Intelligent Hunting

Using Threat Intelligence to Guide Your Hunts

whoami

Keith Gilbert

Security Technologist @ sqrrl

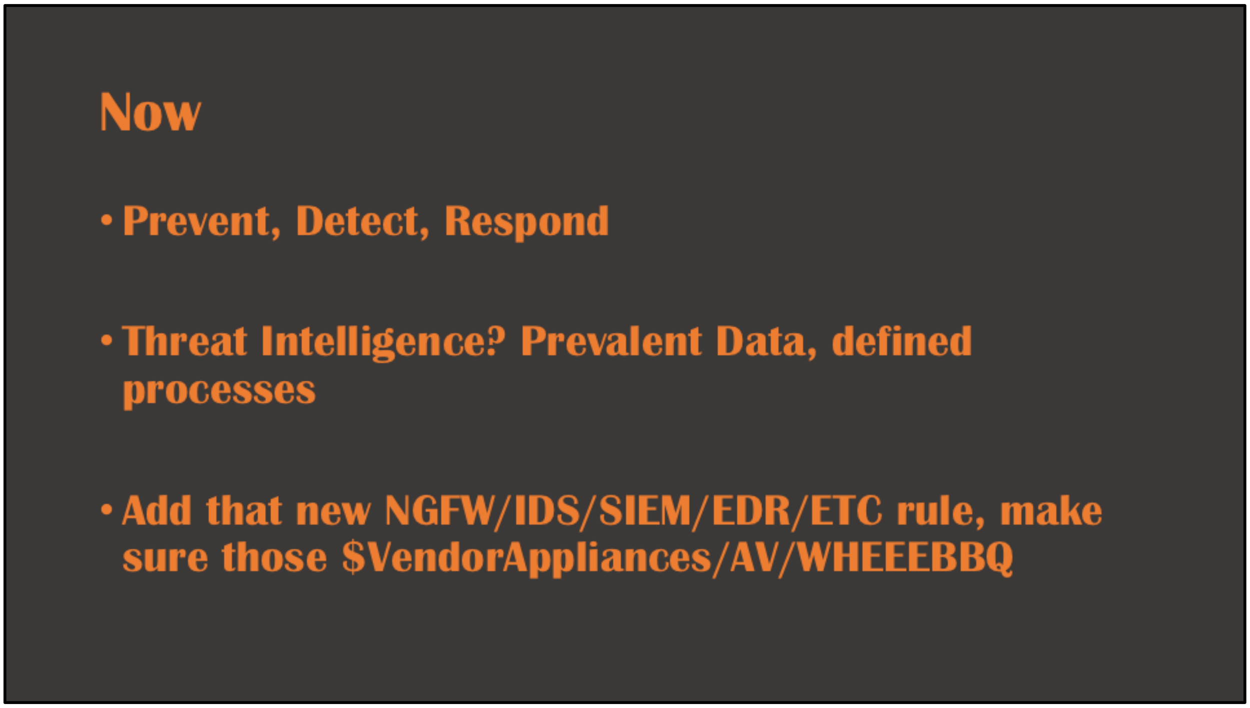
Founder @ MALFORMITYLABS

Yes, my office walls are orange and grey

Why This Talk?

Then

- Prevent! Prevent! Prevent!
- Add that new Firewall & IDS Rule, Update AV/HIPS
- Threat Intelligence? You mean Government?

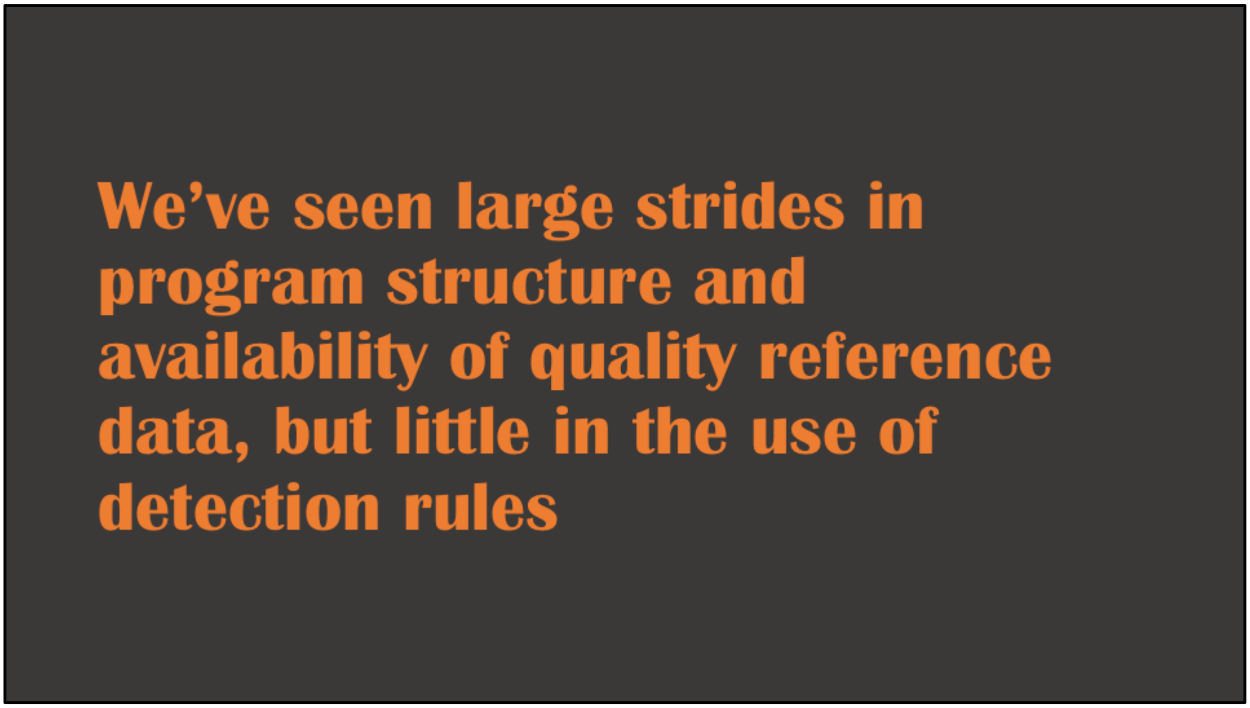


Now

- Prevent, Detect, Respond

- Threat Intelligence? Prevalent Data, defined processes

- Add that new NGFW/IDS/SIEM/EDR/ETC rule, make sure those $VendorAppliances/AV/WHEEEBBQ

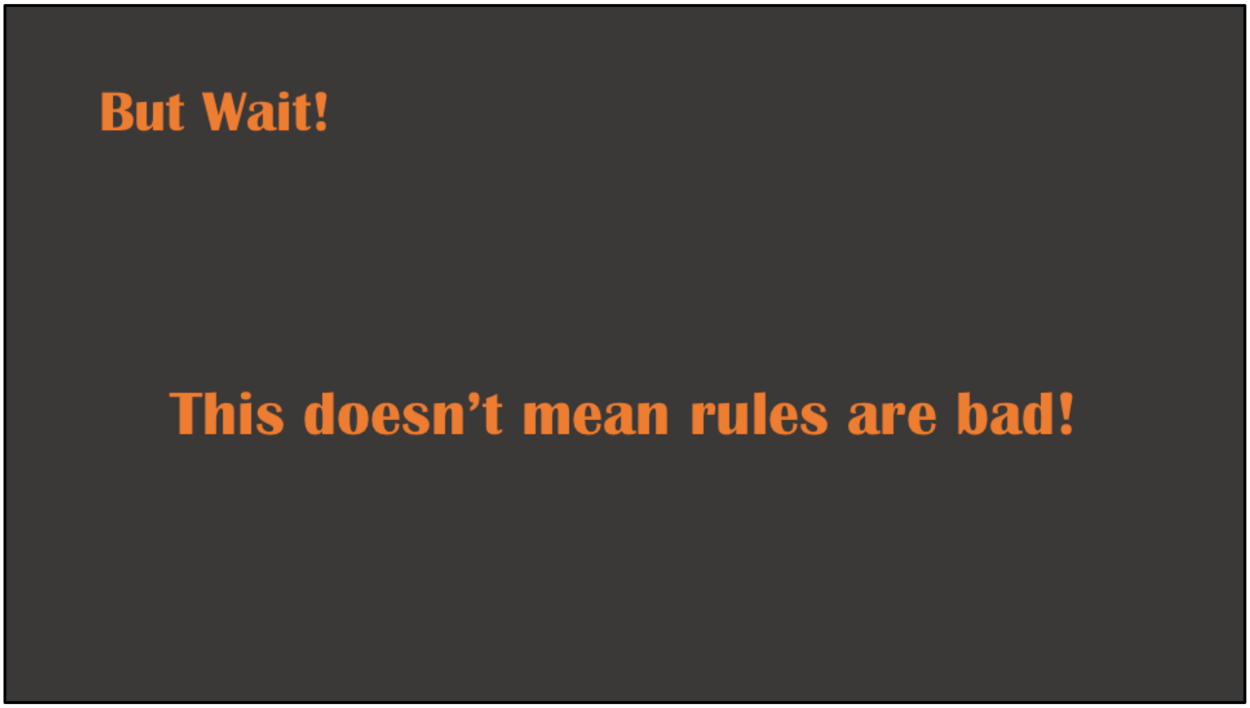


We’ve seen large strides in program structure and availability of quality reference data, but little in the use of detection rules

Transitioned from prevention focused mentality to widely accepted knowledge that detection and response are as, if not more, important than prevention

The level and quality of data available or shared via both open and closed methods is far better.

