Birthday Hunting

Jack Burgess

Birthday Hunting

GIAC (GCFA) Gold Certification

Author: Jack Burgess, j.burgess.mail@gmail.com
Advisor: Hamed Khiabani, Ph.D.
Accepted: April 11th 2020

Abstract

The Birthday Problem has a number of applications to incident response. Existing tools can both narrow the focus of the incident response team and limit their experience to a small subset of alerts. This leaves specialized tools to do the analysis before anything is investigated, imposing a range of biases. We show the use of randomly selected investigation of nodes in the environment has a significant likelihood of finding the adversary. This allows for the evaluation of threat hunting and security operations. The approach is then extended to the evaluation of cybersecurity machine learning products. These products may be complicated and opaque. The approach presented avoids the need to understand the internals, shifting analyst focus to business as usual operations.

1. Introduction

For this paper, we consider a simplified model of a Security Incident Response Team (SIRT). We are interested primarily in the part of the SIRT that investigates indicators, "a sign that an incident may have occurred or may be occurring now" (Cichonski, 2012, p. 26). The indicators are then qualified and turned into leads. These leads can come from matching or hunting (Bejtlich, 2013). "Matching" might start from known conditions like an alert from a security tool. For instance, "Command and control indicator of compromise has been identified by the firewall". For our purposes we simplify "hunting" to an intelligence driven, human led activity that produces leads.

Ultimately, the outputs of both matching and hunting lead to an incident being identified. As far as our SIRT is concerned, the operational cycles following up a lead are the same. We assume the containment, eradication, and recovery (Cichonski, 2012) or Fix, Finish, Exploit, Analyze, and Disseminate (Roberts, 2017) don't distinguish between the source of the lead and are worked by the same people.

Machine learning has been a tool used by incident response teams since at least 1985 (Denning, 1985). In recent times several products have been receiving significant press, followed by management attention, with machine learning as a key component. This machine learning typically proposes to do some of the work of the hunter. Moving the domain of what a security appliance can detect from regex type matching closer to the human lead hunting. The SIRT, we assume, is geared to evaluate a matching type appliance—the regex matched, the packet, IP, and so on—a key feature of matching is the existence of a ground truth. It matches, or it does not.

At some level, the SIRT needs to evaluate the effectiveness of both approaches to allocate resources appropriately. Questions like "Is my hunting methodology better than random?" and "Is this security appliance performance better than random guessing?" allow us to consider detection in a holistic, unified way. In this paper, we show random guessing is an unintuitively high bar for detection methodologies to pass. After establishing how to measure this bar, we go on to show how it can be used as a yardstick. This enables the measurement of both hunting and security appliance performance. In the

Jack Burgess, j.burgess.mail@gmail.com

case of hunting we show that *random guessing* itself may be an effective detection strategy. While for security appliances, we show that this is a significant factor in the buy vs. build decision – the cost to buy a random number generator being negligible.

1.1.1. Evaluating Significance

For better or worse, the scientific community frequently compares the results of a hypothesis to the likelihood of obtaining those same results by chance (Wasserstein, 2016). When a hypothesis is considerably "better than random" typically a p value less than 0.05, the scientist will move to publish the paper. The risk for the scientific community is the work might not actually be better than random. When an unreproducible paper is published future work is at risk of being thrown off-track. For an incident response team an incident is still an incident. Whether it is discovered by hook or by crook, the value to the organization is the same. The SIRT moves from detection to containment.

1.1.2. Security Operations Challenges

One of the challenges security operations that may face is a sort of tunnel vision. This can be shown as a corollary from the base rate fallacy (Bar-Hillel, 1977): Things like DLP, Phishing, and certain specific IPS alerts can take up the bulk of day to day operations. A few signatures cast very wide nets over the environment. This then dominates SIRT expertise. The false positive rate is heavily dependent on the size of the net that is cast. The rate of malicious activity depends on the number of threat actors. The number of events considered by these tools can be hundreds of times larger than events related to credible threats. A 99% accurate alert where 1 in 1000 people are malicious will produce about ten times as many false positives as true positives[1]. While hard work goes into doing much better than this by vendors, only one security tool needs to have the tuning a little off to cause trouble.

[1] As an aside "Accuracy" does not handle classification where there are imbalanced classes well. Precision and recall are much less sensitive to this. Unfortunately, vendor material refers to these much less in the author's experience

In this way, these "matching" tools draw a map of the network. The danger, as with any map, is it becomes confused for the terrain. In the case of our SIRT this is what guides our focus. Our SIRT then becomes very good at closing phishing tickets, very good at closing minor DLP events. While these are worthwhile, these are different skills than those asked by an existential threat—the threats that might justify much of the SIRT's funding. Additionally, the volume of false positives tells us little about threat actor activity—just the how quickly the SIRT can tune an appliance. In the meantime, there is no guarantee that the threat actor is even present on the SIRT map.

