Hacking the Tech in *Little Brother*

Gerald Kruse

Bookend Seminar, October 28, 2015

Gerald Kruse is Assistant Provost at Juniata College, and he holds the John ’54 and Irene ’58 Dale Professorship in Mathematics, Computer Science, and Information Technology.

Juniata College’s 2015 Summer Reading was *Little Brother* by Cory Doctorow.[1] The announcement e-mail describing the book said, “The book takes place in the future (near future) and explores what types of compromises our society and government are willing to make in the aftermath of a terrorist attack.” One of the main themes in *Little Brother* is privacy, a topic Cory Doctorow is quite passionate about. While I appreciate this theme of the book, what really resonated with me were the elements of the book that we address in my classes.

The protagonist in *Little Brother* is Marcus Yallow, a talented but disillusioned teen with a variety of interests, many involving hacking and unpermitted gaming and electronic communications. His hacker name, ***w1n5t0n***, resembles the name Winston in the leet (for “elite”) lingo. Marcus is adept as a hacker and takes pride in confounding his high school’s surveillance technology. After Marcus and his friends skip school to participate in a live action game, they are near ground zero in a terrorist attack. Some of his friends escape, but Marcus gets detained by Homeland Security. This sequence in the book is pretty intense. After Marcus spends several emotional and humiliating days in captivity, he’s released and warned that he’s going to be under surveillance. Marcus decides to fight back and stage a revolt.

I appreciated the way the book unfolds with clues hinting at how its near future is different from today’s society. For example, Marcus’s high school has many familiar elements, but it utilizes gait-recognition software to track students. While there are current efforts to develop this software, the publicly available technology is not as mature as it is in the book. So, what do Marcus and his friends do to try to confound the gait-recognition software? They put gravel in their shoes, causing their stride to vary from step to step. They also crack their “SchoolBook” laptop computers that all students are issued. You may recall that the trend of providing mobile devices to students in schools began about ten years ago. The intent was for the students to use these mobile devices to interact throughout their school day. But what was one of the first things that happened? Students hacked their mobile devices to do things like send messages to each other and change the channels on the televisions in their classrooms. This hacking was certainly an unintended consequence of trying to provide all students access to technology. A similar

series of events happens in *Little Brother*, where it's fairly easy for the students to crack their school-issued laptops. It's done via something called a rootkit. Any file name with a prefix of ***sys*** will be hidden, and placing a ***sys*** file in the computer's root directory will allow a hacker to avoid the surveillance measures on the laptops. Marcus was able to be in class and appear to be taking notes, while in fact he was texting his friends or reading his e-mail or doing whatever else he shouldn't be doing according to the school rules. These two vignettes, confounding the gait-recognition software and hacking the SchoolBooks, set the tone for near-future surveillance society of the book.

After Marcus is released from his detention and decides to fight back, he utilizes a paranoid Linux operating system and runs it on an Xbox. If you are familiar with the proprietary Unix operating system, Linux is the open-source, freely available version. The paranoid operating systems are designed to protect the privacy of the users. Once again, these versions are not quite as mature in real life as in the book. The questionable plot twist, though, involves the system running these paranoid versions. What seventeen-year-old has an unopened Xbox in their closet? The book was well done, but this was one of the too-convenient plot twists that I thought were implausible, along with Marcus being rescued when the local police break down the door at the very moment when he's going to be water-boarded. I'm glad Marcus was rescued, don't get me wrong. I didn't want to see that through, but the timing certainly seemed serendipitous.

Another ubiquitous surveillance technology in the book is Radio Frequency Identification (RFID) tags, and in the book they are referred to as "arphids." Basically, all the students' books and school materials have these tags embedded and can be tracked. So, if students want to leave school during the day, how do they neutralize the tags? One way is to put them in some kind of Faraday bag, a special bag that doesn't allow any kind of kind of signal to be emitted. Another fun way is to simply destroy them in a microwave oven. This approach is celebrated in several videos on YouTube.[2] Later in the book, Marcus's group surreptitiously reprograms the RFIDs in people's touchless subway tags and gasoline cards as they walk by them. This is something of a possibility in the real world. It depends—some RFIDs can be reprogrammed, some can't. In the book, reprogramming the RFIDs confounds the surveillance system. Here's the passage from the book where they hatch the plan:

> "It's the arphid cloners," I said. "They're totally easy to make. Just flash the firmware on a ten-dollar Radio Shack reader/writer and you're done. What we do is go around and randomly swap the tags on people, overwriting their Fast Passes and FasTraks with other people's codes. That'll make *everyone* skew all weird and screwy, and make everyone look guilty. Then: total gridlock."[3]

When Marcus and his group are apart, they use The Onion Router (TOR) to communicate electronically. When a message is sent via TOR, it gets wrapped in layers of encryption as it progresses

through a network of volunteer-owned computers. TOR is available now, if you would like to surf the web in private or communicate securely by e-mail or instant messages. In fact, it's rumored that some of our intelligence operatives use TOR in hostile countries. One thing to note here is that a message doesn't always get wrapped in another key as it is forwarded along. There is a bit of randomness to it. If messages were always wrapped, then a packet of information might catch the attention of surveillance. It's analogous to whispering a message, rather than sending a message via a hidden signal. The very act of whispering draws attention to the existence of a secret message, but if no one is aware that a secret message is being transmitted, they're much less likely to look for it. This concept is steganography, or hiding a message in plain sight. This came up in the book when Marcus was considering ways to communicate with his friends. He had to make sure nobody knew he was trying to communicate with them.

Steganography, historically, is traced to Julius Caesar. Legend has it that when Julius Caesar wanted to send messages to the battlefield, he shaved a messenger's head, had a message tattooed on his head, waited for the messenger's hair to grow back, and then sent him to the recipient. It wasn't necessarily the timeliest of messages, but it was the last place anyone who captured the messenger would have looked.

Modern steganographic messages are typically sent via images, audio files, or video files. The concept is presented in Figure 1. Think of the colored squares on the top of the cube as pixels in some two-dimensional digital picture, perhaps a JPEG or Bitmap. But then, the image has depth, a third dimension, a column of bits below each colored pixel. For example, the light blue pixel at the front of the cube has a 1 below it, then a 0, and then another 1. This pixel is 101 in binary, which translates to 5 in decimal, and since there are three bits it has a depth of three. If you've ever seen the product specifications for a monitor and noticed the phrases "8-bit" or "24-bit" color, these are referring to the number of layers, or bits, used to represent each of the three primary colors (red, green, and blue) in a single pixel. A typical image might contain thousands or millions of pixels, each with the same depth. It turns out that the bit in the very last layer, either the 8^{th} or the 24^{th}, is not going to be very significant to the overall rendering of the image. An 8-bit color scheme results in a range of 256 different decimal numbers to define a color. If one of the pixels is a gray, say represented as 128 in decimal, changing the last bit to make the pixel 129 won't result in a noticeable change in the image. So a steganographic message could be made available by merely posting a new image on social media, where no one would expect to have a message embedded in the encoding of the image. The person receiving the message would simply decode the last layer of bits.

