Chapter 7: Learning From (security) Failures

In times like these when unemployment rates are up to 13%, income has fallen by 5% and suicide rates are climbing I get so angry that the government is wasting money on things like the collection of statistics!

Hans Rosling, quoting a caller on a radio talk show, *The Joy of Stats*

When organizations experience a security event, their natural reaction is to focus on getting back to normal as fast as possible. They see the event as a sign of failure or an embarrassment and everything they do is to minimize the impact and put the event behind them. In that environment, an important task is often overlooked and the silver-lining is often missed. During such an event, a rich set of a data is generated and just waiting to be collected and analyzed. Think of it, if we could somehow gather that data, make sense of it, perhaps even compare and contrast it with other security events, we could learn how we may prevent the next attacker. Maybe even more than that, perhaps we could identify trends and patterns, enough so that we could address multiple common attacks with a single project. Achieving that kind of benefit is the goal of this chapter, we want to figure out what data to collect and how we can collect and manage it. We will also discuss how we can analyze this data and even share it to get the most benefit from the data.

In order to tackle the challenge of learning from breach data, we’re going to leverage the Vocabulary for Event Recording and Incident Sharing (VERIS) framework. One of the authors of this book (Jay) and the RISK team at Verizon have been developing and evolving VERIS in order to produce there Data Breach Investigation Report (DBIR)[[1]](#footnote-1). In an effort to promote adoption and use, Verizon has opened up VERIS and all the details of its use and implementation are hosted at <http://veriscommunity.net>. We won’t just be focusing on implementing VERIS though. We will use it as a foundation and case study as we discuss developing a data collection and analysis effort for breach data.

Besides being an open framework, using VERIS as a case study has another benefit. There is a project called the VERIS Community Database (VCDB), which offers a free and downloadable data set of publicly disclosed security events using the VERIS format. This means we’ll have thousands of VERIS records we can download and analyze throughout this chapter. At the time of this writing, VCDB is being housed at Github (<https://github.com/vz-risk/VCDB>).

Setting up the Research

First and foremost, we want to approach our breach analysis as a research project. If we think of this as a “metrics program” or a “security project”, we may fool ourselves into thinking this is somehow unique to information security and it isn’t. This is all about data collection and analysis, something that has been done countless times before across many different disciplines and generations. Approaching this as if it is a unique project and trying to reinvent the (data analysis) wheel, would not only be wasteful of our time and resources, but we’d be laughed at and ridiculed by all the grown-up data scientists. Let’s avoid all that and call this what it is, a research project.

Most of our work in this book has been of an exploratory nature. We worked with the data to see what it contained and then formed the questions we want to answer with the data and went back into the data. This effort is different because we are starting with no data at all. If we jumped right in and started to collect the breach data that seemed good, we’d waste countless resources, capture data that we’d discover later to be meaningless and we’d end up wishing we had data we didn’t collect. Therefore, we’re going to set a frame for this effort and define a handful of questions we’d like to explore. From that, we will be able to determine what data points we want to collect.

VERIS was developed to support the strategic decision process. Where can we focus our limited resources to get the biggest benefit for our security spending? Given a list of audit findings or remediation projects, how can we prioritize those, so we fix the most critical first? Perhaps even more importantly, we also want to find the opposite of those questions. In other words, can identify areas and tasks where do we *not* want to spend our time and money? Supporting these questions therefore, is our goal in this chapter and can be summarized as:

Goal: Improve security management by supporting informed decision making

**Our goal in collecting and analyzing breach data is to support the decision making process within security leadership.**

Notice how used the word “support” in there. This research will exist to *support* a decision process. It is not intended to be or replace the decision process. We need to have the wherewithal to recognize that security prioritization is a complex issue and we are just beginning to scrape away at it. At this point in that scraping, where we have very little data, we should not make the assumption that we’ll get it perfectly right out of the gate. Even though there may be influencing variables that we don’t collect, we have to support the decision process, and we do that by reducing the amount of uncertainty in that decision.

type="note"

Breach Data is for Reducing Uncertainty

While it would be great to collect breach data to create a perfect and prescriptive list of priorities, it just won’t happen. The data will simply help us know a little more than we currently do, but it won’t be able to definitively show us the path forward. This raises the question for some whether or not it is worth it. Is it worth spending the time and resources to create information that doesn’t tell us what to do?

The answer is an emphatic yes.

Uncertainty exists in the gap between what we know and what we would need to know to make the best decision. While it’s tempting to toss out imperfect information because it contains uncertainty, the value of the information should be assessed by comparison. Not between the perfect information we’d want and the information we’ll get, but instead between the information we *currently have* and the information *we will have*. This is where we see the value of this type of data analysis. Data will help us reduce our uncertainty by reducing the gap between what we know and what we need to know. Plus we will be making progress and setting a foundation for further reduction. This is how science has evolved our knowledge: a series of small steps each reducing our uncertainty a little more. Therefore, our goal should be to use data to reduce our uncertainty, not to give up when the data is less than perfect.

Considerations in a Data Collection Framework

As VERIS has evolved, we’ve developed some guidelines that help us as we consider new data points or evaluate the existing questions we’re asking. These guidelines are not just for VERIS or even just for breach data. If you are looking to collect data manually, these guidelines should help shape that effort and make the process easier for all involved.

First and foremost we want to **aim for objective answers**. If we ask questions that require opinion we are going to get a lot of variety in the answers. In some cases this may be okay because inconsistent answers may be better than no answers at all. However most of the time we want to ask things that are observable of deducible. For example, it’s far better to ask if malware was involved in the attack and the functions it performed instead of asking how advanced the malware was. The investigator during the breach can answer yes or no (see below on “unknown”) to whether or not malware was used. If the investigator has the resources to do malware analysis (or the malware is identifiable), there isn’t a lot of guesswork around what’s it’s capable of.

