.0"

src_ip=16.180.70.237 http_method=GET http_response=200
http_domain=bestcyclingreviews.com

**DBTHING**

**DBTHING API REQUEST**
GET https://dbthing.com/dbthing/16.180.70.237

**DBTHING API RESPONSE**
{"et_match": "false",
"fowler_match": "false",
"k_match": "false",
"match": "true",
"match_count": 2,
"paid_provider_detectedat": '2016-05-29T13:37:55.000-0700',
"paid_provider_ip": '16.180.70.237',
"paid_provider_id": '1699279',
"paid_provider_malware": '1',
"paid_provider_match": "true",
"paid_provider_matchtype": 'host,host',
"paid_provider_rank": '2147483647',
"paid_provider_score": '46',
"paid_provider_spam": '',
"siirt_block": 'true',
"siirt_blockloc": 'pxy',
"siirt_comments": '',
"siirt_confidence": 'high',
"siirt_created_by": 'bobama',
"siirt_description": 'Angler',
"siirt_effective_date": '2016-05-31 00:00:00',
"siirt_indicator": '16.180.70.237',
"siirt_match": 'true',
"siirt_source": 'intel sharing',
"success": 'true',
"torexits_match": 'false'}
the splunk side of dbthing - how it works

SPLUNK EVENT
"GET /handle-bars HTTP/1.0" 200 2527 "http://bestcyclingreviews.com/top_online_shops"
"Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/20100101 Firefox/31.0"

src_ip=16.180.70.237 http_method=GET http_response=200
http_domain=bestcyclingreviews.com

DBTHING CMD
| dbthing src_ip
| search src_ip_match=true

DBTHING PYTHON
"GET /handle-bars HTTP/1.0" 200 2527 "http://bestcyclingreviews.com/top_online_shops"
"Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/20100101 Firefox/31.0"

src_ip=16.180.70.237 http_method=GET http_response=200
http_domain=bestcyclingreviews.com src_ip_match=true
src_ip_success=true src_ip_match=true src_ip_match_count=2
src_ip_fowler_match=false
....

DBTHING API REQUEST
GET https://dbthing.com/dbthing/16.180.70.237

DBTHING API RESPONSE
{"et_match': 'false',
'fowler_match': 'false',
'k_match': 'false',
'match': 'true',
'match_count': 2,
'paid_provider_detectedat':
'2016-05-29T13:37:55.000-0700',
'paid_provider_ip': '16.180.70.237',
'paid_provider_id': '1699279',
'paid_provider_malware': '1',
'paid_provider_match': 'true',
'paid_provider_matchtype': 'host,host',
'paid_provider_rank': '2147483647',
'paid_provider_score': '46',
'paid_provider_spam': '',
'siirt_block': 'true',
'siirt_blockloc': 'pxy',
'siirt_comments': '',
'siirt_confidence': 'high',
'siirt_created_by': 'bobama',
'siirt_description': 'Angler',
'siirt_effective_date': '2016-05-31 00:00:00',
'siirt_indicator': '16.180.70.237',
'siirt_match': 'true',
'siirt_source': 'intel sharing',
'success': 'true',
torexits_match': 'false'}
the splunk side of dbthing
how

the splunk side of dbthing - nuance

- name of query field is prepended to all results
  - this permits multiple queries into dbthing within a single result set

- the speed of light is rough on splunk queries
  - live data APIs are disabled with default args; must be explicitly requested
  - use encouraged toward end of query lifecycle for best results

- only flat json is useful :(
  - complex documents make for poor splunk k=v pairs

- this enrichment approach works well for minimal data

- falls down hard with large amounts
  - consider adding 23k passive dns records to a single splunk event
the splunk side of dbthing - big data problems

- solution is to invert the data
- events become enrichment for threat data
- provides same functionality with finer granularity
- neat way to overcome flat json limitation
  - recur, recur, recur
- minor gotchas
  - counts get weird
  - result sets get huge
- this can be tricky; happy to talk more offline
the ui side of dbthing

