Using a Collated Cybersecurity Dataset for Machine Learning and Artificial Intelligence

Erik Hemberg
MIT CSAIL
Cambridge, United States of America
hembergerik@csail.mit.edu

Una-May O'Reilly
MIT CSAIL
Cambridge, United States of America
unamay@csail.mit.edu

ABSTRACT

Artificial Intelligence (AI) and Machine Learning (ML) algorithms can support the span of indicator-level, e.g. anomaly detection, to behavioral level cyber security modeling and inference. This contribution is based on a dataset named *BRON* which is amalgamated from public threat and vulnerability behavioral sources. We demonstrate how *BRON* can support prediction of related threat techniques and attack patterns. We also discuss other AI and ML uses of *BRON* to exploit its behavioral knowledge.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Artificial intelligence**; **Machine learning**; **Machine learning approaches**;

KEYWORDS

cyber security, threat hunting, Machine Learning, prediction

ACM Reference Format:
Erik Hemberg and Una-May O'Reilly. 2021. Using a Collated Cybersecurity Dataset for Machine Learning and Artificial Intelligence. In *ACM KDD AI4Cyber: The 1st Workshop on Artificial Intelligence- enabled Cybersecurity Analytics at KDD'21, August 14–18, 2021, Virtual.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Among other enticements, Artificial Intelligence (AI) and Machine Learning (ML) offer attack planning, defensive modeling, threat prediction, anomaly detection, and simulation of adversarial dynamics in support of cyber security [1, 5, 7–10, 12]. Automated security activity presently mainly focuses on lower level malicious activity detection that relies upon indicators of compromise and forensics. Alternatively, AI and ML techniques for cyber that work at the behavioral level are emerging. They typically draw upon threat information that abstractly describes an attacker's tactics, techniques and procedures (TTPs) as well as vulnerability knowledge such as exposed product configurations and system weaknesses. These information sources are typically independent, though they sometimes have external links to one another. Here we demonstrate the use of a single dataset, *BRON*[1], that supports AI modeling and ML inference at a behavioral level, see Figure 1, by using an amalgamated set of key public threat and vulnerability information sources. *BRON* is fully described in [11].

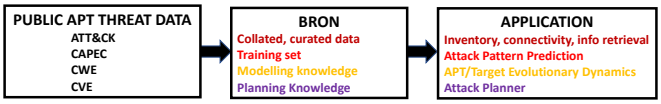


Figure 1: High level concept of *BRON*. Shows the data sources used and possible applications.

Public threat and vulnerability information is, unfortunately, extracted from historic attacks, such as Advanced Persistent Threats (APTs). Post-hoc, APTs are catalogued and framed as the behavior of a specific actor pursuing a goal, posing a threat that has specific tactics, techniques and procedures. The targets of attacks are itemized as hardware or software vulnerabilities or exposures which are sometimes themselves cross-referenced to a type of weakness as found in code, design, or system architecture. Attack patterns are recognized manually and enumerated. According to its type, each unit of information is populated as an entry of a specific database, with some amount of cross-referencing. The combined databases, with irregular, pairwise linkages between them, serve defensive reasoning.

This contribution demonstrates the use of the combined data of four such public databases amalgamated into a single graph database, *BRON* [11]. The four collated databases are:

a) MITRE's ATT&CK MATRIX of *Tactics*, *Techniques*, Procedures and Sub-techniques [14]

b) MITRE's Common Attack Pattern Enumeration and Classification dictionary (CAPEC) [15]

c) MITRE's Common Weakness Enumerations (CWE) [16]

d) NIST's Common Vulnerabilities and Exposures (CVE) [19]

Their collective entries, and links between entries, are stored in *BRON*, a threat and vulnerability graph database. *BRON* adds no new information, while it adds bi-directional links to enable faster and more convenient queries. It is publicly available and regularly updated at http://bron.alfa.csail.mit.edu.

The combined information on APTs within *BRON* expresses, for a threat, who is behind, how it works and what it targets. For a vulnerability, it expresses its type of weakness, and how it can be threatened. The structure of this information allows *BRON* to support statistical ML and inference. We illustrate this in Section 2 with the problem of predicting edges that exist between entries,

[1] *BRON* means bridge in Swedish, referring to how it links data sources

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

but which have not been reported. Solving this pattern inference problem would benefit cyber security experts in partially resolving the ambiguity of missing edges; a trained predictive model could suggest probable edges.