Next we want to **constrain the possible answers** to a short list or small set of options. If we ask for a sentence or description, we should do so knowing that is won’t be useful directly in the data analysis without a lot more effort. Most of the time, free text fields are helpful to record unique aspects or to set context if we ever want to understand why these data points look like they do. With this in mind, we will make judicious use of “notes” fields and a field for the overall “summary” of the event, but all of the real data about the event will be contained in lists or numbers. Having the data constrained to a limited set of values this will make the analysis easier in the long run.

For most every constrained list of answers we create, we’ll usually want to **allow “unknown” and “other” answers**. Even though we may think something is so easy we should always know it, the world will always create a circumstance to prove us wrong. We will want to separate the times we don’t know from the times we know and the list isn’t applicable. This is a subtle distinction, but one that can really mess us up during the analysis. There are a few rare questions we can leave “unknown” off from the answer, but those are rare and you’ll know it when you see it.

The second field we’ll want to add is “other” or depending on the question a “not applicable” answer. We want to avoid from making an exhaustive list. Not only would exhaustive lists become unmanageable (which will slow down data entry), but also we only need to capture *most* of the answers. The common answers create trends and statistics, while the uncommon answers make for interesting stories. Therefore, we want to capture the common things for data analysis and relegate the uncommon to the “other” category and the notes field. We should keep an eye on anything marked “other”, but if we create our list well enough, they should show up few and far between.

It’s okay to be lazy when creating our lists of answers and **seek out standards** to leverage. For example, we will not create our own list of industries to gather. We will leverage the good work of the U.S. Census Bureau and their North American Industry Classification System (NAICS). We won’t attempt to define all the countries, but leverage ISO 3166-1 and store 2-digit codes for the countries. Not only does this offload some of the work, it may also typically be better than anything could dream up.

This last two points may seem subtle, but we want to **avoid conflation and drop the minutiae** where possible. These two concepts are opposites and we have to find the middle ground between them. Conflation occurs when a question (and its answers) are combining more than one concept. For example, the breach types used by Dataloss DB (<http://datalossdb.org/analysis>) conflate the actor, actions and assets into the type. They list a type of “Hack” for a “computer based intrusion” (no asset or actor defined), or “Snooping” which is an “employee … accessing confidential records” (conflating the actor and action) or we can specify “Stolen Media” or “Stolen Drive” or “Stolen Tape”, which are all unique options conflating and repeating the action (physical loss) with the asset. The assignment of a single conflating “breach type” should not be thought of as wrong or bad, it just represents a different goal within the research. Just be aware that conflation of terms like this will create a challenge during data analysis. With conflated terms we will find it challenging to do anything more than simply count the frequency of each breach type.

Where conflation combines more than one concept into a single variable, we have to be careful of the opposite where we split a single concept with minute details. We want to get just enough detail and separation in the list to support our goal. An example of that is when we try to collect how incidents are discovered we want to classify the discovery moethd. While it may be interesting to know if it was an external security researcher, and perhaps amusing to know what color hat they wore (white, black or even grey), those details wouldn’t change our goals. We have split one concept (an external security researcher) across multiple selections. In this case, maybe we just want to drop the distinction of an external security researcher altogether and create one broad field of “an external unrelated party”. But don’t be afraid to go into detail where necessary. As an example, the list of possible assets within VERIS is split into several categories and dozens of detailed assets under each category. There are times we’ll want to split and times we can combine, the trick is getting that balance right.

Luckily, these are all things we’ve been considering as VERIS has been evolving. One of the biggest challenges is saying no to new questions. We’ve found there is always more we’d like to know, but we know that each data point we try to collect has a cost (see below).

type="note"

Consider the Cost per Datum

During a manual data collection effort, it is very tempting to dream up all sorts of questions we’d like answered. Creating such a list isn’t bad and it may even be good to lay out all the questions you’d want to answer. But choose the questions you are going to ask very carefully because every question adds exponential cost across the lifetime of the data. Even before the question gets answered we have to build a method to collect it, so every question must be built into the data collection. We may have data validation going into the method, and then data validation coming out. As a single incident is being entered, each question will require some thought and perhaps even some research before it can be answered, again adding time and effort. That data point may require processing and clean up, and will need to stored and managed. Anytime we want to parse the data (and you’ll want to parse this in many different ways), we may have to consider this field, or worse, consider the interactions of all the fields. Beyond that, there are dozens other subtle interactions that will increase the cost of each data point beyond what we can imagine as we form the research questions.

It’s helpful to pretend you are about to take a long journey to a wise sage who lives on top of a mountain. You will have a limited amount of time to ask questions before the sage says something mysterious and vanishes. What questions will have the greatest impact? You’ll want to identify a handful of questions you really want answered, maybe a handful of questions you’d like to have answered and then you’ll have a mountain of questions you wish you had time to ask, but you’ll just have to make do. The same is true with manual data collection. If the post-incident questionnaire asks too many questions or is too painful, people will lose interest quickly and the answers will end up being of poor quality. You must choose your questions wisely.

An Introduction to VERIS

When there is a security event investigated, a narrative naturally emerges from the process. The investigator will typically try to answer, “Who did what to what (or whom) with what result?” which is a good core set of data points to collect. Therefore, as a starting point, we want to focus on those four points, “Who (threat actor) did what (action) to what or whom (asset) with what result (attribute)?” But that’s not all we may be interested in; we may also want to know how we discovered and responded to the incident and if possible the impact we experienced as a result. Finally, we’ll have some housekeeping items (an identifier, summary, status, etc) and if we aggregate breaches or may share the information, we’ll want to record some victim demographics. Overall, we can break down the following sections of data we want to gather at a minimum.