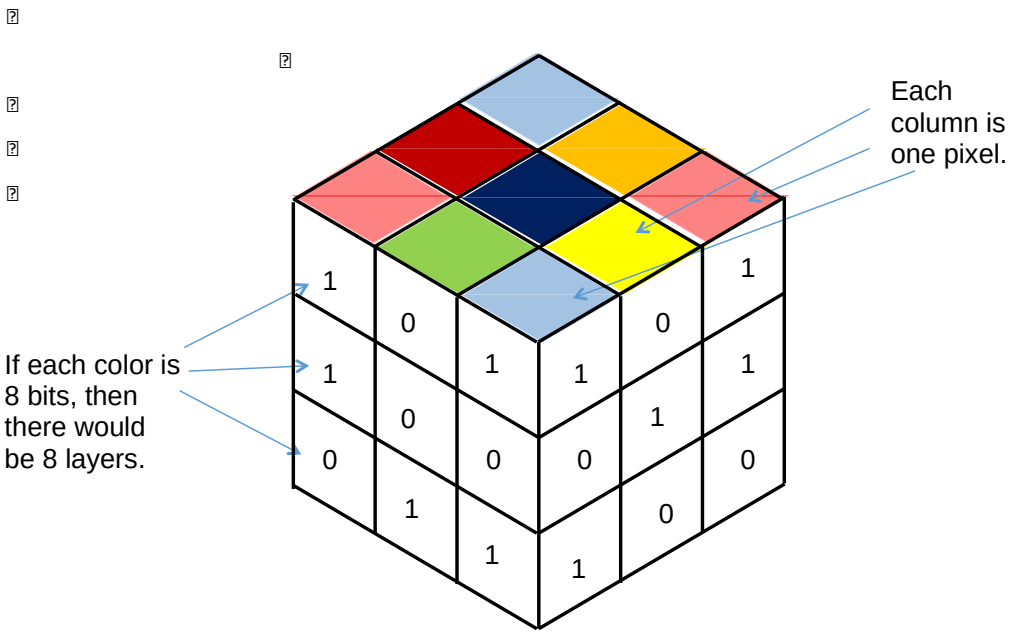

Figure 1. The colored squares are pixels in an image. Each pixel has a column of bits below it that specifies the color.

Figure 2. A website with a steganographic message.

The homepage of my Juniata website also contains a steganographic message (Figure 2).[4] In this case, the message isn't embedded in my picture, but in the Hypertext Markup Language (HTML) used to render the page. We'll explore the HTML source code to find it.

```
<p><a href="http://www.juniata.edu"><img src="junback2.jpg" border="0"
 height="55" width="151"> </a></p>

 <!--  HI THERE BOOKEND ATTENDEES.  HOW DO YOU LIKE THIS BIT OF
STEGANOGRAPHY -->

<p><a href="misc/TheseAreAFewOfMyFavoriteLinks.html">These Are A Few of
My Favorite Links</a></p>
<p>If I have agreed to write you a letter of recommendation: <br>
  <a href="recommendations.htm">Instructions for
Recommendations</a></p>
<p>Helpful advice on summer research and graduate school in Computer
Science:<br><a href="http://www.cra.org/ccc/csgs.php">Computing
Community Consortium</a></p>
<hr>
```

Figure 3. The source code for the website in Figure 2.

Like most programming languages, HTML allows programmers to write comments. These comments aren't executed when the page is rendered; they are there to help programmers who may be upgrading or debugging webpages. They also provide a simple mechanism for sending a steganographic message. In the source code above (Figure 3), you should be able to see the embedded comment "HI THERE BOOKEND ATTENDEES. HOW DO YOU LIKE THIS BIT OF STEGANOGRAPHY."

Another concept from the book is using histograms to identify unusual or unexpected life patterns. As Marcus is travelling around town to plan the next big disruptive event, he is aware he's being tracked. He says, "I wasn't ready to deal with histograms."[5] To understand what he is trying to do, let's go back to pixels. As we mentioned previously, the column of a pixel can be converted to a decimal number, and these numbers can be used to create a histogram. In fact, almost all digital cameras and image processing software have this as an option. Consider Figure 4 below. There is a black-and-white image on the left and a color image on the right. Each image includes its histogram of the pixel values, which was generated in Photoshop. You'll note how different these two histograms are.