In terms of F3EAD (Roberts, 2017) the focus becomes Find. Containment can become reduced a single host and security operations starts to look a lot like pulling from the network and reimaging. Exposure to broad skills and processes are necessary for critical incidents outside of this. The issue is illustrated by two facts; Security incidents represent a credible existential threat (Greenberg, 2018), and there is an aversion to paying market rates for talent, framed as "shortage" (Evans, 2010). In this scenario security operation's then both needs and does not have the same levels of familiarity with "F2EAD".

Another challenge in security operations is evaluating security products. There's a lot of them out there and many do some fantastic stuff. For the analyst on the ground the technical details aren't as visible as the output. The intuitive evaluation approach is to have a look and say oh this looks pretty good, maybe this looks a bit funny, then you must decide if the product is worth it.

Typical output from these tools might be a list of "anomalous" assets with different degrees of explanation and direct relevance to the SIRT's mission—at least from the perspective of the security analyst. Without going into specifics and technical implementation, one of the most common properties seems to be simply being expensive.

We assume the SIRT then must interface with a sales representative of some kind. The pitch might go something like this:

Sales Associate: "We have a great product you should buy it."

SIRT: "I've heard about it but I'm not sure"

Jack Burgess, j.burgess.mail@gmail.com

SA: “Why not run a Proof of Concept (PoC)? We’ll find something”

From a statistics perspective there is a concern here. Only successful PoCs make a sale. A test of the hypothesis that “this product has value” will have different outcomes for the sales associate’s perspective and the SIRT’s. The sales associate talking to an infinite number of SIRTs for a PoC is different to the SIRT buying one product and move forwards with detections. The sales associate isn’t obliged to list all the occasions where the product did not work. From the SIRT’s perspective an unsuccessful implementation, one can imagine, would run up a significant amount of stress across security operations. Information about unsuccessful products is further suppressed by the client; high amounts of noise to deal with means less time to present at conferences. Contrast this with how far a coordinated marketing campaign might propagate the positive reviews.

Through this mechanism, the “Proof of concept that finds something” approach might sustain a vendor, and win awards, while also being no more powerful than a random number generator. This is not to say a random number generator isn’t a powerful tool. On the contrary, we suggest that the value it does have can, and likely is already, acceptable for several vendors and SIRTs. Here we offer the SIRT a way to capture this value themselves, without the overhead.

1.1.3. The Birthday Problem

One of the earliest presentations was by Rouse Ball presenting an idea discussed by Harold Davenport (Ball, 1960, p. 45). The classic presentation goes something like: How many people do we need in a room to have a 50% chance that at least 2 people share a birthday. The number turns out to be 23 for a bit over 50%, and for 35 people, it shoots up to 80%.

For the purposes of mapping the birthday problem to an enterprise network, we refer to each person in the room as a node. This could be an endpoint, an account, some other asset, or an abstract combination of all three. The number of days in the year maps to the number of nodes that we have. The number of people in the room maps to the number of assets investigated. Each room we get with a certain number of people in it then becomes a single hunt.

This can be generalized by considering the probability of no matching birthdays:

$$P(n) \approx 1 - \left(\frac{364}{365}\right)^{\binom{n}{2}}$$

then using Taylor expansion and generalizing the days to m “nodes” as:

$$P(n, m) \approx 1 - e^{\frac{-n^2}{2m}}$$

for *m* nodes existing in the network and *n* of these nodes investigated

1.1.4. Assume Breach

For the purposes of this paper "Assume Breach" is taken to mean, in a technical sense, that at least 1 threat actor is present on 1 node. This provides a lower bound for the expected findings from hunting within the network.

1.1.5. Dwell Time

Dwell time is considered as "The number of days an attacker is present on a victim network from the first evidence of a compromise to detection" (FireEye, 2019). The global median reported by FireEye is 78 days (FireEye, 2019). For modeling purposes this provides a conservative upper bound on the number of nodes it is possible to look at. Each node represents a certain amount of time to investigate. In principle, once the median time to investigate a given number of machines reaches the dwell time this random approach is then at best “as good as” traditional SIRT detection methodologies, assuming equivalent success rates and operational overheads. When the number of machines investigated has a time cost lower than the dwell time Birthday Hunting becomes more effective, given similar operational overheads.

1.1.6. Investigating a Single Node

Investigation of a node can take on many forms. We assume that this typically involves collection of triage data—a smaller than full disk and memory capture to facilitate analysis—then creation of a timeline from this data. For normal security operations a timeline might be used with an IPS alert to pivot to investigation to a specific time.