|  |  |
| --- | --- |
| VERIS Section | Purpose |
| Incident Tracking | Meta-data about the incident for management and tracking purposes. |
| Threat Actor | One or more people that that cause or contribute to an incident |
| Threat Actions | What the threat actor(s) did or used to cause or contribute to the incident |
| Assets | Information assets that were compromised or affected during the incident |
| Attributes | What was affected about the asset during the incident |
| Discovery/Response | Timeline, discovery method and lessons learned |
| Impact | What was the overall effect of the incident to the organization |
| Victim | Demographic information like industry and organizational size |
| Indicators | Optional indicators of compromise (ip addresses, malware hashes, domains, etc.) |
| Plus | Optional section for extending VERIS |

Table 7.1: Sections within VERIS

While it’s tempting to dig into the data (and we will), it’s important to understand the significance of these fields so we don’t misapply them. Therefore we will go through each part of VERIS in more detail and discuss the fields in each section. Keep in mind that the separation of these sections is for people to think about the structure, in the data there is nothing denoting the “incident\_id” field as helping with incident tracking for example.

type="warning"

While we are covering VERIS with some depth, we will not go into every field, and we won’t be able to cover every detail about the framework. For example, we won’t call out all the places the framework specifies a “notes” field (which is almost every section), and we won’t cover the indicators section in detail. Just keep in mind, that the framework is actively maintained and evolving. This chapter is discussing the 1.2.1 release, so be sure to refer to <http://veriscommunity.net/> for all the details and current specification of the VERIS framework.

Incident Tracking

Some of the fields within VERIS exist to simply describe or track the incident. These fields help us keep records straight by identifying each with a unique identifier, tracking the source of the incident and any related incidents. You’d use the source\_id field to compare your unique “source” of incidents to something like the VCDB (which has “vcdb” in that field). Required fields are marked. If something has a value of “**factor**” that means it is a restrictive list and only those values are expected.

|  |  |  |
| --- | --- | --- |
| Field | Value | Description |
| schema\_version | string | VERIS version (currently 1.2.1) |
| incident\_id | string | unique identifier (VCDB uses GUID) |
| source\_id | string | origin of the data (VCDB data has "vcdb") |
| reference | string | URL or internal ticketing system ID |
| security\_incident | factor | Confirmed, Suspected, False positive, Near miss |
| confidence | factor | High, Medium, Low, None |
| summary | string | free text summary of incident |
| related\_incidents | string | free text, other incident\_id’s |
| notes | string | free text |

Table 7.2: Incident Tracking Fields

Just looking down the list, there is only one or two fields here we will use during analysis and those are the two “factor” variable (again, this means they are restricted to a list of expected answers). The security incident is required and will help us split our analysis on whether the event was a confirmed security incident (an asset has a security attribute affected) or not. The confidence rating is a rare subjective field. It enables the analyst to record their subjective assessment of how confident they are in the accuracy of the data they entered. This optional field is not heavily used and won’t appear much in the VCDB incidents we’ll look at.

Threat Actor

Earlier in this chapter we talked about the challenge of conflation. This is something we want to be aware of especially in these next three sections (actor, actions and assets). We talked about the framework DataLoss DB used with a single conflated breach type and we see the same thing with the framework used by Privacy Rights Clearinghouse. Their framework also uses a singular “breach type” to define each event and again it will mix in the actors and actions into the one label. For example, they have an “Insider (INSD)” type, which is defined as “Someone with legitimate access intentionally breaches information - such as an employee or contractor.” And then a type of “Physical loss (PHYS)” which is defined as “Lost, discarded or stolen non-electronic records, such as paper documents.” These simplified labels can quickly become confusing during data entry if, for example, an insider steels paper documents. We may see insiders breach systems, drop malware and social engineer, just as an external actor would and we want to separate the two clearly in the data. VERIS tackles that by separating who from what they did and what was affected. We should mention that we shouldn’t think of the method Privacy Rights Clearinghouse uses as right or wrong. It just has a different focus and represents different priorities and goals. We would like to inform and support security decisions, which benefits from more detail than a single “breach type” label.

|  |  |  |  |
| --- | --- | --- | --- |
| actor | Field | Value | Description |
| external | motive | factor | Helps understand intentions, same enumeration for all instances |
|  | variety | factor | Shapes resources, capability of external actor |
|  | country | factor | ISO-3166-1 2-digit country field |
| internal | motive | factor | Helps understand intentions |
|  | variety | factor | Shapes resources, capability of internal actor |
| partner | motive | factor | Helps understand intentions |
|  | industry | string | U.S. Census NAICS code |
|  | country | factor | ISO-3166-1 2-digit country field |

Table 7.2 Threat Actor Fields

The threat actor section also introduces us to the nesting feature of VERIS. At the top level we are talking about the actor, so we have a section in the data for “actor”, then there are three classes of actors defined, external, internal, and partner, all of which are optional. Within each of those classes we want to add details about that type of actor. Looking down the values we have in this section we see all factors. That means we should be able to include any of these or use them as pivot points. In other words, if we want to support a threat modeling exercise that is comparing different threat communities, we could extract the actions for financially motivated actors and compare that to disgruntled employees.

Threat Actions

This section collects variables to describe what the threat actor(s) did or in some cases, used during the event. Again we have nest variables under top level categories which are:

* **malware**: malicious software, script, or code run on an asset that alters its state or function
* **hacking**: person (at a keyboard) attempting to access or harm an asset without authorization
* **social**: exploiting the human element (phishing, pretexting, etc)
* **misuse**: abusing resources or privileges contrary to that which was intended
* **physical**: personal actions involving proximity, possession, or force
* **error**: anything done (or left undone) incorrectly or inadvertently
* **environmental**: natural events and hazards within the immediate environment or infrastructure of assets

We have to be careful as we work with these categories. There are many opportunities for misinterpretation and misclassification across categories. These categories and the factors in each category are explained in detail along with use case examples at the VERIS website (<http://veriscommunity.net/>). Once you spend some time and look at a few examples though, these get to be a bit easier and eventually will become intuitive.