- data needs curation
- data should be useful outside of splunk
- soc people need love too
- provides a great mechanism for deploying additional tooling
how

The UI side of dbthing
how

the ui side of dbthing

<table>
<thead>
<tr>
<th>registrant</th>
<th>adam nichols</th>
</tr>
</thead>
<tbody>
<tr>
<td>created</td>
<td>2014-05-14</td>
</tr>
<tr>
<td>expires</td>
<td>2018-05-14</td>
</tr>
<tr>
<td>updated</td>
<td>2016-05-09</td>
</tr>
<tr>
<td>date</td>
<td>2016-05-12</td>
</tr>
<tr>
<td>record source</td>
<td>ministryofpromise.co.uk</td>
</tr>
<tr>
<td>parsed url</td>
<td><a href="http://www.enom.com">http://www.enom.com</a></td>
</tr>
<tr>
<td>parsed registered on</td>
<td>14-may-2014</td>
</tr>
<tr>
<td>parsed expiry date</td>
<td>14-may-2018</td>
</tr>
<tr>
<td>parsed last updated</td>
<td>09-may-2016</td>
</tr>
</tbody>
</table>
the ui side of dbthing - some features

- full dbthing query stack
- ioc addition with auto-extraction
- ioc filtering
- remote deployment of netflix sketchy for screenshots
- local copy of regexr for testing without leakage
- other random bits we like
the ugly side of dbthing - what we screwed up

- no solid metrics or visibility into usage
- filtering could be way better
- ui could be way, way better (terrible bootstrap template)
- data storage could be more efficient
- data could be better organized, and much more useful
- collision paradigm could be used as both a +/- filter
dbthing - next gen

- full mean stack
- mongo/redis for data storage
- express/angular/node for api/ui
- homogeneous javascript makes your ocd sing
- full ioc audit trails, data tagging, grouping, type-less indicators
- data inversion is the new hotness
- way more...
how

ui side of dbthing - future state
that sounds swell, but how in the world am I even supposed to begin?

you (maybe)
why don’t you buy this 2u thing that will solve all your whiz-bang-threat-intel-uba-heuristical needz?
why we didn’t just buy a thing
why didn’t we just buy a thing?

- there really wasn’t anything to buy (c. 2011-ish)
- subsequent solutions don’t measure up
- surprisingly expensive
- heterogeneity makes workflow terrible
- homogeneity is even more expensive
- (feature requests + startups) / vc demands = you lose
- “if you want something done right...”
why shouldn’t i just buy a thing?

- you probably should
- it might be more expensive...
- ... but the experience is priceless
- very difficult to solve for x if you don’t know algebra
- necessity is the mother of invention
but i really want to build a thing

- now you’re talking!
- what brought you to that conclusion?
- are you positioned to use your dbthing effectively?
- do you have skills in-house to do that work?
  - if the answer here is “no” you should reconsider
  - timely support of detective controls is sorta mission critical
- can you split the difference, and start with an open-source tool?
why

using foss to solve the problem

- literally tons of tools available
- conversations on this topic happen here often
- mitre crits, misp, many others
- excellent first foray into threat/ioc data management
  - low initial cost
  - great test of resources, skills
  - minimal risk way to dip a toe
- we open-source what we can!
- hopefully more soon...
why

what you’ll need to start

- a siem/analytics platform of some type (probably splunk)
- a solid understanding of your data
  - good storage/query/efd/efs strategies
  - normalization!
  - strong logic/detective foundation
- some (preferably free) threat data to correlate
- a system to house that data
literally any system will do

actual dbthing v1 production infrastructure platform solution
why

ok i’m all in, now what?

▸ start slow
  ▸ get a single, simple, high-fidelity integration working
  ▸ ease into volume/scale
  ▸ tune as necessary
  ▸ rinse/repeat

▸ refactoring is ok, necessary, and to be expected

▸ you’re never done

▸ you’re never happy

▸ enjoy the journey! building cool stuff is fun!

▸ feel free to reach out!
  ▸ always happy to chat, talk shop, or commiserate over beer (mmm, beer)
good luck!