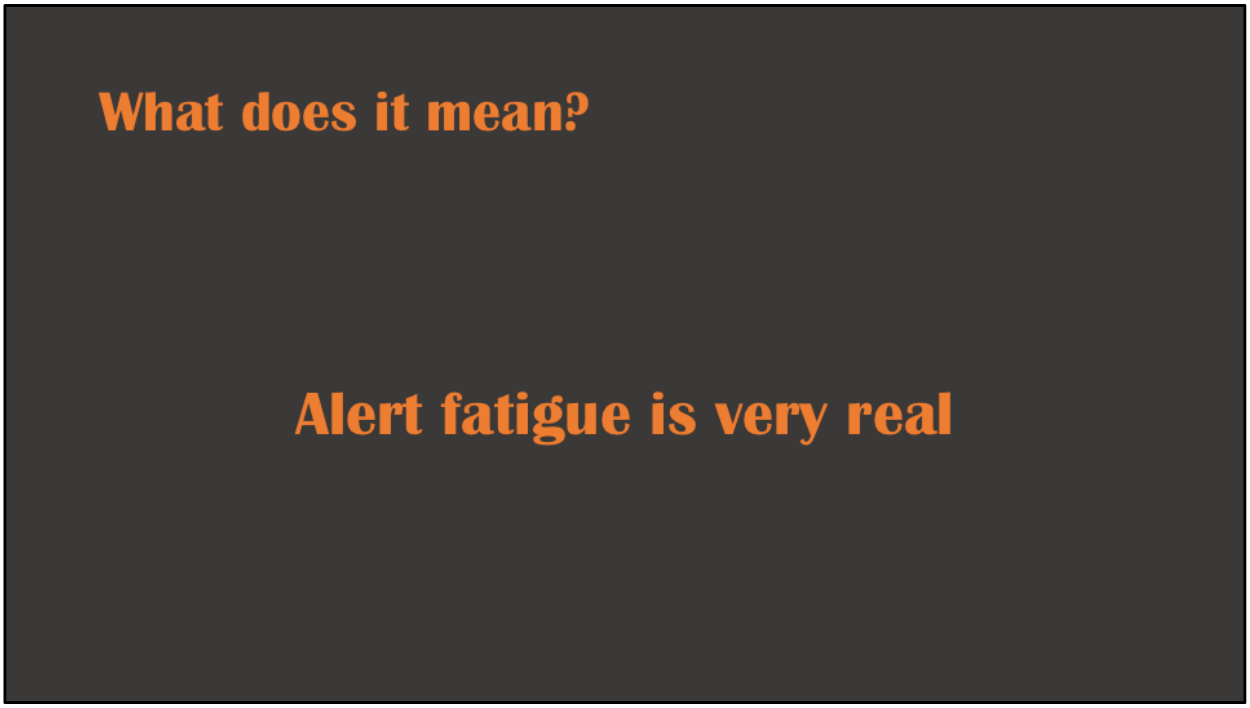
While the types of rules and where they can be deployed has changed, there has been less focus on improving their practical application.



But Wait!

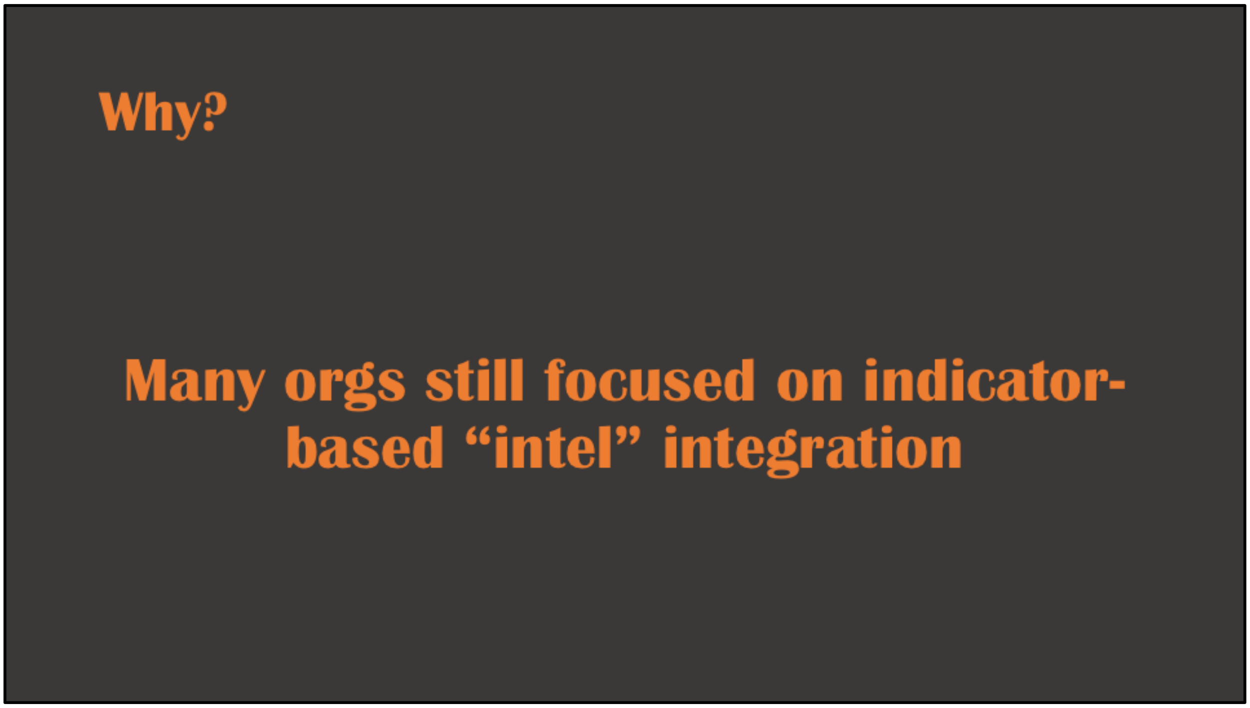
This doesn’t mean rules are bad!

There are certain areas where technologies like Machine Learning may make an impact, but it’s not well suited for all areas.



What does it mean?

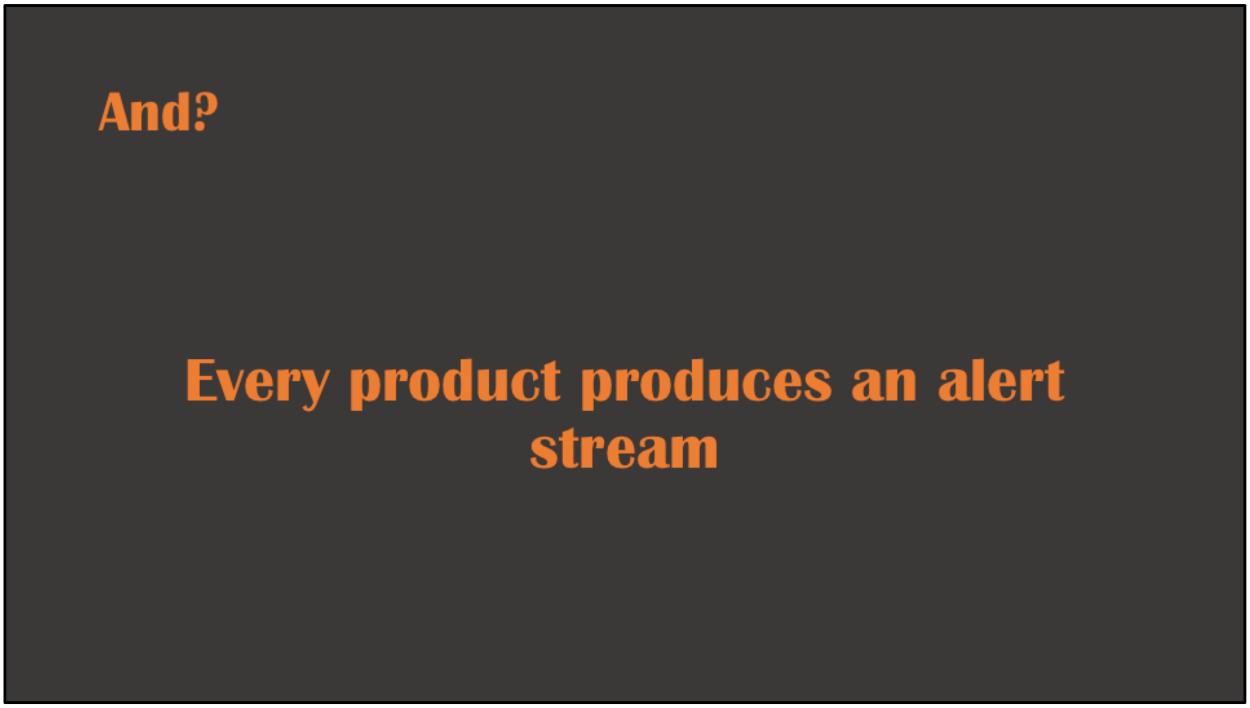
Alert fatigue is very real



Why?

Many orgs still focused on indicator-based “intel” integration

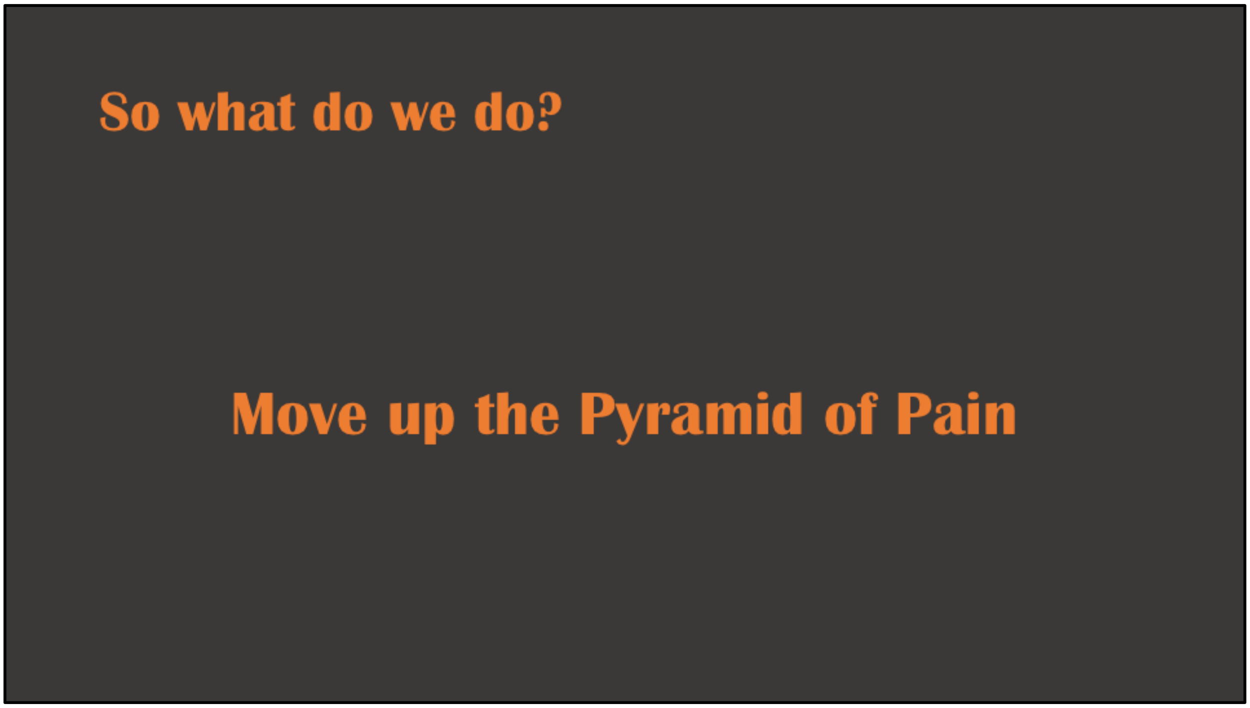
Low on the POP, these orgs may struggle to move up because of visibility, resources, experience or a combination of all three.



And?

Every product produces an alert stream

This is why SIEMs were born, right? The question though, is whether those SIEMs have effectively met the challenge. How many people in the audience would say their SOC uses a SIEM and it has solved the problem of alert fatigue?