|  |  |  |  |
| --- | --- | --- | --- |
| action | Field | Value | Description |
| malware | variety | factor | functionality of malware |
|  | vector | factor | how the malware was installed/infected |
| hacking | variety | factor | type(s) of hacking action |
|  | vector | factor | path of attack |
| social | variety | factor | type(s) of social action |
|  | vector | factor | path or method of communication |
|  | target | factor | role of targeted person |
| misuse | variety | factor | type(s) of misuse action |
|  | vector | factor | path or access method for misuse |
| physical | variety | factor | type(s) of physical actions |
|  | vector | factor | method of physical access |
|  | location | factor | physical location of action |
| error | variety | factor | type(s) of error actions |
|  | vector | factor | cause of error |
| environmental | variety | factor | type(s) of environmental actions |

Notice how “variety” and “vector” are repeated over and over? Every action category has variety field with unique enumerations for each category. All but the environmental actions have a vector field, again with unique enumerations for each category. Finally social actions also ask for the target of the social action and physical actions ask for a location of the action and that explains the whole section! Again notice how every field here is a factor, meaning we’ll be able to split, pivot and/or filter based on these fields.

type="note"

Multiple Events in the Attack Chain

Anyone who has been around information security knows that breaches tend to not be simple and single events. Often times the attacker will perform multiple actions and this complicates the recording process. Most of the factors in the VERIS framework support multiple answers. On one hand, this is very nice, because we don’t have to pick “the one best answer” for a complex security event, but on the flip side, this adds complexity for data management and analysis. As we’ll see later in this chapter, this isn’t as hard as it first seems.

As an example, suppose an attacker sends a phishing email to an executive’s assistant and quickly follows that up with a phone call pretending to be a business partner who sent the email. These are two actions and we should see both “Pretexting” and “Phishing” selected in the social.variety field. If the phishing email contained malware that is installed, we’d also see the malware action along the variety and vector of “Email”, since it was installed via the phishing attack. When we represent this data, if we count up the actions, we’ll usually have more actions then events. This naturally precludes the use of pie charts, which ends up being beneficial for all parties involved.

There is also a common notion within information security of the “attack chain” or “kill chain”. The concept is to establish the actions of the attacker in the order they happened. While VERIS allows multiple actions, it does not record the order they occurred in. This was a conscious trade off of cost versus benefit. Attempting to put the events in order created substantial overhead for the analysts and was taking too long to enter. Most of the time the order of attacks in reports and tickets (definitely in media articles) are either vague or missing completely. As a result VERIS simply records the presence of attacks in order to reduce the amount of effort during data collection.

Information Assets

Assets are the information containers (servers or other devices) that we are trying to protect. Like the others we have a top-level category, which are as follows:

* Server (S): system providing service(s)
* Network (N): infrastructure device or appliance
* User Device (U): end user equipment (laptop, desktop)
* Media (M): data storage devices or physical documents
* People (P): since people can be affected
* Kiosk/Public Terminal (K): public-use devices

Within each category there are several varieties of assets, but the category and variety are stored in the same field. For example a Mail Server is stored as “S - Mail” and a desktop computer is a “U - Desktop”. Associated with each asset is an optional “amount” field, which allows us to record multiple assets with the same variety when they are involved in one event.

|  |  |  |  |
| --- | --- | --- | --- |
| asset | Field | Value | Description |
| assets | variety | factor | specific type of asset, prepended with letter for category |
|  | amount | integer | count of the above asset |
| asset | accessibility | factor | how accessible the assets are |
|  | ownership | factor | who owns the assets |
|  | management | factor | who manages the asset |
|  | hosting | factor | where (physically) is it hosted |
|  | country | factor | location of assets (if different from victim) |
|  | cloud | factor | type of cloud service, if cloud |

There is quite a bit packed into the assets and these are relative recent additions to the VERIS framework. There is a lot of focus around mobile devices and employees bringing their own device into the corporate environment. Also, there may be unique exposures from cloud hosted applications and assets, so that will be captured here as well. Note also that these are all factors so there are only a handful of possible answers, we cannot write in “very” for accessibility of the asset as an example.

Attributes

The attributes of the above assets are what we work hard in information security to not have affected. Attributes are based on the C.I.A. triad, which stands for confidentiality, integrity and availability. For a while VERIS extended these three with three more attributes to record the Parkerian Hexad (named after their originator, security pioneer and long time security researcher, Donn Parker). The extra three attributes included possession and control, authenticity and utility. But the added fields just did not yield enough benefit for the added cost of separate categories so they were combined with the three top categories. For simplicity, when a VERIS record is stored, the sections are just labeled with the three primary categories (in bold below). The three main sections of attributes are:

* **confidentiality**, possession and control: data was observed or disclosed to an unauthorized actor, owner may no longer have exclusive custody
* **integrity** and authenticity: asset is incomplete or changed from authorized content and function, conforms to expected state
* **availability** and utility: asset is not accessible, useful or fit for use

The use of these categories can be quite helpful in separating out the areas to focus on for a security team. The Verizon Data Breach Investigation report has exclusively focused only on breaches where the confidentiality attribute was affect and there was a confirmed data disclosure.

|  |  |  |  |
| --- | --- | --- | --- |
| attribute | Field | Value | Description |
| confidentiality | data\_disclosure | factor | status of confidentiality breach |
|  | data\_total | integer | number of records (see below) |
|  | data.variety | factor | type of data disclosed |
|  | data.amount | integer | number of records |
|  | state | factor | state of data when disclosed |
| integrity | variety | factor | nature of effect |
| availability | variety | factor | nature of effect |
|  | duration | time range | duration of availability/utility loss |

There is a new field type here of “time range” which is actually two fields, a “unit” of time and a “value” for that unit of time. The unit has general measurements of time: seconds, minutes, hours, days, weeks, months and finally years. The value represents how many of those, so 3 weeks or 6 months. Within VERIS it was found that knowing a specific amount of time first was often difficult to get precisely and this method of generalizations was usually possible to discern between. As an example, it may be known that the server went offline during the DDoS attack, but the specific quantity was more than 60 minutes and definitely not a full day. In that case, we would see “hours” in the unit, and if the specific number of hours is known we’d see a value, otherwise it may be blank if the precision is unknown.

