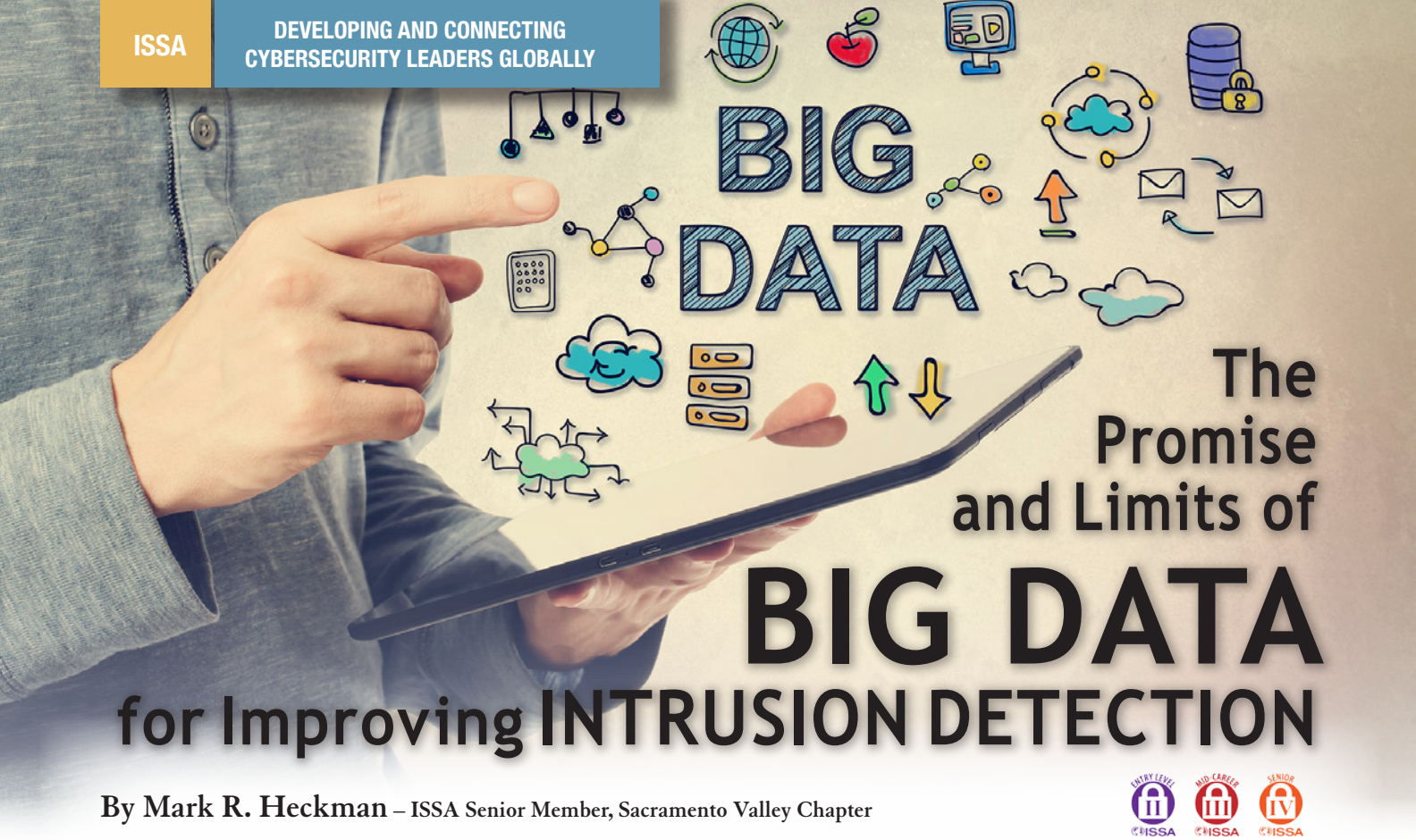

The Promise and Limits of BIG DATA for Improving INTRUSION DETECTION

By Mark R. Heckman – ISSA Senior Member, Sacramento Valley Chapter

This article discusses the essential problems in intrusion detection and how big data techniques have been successfully applied to overcome some of the problems, but also explains some fundamental limits that could prevent big data from achieving all of the promises.

Abstract

The usefulness of intrusion detection systems frequently suffers from a high rate of false-negative alerts—failure to generate alerts when an attack occurs, and false-positive alerts—alerts when no attack is taking place. False-negative alerts mean analysts will not detect an attack, while false-positive alerts distract analysts and may obscure alerts that indicate real attacks. Advances in big data analytical techniques raise the hope that these techniques could be used to vastly improve intrusion detection. This article discusses the essential problems in intrusion detection and how big data techniques have been successfully applied to overcome some of the problems, but also explains some fundamental limits that could prevent big data from achieving all of the promises.

Big Data is a term used to describe techniques for extracting useful information from large volumes of data. Big data techniques such as data mining and machine learning have already proven their usefulness in complex predictive analytics, pattern recognition, and classification problems in many different fields. Advances in big data techniques raise the hope that they could be applied to solve one of the most intractable problems in cybersecurity: high false-negative and false-positive alert rates in intrusion detection systems.

Intrusion detection is the monitoring of system events to detect activity that violates the system security policy. An intrusion detection system (IDS) must analyze large quantities of data, usually in real time, and attempt to pick out and correlate events that indicate an attack is occurring. But that is not an easy task in large, complex, and chaotic systems, which means that most IDSs suffer from high rates of false-negative alerts, false-positive alerts, or both:

- False negatives occur when an IDS fails to alert on a real attack. An IDS that has too many false negatives is simply not very good at detecting attacks.
- False positives occur when an IDS alerts on a suspected attack when no attack has happened. An IDS that has too many false positives will waste the time of security analysts who may miss real attacks—true positives—in the false-positive noise.