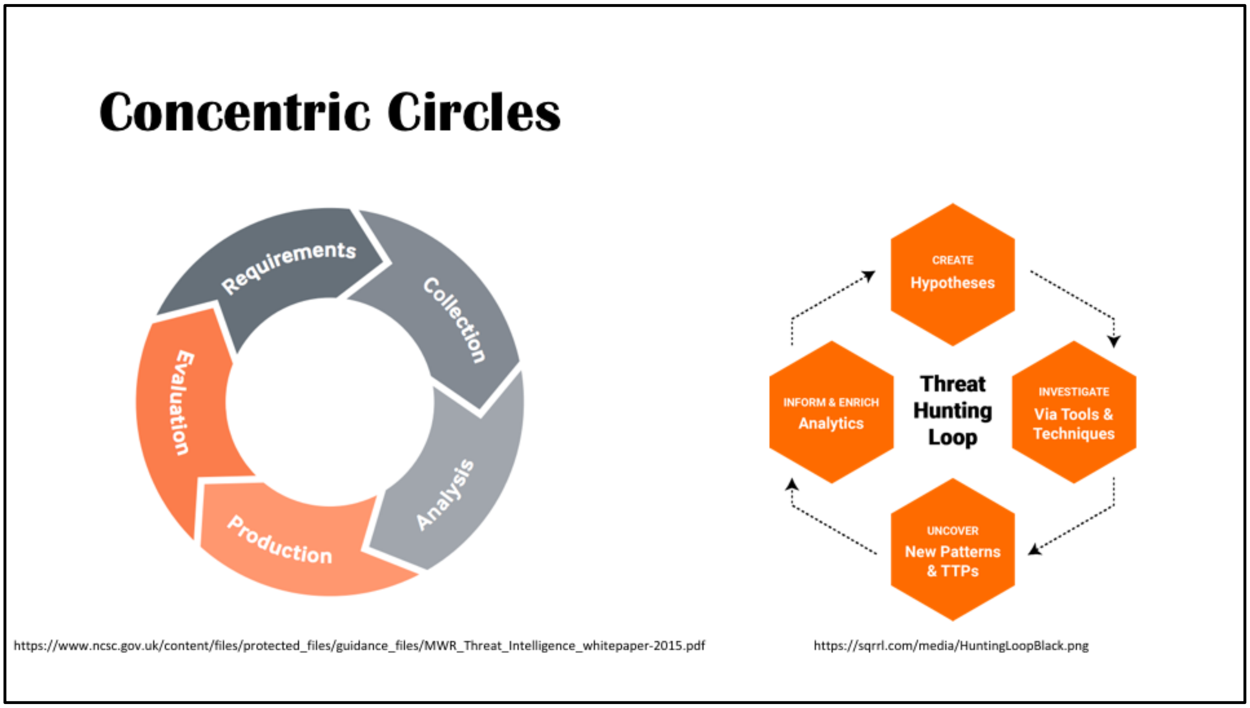


So what do we do?

Move up the Pyramid of Pain

The first goal is to move your organization up the PoP. I realize this is more difficult than this statement allows for. However, it’s crucially important if your aim is to start an effective hunt operation, as well as if you want to make more effective use of the threat data available to your org (especially if you’re paying for it!). For the purposes of the talk, we’re referring to low-level TI, with a general goal of moving from strictly technical intel to tactical intel.

For an organization just starting a hunt capability, this goal is a great first step.



Concentric Circles

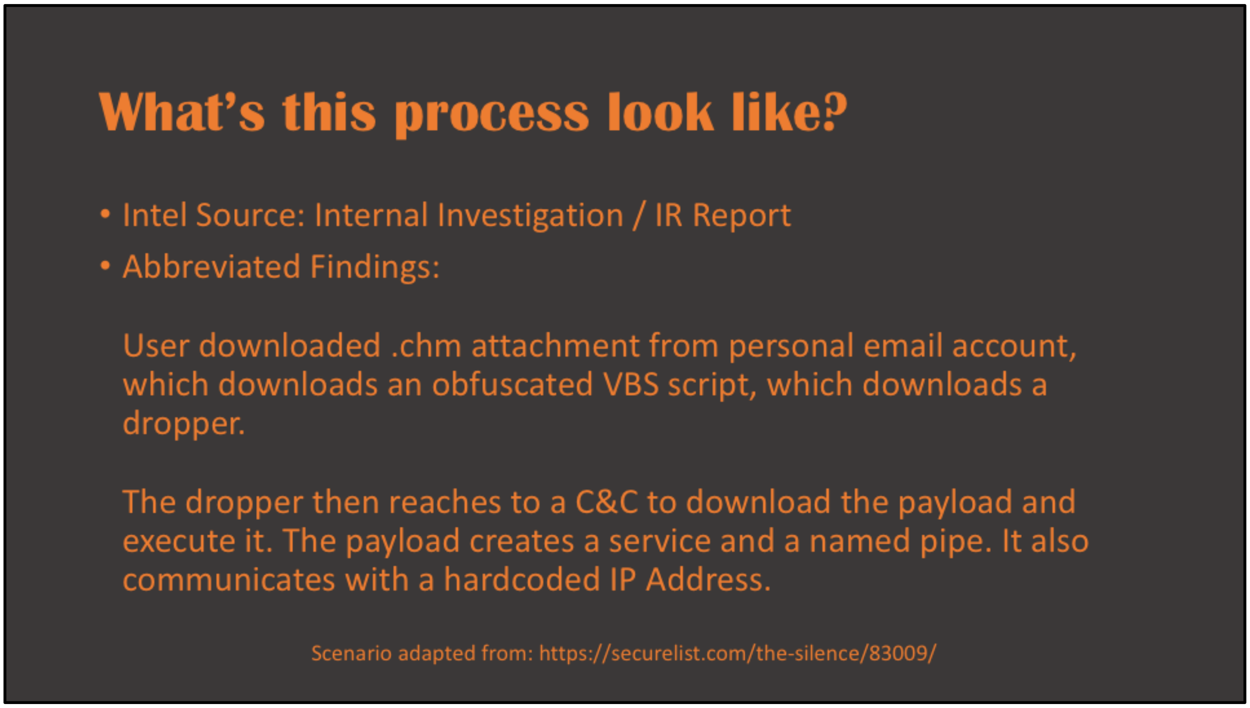
https://www.ncsc.gov.uk/content/files/protected_files/guidance_files/MWR_Threat_Intelligence_whitepaper-2015.pdf

https://sqrrl.com/media/HuntingLoopBlack.png

Some folks may recognize the loop on the left as the intelligence cycle. On the right, we have the Threat Hunting Loop.

In practice, these two processes should often be intertwined as they feed off of each other very well. The evaluation of threat intelligence should lead to organizationally relevant hypothesis and subsequent hunts. Depending on the source of the intelligence, there's a high likelihood that your investigation will uncover new collection requirements that are currently not met, which can then help prioritize projects and capabilities in your program.

At the end of your hunts, there should be two goals in mind. The first is to feed information back to your intelligence program to bolster analysis. The second is to implement preventative or detective controls based upon the outcome of your hunt. While doing this, we want to make sure that we keep our overall goal in mind, to help move up the pyramid of pain.



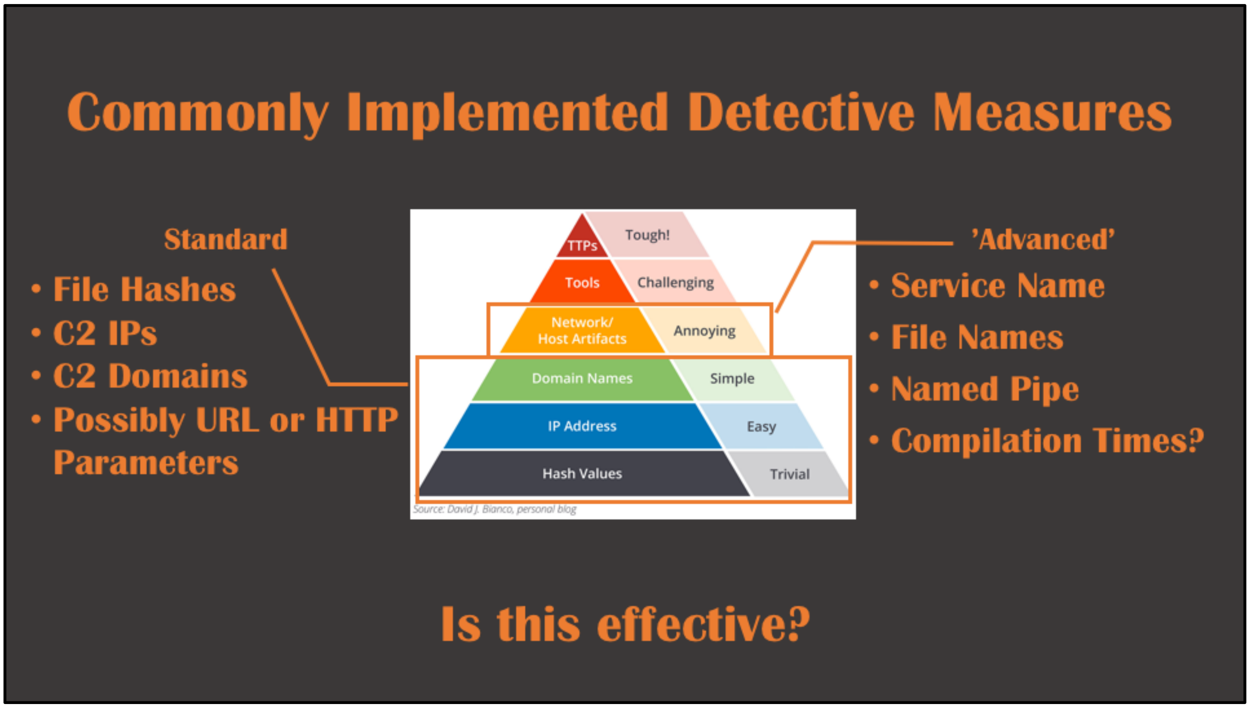
What's this process look like?

- Intel Source: Internal Investigation / IR Report
- Abbreviated Findings:

User downloaded .chm attachment from personal email account, which downloads an obfuscated VBS script, which downloads a dropper.

The dropper then reaches to a C&C to download the payload and execute it. The payload creates a service and a named pipe. It also communicates with a hardcoded IP Address.