We then, in Section 3 discuss how *BRON* can be used for other benefits. We include information retrieval (Section 3.1) showing how *BRON* can be queried to inform users of the original sources or *BRON* about connections that are not present. Connections that are not present are ambiguous: Is the relational behavior non-existent, or existent but missing? We include modeling and simulation (Section 3.2) describing two types of use and a coevolutionary simulation of APT threats and mitigations. We finally cover AI planning (Section 3.3) where *BRON* functions as a knowledge base for attack planning or planning on attack graphs. This could, for example. assist with automation of red-teaming. Finally, future work directions are presented in Section 4.

2 PATTERN INFERENCE

BRON can be a source of training data for machine learning [2–4]. Graph properties such as number of incoming or outgoing edges, or number of paths of different connections and lengths can be used as features or labels. As well, the natural language part of entries in *BRON* offers semantic value that can be featurized.

One particularly challenging ML problem exists within *BRON* (and among the independent databases) due to the irregular nature of the cross-database linkages. Links only exist if they have been noted and reported. *BRON*, *circa* May 2021, has 666 *Techniques* and 740 *Attack Pattern*s, so, in theory, there are a total of 492,840 possible *Technique-Attack Pattern* connections. *BRON* reveals a total of 157 connected *Technique-Attack Pattern* connections, a percentage of merely 0.032%. This is accurate, to the extent that most of the possible connections would never be semantically sensible. However, given about 74% of *Techniques* are not linked to a *Attack Pattern* and the stealthy nature of APTs, there are very likely connections that are not noted or undetected. It also follows then that the goal to predict links (edges) between ATT&CK *Techniques* and *Attack Pattern*s (nodes) is important but complex and one of imputation.

Use-case: BRON for Technique-CAPEC edge prediction. Th goal is to predict links (edges) between ATT&CK Techniques and CAPECs (nodes). Each node has a textual name (e.g. "Interception"). As an initial study we use this textual information to predict Technique-CAPEC edges.

We are interested in: (1) the difference in performance due to feature selection, i.e. when more data is used from BRON, (2) the change in performance due to the feature representation, (3) a baseline classification performance established from untuned classifier models. Thus, we formulate a supervised binary classification problem: given information on pairs of one Technique and one CAPEC entry, train an inference model that predicts if there is a link between the pair of entries.

We encode the text information on Techniques and CAPECs using natural language processing into a vector as the input to the model. The entity names of different *BRON* data sources as *feature selection (data sources)* we use combinations of: *a)* CAPEC *b)* Techniques *c)* Tactics *d)* CWE *e)* CAPEC_Techniques, refers to the names of all known Techniques connected to the CAPEC but not including the name of the CAPEC itself. For example, we create a string for each of the selected features of Tactic, Technique, CAPEC and CWE: *Discovery, System Network Configuration Discovery, Network Topology Mapping, Exposure of Sensitive Information to an Unauthorized Actor*. Each string is encoded according to some *feature representation*. We experiment with two different feature representations: *a)* Bag-of-Words (BOW), *b)* Transformer Neural Network, BERT [6]. Seven different *Classification methods* with default settings from SciKit-Learn [21]: *a)* Multi Layer Perceptron (MLP), *b)* Random Forest, *c)* Logistic Regression (SGD), *d)* K-Nearest Neighbor (KNN). *e)* Naive Bayes (NB), *f)* Support Vector Machine, linear kernel (SVM) *g)* Support Vector Machine, radial basis function kernel (RBF-SVM) In total we have 84 different combinations of features selections, features representations and classification methods.

Table 1: Mean performance measures for top 5 experiments. Bold indicates the best value.