The promise of big data for intrusion detection is obvious: Advanced data analytic techniques will, it is hoped, sharply reduce the rate of false positives and false negatives, making IDSs much more accurate at detecting attacks and ignoring benign behavior. But the use of machine learning for intrusion detection is not new. Automatic generation of statisti-

cal models of normal and abnormal activity, for example, goes back 30 years or more (e.g., see D.E. Denning, An Intrusion-Detection Model, 1987 [11]). What is it about today's technology that promises to do so much better? The following sections will explain the challenges of intrusion detection, discuss how big data is already being used to improve results, and clarify some limits on what big data can do for improving intrusion detection.

Intrusion detection techniques

An IDS works much like a burglar alarm for computer systems. The IDS monitors system events—network, host, or application—and generates alerts when it detects behavior that violates the system security policy. The chief goals of an IDS are to detect as wide a variety of attacks as possible, including both previously known and unknown attacks, in a timely fashion while maintaining high accuracy (i.e., minimizing false negatives and false positives).

It was proven decades ago that the problem of detecting viruses is undecidable – that is, there is no possible generic algorithm that will perfectly detect all viruses (i.e., no false positives or false negatives) [7]. A similar argument can be made that there is no way to perfectly identify intrusions in general (see sidebar "Why a Perfect IDS Is Impossible"), but just because a detection system isn't perfect, doesn't mean it can't be good enough to be useful.

There are three general approaches to intrusion detection: 1) misuse detection—defining specific bad behavior so that, implicitly, everything else is considered to be good, 2) specification-based detection—defining specific good behavior so that, implicitly, everything else is bad, and 3) anomaly detection—automatically generating a specification for "normal" behavior with the expectation that normal behavior is more closely associated with "good" and abnormal behavior is more closely correlated with "bad" [3]. An IDS might use one or a combination of these methods.

The following sections briefly explain misuse and specification-based detection. Because anomaly detection is the intrusion detection approach that may be most amenable to improvement using big data techniques (for reasons explained below), the description of anomaly detection is more in-depth.

Misuse detection

Misuse detection systems work by comparing observed data against a database of stored "signatures" or "rules"—data previously shown to correlate with attacks. When observed data matches a signature, the system raises an alert. A classic example of a misuse detection system is antivirus software. The virus definition database constitutes the signatures. Signatures may consist of strings of bits unique to particular malicious software (malware) object code, a unique command or set of commands executed by malware, or a suspicious pattern of commands.

For example, the Code Red worm of 2001 used a buffer overflow attack containing the following unique data string that could be used as a signature [13]:

/default.ida?NNNNNNNNNNNNNNNNNNNNNNN-
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN-
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN-
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN-
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN-
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN-
NN%u9090%u6858%ucbcd3%7801%u9090%u6805%ucb-
d3%u7801

The Solaris Sadmind/IIS worm (also 2001) executed the following unique command that could be used as a signature [5]:

```
GET/ scripts/../../winnt/system32/cmd.exe /c+ copy+\wint\
system32\CMD.exe+root.exe
```

The creation of new signatures has generally been a manual (i.e., human) process that calls for a high level of expertise. A signature developer examines captured malware and tries to develop a signature that is neither too general nor too specific. If a signature is too general, it will lead to false positives. If a signature is too specific, it may miss some instances of the same attack, leading to false negatives. But typically signatures are very precisely defined, so misuse detection systems usually have a low false-positive rate. A misuse detection system, however, can only detect attacks for which a signature has been created. If there is no matching signature, the system will not detect an attack. Even slight variations in malware can mean that old signatures will no longer match, and misuse detection systems are completely unable to detect new attacks because no signatures can yet exist, leading to high false-negative rates. For this reason, signature databases quickly become obsolete and a misuse detection system must frequently download new signature databases.

SIDEBAR

Why a perfect IDS is impossible

Fred Cohen in 1984 proved that the problem of detecting viruses is undecidable—that is, there is no possible generic algorithm to accurately detect all viruses [7]. This argument was later extended to all malware in general. Thus, a perfect IDS (no false positives or false negatives) is impossible. The argument is based on the most famous of all computer science proofs, called the Halting Problem. The halting problem, created by Alan Turing in the 1930s, asks if there is a generic algorithm that takes as input a program and input to that program and that can determine if the program will ever terminate (i.e., halt) on that input. While it is certainly possible for some programs and inputs to detect if the program will halt or not, Turing proved that it was impossible in the general case [21]. Ever since in computer science it is sufficient to show that a problem is undecidable if solving it would require a solution to the halting problem.

That is what Cohen did. He showed that the problem of detecting all viruses is solvable only if there is a solution to the halting problem. But the halting problem is known to be undecidable so the problem of detecting all viruses is also undecidable. Cohen formulated the problem in this way: A perfect virus detector should emit an alarm if and only if the potential virus passed as input to the detector can ever infect and damage the host (i.e., the virus detector is perfect if it has no false positives and no false negatives). Consider the following program:

```
f();
infect();
```

If the function f() can return, then this is a virus and the detector should alert. If, however, f() is in an infinite loop and will never return, then no infection is possible, so the virus detector should not alert. But coming up with the correct answer would require the virus detector to know if f() will halt or not, so the detection problem is equivalent to the halting problem, which is known to be undecidable.

Note that it is easy to create a system with a 0 percent false-negative rate: simply alert on every event. But, of course, that will yield a 100 percent false-positive rate. Conversely, a system can have a 0 percent false-positive rate simply by never alerting on any events, but that obviously will yield a 100 percent false negative rate. There is no way, in general, to simultaneously have both a 0 percent false-positive and 0 percent false-negative rate. Because perfection is impossible, an IDS must make a tradeoff between false-positive and false-negative rates.