From here the attacker's activity can be observed. This process of observation and collection of evidence goes on to form the raw material of the eradication phase. For a Birthday Hunting program, the key difference is the lack of pivot. With a colorized timeline this might involve the analyst identifying chunks of "interesting" behavior. This behavior would be evaluated and recorded as a baseline that can later be used to separate common from rare behavior in other nodes that are investigated. The value of this is considered further later in the discussion. Alternatively, investigation could be as simple as running “pescan” on the entire disk and sifting through the results.

In the context of following up an alert to evaluate a machine learning appliance this is not too different from what the SIRT might do already and not completely different from taking a box at random. "Top talkers" for instance, is an intuitive use of NetFlow data where it is aggregated by source to find which nodes are using the network the most, then investigating any changes or suspicious rankings. This can be aggregated over hours, days, weeks, months and so on. In a similar way the initial pivot provided by machine learning based indicators can range from a time window of seconds to an upper limit of the current lifetime of the node, the "Something is anomalous with this box" scenario. It follows that a security analyst already has some degree of familiarity with pivotless investigation. A directionless hunt with a loosely defined hypothesis and an anomalous alert that says "something is weird" differs from the machine learning product alert only in of the number that can be generated and associated costs to investigate. The value implied from investigating both is the basis for this paper. The directionless hunt, the "Birthday Hunt" provides value to the SIRT though it's directionless properties that allow it to shed existing SIRT biases. The value of the security machine learning appliance exists to the degree that it beats "Just picking a bunch of assets from the network".

2. Birthday Hunting

2.1. Simulating a “Hunt”

There is no time dimension in the birthday problem. The likelihood depends only on the nodes and the number of nodes investigated. The structure we model is generated

iteratively, analogous to people entering the room. A key deviation here is that a threat hunting team will not have overlapping birthdays. We assume each investigation is thorough and there's no need to investigate a box twice. From this perspective a hunt that has nodes overlapping have identified an adversary. A corollary of this is that each additional compromised machine represents an extra hunt. The modelling results assume the number of hunts is significantly larger than the number of compromised machines.

