Computing in Science and Engineering, 20 (6):54-65, November/December 2018

Evidence-based Detection of Advanced Persistent Threats

Gheorghe Tecuci
Dorin Marcu
Steven Meckl
Mihai Boicu

Learning Agents Center,

Volgenau School of Engineering,

George Mason University

This paper presents an approach to the automation of Cybersecurity Operations Centers with cognitive assistants that capture and automatically apply the expertise employed by cybersecurity analysts when they investigate Advanced Persistent Threats. The goal is to significantly increase the probability of detecting intrusion activity while drastically reducing the workload of the operators.

Despite the tens of billions of dollars spent on cybersecurity every year, the number of successful attacks has increased steadily year over year, due to the fast evolution of attacker methodology.

Cyber defense is done in a cybersecurity operations center (CSOC) where analysts monitor alerts and log data from available information sources, each having differing levels of credibility, and use them to make a determination about the presence or absence of intrusion activity.[1] However, because a single alert alone does not provide sufficient evidence that an incident has occurred, and modern detection technologies are error-prone, each alert must be carefully examined and validated by a human analyst.[1] Both missed detections and false positives are costly, resulting in real damage to an organization or wasted incident response resources respectively. In a large enterprise, thousands of alerts can be reported daily. Therefore, even sensors with a false positive rate of one percent may have enough missed detections and false positives to be unmanageable by even mature CSOCs.[2]

Among the most sophisticated cyber threats faced by an organization are those known as Advanced Persistent Threats (APTs). These are computer network exploitation groups (many of them state-sponsored) that leverage superior resources, knowledge, and tactics to gain and maintain access to targeted networks and adapt to defenders' efforts to resist them. [1,3] FireEye/Mandiant has published reports on more than 37 APT groups since 2013, naming them simply APT1 through APT37 (FireEye 2015).[4]

APT1 is the name given by Mandiant[5] to a group of APT actors, attributed to China's People's Liberation Army unit 61398, who led a years-long campaign of cyber espionage dating back to at least 2004. APT1 is known for a regimented approach to computer intrusion activity. An

APT1 intrusion typically consists of the following phases: (1) gain access to a network by sending fraudulent, malicious email messages to specific users (spearphishing); (2) use multiple types of backdoor programs to maintain presence and provide remote connectivity to the target network; (3) use a collection of command-and-control (C2) servers to obfuscate the source of their attacks; (4) escalate privileges and acquire legitimate login credentials to access network resources; (5) move laterally within the target network using legitimate credentials to gain redundant points of presence and identify information of interest; and (6) exfiltrate targeted information through their C2 infrastructure.[5]

Automatic security systems deployed today do not reliably detect APTs because this requires reasoning over a large set of weak indicators. Therefore the current practice relies on experienced cyber analysts to manually investigate such indicators. However, as discussed above, the large and increasing number of alerts and the time required for their manual analysis creates a very complex, expensive, and non-sustainable security environment for network defense organizations.

The resource strain is exacerbated by the fact that intrusion detection systems have a relatively high false positive rate. A network connection with 100 Mbps sustained throughput transmits approximately 1.8 billion packets per day. Even an IDS with a false positive rate of one percent could result in an overwhelming number of security events for CSOC analysts to investigate. A skilled security analyst can investigate 10-20 incidents per day[2] and the number of incidents a CSOC can investigate scales linearly with the number of analysts. Because each alert must be investigated as if it were a true positive, investigation of false positives can be very expensive to an organization.

To alleviate these problems, we research the development of cognitive assistants capable of being taught by expert cybersecurity analysts how to investigate potential APT intrusions. Once taught, these agents can automatically investigate alerts, leading to a significant increase in the probability of detecting intrusion activity, and a drastic reduction in the workload of the operators.