Specification-based detection

A specification-based system compares observed data against a database of legitimate behaviors. Whenever the observed behavior does not match a record in the database of legitimate behaviors, the system raises an alert. This approach is excellent at detecting previously unknown attacks because, by definition, attacks are not legitimate activity and will not match the specification, so specification-based IDSs can have low false-negative rates.

Given complete and detailed specifications for the complete range of legitimate activity in a system, specification-based IDSs can also have low false-positive rates. But creating such specifications in all but simple systems is difficult and error-prone, so specifications in complex systems are likely to be inaccurate, leading to very high false-positive rates. For this reason, specification-based IDSs are chiefly used in systems with well-defined, regular processing activity such as medical [19] and SCADA devices [6].

An example of behavior specifications in a medical device is a set of 11 rules for an IDS that monitors a medical cyber physical system (MCPS) [19]. The MCPS consists of vital sign monitor, patient controlled analgesia, and cardiac devices. The rules require that actual sensor readings (for pulse, blood pressure, etc.) match the readings shown on the monitor, that the patient is in a debilitated state (e.g., his heart is fibrillating) before treatment (in this case, defibrillation) is given, and that the automatically provided treatment is in the safe range (e.g., the analgesic request rate is below a safe threshold). The IDS alerts any time the observed state of the system violates any of the rules.

Specifications are created from theoretical reasoning about legitimate behavior, rather than by empirical observation of data as the system is being used. But big data techniques are aimed at extracting information from empirical observation of large data sets. For this reason, specification-based intrusion detection is the least likely intrusion detection method to benefit from big data techniques.

Anomaly detection

Anomaly detection is the identification of activity in observed data that does not conform to expected behavior. Like

specification-based intrusion detection, observed behavior is compared against a database of expected behaviors, and outliers cause the system to alert. Unlike specification-based intrusion detection, however, where the specifications are defined ahead of time and remain fixed, the expected behavior in anomaly detection is defined through analysis of empirical data and can adapt as "normal" behavior changes over time.

The key assumption of anomaly detection is that attacks exhibit characteristics that are different than those of normal behavior [11]. Anomaly detection works by analyzing a set of system characteristics and comparing its values against a recorded baseline, or profile, that represents what is normal for the system. Outliers are labeled "anomalous." Anomalous events are assigned a score based on the degree of anomalousness. When the degree of anomalousness exceeds some threshold parameter, the system raises an alert. Figure 1 abstractly depicts a profile (the circle) around behaviors considered "normal." Behaviors a1 and a2 are anomalous, where a2, being farther away from "normal," has a higher degree of anomalousness.

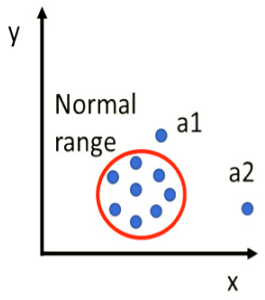

Figure 1 – "Normal" profile with anomalous outliers

The process of calculating a profile is called *training*. Training typically uses values observed while the system is running. Because normal behavior can vary quite widely, a profile must be developed over the course of many observations. For example, consider if the ISSA wanted to use anomaly detection to detect buffer overflow attacks on the quick search field of the ISSA home page (www.issa.org). The IDS might observe the length of search terms over many searches, calculate the average input length, and store that average as a profile. An input value that greatly exceeds that average could represent a potential buffer overflow and would trigger an alert.

Not all data is equally useful in differentiating normal from anomalous behavior. Choosing the attributes to train on—a process called *feature selection*—is as important or more than the algorithm used to create profiles. For example, while the length of search terms in the ISSA quick search field is correlated with buffer overflow attacks, the type of characters in the input string—numbers or letters, say—may not be. It would likely be impossible for the system to create meaningful profiles if it trained on the type of characters.

Anomaly detection models

Denning identified three metrics and five statistical models for anomaly detection [11]. Most, if not all, anomaly detection systems used today still depend on these same basic metrics and statistical models. The metrics are an event counter that represents the number of occurrences of an event during a period of time (such as the number of failed logins in one minute), an interval timer that represents the length of time between two related events (such as the length of time between two successive logins to the same account), and the quantity of resources consumed by an action during a period

The Council Gets a Clue

By Jeff Hall – ISSA member, Minnesota Chapter

FEBRUARY 2017, the PCI Security Standards Council issued a new information supplement titled "Multi-Factor Authentication"[1] after the brew-ha-ha that occurred last fall at the community meeting in Las Vegas. For once, the Council has issued a great reference regarding the issues of multi-factor authentication (MFA). I still have a couple of minor bones to pick about this document, but more on that later.

If you understand the concepts of MFA, you can skip through the document to the end where the Council presents four scenarios on good and bad MFA. These are well documented and explain the thought process behind why the scenario works or does not work for MFA. The key take away of all of this is the independence of the MFA solution from the logon process. The Council is getting in front of the curve here and stopping people from creating insecure situations where they believe they are using MFA that minimizes or stops breaches through administrators or users with access to bulk card data.

Now for a few things that I do not necessarily agree with in this document.

The first involves the Council's continued belief that hardware security modules (HSM) are actually only hardware. On page four, the following statement is made.

> "Hardware cryptographic modules are preferred over software due to their immutability, smaller attack surfaces, and more reliable behavior; as such, they can provide a higher degree of assurance that they can be relied upon to perform their trusted function or functions."

The Council has made similar statements over the years in the mistaken assumption that HSMs are only hardware. HSMs are hardware that use software to manage keys. There are standards that are followed (e.g., FIPS 140) to ensure that the HSM remains secure, but these devices are predominately software driven. That is not to say that just any device can serve as an HSM, but a lot of us in the security community are concerned that the Council continues to perpetuate a myth that HSMs are only hardware, which is patently false.

My other issue comes on page six as part of the discussion regarding the use of SMS for MFA.