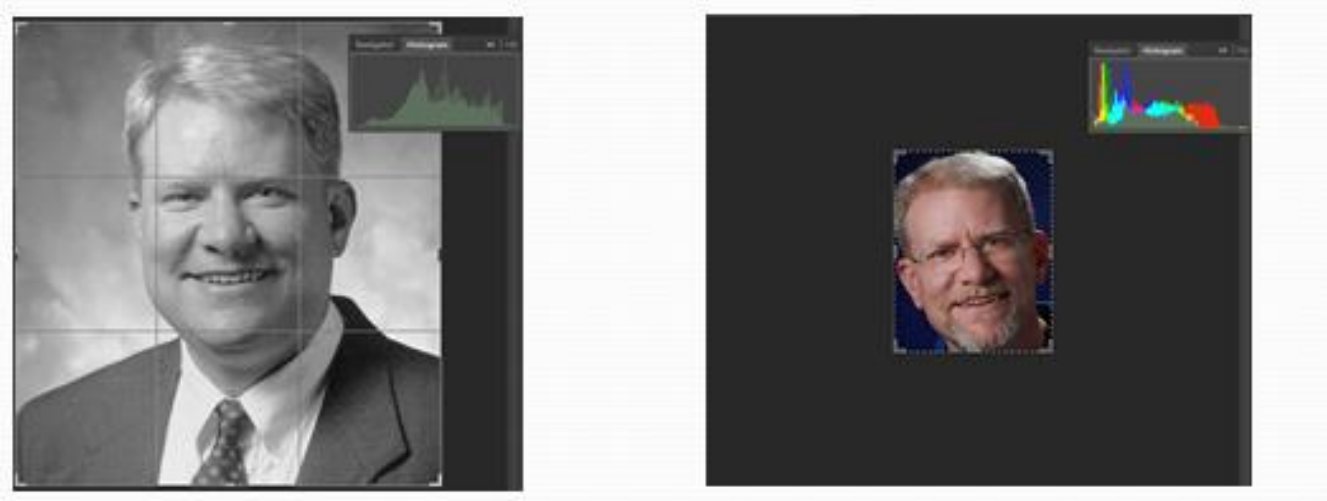

Figure 4. Two images and their associated histograms.

So, what does this have to do with Marcus confounding his surveillance? Well, Marcus is referring to the histogram that's tracking his motion. The surveillance software quantifies his movement somehow and creates histograms from this numeric data. Assuming someone follows the same patterns most days, their histograms will appear roughly the same, and this will establish a baseline of their normal or expected movement throughout the day. So an outlier histogram would alert surveillance to the possibility of suspicious movement, similar to finding the histogram for a color image in a stream of histograms from black-and-white images. The subject is not just going to work, going to the gas station, and going back home. There's something else in their pattern that is different.

In another application of histograms, NASA uses histograms of images to look for changes in landscapes on different planets, like Mars. They place a camera and compare histograms of the images it sends back to detect changes, instead of looking for minute changes in the images themselves. If the histogram of an image changes, then the analysts will follow up and study that area more closely. This method of hunting for outliers came up twice in the book—one time with people's patterns walking around, and then another time with Marcus's web traffic. Since he was using TOR and paranoid Linux, he understood that he might appear as an outlier, because normally a very small percentage of web traffic is encrypted. TOR and paranoid Linux users will have a much higher percentage of encrypted web traffic, and so a histogram of this web traffic will show up as an outlier. The author, through Marcus, also makes the point that the histograms tracking these movements of people might identify abnormal life patterns that aren't necessarily problematic. I think he used the example of discovering a person going to the drugstore for a drug that indicated they had a certain disease. Somebody else was visiting a particular house frequently, and that indicated there might be an illicit relationship. In the book, these abnormal life

patterns appeared as outliers, the Homeland Security department followed up, and people who were not breaking the law but had a secret now had that secret exposed.

Let's continue this privacy discussion with an exploration of Google. I have a "free" Gmail account. Why am I putting "free" in quotes? What is not free about my Gmail account? I don't pay any fee. Yes, it's full of advertisements, but more concerning is that Google is scouring my account to base their advertising on the content of my emails. Just the idea that my email is being searched for patterns is disconcerting. We don't usually think about it since there is no fee. We're paying by sharing our information, which results in the targeted advertising. Generally, it's pretty innocuous, but there are some really terrible failures on Google's part with this technology, which Microsoft is very willing to share.

Here are two examples that I found on Microsoft's website, which was part of their efforts to publicize their own "free" and "more private" email option. In the first Gmail message, there is a discussion of a chemo treatment. The word "chemo" is scanned, resulting in targeted advertising for makeup. That's insensitive at the very least. Next, there is an email to a friend to let them know that their beloved pet was put down. Google incorrectly targeted ads for cat food, cat toys, and other cat products.

Marissa Meyer is now the CEO at Yahoo, but she was one of the earliest employees at Google. Several years ago, when she was still an executive at Google, I attended her keynote address at the Special Interest Group of Computer Science Educators (SIGSCE). The mantra she kept saying was, "We don't want to do evil." She was basically saying, "We have all this data on people, but we want to use it wisely."