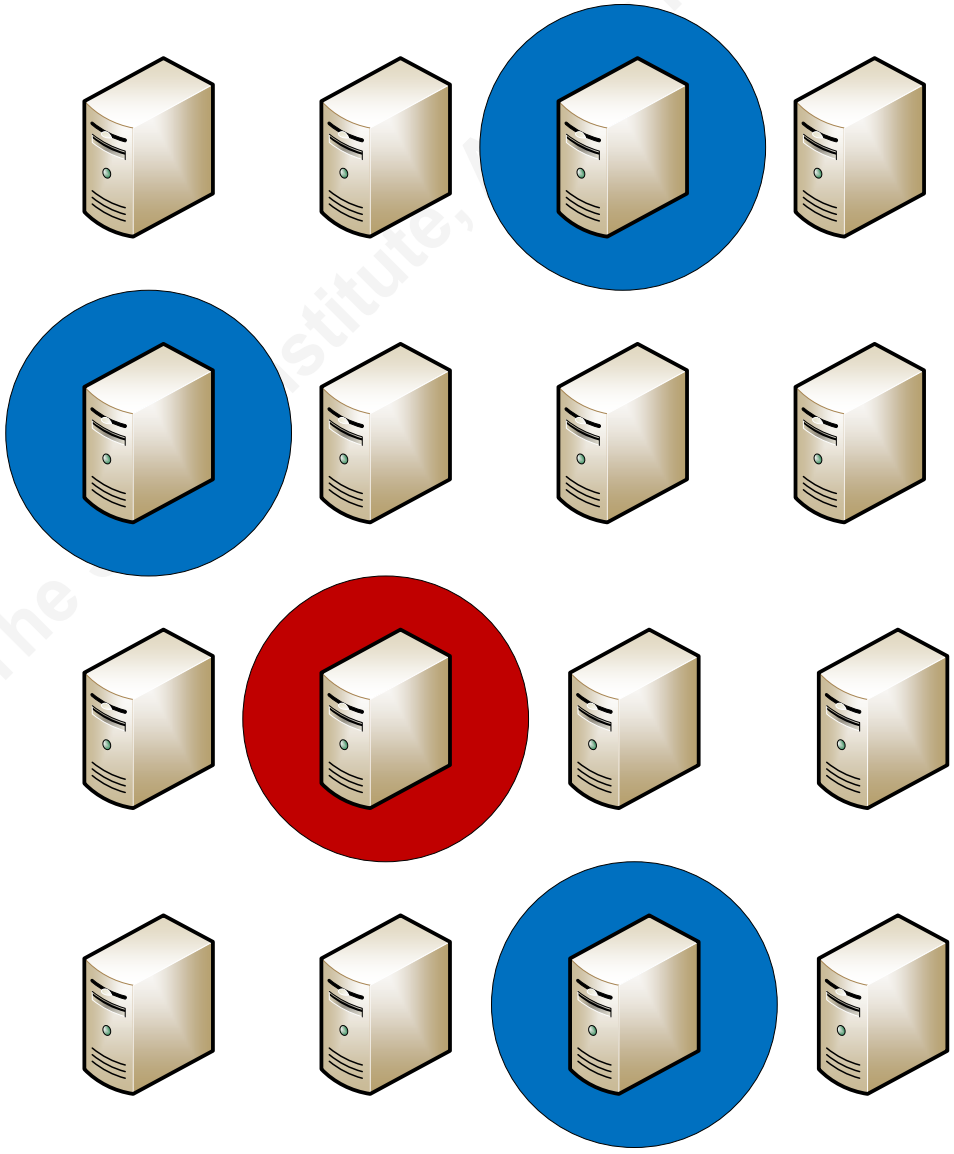

Figure 1: The hunt team chooses randomly (Blue). In a statistical sense all assets are investigated at the same time. As the hunt team will not investigate the same node twice, it follows that two activity markers on the same node must be adversary and hunt activity (Red).

Jack Burgess, j.burgess.mail@gmail.com

2.2. Modelling

We define 3 key variables; the number of known assets in the environment m, the number of assets investigated as part of the hunt n, and the number of hunts H. Each asset investigated n_i has either 0 for no threat actor activity, or values greater than 0 indicate a threat actor was discovered. The maximum value is taken for each hunt H_i and normalized with H to provide a density. We assume that any discovered attackers are immediately replaced in the network from a reservoir of attackers. In practice there may only be so many adversaries with capability of compromising a network. In this sense number of hunts conducted could have no loss in effectiveness being spread over a longer period—up to orders of magnitude comparable to the adversary dwell time where existing security tools might be expected to take over

2.3. Recovering Birthday Problem Results

Using the simulation code in the appendix, we attempt to recover the traditional results from the Birthday Problem where $m = 365$.

Figure 2 plots the simulated results against approximate results. As expected at 20 – 23 investigations, the likelihood is close to 50%.

Figure 3 looks at hunts where 23 nodes are investigated specifically. A smaller density of hunts (~0.025) have greater than 1 threat actor discovered—from the perspective of at least 2 people in a room having the same birthday this is counted simply as meeting the condition—we consider this for discussion later in the context of security operations. These combine to give a total density greater than 0.5, the expected result from the traditional presentation.

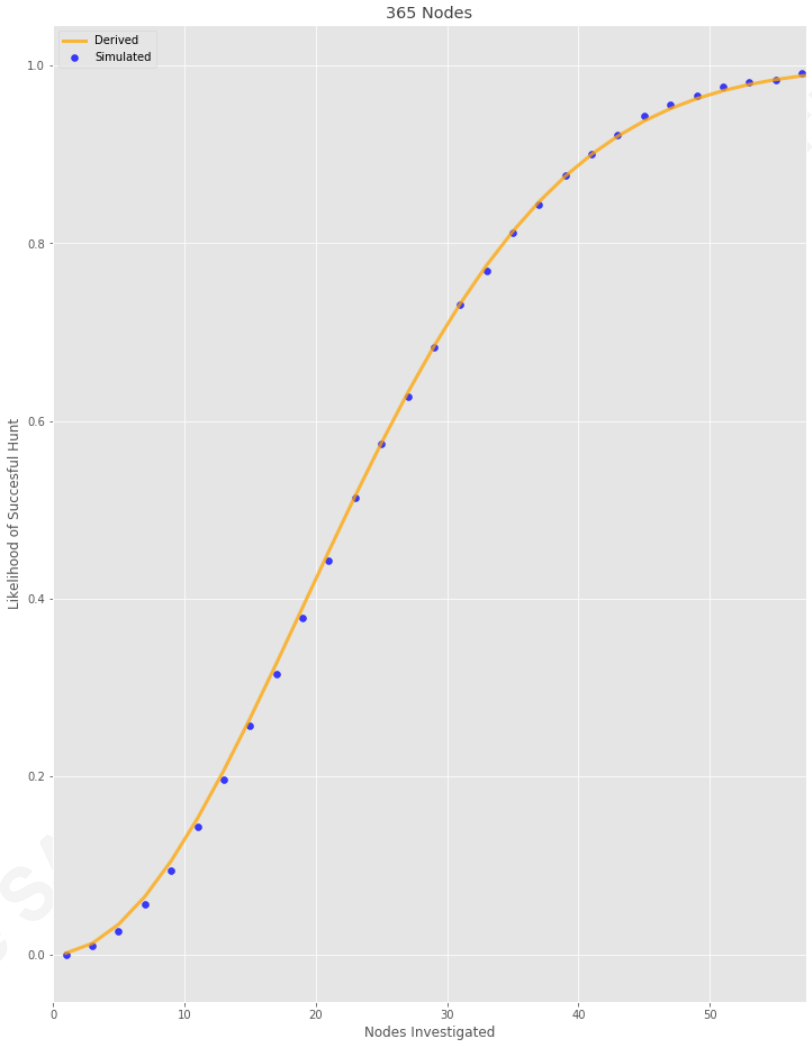

Figure 2. The original Birthday Problem numbers (365 nodes) recovered in the context of a threat hunt simulation (Blue). The Orange line represents the approximation discussed in 1.1.3