> "PCI DSS relies on industry standards—such as NIST, ISO, and ANSI—that cover all industries, not just the payments industry.

1 "Multi-Factor Authentication," PCI Security Standards Council, February 2017 – https://www.pcisecuritystandards.org/pdfs/Multi-Factor-Authentication-Guidance-v1.pdf.

of time (such as CPU time used by a program during a single execution).

Denning's five statistical models are Operational, Mean and Standard Deviation, Multivariat, Markov Process, and Time Series.

The Operational Model is based on determining fixed upper and lower bounds on observed characteristics. A limit of three failed logins before locking a person out of a system is an example of the operational model, where the lower and upper bounds on normal are (0, 2).

The Mean and Standard Deviation Model uses the first two statistical moments (i.e., mean and standard deviation) of the distribution of observed behavior. A new observation is determined to be abnormal if it falls outside a specified confidence interval, which is some number of standard deviations on either side of the mean. The confidence interval is a bound on the probability of the occurrence of that particular behavior. For example, as shown in figure 2, observed behaviors that fall outside a confidence interval of three standard deviations only happen 0.2 percent of the time. The ISSA quick search input length example mentioned above is an example of applying the mean and standard deviation model. If the profile were set at three standard deviations (3σ) above the mean, the IDS should alert on only 0.1 percent of the accesses to that form field (we don't care about shorter input lengths below the mean in this case, because that is not correlated with buffer overflow attacks).

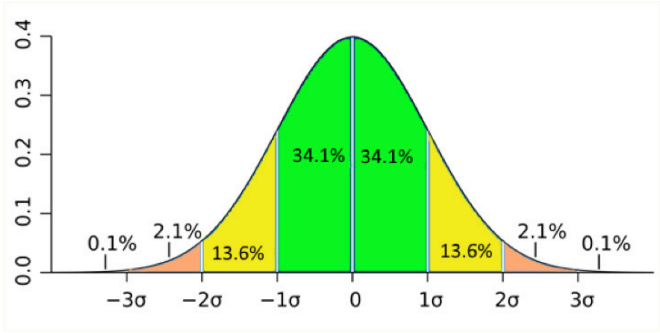

Figure 2 – The Mean and Standard Deviation Model

The Multivariate Model is similar to the mean and standard deviation model, but is based on the correlation of multiple metrics rather than just a single metric. For example, buffer overflow attacks typically use not only a large input, but the byte values in the input are often highly repetitive (e.g., as in the Code Red Worm buffer overflow signature described above, which contains a long string of ASCII "N" characters 13). Instead of just using the length of an input to a web field, our hypothetical ISSA quick search anomaly detector might look at both length and the character distribution of the input in order to reduce false positives [17].

The Markov Process Model calculates profiles based on the probabilities of sequences of events, rather than on the statistics of single events. An example of an anomaly detection system that uses the Markov process model is the time-based inductive-learning approach described by Teng, et. al [20]. The system observes the ordering of events and creates prob-

> While NIST currently permits the use of SMS, they have advised that out-of-band authentication using SMS or voice has been deprecated and may be removed from future releases of their publication."

While everything in this statement is accurate, it gives the uninitiated the impression that SMS or voice is no longer a valid MFA solution. I know this to be true because I have fielded a number of questions from clients and prospects on this subject, particularly about SMS. The key is that this is not SSL and early TLS where NIST called them out as insecure and to no longer be used. This is a "heads up" from NIST to everyone that there is an issue that makes SMS and voice not secure enough for MFA.

But while there is a risk, a lot of us in the security community question the viability of that risk when matched against merchant risk versus a bank or a government agency. While I would not want any bank or government agency to use SMS or voice for MFA, a small business may not have a choice given their solution. The reason is that the risk of an attack on SMS or voice is such that only a high value target such as a bank or government agency would be worth such an effort. In my very humble opinion, while a total ban is the easy solution, this is an instance where the Council should take a more nuanced approach toward the use of SMS and voice for MFA. The bottom line to me is that small merchants using any MFA solution, even if flawed, is better than using no MFA solution.

I would recommend the following approach to manage this risk:

- Level 4 merchants can be allowed to use SMS or voice for MFA.
- Level 1, 2, and 3 merchants would be allowed to transition away from SMS and voice to a more secure MFA solution within one year of NIST stating that they are no longer acceptable.
- All service providers would not be allowed to use SMS or voice for MFA once NIST states that both are no longer acceptable. This means service providers should start transitioning now if they use either.

Those are my thoughts on the subject. I look forward to the comments I am sure to receive.

About the Author

Jeff Hall, CISSP, CISM, is a Principal Security Consultant in Optiv Security's Governance, Risk & Compliance practice and focuses on payment card industry and related security projects. Jeff has over 30 years of technology and compliance experience and is certified in the governance of enterprise information technology and a PCI QSA. Check out PCI Guru blog or contact him at Jeff.Hall@optiv.com.

ability-based rules. Having observed, for example, this series of events: A-B-C-S-T-S-T-A-B-C-A-B-C, the system will generate the following rules:

R1: A - B → (C, 100%) C follows A-B with 100% probability

R2: C → (S, 50%; A, 50%) C is followed by S 50% of the time and A 50% of the time

R3: S → (T, 100%)

R4: T → (A, 50%; S, 50%)

R2 and R4 have very poor predictive power and will not become part of the profile, but R1 and R3 have good predictive power. If the system were to ever see the sequence A-B-D, for example, that would trigger an alert because it violates Rule R1.

Markov models are frequently used in systems where the statistical difference between normal behavior and attacks is not large, which could lead to high false-positive or high false-negative rates. For example, the HMMPayl system uses a Markov model based on sequences of characters to detect common attacks against web applications (such as XSS and SQL-Injection) where statistical measurements, such as input length or character distribution are insufficiently precise [2].