Name	Error	AUC	F1
CWE-TACTIC-BOW-SGD	0.238	0.821	0.768
CWE-TACTIC-BERT-MLP	0.242	0.827	0.769
CWE-TACTIC-BERT-RANDOM_FOREST	0.231	0.840	0.770
CWE-BOW-RANDOM_FOREST	0.226	0.850	0.773
CWE-TACTIC-BOW-RANDOM_FOREST	**0.197**	**0.870**	**0.802**

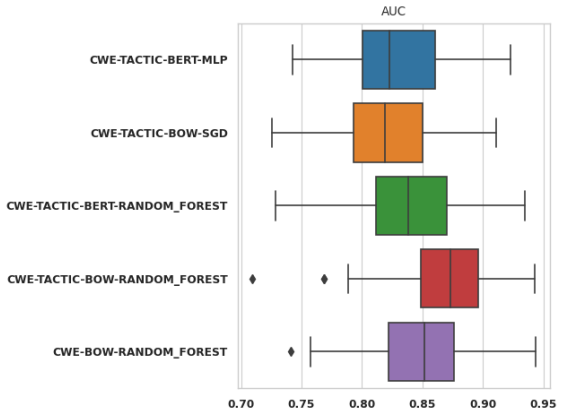
For all combinations we measure *performance* with: *a)* Error (1.0 - accuracy), *b)* AUC, *c)* F1-score. We perform 100 independent trials with different 70-30 train-test data splits. The data for each trial has 314 exemplars with 50-50 class balance (from under-sampling the majority class).

Results & Discussion. Figure 2 and Table 1 show results from predicting Technique-CAPEC with data from *BRON*. We see that using data from *BRON* can improve the performance. The experiment name indicates what data sources, features and classifier was used. For readability and space considerations we only show the top 5 based on F1-score. The significance tests reveal that CWE-TACTIC-BOW-RANDOM_FOREST has better performance on each of the measures (Error, AUC and F1). We measure if the differences in mean are statistically significant with a Wilcoxon-ranksum test, Bonferroni post-hoc adjustment and a p-value threshold of 0.05.

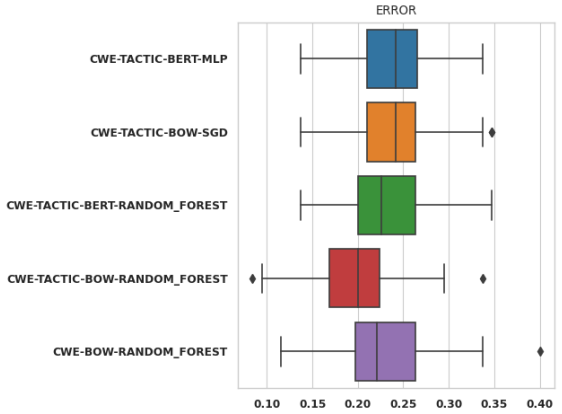
The feature selection experiments showed that in general more data improved performance, however the CAPEC_TECHNIQUE feature was not used by any of the top five experiments. In regard to feature representation BOW seem to work well, however BERT might improve with more cybersecurity specific training vocabulary and more data. The best classifiers were Random Forest. We see that using linked data sets from *BRON* can improve the performance. As expected, there is a difference between default classifier performance, and performance can hopefully be improved with parameter tuning. These experiments can be extended with more or other data.

3 DISCUSSION

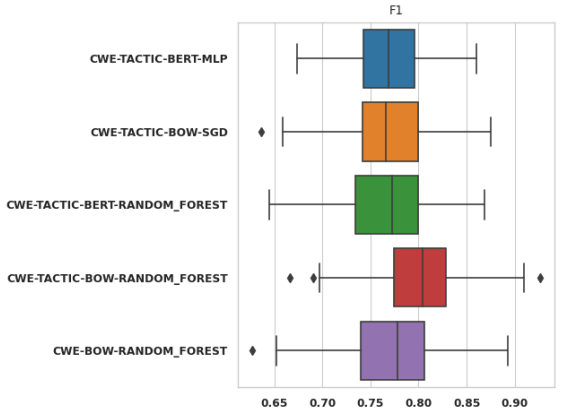
In this section we discuss how *BRON* can be used for information retrieval, modeling and simulation, and AI planning.



(a) AUC, higher is better



(b) Error, lower is better



(c) F1, higher is better

Figure 2: Box-plots of the performance measures for different Technique-CAPEC link predictions. The experiment name indicates what data sources, features and classifier was used (- is a separator of features, representation and classifier).