It's also fun to consider Google's searches. If you know the algorithm, and if you know how they build their searches, you can find some interesting results. Google bases search results on the content of pages, i.e., what words are on a page. They maintain that there is no weight on the meaning or context, merely the words, and the pages are ranked based on who is pointing to the web pages with all these words. The link structure on the web is really what defines the page ranks, at least initially. A bunch of hackers realized that if they all linked to a certain website, then they could affect Google's search results. The key to this is picking words or a pair of words that aren't too common. The first "Google Bomb" involved only about thirty-five people, each of whom placed a link to the White House's homepage with the associated click text "click here for a miserable failure." And it caused quite a firestorm when people realized that the number one search result for "miserable failure" was the homepage of the president at the time, George W. Bush. The right-wing response was to retaliate with a "miserable failure" link to Jimmy Carter. You can see these two results in Figure 5. The third result is a BBC article about the Google Bomb.

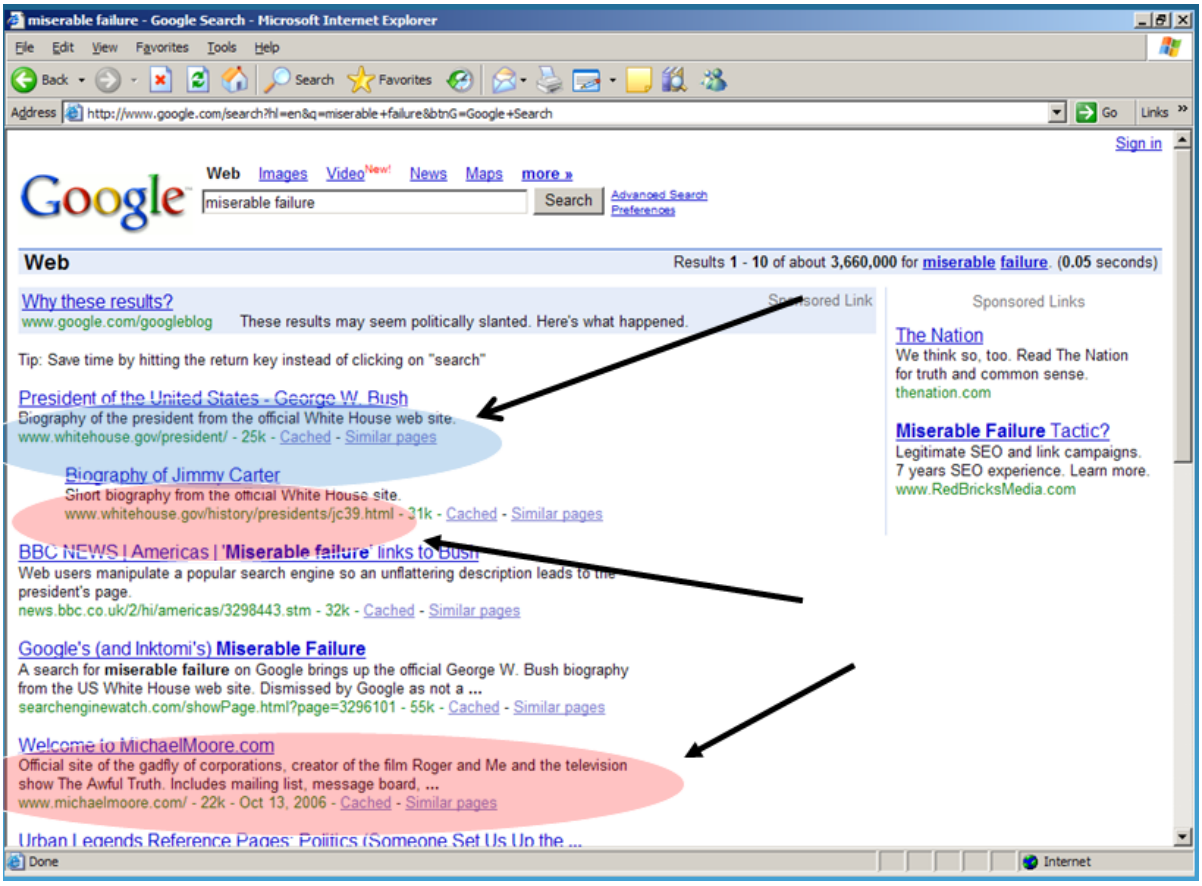

Figure 5. Search results for “miserable failure.”

