Developing A Security Data Science Mindset

Figure 12- [f0#.eps]

Figure 12-# is a slight modification of Drew Conway’s “Data Science Venn Diagram” (<http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram>) and a simple tool for a quick self-evaluation of where you currently are on your journey to becoming a security data scientist. We’ll take a look at each major component along with the interactions between each component to give you an idea of where top topics in this book fit and provide pointers for delving into or shoring up areas that may not currently be strong points for you.

The Hacker

The term “hacker” has a great deal of confusion surrounding it as it has been usurped by news media and manipulated by marketing firms. In the context of a security data scientist, “hacker” means:

* Being able to code, either scripting in a language like Python or full on programming in something like C;
* Knowing a wide variety of data formats and understanding how to slice, dice, and bend them to your will;
* Having the ability to think logically/scientifically (essentially, not jumping to conclusions) as well as algorithmically (break apart a problem into its composite parts)

If you are an information security professional who isn’t a coder, Chapters 2, 3 and 4 have been designed to help you bootstrap into that skill. If you are a coder, those same chapters cover a language that is most likely new to you (R) and place coding in the context of data analysis versus application building or systems administration, which may be more familiar problem domains for you. Whether you’re at the top of your game as a programmer or just getting started there is always more to learn and no shortage of resources available to do so, including:

**Codeacademy** (<http://www.codecademy.com/>). This is an especially good resource for those new to programming in general or those unfamiliar with a particular language. It would be worthwhile to take a look at the JavaScript & jQuery offerings given the emphasis on JSON in Chapter 8 and D3.js in Chapter 11, and if you don’t know Python well (or at all), their Python course can definitely help.

**Code School** (<https://www.codeschool.com/courses>). The offerings at Code School can be a bit overwhelming and not all are free. However, their R course is freely available at the time of writing and will help you navigate the syntax and nuances of the language.

**W3Schools** (<http://www.w3schools.com/>). If you haven’t had the opportunity to shore up your HTML/CSS/JavaScript skills, W3Schools provides an extremely friendly environment to both learn and experiment. You’ll need at least a basic understanding of these client side components if you want your analyses and results to reach the widest audience.

**StackExchange** (<http://stackexchange.com/>). While you won’t necessarily learn how to code at the StackExchange family of websites, you will have a place to look for answers or ask questions when you’re stumped. Whether it’s trying to understand some esoteric option in ggplot2 or how to do something a bit more complex with a pandas data frame there’s a very good change the answer will be in StackExchange.

When it comes to data formats, security professionals are in the unenviable position of having to be able to manipulate everything from NetFlow captures, to full packet capture (PCAP) dumps and almost every log format known to humankind. The IronPort log file snippet in the MongoDB section of Chapter 8 is an example of how “imperfect” your data world is. While that log file contains highly useful data, it’s in a format that you must parse and convert to make it useful. The only way to get good at that is to actually do it over, and over, and over again, building up reusable bits of code and techniques along the way to save you some time later on.

Learning how to think logically, scientifically and algorithmically requires time, effort and practice. Formal, in-person, instructor-led education may work best for some students, especially those who have shied away from programming. However, introductory sites like Project Euler (<http://projecteuler.net/problems>) can get you started down this path, more advanced and diverse problem sets can be found at Kaggle (<http://www.kaggle.com/competitions>), and you can delve into wide and deep security domain problems at the VAST Challenge (<http://vacommunity.org/VAST+Challenge+2013>) site (look in both current and previous years’ sections).

Overarching these three traits is the need to develop and hone a sense of *curiosity*. In fact, curiosity may be the **single most important trait** of a “hacker”. The need to know *why* or *how* something works the way it does from start to finish is an invaluable driving force when faced with a complex data science problem. When combined with the other two security data science primary skills (statistics knowledge and security domain expertise), you can eventually get to a place where developing a successful NetFlow-based malware traffic clustering algorithm is as rewardgin as TO DO

The Statistician