3.1 Information Retrieval

Example: Analyzing top 25 CWEs with BRON. The *2020 Common Weakness Enumeration (CWE) Top 25 Most Dangerous Software Weaknesses* list [17] is compiled by considering the prevalence and severity of CVEs and their associated *Weaknesses* (implemented by linking). These *Weaknesses* highlight "the most frequent and critical errors that can lead to serious vulnerabilities in software" [17]. For example, an attacker can exploit the vulnerabilities to take control of a system, obtain sensitive information, or cause a denial-of-service. The CWE Top 25 list is a resource that can provide insight into the most severe and current security weaknesses [17].

We use *BRON* to answer the following questions. What are the *Tactics*, *Techniques* and *Attack Patterns* linked to these *Weaknesses*? What commonalities in these features are there among the Top 25?

Our analysis of the Top 25 CWE [17] is summarized in Table 2. We observe 4 of 25 *Weaknesses* lack a presence in any *Attack Pattern*. Reflecting diversity in threats that could target the *Weaknesses*, there are 8 distinct *Tactics* associated with the Top 25. In terms of commonalities, the most frequent *Tactics* associated with the Top 25 weaknesses are `Defense Evasion, Privilege escalation, Discovery`. Only 2 *Techniques*, `T1148, T1562.003` occur more than once. The three most frequent *Attack Patterns* are `Using Slashes in Alternate Encoding, Exploiting Trust in Client, Command Line Execution through SQL Injection`.

The three most frequent *Vulnerabilities* (i.e. top 3) occur 3 times each and are `CVE-2017-7778, CVE-2016-10164, CVE-2016-7163`. The three most frequent *Affected Prod Conf*(s) are 3 different linux versions occurring 23, 24, 25 times respectively. We also analyzed the *Weakness* text descriptions with a frequency analysis of unigrams and bigrams. `Buffer Overflow` emerged as a common *Attack Pattern* bigram.

We note that not all weakness are linked with the same frequency to *Attack Patterns*, *Techniques* and *Tactics*. The ambiguity of this finding prompts: is absent data due to non-existence or being unreported? In addition, each of the source datasets has some bias. E.g. ATT&CK is from APT groups and include only common tactics and techniques. CWE and CVE include unexploited software vulnerabilities. BRON can help make sense of connected data before using them for AI/ML.

3.2 Modeling and Simulation

BRON can also be used for modeling and simulation (ModSim). Related works in cybersecurity use ModSim to conduct sensitivity analysis of network vulnerabilities and threats, or to investigate dynamics between threat and defense adaptations. The latter class of works intersects with studies of the coevolution of attacks and defenses[20]. While it does not use *BRON*, [13] models the reconnaissance stage behavior of an APT and the deceptive cloaking of a software-defined network. It simulates the behaviors coevolving through using feedback to adapt after engagements where a reconnaissance scan tries to operates within a defensive overlay.

A *BRON*-based example is EvoAPT[23]. This evolutionary algorithm system incorporates known threats and vulnerabilities from *BRON* into a stylized "competition" that pits cyber *Attack Patterns* against *mitigations*. The outcome of a competition is quantified using the Common Vulnerability Scoring System - CVSS, values