Jack Burgess, j.burgess.mail@gmail.com

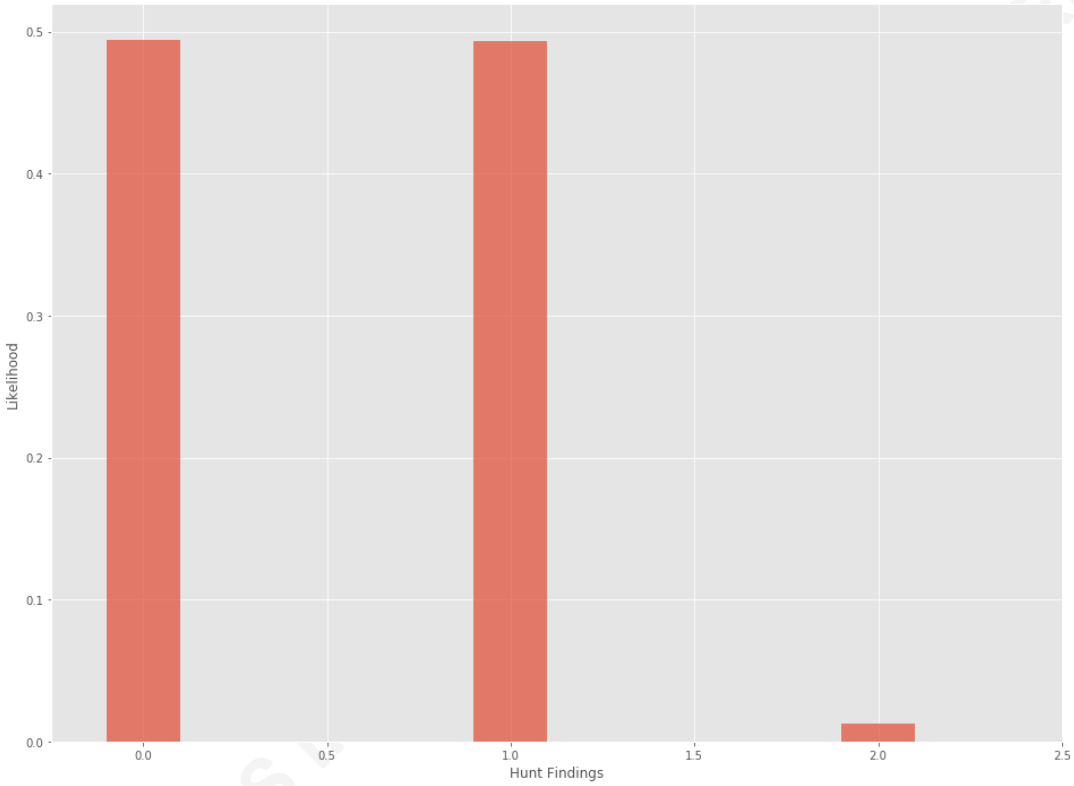

Figure 3: Density of hunt findings for 23 nodes investigated from a total of 365.

2.4. A Credible SIRT Hunt

Having recovered the expected values from the Birthday Problem we move on to values representative of a hunting program. We consider a SIRT in a large enterprise with n = 10 k nodes, m = 100 nodes investigated, and H = 10 hunts. It is assumed that the time to collect the data for m is insignificant on the timescale of threat actor dwell time—the time taken to investigate this data can be considered against median dwell time to benchmark existing SIRT detection methodologies. In this case H is chosen as 10 hunts to be on the order of 1 hunt per month. With this model we find a typical precision of 40%. For 10 hunts per year this represents 4 incidents, and 6 hunts without any findings. Later we consider the value, both gained and lost, for discovery of each incident and a hunt without findings.