One of the things that I do in my CS 315 Algorithms and Analysis class is have the students try a Google Bomb, in a good way, promoting Juniata College. Our first idea was to use the school’s slogan, “Think, Evolve, Act.” But the challenge was that those words weren’t unique enough. All those words were part of other slogans, such as “Think Globally, Act Locally.” So our Google Bomb ended up pointing to various conservation websites, and not any at Juniata. Our second choice was the phrase “Zestful Algorithms” (Figure 6). I even compelled some of my colleagues to create links to my webpage as well, with “Click Here for Zestful Algorithms.” For a while, we had some local glory with our Google Bomb. Unfortunately, Google has removed the ability to manipulate search results this way, since so many people were doing it. We did have a little fun at Google’s expense. I didn’t know who Marcus Yallow was at the time, but I guess I would have felt a little like Marcus Yallow when I did this.

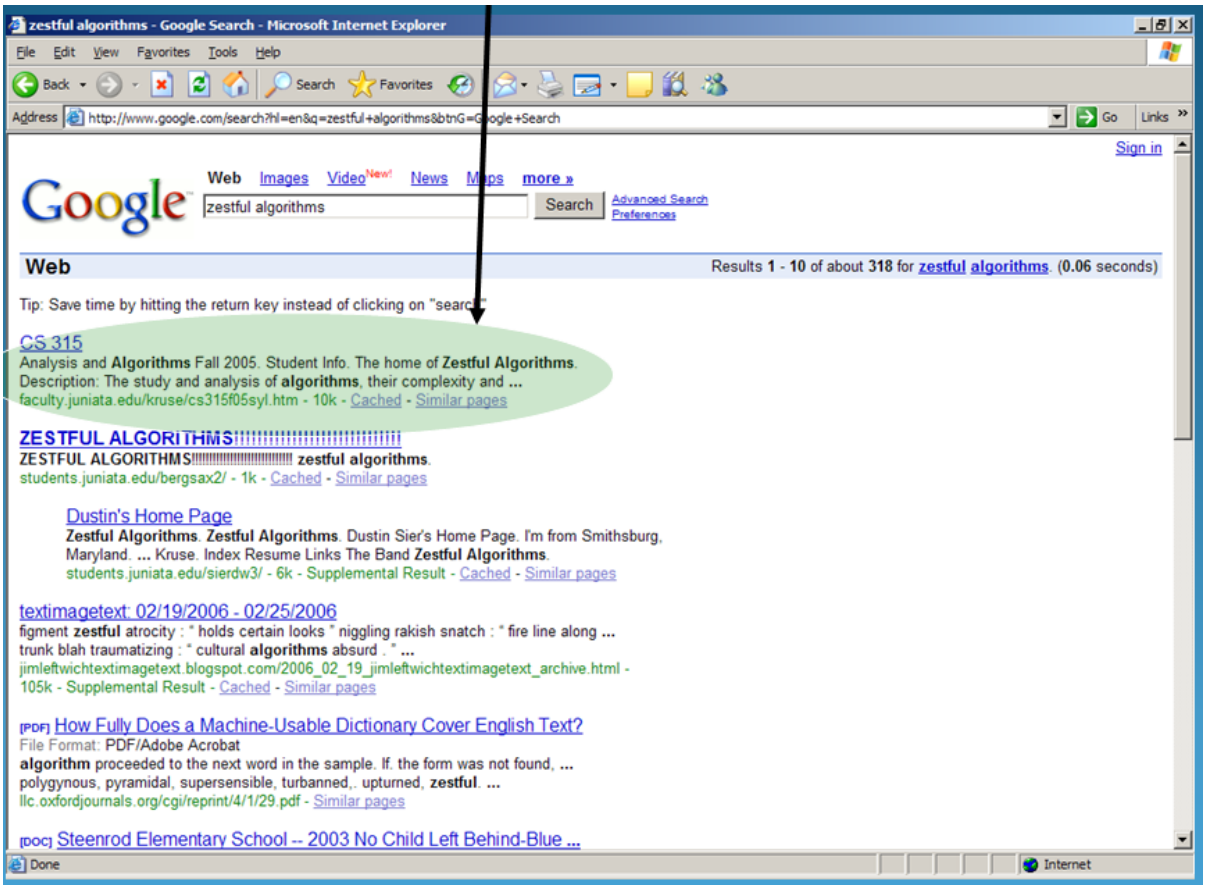

Figure 6. A Juniata Google Bomb: search results for "zestful algorithms."