The Time Series Model is like the Markov process model but takes into account both the order of events and the inter-arrival times. An example of an anomaly detection system that uses the time series model is a system that detects an irregular heartbeat. In figure 3, the electrocardiogram depicts an irregular heartbeat. None of the readings in the abnormal flat region are anomalous by themselves, however, because they also occur during the course of normal heart rhythm. Only the order and relative time between the readings make that region anomalous.

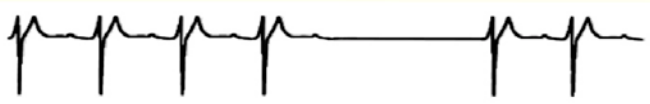

Figure 3 – Time series model example detecting an irregular heartbeat

Challenges to anomaly detection

For all of the potential of anomaly detection, there are some real challenges that must be overcome to build a useful anomaly detection system. What is anomalous, for example, is not necessarily bad. If while training the system has not seen the complete range of allowed behaviors, or if some allowed behaviors were rare in the training data, the previously unseen—but good—behaviors will likely cause many false positives. Similarly, what is normal is not necessarily good. What if, for example, an attacker is already in place? A backdoor port might "normally" be open and the anomaly detecti on system would never alert on it, resulting in many false negatives.

Defining normal in complex, chaotic systems is not easy. Data is noisy and boundaries between normal and anomalous behavior may not be precise. In figure 1, for example, why is anomalous event a1, which is very close to the normal range, outside of it? If a different set of training data was given to the system, a1 might very well have been considered part of the normal profile. A slight change in the trained profile can have a large effect on the number of false positives and false negatives.

Normal activity in many systems is highly variable. It may vary by time of day, day of week, or season. An anomaly detection system unable to identify cyclical, often complex patterns will have a large number of false positives or false negatives at different times for the same training profiles. But as users, installed programs, hosts, and other factors change over time, so too will the profile of normal activity. Anomaly detection systems typically handle change to normal system behavior over time by continually recalculating profiles, giving more weight to recent activity (see the sidebar "Weighting") than to older activity. If, however, behavior is cyclical then the system might always be chasing normal, constantly playing catch up, with the profiles always inaccurate. In such a case, the accuracy of the system might have been better if the system had simply kept the original profiles constant (much like a broken clock is correct twice a day).

Moreover, the accuracy of anomaly detection systems depends heavily on the choice of monitored events. While some types of events may be strongly correlated with good behaviors (or correlated with bad behaviors), others may be completely independent. If a system is trained on independent data, the profiles will be irrelevant and potentially result in many false positives and false negatives. But determining in advance the usefulness of particular features in the data is not a simple problem. One solution is to increase the variety and number of event records that are included in the training. But simply collecting and examining every possible type of log record that a system can produce is difficult and time consuming. It can be difficult to collect, store, search, analyze, and correlate so many events and to determine which, if any, provide useful information. The problem of determining correlations between different varieties of data, moreover, increases exponentially with the number of varieties.

Within the vast quantity of data that must be continually scanned by an IDS, the number of events that correspond to real attacks is typically very small. This means that even a low false-positive rate can produce a very large number of bogus alerts. While it may seem more useful for an IDS to identify as many attacks as possible, a large number of false positives can have a greater impact on the perceived usefulness of an IDS to security analysts (see the sidebar "The Problem with False Positives"). Anomaly detection systems that depend on statistical models are particularly susceptible to false positives in this way.

SIDEBAR

Weighting

An anomaly detection system must be able to adjust the profile to compensate for changes over time, a process that is usually accomplished by weighting recently observed behavior more heavily than older behavior. For example, consider an anomaly detection system that counts the number of occurrences of a certain event *Cn* for each day *n*. The system compares day *n*'s count *Cn* against the expected, "normal" value for that day *En* to see if that day's count is anomalous (e.g., using the mean and standard deviation model described in the article). We can calculate the expected count for day *n*, *En*, as the weighted summation of previous days:

$$En = 2^{-1} * C_{n-1} + 2^{-2} * C_{n-2} + 2^{-3} * C_{n-3} + \ldots$$

In words, the expected value for today (day *n*) is calculated as one-half of the observed value for yesterday (day *n-1*) plus one quarter of the observed value for two days ago (day *n-2*) plus one-eighth of the value for three days ago (day *n-3*) plus … and so on. This gives yesterday's observed count the same weight as all of the counts observed on previous days together. The summation works well for weighting because 1/2 + 1/4 + 1/8 + 1/16 + ⋯ = 1, as depicted in figure 4.

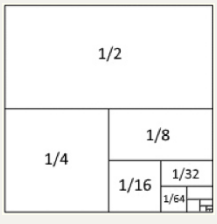

Figure 4 – The summation of 1/2 + 1/4 + 1/8 + 1/16 …

Because the expected value for a day is calculated using the observed counts from all previous days, the expected value is easy for a computer to update each day using the simpler formula

$$En = 0.5 * (C_{n-1} + E_{n-1})$$

Coming up with a meaningful and accurate weighting scheme in complex systems, however, is generally not this simple.

Big data techniques and their application to intrusion detection

Data is records, statistics, and other objective facts or empirical observations. *Information*, on the other hand, is useful interpretations of data. Big data techniques cover the collection, storage, and analysis of large data sets in order to generate useful information.

Big data is typically defined in terms of the "3 Vs"[18]:

1. **Volume**—the sheer scale of data there is to collect
2. **Velocity**—the rate at which new data can be analyzed
3. **Variety**—the tremendous diversity of data sources

The ability to store, search, and analyze large amounts of log data, particularly log data from heterogeneous sources, is essential for reducing false positives and negatives, yet organizations using traditional data management techniques, such as relational databases, may still struggle to handle the flood of log data.