Scenario adapted from: https://securelist.com/the-silence/83009/



Commonly Implemented Detective Measures

Standard

- File Hashes
- C2 IPs
- C2 Domains
- Possibly URL or HTTP Parameters

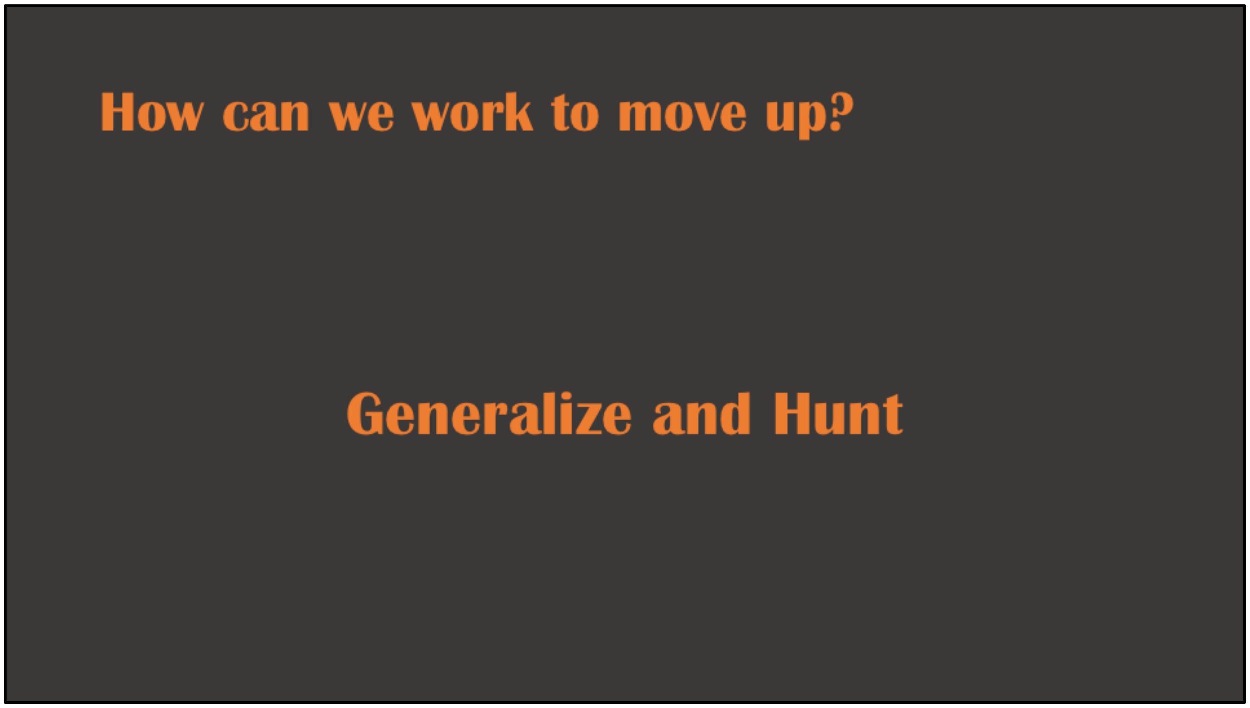
Source: David J. Bianco, personal blog

'Advanced'

- Service Name
- File Names
- Named Pipe
- Compilation Times?

Is this effective?

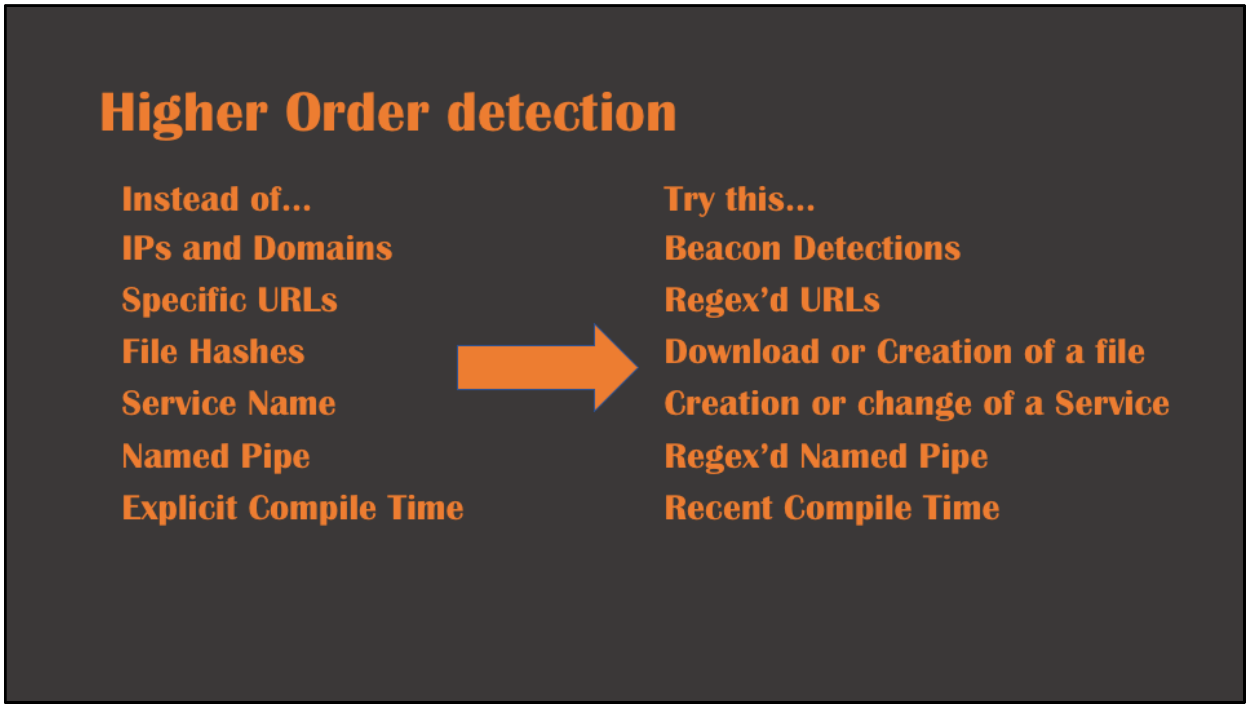
From our prior scenario, these are common detective measures that may be implemented. All are middle or below on the Pyramid of Pain and on top of that, are focused on specific values in the detections. Depending on the relationship between the actor and the source of the malware, presence of a builder, or other factors, changing the host artifacts themselves can still be a trivial task for the actor on the next run.



How can we work to move up?

Generalize and Hunt

Given that the measures listed on the previous slide are specific values, we can automate those lookups in our data sets. However, we can work to generalize those same factors and use the outputs to inform some hunts. The goal of the generalization is to break down the malicious activity in to small building blocks of distinct actions.

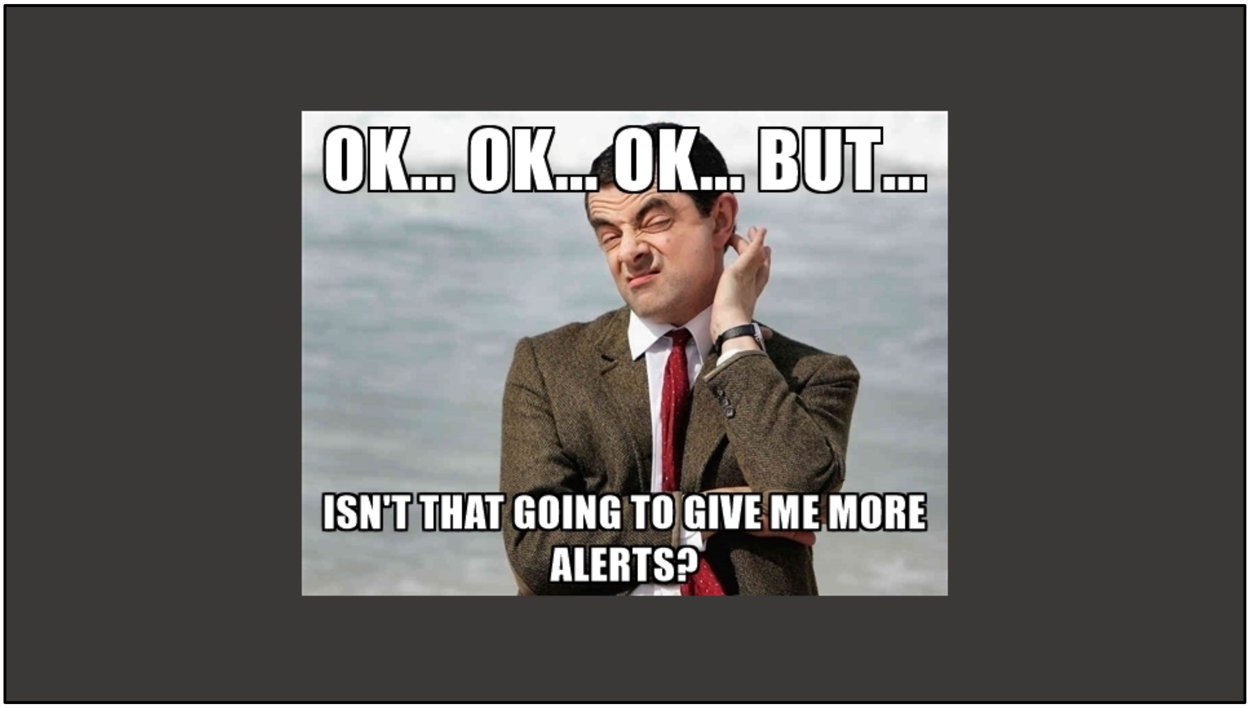


Higher Order detection

Instead of…	Try this…
IPs and Domains	Beacon Detections
Specific URLs	Regex'd URLs
File Hashes	Download or Creation of a file
Service Name	Creation or change of a Service
Named Pipe	Regex'd Named Pipe
Explicit Compile Time	Recent Compile Time

Each of the prior detection examples from our scenario has a generalized option for detection. Each of these is tied to actual functionality, characteristics, or behaviors of the malicious samples involved. Changing all of these to use completely different mechanisms crosses the border from simple or possibly annoying to challenging or tough.

One thing to keep in mind here is that each of these higher order detections may still have different lifetimes or require different levels of care. However, they should still last longer than any of the atomic indicators.