One of the discussions that Marcus Yallow has in the book (on page 47) is about the false positive paradox. Do we react when we hear a car alarm? No. Why not? Could it be that we hear them all the time and usually there's nobody actively stealing a car? There are approximately 250,000,000 cars and trucks registered in the U.S.[6] Approximately 700,000 vehicles are reported stolen each year, which is about 0.3 percent of the registered cars.[7] Let's use this car alarm example to explore what Marcus is referring to. For the sake of argument, we will pretend that we have a parking garage with 1000 cars. First, according to our national statistics, of the 1000 cars in this big parking garage, 997 of the cars are okay and 3 are stolen in the course of a year. So the first question we ask about a car is whether or not it is actually stolen. The second question is whether or not the car alarm sounds. If the car alarm sounds, that's a positive test, and if it doesn't sound, that's a negative test. We can represent the results in a matrix, with the answers to first question in the columns and the answers to the second in the rows (Figure 7).

	Car Stolen	***Car NOT Stolen***	ROW TOTAL
Car Alarm Sounds (Test is Positive)			
Car Alarm Does NOT Sound (Test is Negative)			
COLUMN TOTAL	3	997	***1000***

Figure 7. Matrix to be used for the car alarm results.

If a car is being stolen and its alarm sounds, that is a true positive test result. If a car is sitting there quietly and is not being stolen, that is a true negative. We will assume that the sensitivity, or true positive rate, which is the proportion of cars being stolen that the car alarm detects accurately, is 99%. We'll also assume that the specificity, or true negative rate, which is the proportion of cars NOT being stolen whose alarms do not sound, is also 99%. If the specificity is 99% and there are 997 cars that are not stolen, then there are approximately (due to rounding) 987 true negatives—cars that are sitting quietly and not being stolen. If the sensitivity is 99% and 3 cars are stolen, then the number of true positives is 99% of 3. That's 2.97, but we're just going to just round it to 3. These are the cars that are being stolen and whose alarm is going off. Finally, let's consider the full table (Figure 8), including the false positives and false negatives. Note that there are 997 – 987 = 10 false positives, where a car is not being stolen but its alarm is going off. And if we have 3 of 3 stolen cars as true positives, then no cars are false negatives. There are no cases where the alarm doesn't go off and the car is still stolen. The key here, which follows along with what Marcus discusses in the book, is that of the total number of positive car alarms is 13, and 77% of these, the 10 false positives, are incorrect.

	Car Stolen	***Car NOT Stolen***	ROW TOTAL
Car Alarm Sounds (Test is Positive)	3 "True Positive"	10 "False Positive"	13
Car Alarm Does NOT Sound (Test is Negative)	0 "False Negative"	987 "True Negative"	987
COLUMN TOTAL	3	997	***1000***

Figure 8. Car alarm results.

That's why we don't pursue whenever a car alarm is going off: most alarms don't indicate that a car is actually being stolen. This false positive paradox is even worse for things that are more rare than the car alarm example. We run into problems when there is a huge proportion of false positives. Marcus makes the point that if you're going to track people and try to find people that are doing bad things, then you'd better have a pretty accurate screening, with some kind of backup screening, before making any conclusions.

NOTES

1. Cory Doctorow, *Little Brother* (New York: Tor Teen, 2010). The book is also freely available on Doctorow's website: http://craphound.com/littlebrother/download/.
2. See "RFID: Don't Microwave," https://www.youtube.com/watch?v=bBk8yZcyt5g, for example.
3. Doctorow, *Little Brother*, p. 46.
4. Gerald Kruse's Juniata College homepage, http://jcsites.juniata.edu/faculty/kruse/ (accessed October 28, 2015).
5. Doctorow, *Little Brother*, p. 37.
6. Jerry Hirsch, "253 million cars and trucks on U.S. roads; average age is 11.4 years," *Los Angeles Times*, June 4, 2014, http://www.latimes.com/business/autos/la-fi-hy-ihs-automotive-average-age-car-20140609-story.html.
7. Federal Bureau of Investigation, "Motor Vehicle Theft," *Crime in the United States, 2012,* https://ucr.fbi.gov/crime-in-the-u.s/2012/crime-in-the-u.s.-2012/property-crime/motor-vehicle-theft.