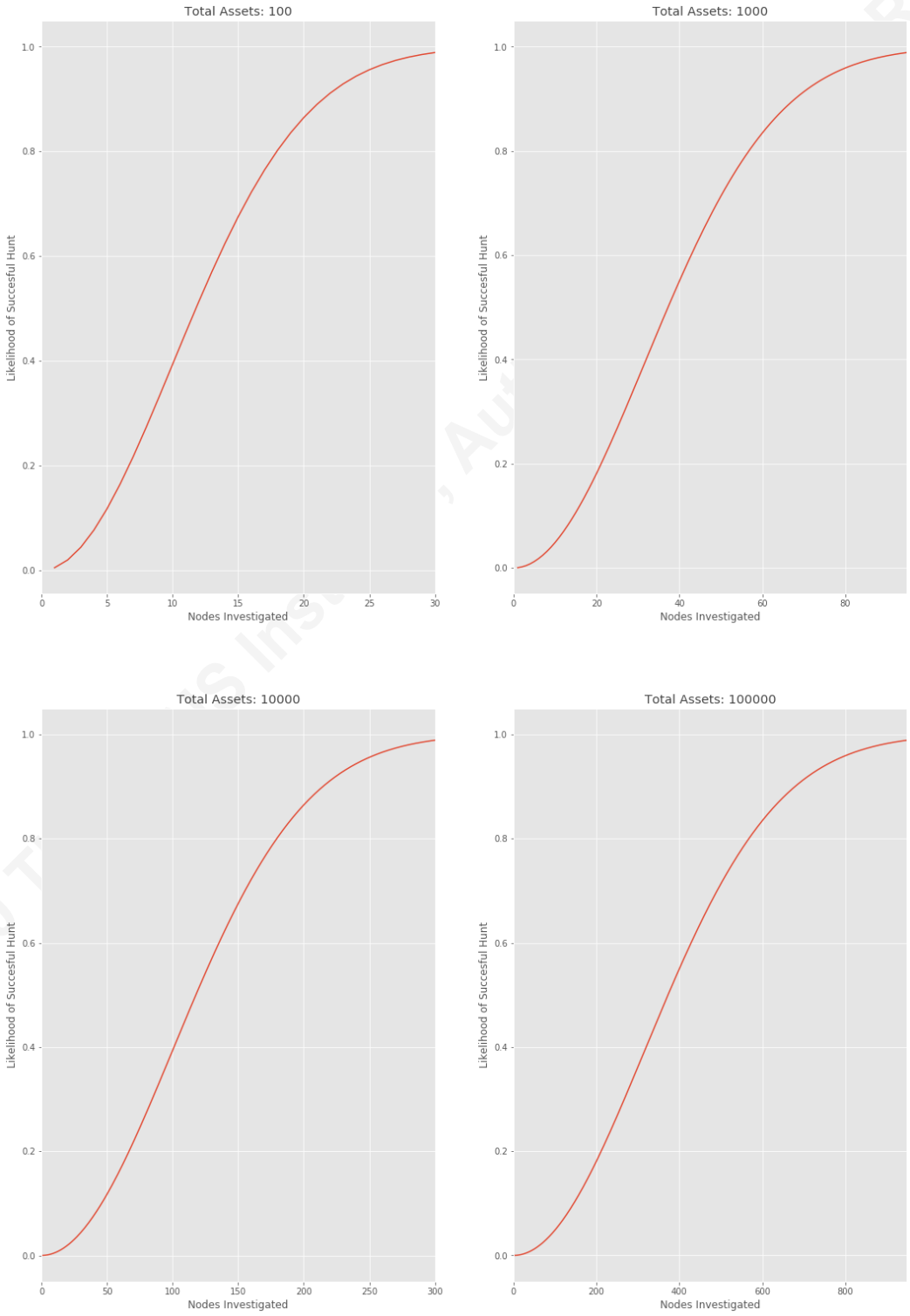

Figure 4: Birthday Hunting likelihood of Success for 100, 1000, 10k, 100k Assets

Jack Burgess, j.burgess.mail@gmail.com

3. Discussion

3.1. Developing a Birthday Hunting Program

These results confirm that a small fraction of investigated nodes can reveal a compromise is a larger environment. The practicality of this random selection based "Birthday Hunting" program depends largely on the cost to investigate vs the outcome.

The cost of a hunt without a finding is proportional to investigation costs for the nodes investigated. Reducing investigation costs allows for more investigations in the same timeframe, increasing the likelihood of a successful hunt. Alternatively, the likelihood of a successful hunt may already be acceptable. A starting value for this could be on par with the network's IPS with a similar dwell time and operational overhead. In this case reducing investigation times frees up analysts to work in other areas of the SIRT. As mentioned earlier, traditional timelining might involve starting from a specific IPS alert as the initial pivot, then validating and tracking attacker behavior. Without the initial pivot the analyst has several options. The first would be to step forward through the timeline looking at clusters of activity. In a colorized timeline this might be finding red evidence of execution close to green writing to the registry. A baseline of common behaviors can be built up and shared, removing these combinations from analysis in future timelines created as part of the hunt. Anecdotally this "Baselining" to highlight novel temporally related artifacts scales well with higher values of *H*. If we treat each combination of temporally related activity as a "word" this is consistent with Zipf's law. Alternatively, scope can be reduced. Lateral movement, privilege escalation, and persistence have several well-known techniques (Strom, 2018). The artifacts associated with these tactics are also well known. Focusing on these can reduce investigation time and still have acceptable range of threat actors in scope. A combination of collecting baseline data and focusing on common behaviors creates a reference of known good activity that can inform future hunts, adjacent detection technologies, and security operations processes.