type="note"

Counting records

One of the more common pieces of information in publicly disclosed breaches is the number of records. Perhaps reporters and the general public demand this and the victims are forced to provide a number, even if it’s all the records in the database. Records can be relatively easy to count when the data comes with obvious separation. Payment (credit) cards, identities or medical records are quite clear in their separation and lend themselves to being counted. But when we get into more complex types like classified information or trade secrets, the ability to count records becomes a little less defined. Perhaps the number of physical documents could be used, or number of files disclosed, but oftentimes it’s difficult to count them. Overall, analysts struggle to record a precise number for the data varieties of classified or internal information and trade secrets and we will have to account for that in the analysis and visualizations.

Discovery/Response

We just saw our first time range in the availability attribute and we’ll see that a lot more in this section for the timeline data. Some of these fields are not in a section like the previous four sections, but the timeline does have its own section.

|  |  |  |  |
| --- | --- | --- | --- |
| section | Field | Value | Description |
|  | discovery\_method | factor | how event was discovered |
|  | control\_failure | string | free text field to describe what, under the victims control, failed |
|  | corrective\_action | string | free text field, describing what the victim should do |
|  | targeted | factor | targeted or opportunistic attack |
| timeline | incident\_date | date | date of incident |
|  | compromise | time range | time to initial compromise |
|  | exfiltration | time range | time from initial compromise to data exfiltration |
|  | discovery | time range | time from initial compromise to discovery |
|  | containment | time range | time from discovery to containment |

We have a new field of “date” here and this is not a standard date field. Because VERIS has to account for unknown values the date fields are in separate variables. Too often then we’d like for our analyses, the precise date of the incident isn’t known or isn’t reported clearly. The framework assumes at least the year is known, but the month, day and time fields are all optional in that date field. The other fields in the timeline are the same time range values we saw in the availability attribute.

Notice also that the control failure and corrective action suggestions are free text. This makes them difficult to include in our data analysis without more effort. Finally, the discovery method is one of the rare enumerations that cannot have multiple answers. The framework assumes that once the incident was discovered it could not be discovered again, so only one method of discovery is allowed.

Impact

The impact section is perhaps one of the most, if not the most, sparsely populated section in the incidents. This has nothing to do with the framework and everything to do with the lack of accurate data to collect and record about the impact. The result is that this section has some subjective measurements and estimates.

|  |  |  |  |
| --- | --- | --- | --- |
| section | Field | Value | Description |
|  | currency | factor | ISO 4217 currencies for monetary estimations |
| overall | rating | factor | qualitative rating of overall impact |
|  | min\_amount | number | min estimated monetary amount |
|  | amount | number | most likely estimated monetary amount |
|  | max\_amount | number | max estimated monetary amount |
| loss | variety | factor | specific category of loss |
|  | rating | factor | qualitative rating of overall impact |
|  | min\_amount | number | min estimated monetary amount |
|  | amount | number | most likely estimated monetary amount |
|  | max\_amount | number | max estimated monetary amount |

Notice the repeating rating and monetary estimations. There is a dedication “overall” field here for those fields, but the loss section is defined in the data as an array. This means that the analyst can add multiple loss sections in the data for each variety of loss being recorded. The loss varieties are specific types of loss, for example “response and recovery” costs or “legal and regulatory”.

Victim

The last section we should cover is the victim section. If VERIS is being implemented inside a single organization, the fields in this section could be skipped (or hard-coded) since the victim will always be the same. But for cases like the VCDB where it is aggregating across many victims, this section is vital. We want to capture data about the victim with the intention of contrasting and comparing breach data when we split on these fields.

If you remember back to Chapter 5 where we touched on regression analysis, we attempted to find independent variables that could help describe the outcomes we observed. The data we are collecting about the victim can go a long way to describe the types of threat actors and their actions. For example, in the 2013 DBIR, Verizon saw state-affiliated espionage in at least 3 out of every 4 cases within the manufacturing industry and yet none in the retail industry. While industry alone is not a perfect predicting variable, it does help reduce our uncertainty, and remember that’s what we are after here.

|  |  |  |  |
| --- | --- | --- | --- |
| victim | Field | Value | Description |
| victim | victim\_id | string | identifier or name of victim |
|  | industry | string | U.S. Census NAICS code |
|  | employee\_count | factor | Label for number of employees |
|  | country | factor | ISO 3166-1 2-digit country code |
|  | state | string | state, province or region in country |
|  | locations\_affected | integer | number of locations affected |
|  | revenue | integer | annual revenue of the victim |
| secondary | victim\_id | string | list of secondary victim\_id or name(s) |
|  | amount | integer | and/or count of secondary victims |

The most recent change to the VERIS framework (version 1.2.1) changed how this section is stored. In version 1.2 and before, the entire victim section could be repeated for each victim involved in the incident. For example, if an organization is breached and they were processing data on behalf of another organization, they would become a victim of the same breach. This was found to be confusing though and the victim was reduced to just supporting one single victim. The fields in the “secondary” section where added in 1.2.1 to capture what was treated as a multiple victim breach.

Anywhere we have an industry (which is here and in the threat actor partner section), they are listed as a “string” but they should not be free text. Following one of our guidelines to leverage other resources wherever possible, VERIS leverages the U.S. Census Bureau’s North American Industry Classification System (NAICS). Doing so adds flexibility and a level of detail not possible with other industry classification systems. If people were going to create a list of industries, they’d probably come up with a dozen or so high level categories and call it good. NAICS started there (20 top-level categories actually), but then made it extendible and enables more and more detail to be put into the industry specification. Industries within NAICS are represented by a 2 to 6 digit integer which is why VERIS stores them as a string and not a factor, the list is enormous (but is listed at http://www.census.gov/eos/www/naics/).