A 2013 report from the Cloud Security Alliance, for example, says that "it is estimated that an enterprise as large as HP currently (in 2013) generates 1 trillion events per day, or roughly 12 million events per second" [4]. Organizations using traditional data management techniques, such as relational databases, cannot handle the growing flood of data. New big data techniques, however, such as Hadoop and Hive permit the management, search, and analysis of much greater amounts of log information than previously possible [9]. Hadoop architecture is a software framework for storing and processing large quantities of unstructured, distributed data [15]. Hive is a structured query infrastructure for making queries on top of Hadoop [1]. Big data techniques such as these are starting to be used in intrusion detection [22]. In one case study, Zions Bank reported that their security systems generated three terabytes of data per week. Storing and searching the data using Hadoop and Hive cut the search time for a month of log data from up to an hour down to a minute [9].

Data mining is a catchall term for the process of collecting, searching, and analyzing large amounts of data to discover useful patterns or relationships. It is difficult for humans to find meaningful associations between disparate data items in huge data sets from many different sources, so big data techniques include machine learning—a type of artificial intelligence where computers can learn without being programmed. Machine learning can be used in intrusion detection to automatically find meaningful relationships between

Continued on page 43

The Promise and Limits of Big Data for Improving Intrusion Detection

Continued from page 29

data items. A machine learning system observes collected data, applies analytical algorithms to determine patterns, and creates analytical models. Hui Wang, senior director of global risk and data sciences at PayPal, which uses machine learning to identify fraud, praises advanced machine learning, saying it can "see" things that "even human beings might not be able to see" [8].

SIDEBAR

The Problem with False Positives

At first glance, it may seem obvious that the ability of an IDS to alert on as many intrusions as possible, minimizing the number of attacks that are missed (false negatives), is more important than minimizing the number of false alarms (false positives). But false positives tend to have a greater impact on the perceived usefulness of an IDS than do false negatives because false positives are false alarms that waste analysts' time. An IDS that has too many false positives will eventually be ignored or simply unplugged.

Let us define the "utility" of an IDS as the probability that an attack really happened when the IDS alerts. That is, Utility = TP / (TP + FP), where TP is the number of true positives (alerts due to a real attack) and FP is the number of false positives (alerts when there is no attack). Similarly, we will also label the number of false negatives FN (failures to alert when there is a real attack) and the number of true negatives TN (lack of alerts, correctly, when there is no attack). To increase the utility of an IDS, you must either increase TP or decrease FP. Let us assume that we have a very accurate IDS with a 99 percent TP rate and only a one percent FP rate. This means that 99 percent of the time that an attack happens, the system correctly alerts, and only one percent of the time that a non-attack event happens, the system incorrectly alerts. (The corresponding values for FN and TN rates: FN rate = 1 – TP rate = 1%, TN rate = 1 – FP rate = 99%.)

Let us further assume that there is a fairly constant attack rate and that one out of every 10,000 events is malicious (i.e., the attack density is 1/10000).

We will calculate the utility using Bayes' rule, where Pr(X|Y) is the probability of X given Y.

Bayes' rule: $Pr(X|Y) = (Pr(Y|X) * Pr(X))/Pr(Y)$

If we designate attacks by the letter A and alerts by the letter L, the utility—the probability that an attack really happened when the IDS alerts—is denoted by Pr(A|L).

The probability of an event being an attack—Pr(A)—is 1/10000 = 0.0001. The probability, therefore, that an event is not part of an attack—Pr(¬A)—is (1 - 0.0001) = 0.9999. We are assuming that the probability of alerting on a real attack (TP) is 99 percent, which means Pr(L|A) = 0.99, and the probability of alerting on a non-attack (FP) is one percent, so Pr(L|¬A) = 0.01. The probability of an alert (whether a TP or FP) on any given event Pr(L), is the probability of alerting on a real attack times the probability of the event being a real attack plus the probability of alerting on a non-attack times the probability of the event not being an attack:

$Pr(L) = Pr(L|A)*Pr(A) + Pr(L|\neg A)*Pr(\neg A) = (0.99)(0.0001) + (0.01)(0.9999) = 0.010098$

Now we are ready to apply Bayes' rule to calculate the utility of the IDS, Pr(A|L):

$$Pr(A|L) = \frac{Pr(L|A) * Pr(A)}{Pr(L)} = \frac{0.99 * 0.0001}{0.010098} = 0.0098$$

So a 99 percent accurate detector (i.e., both a 99 percent TP rate and 99 percent TN rate) would have less than one percent utility. Another way of stating that result is that out of 100 alerts, only one alert would reflect a real attack. This is a major problem with IDSs: The inability to suppress false positives can greatly reduce the usefulness of the system.

But which are the most significant factors in the high false-positive rate and can we easily fix those factors to improve the utility of the IDS? To find the factors, let us change one parameter at a time in our assumed IDS. In order to get the utility of the system to 50 percent, table 1 shows that we either have to increase the attack density from 1/10000 to an unrealistically high 1/100, or else drop the FP rate from 1/100 to an extremely low 1/10000. To achieve a 99 percent utility, the FP rate has to drop to the proverbial one in a million.

What these calculations show is that, because intrusions are typically rare events and non-intrusions are vastly more common, true positives are generally swamped by false positives. The rate of attacks is out of our control, so our only hope for increasing the utility of our IDS is to be able to decrease the false positive rate of our system. But, as the article explains, there are many challenges to reducing the false positive rate in IDSs.