Analogous to Chaos engineering, a different perspective is to simply rebuild each machine at random (Basiri, 2016). For the threat actor this looks like they were discovered. From Mandia 2014, Ch. 17, on being discovered the threat actor might attempt change TTPs, go dormant, become destructive, or increase activity. These can be

devastating for a SIRT that detected the activity specifically. In the case of a SIRT that had machine wiped by luck the implications can read differently. The threat actor may assume the SIRT has visibility where it does not, changing TTPs from undetectable to something the SIRT can see. Becoming dormant does will not cause the SIRT to consider the incident contained, instead it buys time for the SIRT to hone their skills.

3.1.1. On Prioritizing Nodes

Intuitively threat actors will target more valuable nodes. In the case of the Birthday Problem an unequal distribution *decreases* the number of nodes to investigate for 50% likelihood (Borja, 2017).

This can also be modelled as a simultaneous 2 player, 2 strategy, repeated game with blue team strategies of “Investigate” or “Don’t investigate” and red strategies of “Compromise” or “Don’t compromise”. From this it can be shown that the likelihood of an asset being compromised is proportional to the red team payoff—this same game can be further used to estimate the false negative rate when evaluating the precision and recall of security appliances and hunting programs.

3.1.2. Value of a Pivotless Timeline

A pivotless timeline—based on Kape triage data for instance—allows for investigation independent of the common SIRT alerts. The idea itself is long associated with using timelines, being proposed by Guðjónsson in the original supertimeline paper, to use an activity density approach to create a pivot. While common alerts are well understood, they are unlikely to represent existential risk to a firm—if common alerts did represent a significant existential risk the firm would likely no longer exist. Instead the analyst can concentrate on what might represent existential risk. Meanwhile, recording normal patterns as they move through evidence of execution, registry modification, file deletion etc. a baseline of normal can be developed speeding up future analysis. The tools required to support this analysis can be evaluated at the same time. The time it takes to acquire and prepare triage data can be evaluated and process honed in a low-pressure environment. Opportunities to improve this can be tested with less risk if something goes wrong than would happen during an incident in full flight. Treating a birthday hunt as a drill provides an opportunity for analysts to better understand the baseline network

behavior. It is only slightly tongue in cheek to point out that this randomized hunting approach is agentless, has no end of life, non-parametric, and has 100% uptime.

3.1.3. More Than One Birthday on the Same Day

By considering the cases where more than one successful hunt occurs as one hunt this suggests that for some percentage of hunts less that the estimated number of nodes to be investigated are needed. In practice investigation may be halted and resources assigned to containment of the discovered incident.

3.1.4. Metrics for existing hunting programs

The purest form of birthday hunting is perhaps as thought experiment, a *gendanken*. An existing hunting team can be compared to an imaginary hunting team. The hypothesis for a real threat hunt will have a certain number of nodes in scope. Over several real-world hunts, the expected number of findings for birthday hunts should be greater. This delta from birthday hunting contains information about intelligence used to form hunt hypotheses and investigative methods used by the hunt team. Is the delta negative? Perhaps the intelligence being used does not reflect the environment. Is the intelligence well vetted? Maybe something is being missed in investigations. Each of the variables (Nodes investigated, Nodes in environment, existence of threat actor activities) can be validated independently to reverse engineer the others—if the data is reliable and we're sure the attacker activity is there, the hit rate implies how many nodes are in the environment—this reverse engineering can be done by rearranging the equation from 1.1.3. These "shadow" variables can become metrics for broader security operations concerns; shadow IT, shadow data and shadow intel. These concerns may exist already but are difficult to measure. "How many of our assets do we know about?", "Are we receiving the data we should?", and "How much threat actor activity might be invisible to us?" are all important operational concerns that have no doubt been given some thought. This birthday hunting approach can allow the SIRT insight how much of the picture might be missing.

3.2. Evaluating Machine Learning Products

3.2.1. Establishing an Acceptable Performance Baseline

The results in Figure 4 can also be interpreted in the context of baselining a proof of concept. What are the odds of successful PoC if the product did nothing more than hand you a random list of results? In this context what was a complicated vendor evaluation process is reduced to the cost of implementing a random number generator and a list of assets.

3.2.2. Validation vs. Replication

Given that only successful PoCs make a sale, and there is a significant chance of successful PoCs making that sale, vendor documentation and testimonials may be blind to the impact provided by the content. In this light attempting to understand the tool's inner working for verification purposes becomes not only irrelevant, but in some cases impossible. There is significant effort in untangling complex language, foreign to many without a statistics background as well.