As an example, let’s take the pizza shop down the street. The NAICS code for that is 722511 which represents “Pizza Parlors, full service.” But sometimes, maybe the analyst just knows it’s a restaurant, then they would just record “7225”, or maybe they know the victim offers some type of food or beverage service, then they may enter “722”, but if they are really unsure exactly what type of service establishment it is, we may just see “72” for “Accommodation and Food Services”. When we do analysis on this field we can drill down or up depending on the level of detail we want. But be careful, the more detail we put into this field, the smaller we will be dividing up our samples.

Extending VERIS with Plus

Finally we have the catch all section labeled as “plus”. Within the VERIS framework there technically is nothing specified in this field and the data schema simply allows anything to exist in this section. It exists to allow individual implementations to record additional fields not in the base VERIS schema. If we look at the VCDB repository for example, each incident has a plus section with the analyst who recorded the incident and the time it was created along with a few other fields being considered. Any implementation can apply the guidelines (or not at your own peril) and add their own fields here. If they fields are useful, feel free to suggest the change to the core framework!

Seeing VERIS in action

It’s always nice to take some time before we jump into the analysis to look directly at the data. It helps set the context in our mind and may help shape our approach to the analysis. Since the average incident is about 100 lines of JSON, we won’t include the whole incident. Please take some time to surf around the VCDB repository and look at the data there for full records. As a good example, here are the actor and action sections from an incident from VCDB:

"actor": {

"external": {

"country": [ "SY" ],

"motive": [ "Ideology" ],

"variety": [ "State-affiliated" ]

}

},

"action": {

"hacking": {

"variety": [ "Use of stolen creds" ],

"vector": [ "Web application" ]

},

"social": {

"target": [ "End-user" ],

"variety": [ "Phishing" ],

"vector": [ "Email" ]

}

}

If you have never seen JSON before, this is what it looks like. One thing to keep in mind is that rarely if ever would we want to edit the JSON by hand. It’s not that JSON is terribly difficult, but it’s terribly easy to mistype a factor, forget a comma or a quote or something. Just be sure that if you are attempting to create a JSON file that you have a way to validate the JSON and if possible the entire contents and factors within the data.

The best part apart working with JSON is that it typically imports right into native objects in the languages we use. Within python, an incident in JSON is imported directly to a python dictionary. The code to load up a JSON object and view the hacking variety in this object is relatively simple:

# python to load JSON and read hacking variety:

import simplejson as json

# Open the JSON file and read the raw contents into jsondata

jsondata = open("some\_veris\_file.json").read()

# convert the contents into a python dictionary

incident = json.loads(jsondata)

# now access the hacking variety:

print(incident['action']['hacking']['variety'])

Which would print out the python list object for hacking variety and display ['Use of stolen creds'] the example above. In reality we should wrap the json.loads() command with try-except, if the file has any errors in the JSON syntax, they will be caught that way. Plus the hacking action is optional, and we’d want to test if the “hacking” key existed before we attempt to read it. But it helps show how easy JSON can be to load and work with.

Within R, json files are converted to a native list object. Performing the same function of loading the file and printing the hacking variety is done like this:

# use the rjson library

library(rjson)

# fromJSON accepts a filename to read from

incident <- fromJSON(file="~/Documents/json/newfinal/jay.json")

# print the hacing variety

print(incident$action$hacking$variety)

[1] "Use of stolen creds"

The R code returns a one-element vector with the value in the hacking vector. Again, in full-featured code, we’d want better error checking than this, but it does show how easy this data is to load into native objects and work with.

Creating research questions around the asset

Since our goal is to support the decision around setting priority for remediation efforts, we have to think through the types of questions we will be asking during the prioritization. As a starting point, every remediation effort is about addressing one or more **asset** in our environment. An asset is something we value. Examples include servers, laptops, printed documents and so on. We know we want to form one or more research questions around the asset:

Let’s start out with an obvious question:

**Are some types of assets more likely to be involved in a security incident than others?**

Notice how the question is worded to be a good research question. It clearly defines what we want to measure, how we want to gather the data, and what we want to test in this data once it’s gathered. We can then break it down like this:

What we want to measure: **type of asset**, categorical list

How to gather: **yes/no**, if asset was involved or not

Test: compare **proportions** of incidents where type of asset was involved

There is an assumption going into this question which is some types of assets are more targeted than others. If we thought that all assets have an equal chance of being involved, this may not be worth our time. However, there’s a good chance that there is a difference in the types of assets involved in security incidents and this is interesting enough to include in our list of research questions. While this question is interesting, there are many other questions we could ask around assets to help reduce our uncertainty. For example, the type of data it houses and who manages the asset, whether it’s an employee owned device, a third party partner or perhaps even a cloud-based solution we’re talking about. With those in mind, let’s generate a few more questions.

Are assets managed by others more likely to be involved in a security incident that assets managed by us?

What to measure: **management responsibility** for asset (categorical list)

How to measure: **yes/no** if asset was managed by other party

Test: compare **proportion** of incidents where assets were managed by other party

Are assets in the “cloud” more likely to be involved in a security incident than non-cloud assets?

What to measure: “cloud” technology contributed to the incident (categorical list)

How to measure: **yes/no** if cloud technology contributed to the incident

Test: compare proportion of incidents where assets involved cloud-specific weaknesses

We will hold off for now on the type of data, because that will have more meaning for us than it’s association with the asset. For now these are a good set of questions. This will help us get a picture for how this notion of an asset can play into incidents and some attributes that may or may not contribute to its involvement in a breach. We could easily develop dozens of questions around the asset (type of OS, version, software, and so on), but there is a very important lesson about this type of data collection. Each question we ask and each data point we want to capture comes with a rather high price tag.