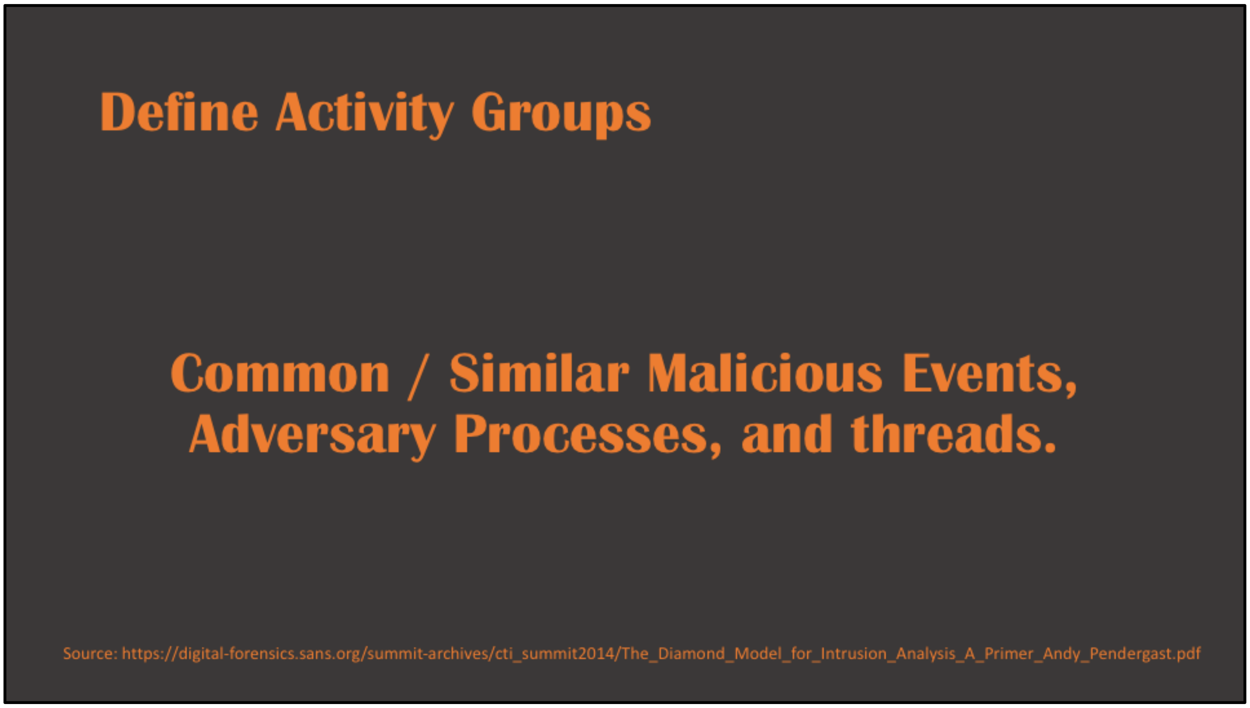
Yes, it will. However, I'm not proposing that each of these generalized detections be implemented and acted on separately.

What we want to do instead is take these building block behaviors and use them to seed our hunts. During the hunt, I want to pay particular attention to any additional characteristics along the way, particularly if I identify malicious behavior that isn't tied to the original incident that I seeded my hunt with. For instance – not every file download is malicious. What about file downloads from domains or IPs that I've only ever seen from one host, or less than 5 times? That might be another generalized rule for you to add to your collection.

This process will help us build out a good library of generalized rules.

So I have a list of generalized rules to implement, what next?

We determined that we don’t want to address each of these rules individually. What I want to do is relate the rules together. Your hunting activities in the prior steps should inform this activity.



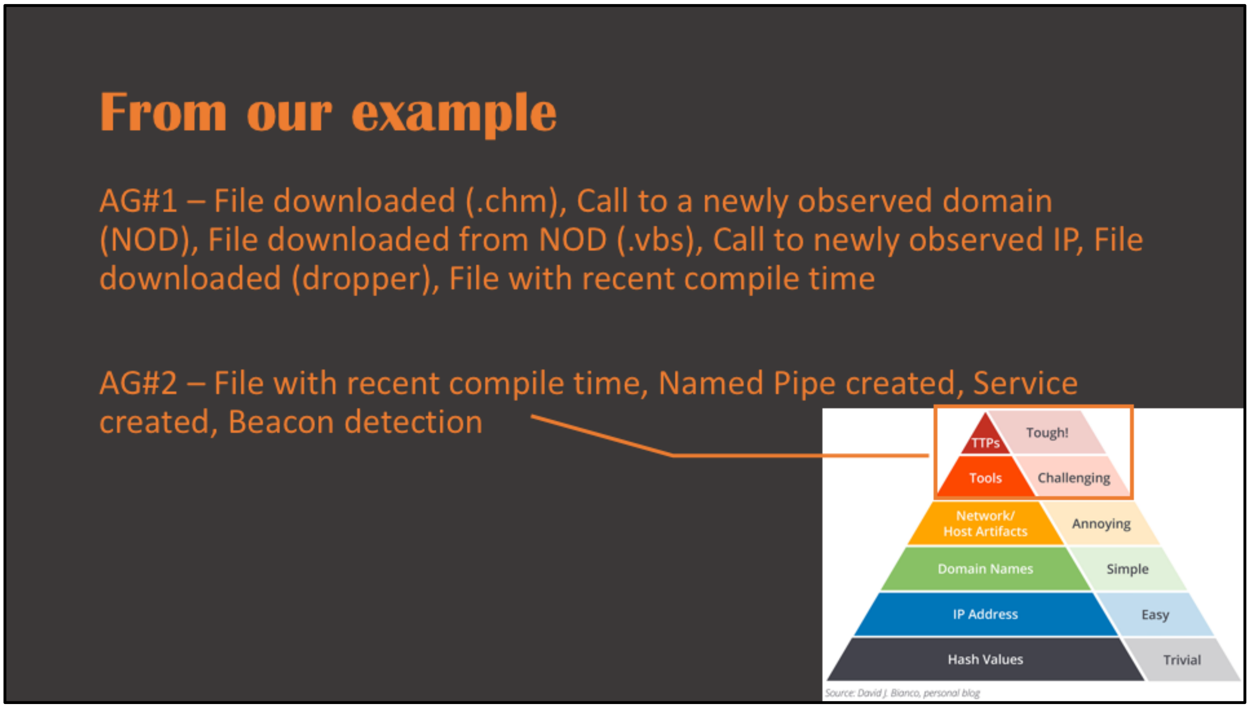
Define Activity Groups

Common / Similar Malicious Events, Adversary Processes, and threads.

Source: https://digital-forensics.sans.org/summit-archives/cti_summit2014/The_Diamond_Model_for_Intrusion_Analysis_A_Primer_Andy_Pendergast.pdf

I'm going to refer to these collections as Activity Groups. This definition is from a Diamond Model presentation that Andy Pendergast gave at this summit 4 years ago. Not only is the Diamond model awesome, he has some slides on other uses of activity groups and some on creating activity groups. We're going to do that as well, but it can be useful to see how they apply in different use cases and abstraction levels.

The important takeaway is that Activity Groups aren't static. They evolve as new information becomes available



From our example

AG#1 – File downloaded (.chm), Call to a newly observed domain (NOD), File downloaded from NOD (.vbs), Call to newly observed IP, File downloaded (dropper), File with recent compile time

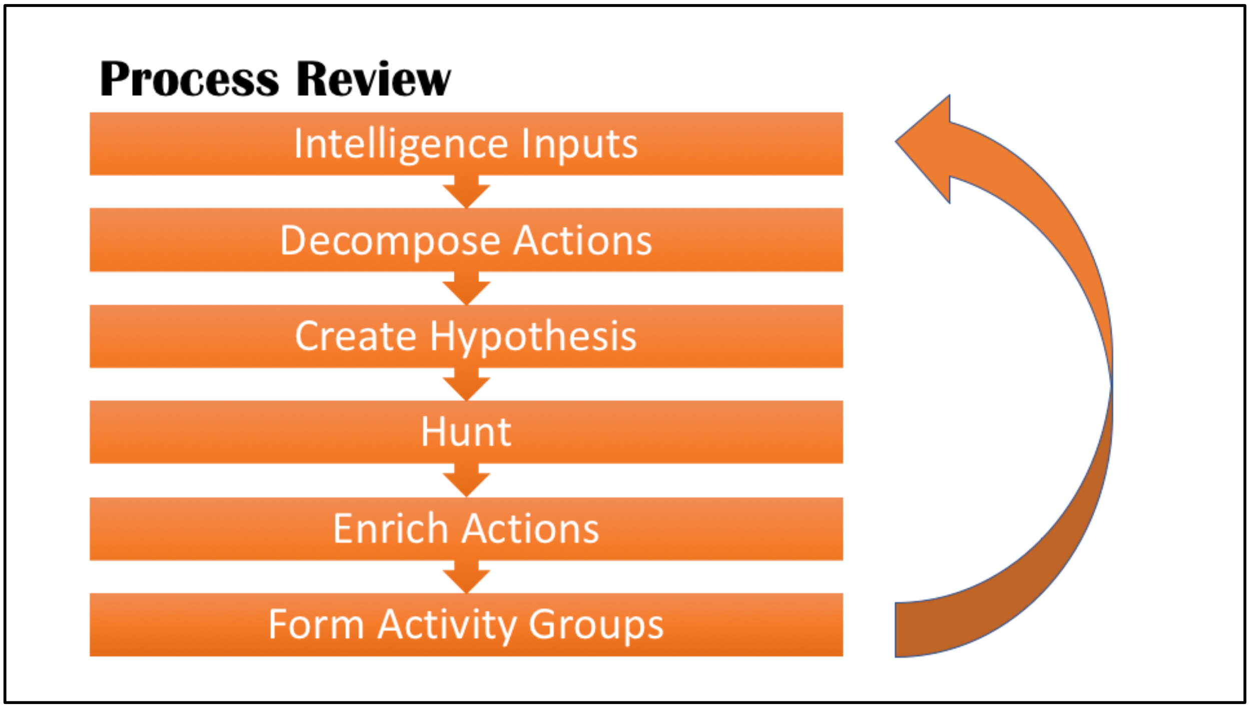
AG#2 – File with recent compile time, Named Pipe created, Service created, Beacon detection

Source: David J. Bianco, personal blog

These two activity groups are examples based on our original scenario, with a couple of additional features included that we determined in our hunt. Each of these individual pieces can be grouped, used in other groups, or in some cases, may be high enough to look at on their own on some schedule.

When you define your groups, one item you want to keep in mind is whether or not order matters in your specific group. I find it can be helpful to think of these groups in rough boolean logic. If Activity 1 and Activity 2, then activity 3.

The idea here is that we can create activity groups based on information gleaned from our Intel team and our hunts. This is a fantastic way to classify and document data collection gaps and gaps in detective capability as a whole. Which of the specific behaviors can you not monitor, and consequently, which activities are you prevented from detecting.