Parameter Set	Attack Density Pr(A)	Probability of an alert Pr(L)	True Positive Pr(L|A)	False Positive Pr(L|¬A)	Utility Pr(A|L)
Original	0.0001	0.010098	0.99	0.01	0.0098
I	0.0001	0.010098999	0.99999	0.01	0.0099
II	0.01	0.0198	0.99	0.01	0.5
III	0.0001	0.00019899	0.99	0.0001	0.5
IV	0.0001	0.0000999999	0.99	0.000001	0.99

Table 1

Machine learning systems may engage in what is called "supervised" or "unsupervised" learning. In supervised learning, humans label data to guide the development of models. For example, a machine learning system does not know on its own the difference between legitimate and malicious activity. The system needs to be given training data containing both types of activity, and a human must provide feedback to the system for it to learn the difference. In unsupervised learning, the system tries on its own to identify data clusters based on objective criteria. Supervised learning is necessary for subjective labels such as "legitimate process" and "malware," while unsupervised learning can be used for objective labeling of data, such as "normal" and "anomalous." (For a longer discussion of how machine learning works, see Stephen Jou's August 2016 article in the *ISSA Journal*, "Machine Learning: A Primer for Security." [16])

As mentioned earlier, however, the use of machine learning in intrusion detection is not new. Automatically generated anomaly detection system profiles are an example of machine learning. But today's machine learning systems, built on infrastructure such as Hadoop and Hive, are capable of processing vastly greater quantities and varieties of data at high speed. This, then, is the major promise of big data for intrusion detection: scale and speed.

Use and limits of big data for misuse detection

Development of misuse detection signatures has typically been a "manual" process, where expert human knowledge is converted into an analytical model. Expert system rule models of this type are static and don't automatically adapt over time. Malware writers know this and construct defenses within malware against intrusion detection to prevent the malware from matching a signature. Polymorphic malware is an example defense against detection by an IDS. Polymorphic malware mutates, so the code won't match an existing signature but preserves the original algorithm. There are an almost infinite number of ways that malware can mutate. The code itself can be reordered, or it can be encrypted using different keys, or its purpose can be obfuscated, and it is impossible for signature databases to encompass all of the possible signatures, even if signature writers could keep up.

But now some commercial IDSs are using machine learning to automatically generate more general, abstract signatures for known attacks (Endgame is one vendor that claims to offer an IDS of this type [12]). By looking at common features, the vendors claim, misuse detection systems will avoid being fooled by polymorphic variations of the malware and can even detect new malware that uses the same attack techniques as known malware, sharply reducing false negatives.

On the other hand, while machine learning-generated signatures are signatures on steroids, they can nevertheless only be used to detect previously identified "bad" behavior. Misuse detection systems that use machine learning-generated signatures will still not be able to identify truly new attacks that do not share characteristics with older malware for which signatures already exist. This means that attackers will continue to be able to find ways to prevent detection of malware and that machine learning-generated signatures, although a major improvement over human-generated signatures, may be just one more step in the continuing arms race between cyber attackers and defenders.

Use and limits of big data for specification-based detection

As mentioned above, specifications are created a priori from theoretical reasoning about legitimate behavior, rather than

by empirical observation of behavior. Because big data techniques are aimed at extracting information from empirical observation, specification-based intrusion detection will likely benefit the least of the three intrusion detection methods from big data techniques.

Yet, big data techniques can be used to help guide the creation of specifications. Specification developers can create rough specifications and compare them against a large stored database of empirical observations of system behavior. The results of that comparison can be used to refine the specifications. But using empirical data to help guide the creation of specifications is not itself new; the chief contribution of big data in this case would be increased scale and speed of the comparison.

Use and limits of big data for anomaly detection

Anomaly detection training is a process of collecting, searching, and analyzing data to determine patterns or relationships in the data (i.e., profiles) that indicate normal behavior and that may be associated with legitimate activity. Big data techniques permit the collection, searching, and analysis of much greater quantities of event data than was possible before, providing a foundation for advanced analysis. Big data thus seems tailor-made for improving anomaly detection.

The ability to handle greater quantities and types of log data could help increase the accuracy of profiles by observing larger sets of training data and thereby reducing the effect of noise in the data, but even big data techniques are not a complete panacea. Increasing data volume, velocity, and variety can overwhelm the ability of even a big data-enabled system to analyze the data. Advances in processing data must be matched by advances in feature selection in order to improve the accuracy of profiles and to reduce the data overload. Machine learning can help with this by identifying useful features and eliminating irrelevant and redundant features from training [14]. Machine learning can also be applied to profile even highly variable activity that varies over time. And advanced correlation algorithms could help find subtle and complex relationships in heterogeneous data that were not findable using older techniques [22].

Yet, other fundamental problems of anomaly detection still apply, even if an IDS is big data-enabled. Just because we can collect and analyze more data, it doesn't necessarily follow that the extra data contains any more useful information. And what is anomalous, as mentioned earlier, is not necessarily bad and what is normal is not necessarily good. The scale and speed of big data could, in the worst case, increase, rather than decrease, the number of false positives and false negatives.

Conclusion

The set of techniques collectively referred to as big data offers the promise of collecting, storing, processing, and analyzing data at unprecedented scale and speed. It is hoped that the application of big data to the problem of intrusion detection will greatly improve results by reducing false-positive and false-negative alert rates.

But the belief that there are much more accurate anomaly detection profiles to be created, for example, if only we could examine data in the right way, is still unsupported. Complex, chaotic systems with fuzzy boundaries between normal and anomalous may remain resistant to analysis no matter how many data mining and machine learning algorithms we bring to bear on the problem. And improved signatures created using big data techniques and used in misuse detection systems are still just signatures, leaving systems just as unable to detect new attacks as they were before. It could also turn out that, even if big data techniques can improve intrusion detection, the overall improvement is slight. We are still early in the application of big data to intrusion detection and there is much potential, but fundamental limits in intrusion detection mean that big data cannot be a magic bullet.