Creating research questions around the actors

Knowing that our goal is to support the prioritization of remediation efforts, understanding who is attacking and their motive will help us prioritize where to focus our resources. For example, if we knew that most attackers are financially motivated and are external (non-employees) to our organization, we could prioritize projects that protect can lead to direct financial benefit for the attacker. Even if we just collected asset information and attacker information, we can see how the two fields together are more valuable than both of the fields alone.

attackers are more likely to attempt a brute force attack against passwords, we may wish to prioritize improving systems with weak or no password requirements. On the other hand, if we discover that attackers are more likely to grab credentials with key loggers, we could deprioritize enforcing complex passwords and focus on the multi-factor authentication projects.

Creating research questions around the actions

Knowing that our goal is to support the prioritization of remediation efforts, understanding what attackers are doing will help us prioritize where to focus first. For example, if we know attackers are more likely to attempt a brute force attack against passwords, we may wish to prioritize improving systems with weak or no password requirements. On the other hand, if we discover that attackers are more likely to grab credentials with key loggers, we could deprioritize enforcing complex passwords and focus on the multi-factor authentication projects.

There are two large challenges with recording what happened and we’ll have to address both at some point. First, any given incident may be the result of more than one action. While a SQL injection attack is typically a single action that exfiltrates data immediately, a phishing attack is typically one of many actions the attacker takes before they exfiltrate data. Second, there is no end to the amount of detail we could collect about the actions. Even within SQL injection, did they use an automated tool? What was the language of the web application? Was it custom written or an off-the-shelf application? Was there any application firewall or filtering in place? At some point rather quickly, we will have gone too far with the level of detail we seek (see the call out on the “cost per datum”).

With the knowledge that there are a lot of possible actions, we will want to limit the number of questions we ask, since it may be duplicated across multiple different actions. This short list should be enough to get started.

Are some threat actions more prevalent than others?

We will need a list of actions to select from

Are the attack vectors (pathways) more prevalent than others?

Finding the answers to these questions should help with our overall goal. But we still have a problem here in that we could easily create a list of actions that are long. Most every method of collecting breach data will attempt to capture “what happened”. Let’s take a quick survey of what’s out there.

Privacy Rights Clearing house has the following “types of breaches”:

Unintended disclosure (DISC) - Sensitive information posted publicly on a website, mishandled or sent to the wrong party via email, fax or mail.

Hacking or malware (HACK) - Electronic entry by an outside party, malware and spyware.

Payment Card Fraud (CARD) - Fraud involving debit and credit cards that is not accomplished via hacking. For example, skimming devices at point-of-service terminals.

Insider ( INSD) - Someone with legitimate access intentionally breaches information - such as an employee or contractor.

Physical loss (PHYS) - Lost, discarded or stolen non-electronic records, such as paper documents

Portable device (PORT) - Lost, discarded or stolen laptop, PDA, smartphone, portable memory device, CD, hard drive, data tape, etc

Stationary device (STAT) - Lost, discarded or stolen stationary electronic device such as a computer or server not designed for mobility.

Unknown or other (UNKN)

mple, Privacy Rights Clearing House

we’ve gone to far, and if you reference back to the note “We will have to create a cut-off point somewhere

a lot of In other words, if we ask for a single label for the actions, we are going to lose out

we will find a good deal of information in understanding what attackers are doing so we know what to prioritize One common question across all methods of breach data collection is “what happened?” But how this question gets answered is anything but common. and this is where we have to keep an eye on complexity.

ANOTHER NOTE: Conflated terms

CALLOUT on NAICS

Below are notes, can ignore for now

Within that statement of the goal, we have included two questions. These are not research questions and in fact would be absolutely horrible research questions. They are ambiguous and next to impossible to measure directly but they are serving as a compass. Every research question we form, should serve to eat away at the question.

Given that this is the goal, This type of general goal will help us define We want to understand if there are areas of the organization that are more likely to be targeted Where should we be focusing our assessment efforts efforts and prioritization and risk assessments (in any and all incantations of the concept). In other words we want to focus on spotting the large trends, and some of the details (like recording specific IP address) will be out of scope. Let’s state that

For example, if we want to collect breach data to help with upcoming project selection and budget allocation, we’d want to collect different data and manage it differently than if we want to study breaches for specific indicators of compromise such as IP addresses

Defining an overall goal is going to

miss some data we’d really care about. Perhaps we want to collect data to feed into

and Unfortunately the media does not help much in that regard as they play the “name and shame” As something we need to As if the victim was dumb enough to get robbed, but now that we’re moving into a data-driven world we should be thinking of a security breach as incredibly rich source of data. Where else can we learn about who is attacking, their tactics and practices and targets?

Before we jump into collecting data we have to ask ourselves, what do we want to know? Because if we survey the landscape of breach data (and we will in the next section), we may find that the types of research questions we are able to answer with what is available indicate that the research questions either weren’t well thought out, or were actually that simple. Huh, let’s see.

landscape

Once our defenses fail (and they will), we

We should probably take a moment to mention that one of the authors of this book

and looking at some of the more intangible aspects of computer security.

One example that is close to home for

Humans are wonderful/horrible sources of data

We want to begin this chapter by covering methods of collecting data from the environment and then discuss the challenges we face collecting data on breaches. We will talk about several methods of collecting breach data and then we will look at data analysis techniques on public breach data recording using the Vocabulary for Event Recording and Incident Sharing (VERIS) framework.

Breach data represents an extremely valuable source of information.

We want to focus this whole chapter on VERIS, Identify Verizon and the DBIR. We want to establish this framework below and then walk through what we do with breach data collection. Let’s not focus on the Data in the DBIR, but how we are able to pull information from reports.