Process Review

Intelligence Inputs

Decompose Actions

Create Hypothesis

Hunt

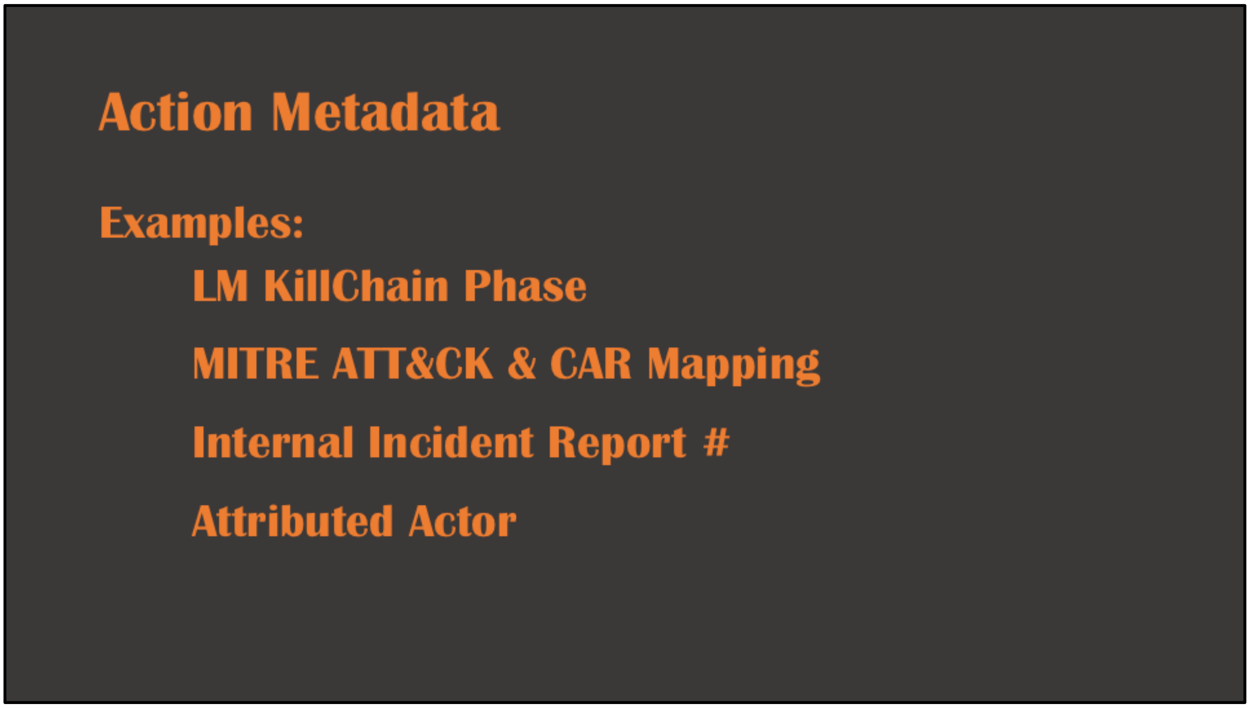
Enrich Actions

Form Activity Groups

All in one slide, this is the rough outline of the process we described. This is an initial definition and I'm sure it'll change over time. For instance – it's likely that step two would fit very well in your intel generation process. Of course, the operational aspect not listed here involves implementation and testing. While our goal is to create a higher fidelity grouping of activity, we'd still want to make sure the activity groups we defined are valuable.

What about unknown Activity Groups?

This problem is something we encounter as defenders in many areas. The discussion thus far has focused on defining activity groups that follow known collections of behaviors. That still leaves us with the task of uncovering our unknowns. While they may not have as high a True Positive rate as ours known activity groups, I think there's still an opportunity to cut the number of alerts generated.

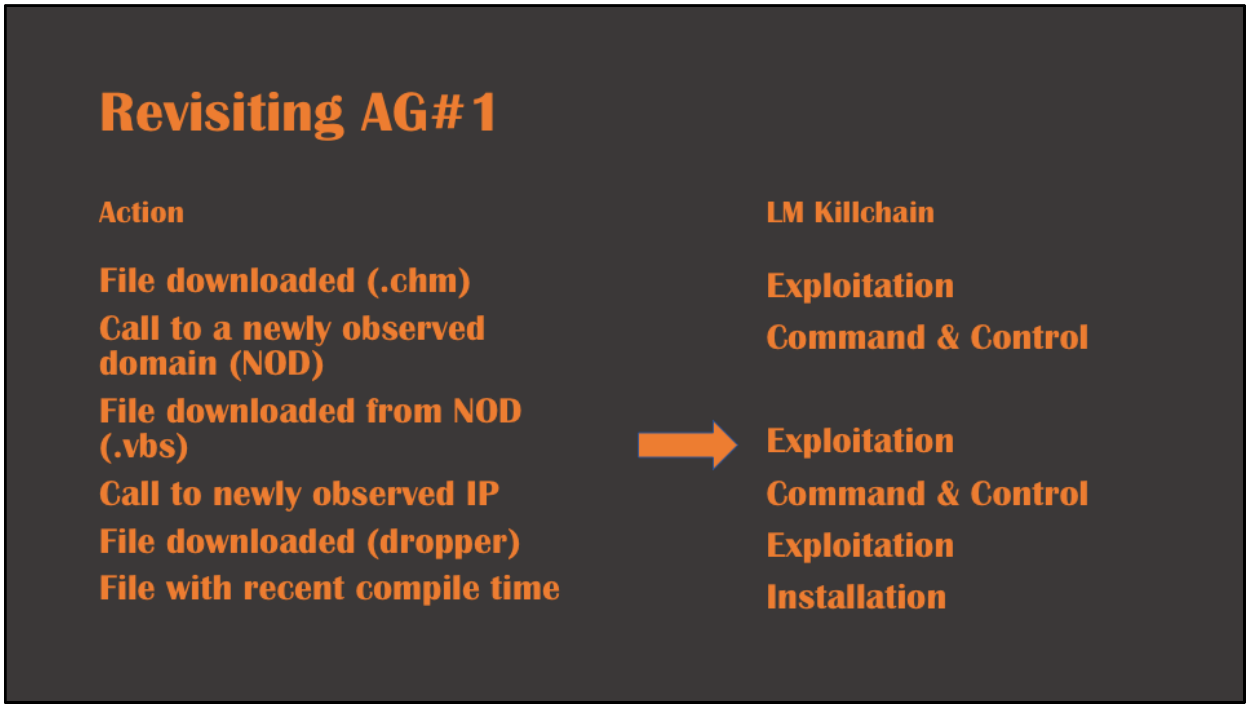


Action Metadata

Examples:

- LM KillChain Phase
- MITRE ATT&CK & CAR Mapping
- Internal Incident Report #
- Attributed Actor

The way I think this can be done is through the use of specific frameworks focusing on adversary activity. These aren't the only options by any means, but they do represent some likely candidates. Using these descriptive pieces of metadata in our Action definitions can allow us to form Activity Groups based on classification, rather than action.



Revisiting AG#1

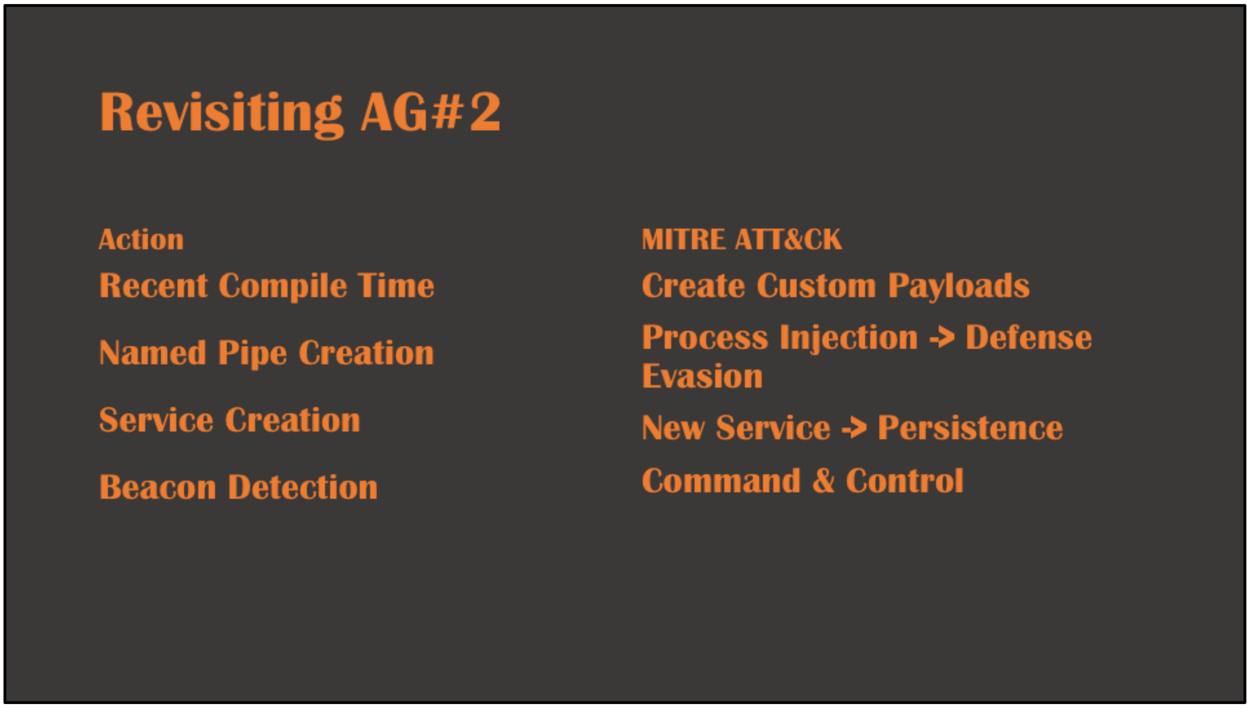
Action	LM Killchain
File downloaded (.chm)	Exploitation
Call to a newly observed domain (NOD)	Command & Control
File downloaded from NOD (.vbs)	Exploitation
Call to newly observed IP	Command & Control
File downloaded (dropper)	Exploitation
File with recent compile time	Installation

When we take a look at the first example activity group that we defined earlier, we can see that we end up with a varied collection of steps based on the actions in the group.

This gives us a new potential way to define an activity group that can account for changes in specific actions within the group. So for instance, here, we may decide our activity group should fire when I see any collection of actions that meets the criteria of:

2 Exploitation Actions
2 C2 Actions
1 Installation Action

Now, that cuts out 1 of the Exploitation actions in our example, but would still alert on the same activity. It affords us a bit more flexibility – now, if the actor removes one of the downloader stages, we can still identify the activity group.