Guides to evaluating machine learning products can focus on questions like "Where does the data come from?", "what is being used for ground truth?", "Are the algorithms supervised, unsupervised, or semi supervised?". These are all relevant questions, and any vendor will have *an* answer. More importantly is what does an analyst, or a CISO, do with this information. Hopefully few products are still trained on the KDD dataset and derivatives given the well-established concerns (McHugh, 2000). Even so, being trained on white noise as presented here can be a net positive for the SIRT—so why spend the time trying to look under the hood of a car nobody understands (Došilović, 2018)?

Instead by treating the system as a black box with a measurable baseline analysts can focus on what they know, validating alerts.

4. Conclusion

Concepts from the birthday problem can be applied to context of SIRT threat hunting. This lens allows for identification of threat actors, broader exposure of analysts to the environment, and opportunities to work on tools and processes in a low stress environment for the SIRT. While at the same time, providing exposure to tools and processes needed for existential threats. With the effectiveness of randomized hunting being shown to be unintuitively high, only a minor fraction of nodes in the environment need to be investigated for a high likelihood of success.

With the number of nodes to be investigated predetermined by a security appliance, these nodes can then be considered against a theoretical a baseline. This enables the black box evaluation of security machine learning products. A black box approach to evaluation abstracts the complexity of the tools themselves, and more importantly marketing material, into an objective frame. This allows analysts to perform the evaluation with analyst skills—providing confidence in the purchasing decision without having to divert resources from the other areas of the SIRT.

Jack Burgess, j.burgess.mail@gmail.com

References

Ball, WW Rouse. (1960). "Other questions on probability." Mathematical Recreations and Essays 45

Bar-Hillel, M. (1977). The base-rate fallacy in probability judgments. DECISIONS AND DESIGNS INC MCLEAN VA.

Basiri, A., Behnam N., De Rooji, R., Hochstein, L., Kosewski, L., Reynolds, J., & Rosenthal, C. (2016). Chaos engineering. IEEE Software, 33(3), 35-41.

Bejtlich, R. (2013). The practice of network security monitoring: understanding incident detection and response. No Starch Press.

Borja, M. C., & Haigh, J. (2007). The birthday problem. Significance, 4(3), 124-127.

Cichonski, P., Millar, T., Grance, T., & Scarfone, K. (2012). Computer security incident handling guide. NIST Special Publication, 800(61), 1-147.

Denning, D., & Neumann, P. G. (1985). Requirements and model for IDES-a real-time intrusion-detection expert system (Vol. 8). SRI International.

Došilović, F. K., Brčić, M., & Hlupić, N. (2018, May). Explainable artificial intelligence: A survey. In 2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO) (pp. 0210-0215). IEEE.

Evans, K., & Reeder, F. (2010). A human capital crisis in cybersecurity: Technical proficiency matters. CSIS.

FireEye (2019), M-Trends Report 2019; Insights into Today's Breaches and Cyber Attacks. FireEye.

Greenberg, A. (2018). The untold story of NotPetya, the most devastating cyberattack in history. Wired, August, 22.

Guðjónsson, K. (2010). Mastering the super timeline with log2timeline. SANS Institute.

Mandia, K., Luttgens, J., Pepe, M. (2014). Incident Response & Computer Forensics, Third Edition, 3rd Edition, McGraw-Hill

McHugh, J. (2000). Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM Transactions on Information and System Security (TISSEC), 3(4), 262-294.

Jack Burgess, j.burgess.mail@gmail.com

Roberts, S. J., & Brown, R. (2017). Intelligence-Driven Incident Response: Outwitting the Adversary. O'Reilly Media, Inc.

Strom, B. E., Applebaum, A., Miller, D. P., Nickels, K. C., Pennington, A. G., & Thomas, C. B. (2018). MITRE ATT&CK: Design and Philosophy. MITRE Product MP, 18-0944.

Wasserstein, R. L., & Lazar, N. A. (2016). The ASA statement on p-values: context, process, and purpose.

Appendix

Birthday Hunting Simulation (Python 3.6+)

```
import numpy as np

def run_birthday_hunt_program(number_of_assets,
number_of_investigations, number_of_hunts=10):
    results = []
    for y in range(number_of_hunts):
        environment = [0 for x in range(number_of_assets)]
        for x in range(number_of_investigations):
            environment[np.random.randint(number_of_assets)] += 1
        results += [max(environment) > 1]
    return results

hunt_results = run_birthday_hunt_program(10000, 100, 10)
```

Jack Burgess, j.burgess.mail@gmail.com

Last Updated: May 30th, 2020

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Amsterdam August 2020	**Amsterdam, NL**	**Aug 03, 2020 - Aug 08, 2020**	**Live Event**
SANS FOR508 Canberra August 2020	**Canberra, AU**	**Aug 17, 2020 - Aug 22, 2020**	**Live Event**
SANS OnDemand	**OnlineUS**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**