When will want to

Four phases of a research project:

Define the goals and research questions

Establish data model and collection method

Collect and manage the data

Data analysis and presentation

*“VERIS is complex and they didn’t want to deal with it.”*

This

**Defining the Research Goals**

Whether we call it risk analysis or predictive analytics, a metrics program or a longitudinal study, or perhaps just simply a research project the effort should always begin with the same question: What are we trying to accomplish? Answering that question, with all its complex and nuanced details, will align expectations and help shape all of our following actions. We cannot stress the importance of this step enough. Far too often we have come across data that was gathered and prepared at great expense for some, yet failed to address even the most basic questions of the organization. In spite of the fancy graphics and glossy reports, the entire effort was doomed before it started because there was not agreement on where it was headed. One standard approach to getting agreement is establishing one or more research questions that the effort will try to answer.

Wade: What questions did you want to answer in the beginning?

When we prepare to collect data about information security events, we have an added complexity. Not only do we have to define what questions we want to answer, but undoubtedly, we will also have to define the questions we won’t answer. Not because we would be incapable from a data perspective, but every data point we collect comes at a price and that price-per-datum is much higher than people imagine. The primary limiting factor in collecting breach data is not the availability of data, but the amount of effort it takes to collect that data. If we attempted to answer every question we can think of, we would be crippled in the data collection step and the whole effort would ultimately fail. Therefore we have to state the first rule/guideline/mantra/principal:

Every data point has a high collection price. -- Every data point costs

One data point to collect is worth a dozen

Think of each data point as adding head count to the budget. The initial cost may be relatively low, but there are hidden costs. We want to limit

Within VERIS, we set our goal to support the decision-making process at a strategic level. This has several benefits and drawbacks. First, it sets the focus at more of a strategic layer than a tactical layer and frees us from the collection of specific indicators of compromise. The benefit of that is that we have less data to collect and the collection can be a bit easier. The drawback is that we lose the ability to inform detailed analysis of log data. With VERIS data alone, we cannot help determine if a specific transaction, host or executable is malicious or not. However, we can help determine if we should focus

So that brings us to

We will get into why this is true. Every question we put on a collection form, or measurement we want to read, comes at a price. While it’s easy to think, “it’s just one data point”, think about it throughout the life cycle of the data. Consider how many times that one data point will be touched in the life of the data. We may have one or more validation and/or verification processes, one or more transformation processes and certainly a handful of analysis and visualization efforts. While it is just one question, that data point may require a minute here and a few minutes there at multiple stages in its lifetime. Now multiply that by hundreds perhaps even thousands of records and it’s easy to see that the price per data point is a critical consideration. The effect of this compounding cost is that every data point must be scrutinized and tied back to one or more research question.

On top of that, it’s never just one more data point. If we become lax, the one more data point turns into dozens. When we consider changing VERIS, we will debate every single data point we include.

The VERIS community database (VCDB) has over 1,500 records in it at the time of this writing. If we spend an extra minute collecting 5 “interesting” data points,

For every data point we should add some type of error checking and/or validation. If we upgrade or migrate the data, we’ll hav Every data point isn’t just collected and we’re done. That one data point becomes part of the defining context in which is was gathered. If we have validation and/or verification, that data point must be considered, if we transform or move the data (and you most likely will do this multiple times), we have to touch that data point. and now must be covered by the data validation process. If relationships are established we

We have the collection, validation and verification (depending on importance)

anyone has, we

Security events offer a wealth of data and the main limitation re is no end to the amount of detail that we can gather.

**Establishing the Data Model**

The term “data model” can mean a variety of things depending on context and perspectives. In this case, we are using the term a bit ambiguously. At a bare minimum, we want to

leave the details up to the specifics of any individual impelementation. At a basic level, what we are describing is what fields we will look to collect and how those will be stored.

up to implementation details.

**Collecting and Managing Data**

Collecting data from the environment typically has a higher cost than collecting data from a log file. If the benefit is great enough, we may even end up recording one data point at a time so we want to be sure we are approaching this type of data collection with a clear intent. This is where our research question (discussed in Chapter 1) will be critical. We want to begin with one or more questions that we want to answer. By jumping right into data collection with one or more research questions, the data collection ends up favoring the convenient and obvious. Take for example a call center, if we leap into data collection, we will capture what’s obvious and available: how many calls, the operator, duration, etc. But what if we want to understand how effective the call center is from the customer’s perspective? That won’t be found in a log file and all the effort in collection and preparation is wasted. We offer the following steps to follow when creating a program of data collection:

Create purpose and direction through a list of questions

Brainstorm a list of questions that drive action or affect decisions moving forward

State the questions in a way data can answer

Prioritize and select one or more critical questions

if we answered this, would we change? If it was affirmative or negative, anything different? (run through possible answers, possibly set a threshold)

Select data collection method

Identify sources and methods of collecting data that address questions

Establish data model and vocabulary (fields and data types - taxonomy/framework)

is this field a singular value? multiple attributes?

Every data point and field has a cost - go back to questions, do we care the specific value or just that it’s above or below threshold.

Avoid pack-rat mentality, while it’d be awesome to have all the data, attempting to gather it is a recipe for disaster

Collect and manage

Varying data collection methods

data storage - dictated by 2b and 2c

Data Analysis

categorical data analysis

atomic data: Boolean, integer, (float), string

collections: list or dict (assoc, object)

We touched on

Once we

Integrating into a process (ticket system)

document/log review (after action reports, forms)

to answer a question, we have secondary review

survey

census (every member in population)

Focus group

Interviews

Observation

Survey

Follow-up interview

(after action report)

Issues:

Validity and consistency Issues

Cost to collect (resources like time, money, complexity)

Inference / Benefit of collection

“ expected quality of the collected data, estimated costs, predicted nonresponse rates, expected level of measure errors, and length of the data collection period “

T

Type of research: confirmatory, exploratory

Research question is the reason we are doing this.

When answered, we’ll know something we didn’t before.

1. The Verizon Data Breach Investigations Report leverages the VERIS framework for its data collection and data analysis and may help the reader get a context for this chapter. The most recent report can be found at www.verizonenterprise.com/DBIR/ [↑](#footnote-ref-1)