Table 2: Top 25 CWE [17]. *APC* is *Affected Product Configuration.*

CWE ID	Name	*#Tactics*	*#Techniques*	*#Attack Patterns*	*#Vulnerabilities*	Sum CVSS	Ave CVSS	*#APC*
CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	0	0	6	12,629	57,890	4.58	43,013
CWE-787	Out-of-bounds Write	0	0	0	1,159	8,667	7.48	1,499
CWE-20	Improper Input Validation	1	3	51	7,820	49,995	6.39	44,399
CWE-125	Out-of-bounds Read	0	0	2	2,172	13,549	6.24	2,029
CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	0	0	12	10,449	81,125	7.76	39,673
CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	0	0	6	5,477	41,309	7.54	14,685
CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	2	14	58	6,824	32,813	4.81	29,874
CWE-416	Use After Free	0	0	0	1,187	8,742	7.37	1,636
CWE-352	Cross-Site Request Forgery (CSRF)	0	0	4	2,377	16,319	6.87	11,646
CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	0	0	5	724	6,131	8.47	3,325
CWE-190	Integer Overflow or Wraparound	0	0	1	1,218	8,268	6.79	2,129
CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	0	0	5	2,964	18,684	6.3	14,368
CWE-476	NULL Pointer Dereference	0	0	0	1,019	5,994	5.88	2,342
CWE-287	Improper Authentication	4	3	10	1,654	11,453	6.92	13,061
CWE-434	Unrestricted Upload of File with Dangerous Type	0	0	1	562	4,315	7.68	1,370
CWE-732	Incorrect Permission Assignment for Critical Resource	0	1	11	427	2,654	6.22	1,151
CWE-94	Improper Control of Generation of Code ('Code Injection')	0	0	3	2,287	17,683	7.73	12,666
CWE-522	Insufficiently Protected Credentials	5	15	9	277	1,548	5.59	923
CWE-611	Improper Restriction of XML External Entity Reference	0	0	1	488	3,381	6.93	1,985
CWE-798	Use of Hard-coded Credentials	0	0	2	244	1,919	7.87	543
CWE-502	Deserialization of Untrusted Data	0	0	1	387	3,151	8.14	1,580
CWE-269	Improper Privilege Management	0	0	3	1,095	7,421	6.78	3,770
CWE-400	Uncontrolled Resource Consumption	0	0	3	728	4,459	6.13	4,303
CWE-306	Missing Authentication for Critical Function	0	0	4	112	793	7.09	504
CWE-862	Missing Authorization	0	0	0	190	1,162	6.12	527

within *BRON*. Variations of *Attack Pattern*s within the simulation are drawn from *BRON*. Mitigations take two forms: software updates or monitoring, and the software that is mitigated is identified by drawing from *BRON*'s entries from the CVE database. Three abstract models of population-level dynamics where APTs interact with defenses are aligned with three competitive, coevolutionary algorithm variants that use the competition. A comparative study shows that the way a defensive population preferentially acts, e.g. shifting to mitigating recent attack patterns, results in different evolutionary outcomes, expressed as different dominant attack patterns and mitigations.

We foresee *BRON* supporting other ModSim environments and studies. We anticipate that it will be plumbed for its APT behavioral structure, its connective structure, and text, offering further possible elaboration of modeled APT behaviors.

3.3 Planning

BRON can be incorporated into traditional artificial intelligence (AI) planning. One use case is driven by a need for automated red-teams which attack a system to gauge its defensive capacity or the competence of its security team [22]. The attacks can be plans derived by planners. The planner, itself, requires structured threat data and guidance on how to make domain-specific adaptations.

One close example is [22]. This system utilizes a complex knowledge base which references APT information from ATT&CK. Another example, that specifically incorporates *BRON* is, Attack Planner [18]. It is a computational vulnerability analysis system that outputs multistage attack model trees that achieve a desired goal on a desired system resource. Attack Planner generates attack graphs to achieve different goals, based on already known tactics and techniques. In order to incorporate ATT&CK and CVE, *BRON* was used via an interface between *BRON* 's graph representation of this data and the Attack Planner. ATT&CK and CVE categorize and organize all stages of an attack campaign at varying levels of depth starting from an overarching goal to down to specific exploits on a specific version of an operating system. By using *BRON* to link the specific exploits with their parent goals, the Attack Planner is able to generate plans with higher detail.

4 SUMMARY AND FUTURE WORK

We have demonstrated and discussed how *BRON*, a collated information dataset supports ML and AI at the behavioral level. Uses of *BRON* include information retrieval, pattern inference, modeling and simulation and AI-based attack planning.

The inference could be improved by tuning the feature representation and classifiers. *BRON* could be enhanced with additional behavioral knowledge, from timely sources such as threat reports. It could also be the basis of an open challenge within a security, knowledge discovery or applied ML workshop. E.g. extend inference or formulate more supervised learning problems around missing data.

Acknowledgments. This material is based upon work supported by the DARPA Advanced Research Project Agency (DARPA) and Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under Contract No. N66001-18-C-4036

REFERENCES

[1] Neda AfzaliSeresht, Yuan Miao, Qing Liu, Assefa Teshome, and Wenjie Ye. 2020. Investigating cyber alerts with graph-based analytics and narrative visualization. In *2020 24th International Conference Information Visualisation (IV)*. IEEE, 521–529.