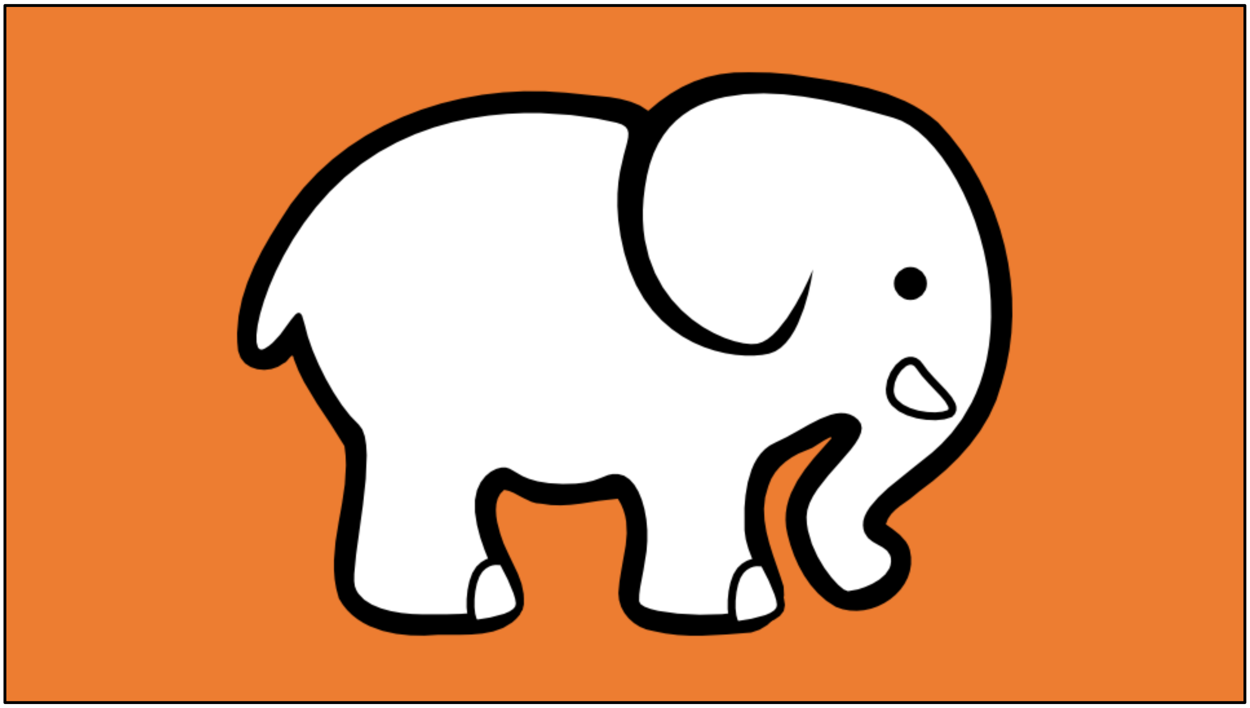


Revisiting AG#2

Action	MITRE ATT&CK
Recent Compile Time	Create Custom Payloads
Named Pipe Creation	Process Injection -> Defense Evasion
Service Creation	New Service -> Persistence
Beacon Detection	Command & Control

This is just another example using a different framework. The end goal is to turn an activity group behavior independent.

In the case of the MITRE ATT&CK framework, we're limited a bit on which stages we can include because ATT&CK focuses on post-exploit actions. For the first, we cheat a bit and pull from the PRE-ATT&CK framework.

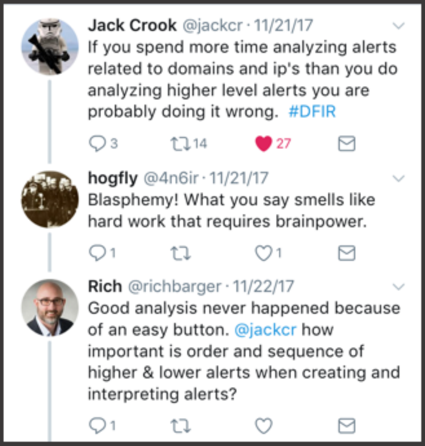


Ok – so here it is. Some of you may be thinking, Keith – man, you're a bit behind the times, these are just correlation rules.

At the basic level, yes, they are correlation rules. However, in a lot of cases, I think the process of taking in information, processing it, and implementing those types of detection capabilities is lacking. On top of that – how many existing solutions or frameworks for creation of correlation rules are easy to use and allow you the flexibility to implement rules at both a technical and more abstracted layer?

The fact is – this is an introductory and base level outline of an area I believe is ripe for improvement. My goal is to help spark conversation and ideas around the theory. As I stated at the beginning, it's the area of practice that I haven't seen change much in the last decade.

This is just the beginning...



Jack Crook @jackcr · 11/21/17
If you spend more time analyzing alerts related to domains and ip's than you do analyzing higher level alerts you are probably doing it wrong. #DFIR

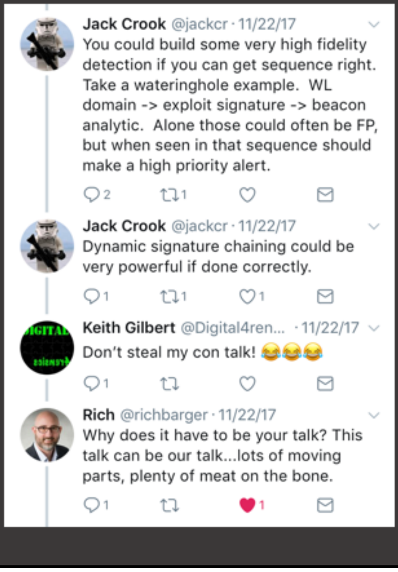
3 14 ♥ 27

hogfly @4n6ir · 11/21/17
Blasphemy! What you say smells like hard work that requires brainpower.

1 1

Rich @richbarger · 11/22/17
Good analysis never happened because of an easy button. @jackcr how important is order and sequence of higher & lower alerts when creating and interpreting alerts?

1



Jack Crook @jackcr · 11/22/17
You could build some very high fidelity detection if you can get sequence right. Take a wateringhole example. WL domain -> exploit signature -> beacon analytic. Alone those could often be FP, but when seen in that sequence should make a high priority alert.

2 1

Jack Crook @jackcr · 11/22/17
Dynamic signature chaining could be very powerful if done correctly.

1 1 1

Keith Gilbert @Digital4ren... · 11/22/17
Don't steal my con talk! 😂😂😂

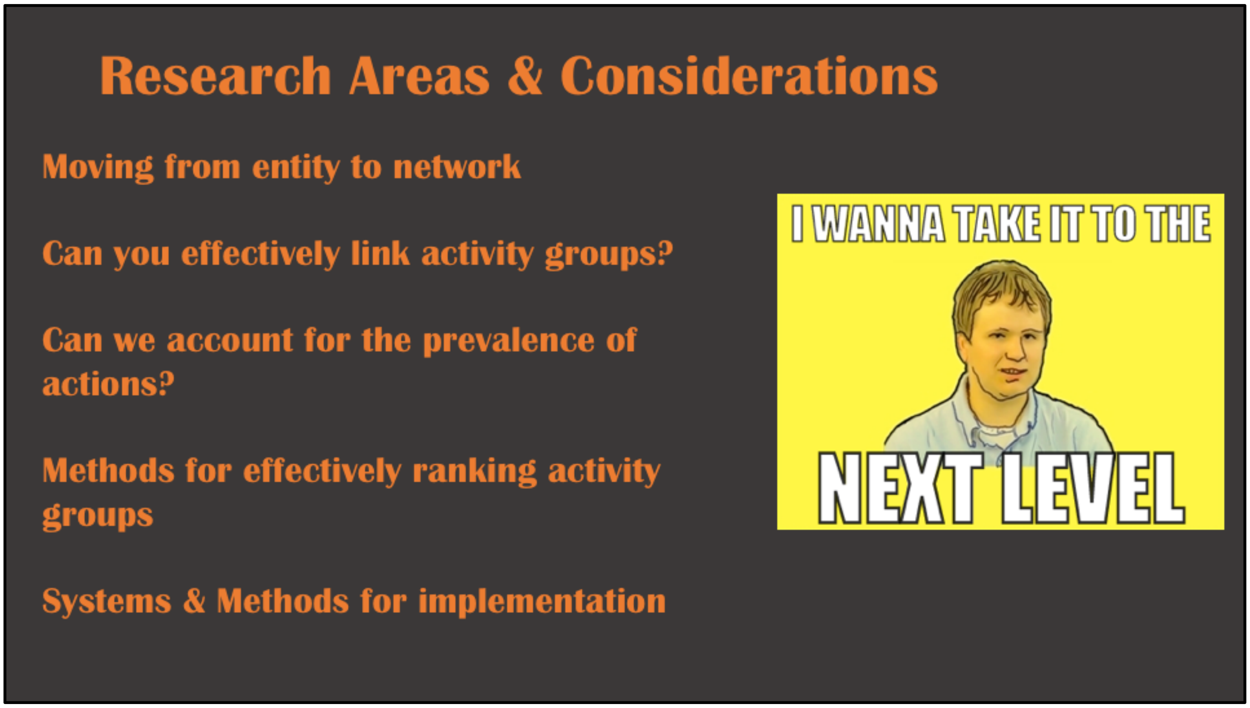
1

Rich @richbarger · 11/22/17
Why does it have to be your talk? This talk can be our talk...lots of moving parts, plenty of meat on the bone.

1 ♥ 1

Just an example of that which presented itself shortly after acceptance emails went out for this conference. I'm sure most of you are familiar with Jack's amazing content both on twitter and his blog. Here, he refers to the concept as Dynamic Signature Chaining – I think this lends itself to the idea of a very flexible framework that has the ability to descriptively link rules or signatures.

After I joked about encroaching on this talk – Rich aptly, and 100% correctly, pointed out that this isn't a solved or well defined theory and that there's plenty of room for input from others.



Research Areas & Considerations

Moving from entity to network

Can you effectively link activity groups?

Can we account for the prevalence of actions?

Methods for effectively ranking activity groups

Systems & Methods for implementation

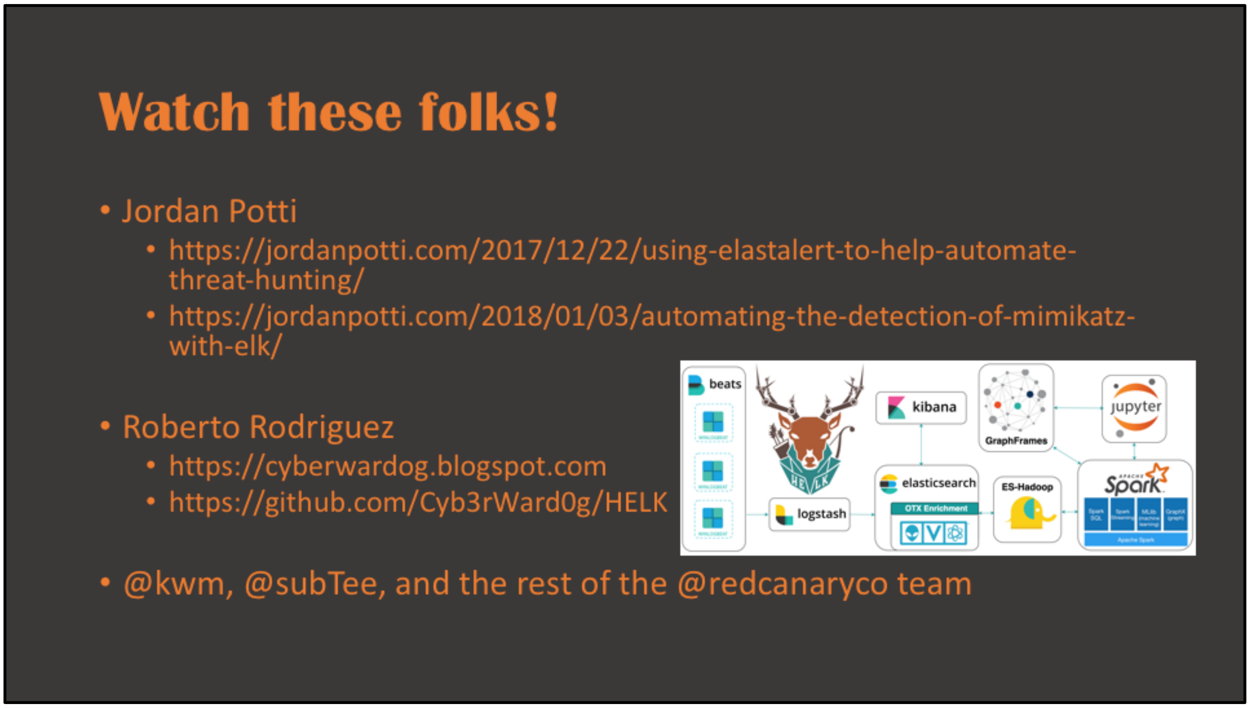
I WANNA TAKE IT TO THE
NEXT LEVEL

While there are certainly a number of areas to be fleshed out within an implementation of these ideas - these are some of the primary areas that I think could provide some very tangible value moving forward.