References

1. Apache Hive – https://hive.apache.org/.
2. D. Ariu, R. Tronci, and G. Giacinto. 2011. HMMPayl: An Intrusion Detection System Based on Hidden Markov Models. *Computers & Security* 30, 4 (June 2011), 221-241. DOI: http://dx.doi.org/10.1016/j.cose.2010.12.004.
3. Matt Bishop. 2002. *Computer Security: Art and Science.* Addison-Wesley Professional, Boston, MA, USA.
4. Cárdenas, A.A., P.K. Manadhata, and S. Rajan, eds. 2013. Big Data Analytics for Security Intelligence, *Cloud Security Alliance*, 2013 https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Analytics_for_Security_Intelligence.pdf.

5. CERT. sadmind/IIS Worm – https://www.cert.org/historical/advisories/CA-2001-11.cfm?.
6. S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes. 2007. Using Model-Based Intrusion Detection for SCADA Networks, in *Proc. 2007 the SCADA Security Scientific Symposium*, pp. 127-134. Miami Beach, FL. Jan. 2007.
7. Cohen, F. 1987. Computer Viruses, *Computers & Security,* Volume 6, Issue 1, 1987, pp 22-35, ISSN 0167-4048 – http://dx.doi.org/10.1016/0167-4048(87)90122-2.
8. Crosman, P. How PayPal Is Taking a Chance on AI to Fight Fraud, *American Banker*, September 1, 2016 – https://www.americanbanker.com/news/how-paypal-is-taking-a-chance-on-ai-to-fight-fraud.
9. Dark Reading. A Case Study In Security Big Data Analysis, 9 March 2012 – http://www.darkreading.com/analytics/security-monitoring/a-case-study-in-security-big-data-analysis/d/d-id/1137299.
10. Dark Reading. A Case Study In Security big data Analysis, DarkReading.com, March 9, 2012 – http://www.darkreading.com/analytics/security-monitoring/a-case-study-in-security-big-data-analysis/d/d-id/1137299.
11. Denning, D.E. 1987. An Intrusion-Detection Model. *IEEE Trans. Softw. Eng.* 13, 2 (February 1987), 222-232. DOI=http://dx.doi.org/10.1109/TSE.1987.232894.
12. Endgame. Our Platform. Endgame – https://www.endgame.com/platform.
13. eEye Digital Security. ANALYSIS: .ida "Code Red" Worm – https://web.archive.org/web/20110722192419/http://www.eeye.com/Resources/Security-Center/Research/Security-Advisories/AL20010717.
14. Guyon, I. and A. Elisseeff. 2003. "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research 3* (2003), pp 1157-1182 – http://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf.
15. Hadoop: What Is It and Why Does It Matter? – https://www.sas.com/en_us/insights/big-data/hadoop.html.
16. Stephen Jou, Machine Learning: A Primer for Security, *ISSA Journal*, August 2016, pp. 14-21.
17. C. Kruegel and G. Vigna. 2003. "Anomaly Detection of Web-Based Attacks," in *Proc. 10th ACM Conference on Computer and Communications Security* (CCS '03). ACM, New York, NY, USA, 251-261. DOI=http://dx.doi.org/10.1145/948109.948144.
18. D. Laney. 2001. 3D Data Management: Controlling Data Volume, Velocity and Variety. Technical Report 949, META Group (now Gartner). February 6, 2001 – http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.
19. R. Mitchell and I. R. Chen. 2007. Behavior Rule Specification-Based Intrusion Detection for Safety Critical Medical Cyber Physical Systems, in I*EEE Transactions on Dependable and Secure Computing*, vol. 12, no. 1, pp. 16-30, Jan.-Feb. 1 2015. DOI=http://dx.doi.org/10.1109/TDSC.2014.2312327.
20. H. S. Teng, K. Chen and S. C. Lu. 1990. "Adaptive Real-Time Anomaly Detection Using Inductively Generated Sequential Patterns," in *Proc. 1990 IEEE Computer Society Symposium on Research in Security and Privacy*, Oakland, CA, 1990, pp. 278-284. DOI: http://dx.doi.org/10.1109/RISP.1990.63857.
21. A.Turing. 1937. On Computable Numbers, with an Application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, Series 2, Volume 42 (1937), pp 230–265, DOI: http://dx.doi.org/10.1112/plms/s2-42.1.230 and On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction, *Proceedings of the London Mathematical Society*, Series 2, Volume 43 (1938), pp 544–546, DOI: http://dx.doi.org/10.1112/plms/s2-43.6.544. Available: http://www.turingarchive.org/browse.php/B/12.
22. R. Zuech, T.M. Khoshgoftaar, and R. Wald. 2015. Intrusion Detection and Big Heterogeneous Data: A Survey, *Journal of Big Data* (2015) 2:3, Springer, DOI: http://dx.doi.org/10.1186/s40537-015-0013-4.

About the Author

Mark Heckman has worked in the field of information security for over 30 years as a researcher, developer, and practitioner. He currently is a Professor of Practice in the Center for Cyber Security Engineering and Technology at the University of San Diego. He may be reached at mheckman@sandiego.edu.

Looking Ahead – Journal Themes

August: Disruptive Technologies – Due: 6/22/17

Waves of disruptive technologies continually threaten to sweep away existing business landscapes. Blockchain, tokenization, 5G networks, quantum cryptography, smart cars, and the Internet of Things (IoT) are names of just a few. All promise the disintermediation of our competitors, and offer us unlimited new opportunities if only we become early adopters willing to accept the risks. This issue of the Journal seeks articles on the security threats and vulnerabilities of all things disruptive, and solutions that can help us to embrace the coming new technology waves and manage the risks.

September: Health Care – Due: 7/22/17

Healthcare is one area of particular focus for information security practitioners as there are very specific security, privacy, and technological issues and mandates one must deal with. These also vary by jurisdiction. There are also many tools security professionals can use in this space that allow for a relatively consistent application of controls. We are looking for your thoughts and ideas on information security in the healthcare space.

Submit articles to editor@issa.org.