[2] ALFA Group. 2021. BRON. http://bron.alfa.csail.mit.edu

[3] ALFA Group. 2021. BRON ML repository. http://github.com/ALFA-group/BRON-ML/tree/1.0

[4] ALFA Group. 2021. BRON repository. https://github.com/ALFA-group/BRON/tree/2.1

[5] Frederico Araujo, Dhilung Kirat, Xiaokui Shu, Teryl Taylor, and Jiyong Jang. 2021. Evidential Cyber Threat Hunting. arXiv:2104.10319 [cs.CR]

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Neil Dhir, Henrique Hoeltgebaum, Niall Adams, Mark Briers, Anthony Burke, and Paul Jones. 2021. Prospective Artificial Intelligence Approaches for Active Cyber Defence. arXiv:2104.09981 [cs.CR]

[8] Aviad Elitzur, Rami Puzis, and Polina Zilberman. 2019. Attack Hypothesis Generation. In *2019 European Intelligence and Security Informatics Conference (EISIC)*. IEEE, 40–47.

[9] Gregory Falco, Arun Viswanathan, Carlos Caldera, and Howard Shrobe. 2018. A master attack methodology for an AI-based automated attack planner for smart cities. *IEEE Access* 6 (2018), 48360–48373.

[10] Ibrahim Ghafir, Mohammad Hammoudeh, Vaclav Prenosil, Liangxiu Han, Robert Hegarty, Khaled Rabie, and Francisco J Aparicio-Navarro. 2018. Detection of advanced persistent threat using machine-learning correlation analysis. *Future Generation Computer Systems* 89 (2018), 349–359.

[11] Erik Hemberg, Jonathan Kelly, Michal Shlapentokh-Rothman, Bryn Reinstadler, Katherine Xu, Nick Rutar, and Una-May O'Reilly. 2020. BRON–Linking Attack Tactics, Techniques, and Patterns with Defensive Weaknesses, Vulnerabilities and Affected Platform Configurations. *arXiv preprint arXiv:2010.00533* (2020).

[12] Ghaith Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. Ttpdrill: Automatic and accurate extraction of threat actions from unstructured text of cti sources. In *Proceedings of the 33rd Annual Computer Security Applications Conference*. 103–115.

[13] Jonathan Kelly, Michael DeLaus, Erik Hemberg, and Una-May O'Reilly. 2019. Adversarially adapting deceptive views and reconnaissance scans on a software defined network. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 49–54.

[14] MITRE. [n.d.]. ATT&CK Matrix for Enterprise. https://attack.mitre.org/ https://attack.mitre.org/.

[15] MITRE. [n.d.]. Common Attack Pattern Enumeration and Classification. https://capec.mitre.org/ https://capec.mitre.org/.

[16] MITRE. [n.d.]. Common Weakness Enumeration. https://cwe.mitre.org/ https://cwe.mitre.org/.

[17] MITRE. [n.d.]. Top 25 CWE. https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html.

[18] Sam Nguyen. 2020. *Automated attack tree generation and evaluation: systemization of knowledge*. Master's thesis. Massachusetts Institute of Technology.

[19] NIST. [n.d.]. National Vulnerability Databased. https://nvd.nist.gov https://nvd.nist.gov.

[20] Una-May O'Reilly, Jamal Toutouh, Marcos Pertierra, Daniel Prado Sanchez, Dennis Garcia, Anthony Erb Luogo, Jonathan Kelly, and Erik Hemberg. 2020. Adversarial genetic programming for cyber security: A rising application domain where GP matters. *Genetic Programming and Evolvable Machines* 21, 1 (2020), 219–250.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[22] Bryn Marie Reinstadler. 2021. *AI Attack Planning for Emulated Networks*. Master's thesis. Massachusetts Institute of Technology.

[23] Michal Shlapentokh-Rothman, Avital Baral, Erik Hemberg, and Una-May O'Reilly. 2021. Coevolutionary Modeling of Cyber Attack Patterns and Mitigations Using Public Datasets. In *Proceedings of the Genetic and Evolutionary Computation Conference*.