The discussions here focused on example activity groups that involve only one host. However, there are plenty of examples of activity groups that would span multiple hosts, which increases difficult of grouping. For instance – if one host has an exploitation event fire, and then an actor moves laterally, we've added actions on objectives. However, we now want to bring in any identified actions on the target host as well.

The next few focus on items that could potentially contribute to higher fidelity activity groups that present more of a holistic picture. If we identify an activity group on two hosts that are then connected via another identified action, can or should we link those? Can we create graduated activity groups that rank themselves based on the severity of the actions in them, the % of actions matched in the activity group, or other factors?

Lastly – the basic requirement for putting this in to practice is either a dedicated system or index for alerts that we can then use to ingest the identified Actions. I’ve been referring to this as sort of a META-SIEM – instead of ingesting raw data, you’re ingesting alerts and then combining them.



Watch these folks!

- Jordan Potti
  - https://jordanpotti.com/2017/12/22/using-elastalert-to-help-automate-threat-hunting/
  - https://jordanpotti.com/2018/01/03/automating-the-detection-of-mimikatz-with-elk/
- Roberto Rodriguez
  - https://cyberwardog.blogspot.com
  - https://github.com/Cyb3rWard0g/HELK
- @kwm, @subTee, and the rest of the @redcanaryco team

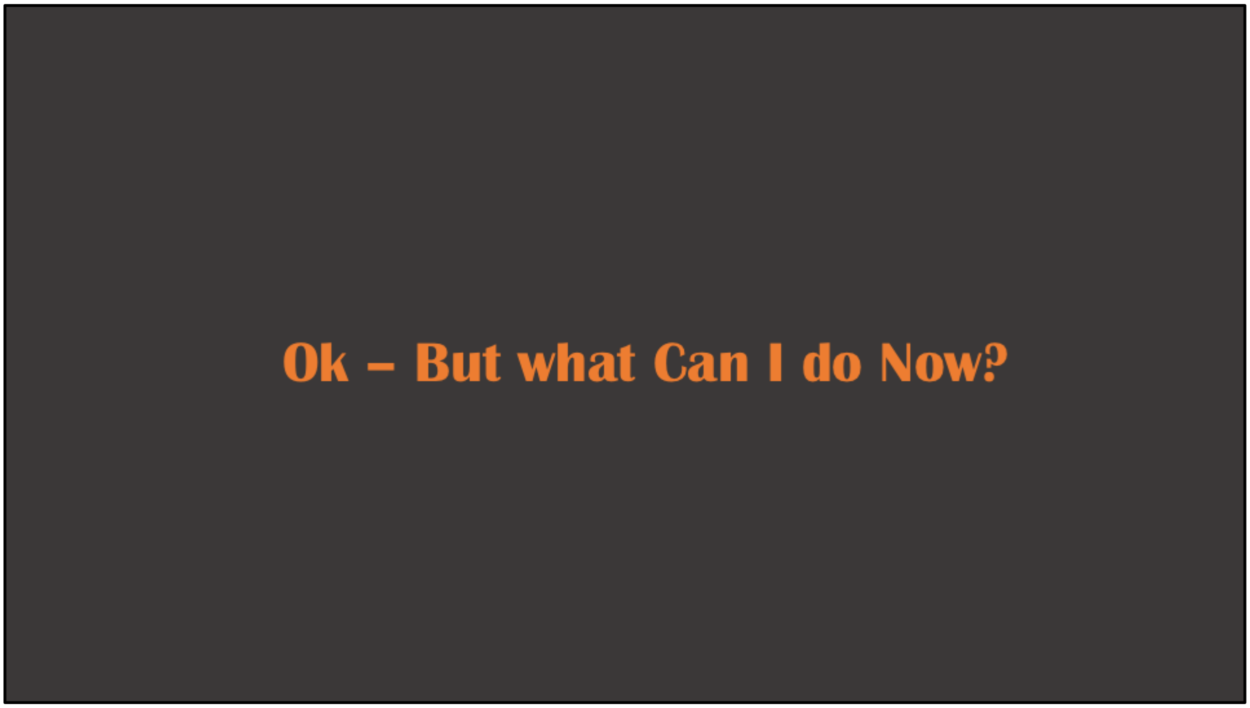
beats
HELK
kibana
GraphFrames
jupyter
logstash
elasticsearch
OTX Enrichment
ES-Hadoop
Apache Spark

There are several projects or blogs that came out during the time period I was putting this topic together and I think these are great places to monitor for advancements in this space.

I came across a couple blog by Jordan Potti on using Elastalert to identify actions of interest, and then one on grouping which only creates an Alert if 5 other occurrences were all detected within a second of each other. Both are great examples on how the process here may actually be put in to practice.

Roberto was already slated to get a shout out due to his awesome references and work on hunting, mapping detection gaps, and working with ATT&CK. As luck would have it, he released HELK while writing this talk and I noted that Elastalert is on the roadmap. Given the Jordan's work – seems like HELK might be a prime place to put these theories in to action.

Keith, Casey, and the rest of the Red Canary team consistently put out great content in blogs and the respective twitter feeds, much of which is directly or indirectly related to items discussed today, attacker methods, detection methods, etc.



Ok – But what Can I do Now?

As we noted – the processes here are not well defined or widely adopted and there are definite research areas of interest. But – there are some things you can start working on now to get you set down a path to start testing or experimenting.

Break Down Your Intel Sources

Instead of pulling atomic indicators straight out and calling it a morning – work to automate that and focus on assessing the behaviors present in the source. Internal, external, commercial, public – any long form report should have some higher level content that you can extract.

Hunt for Activity Groups

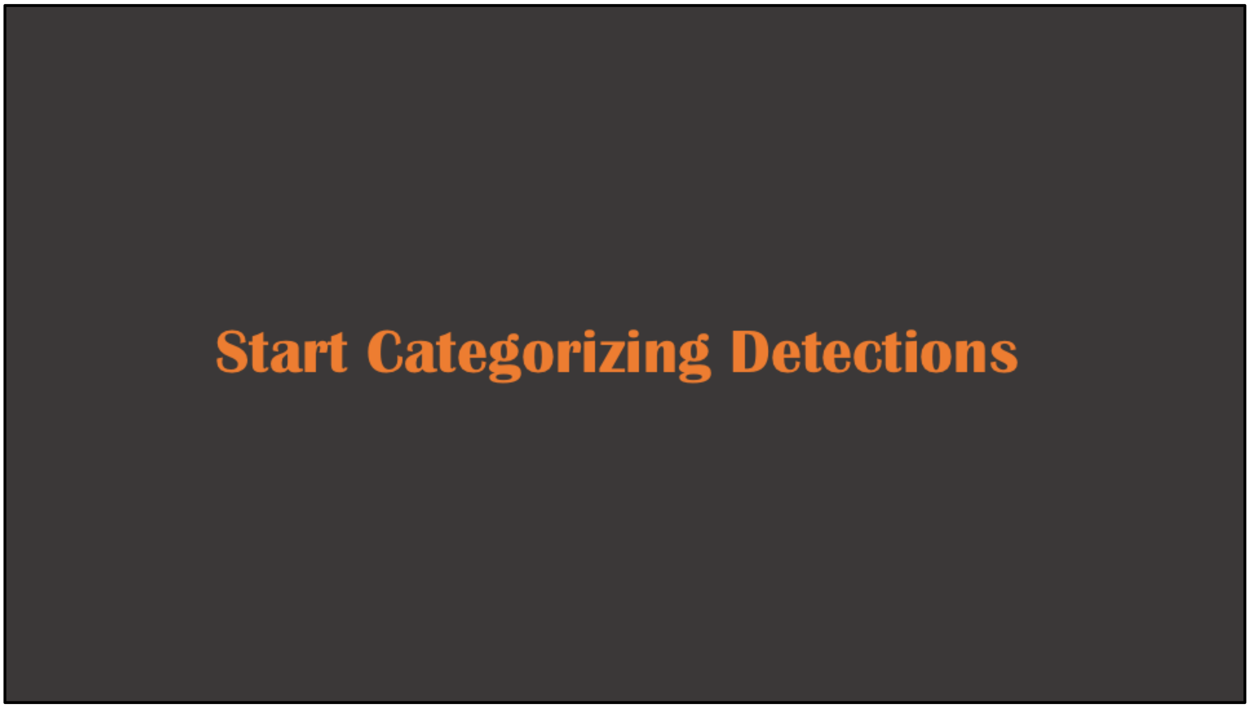
After you pull out all of your distinct behaviors – assess which ones are closely related and could form activity groups. Keep in mind that there may be ways to enrich those groups.

Then you want to go hunt. This is a crucial step to determine if the behaviors and groups that you pulled out are effective on your network.

Identify Data and Detection Gaps

When you do that, it’s near a guarantee you’ll come across TTPs and Activity Groups with some gaps. Inventory them for tracking and develop a plan to reduce the gaps.

The great thing about this step in this process – you now have documented TTPs that are relevant to your organization (assuming your intel sources are relevant) from a defined threat that you can’t see. In aggregate – that collection of data is great to include in a proposal for increased tooling or process changes.



Start Categorizing Detections

After you’ve verified applicability of an activity group for your organization, think about the higher level meta-classes that may be useful for organization, tracking/inventory, unknown activity groups, and versioning.

If you have a way to start implementing this process in an automated fashion, that’s great. If not, you’re building up a library of verified, useful, higher level detections to use later. You can then begin planning to iteratively automate portions of your collection as you work toward full capability. For now, they can be used as hunts.

Questions?