To teach such an agent, the expert cyber analyst follows a systematic evidence-based reasoning approach to APT detection, grounded in the scientific method. This starts with alerts that lead to the generation of alternative hypotheses that may explain them, some representing intrusion activity while others representing legitimate activities (called false positives). Each of these hypotheses is used to guide the search for evidence to confirm it, and the found evidence is used to determine the probability of each hypothesis.[6]

The agent, guided by the expert, learns general rules from examples of such alerts, alternative explanatory hypotheses and their analyses, that allow it to investigate similar alerts in a similar way. These rules are further improved based on such investigations and their critique by the expert, until the agent can accurately simulate the reasoning of the expert.

We will overview and illustrate this approach, also providing information on the developed prototype system. Then we will discuss the applicability of this approach to areas other than cybersecurity.

COGNITIVE ASSISTANTS FOR APT DETECTION

Figure 1 is an overview of the process of training and using the cognitive assistants for advanced persistent threat detection. An expert cyber analyst teaches a learning agent shell, through examples and explanations, how to generate and assess both APT intrusion hypotheses and false positive hypotheses.[7] The trained learning agent is then customized into specialized autonomous collaborative agents for a specific CSOC.

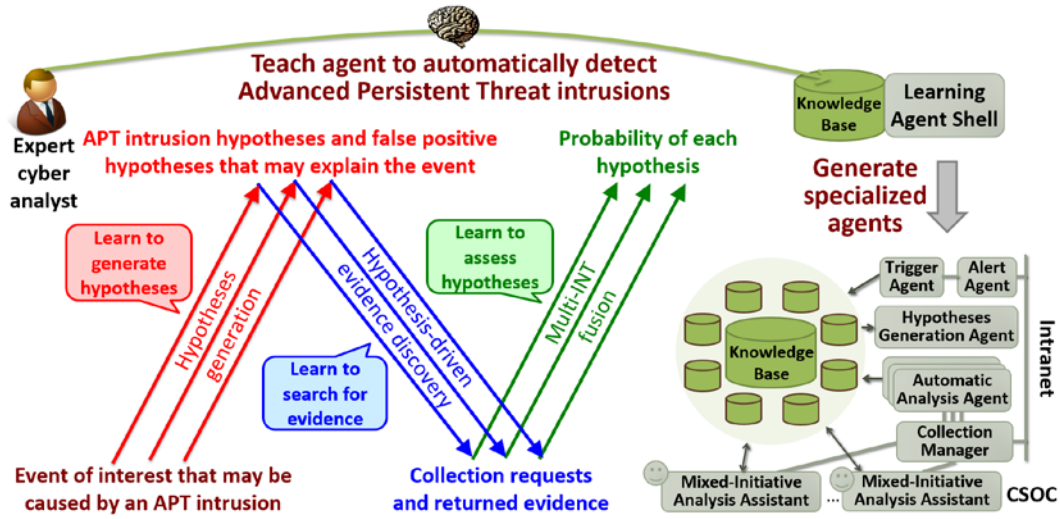

Figure 1. Overview of evidence-based detection of advanced persistent threats.

The left part of Figure 1 illustrates the process of teaching the learning agent shell. An expert analyst teaches the agent how to detect a specific APT intrusion (e.g., by the Auriga malware of APT1)[5], by illustrating all the associated reasoning steps. First, the expert analyst specifies an event of interest that may be caused by this APT1 intrusion. This event is called trigger and is the kind of alert generated by a network intrusion detection system (IDS), such as BRO.[8] Then the expert defines alternative hypotheses that may explain this alert. Some of these hypotheses are APT1 intrusion hypotheses, but others are false positive hypotheses. Each of these hypotheses has to be analyzed, based on evidence, to estimate its probability. Each hypothesis is decomposed into simpler hypotheses that more clearly point to the evidence that needs to be collected to assess them. Evidence is collected and used to assess the probabilities of the generated hypotheses.

From the above reasoning steps the agent, with the help of the cyber expert, learns different types of rules, including: rules to generate hypotheses from triggers issued by IDS systems; rules to search for evidence that is relevant to the generated hypotheses; and rules to assess the probabilities of the hypotheses based on the collected evidence.

Once the learning agent shell has been taught to detect APT intrusions, it is used to generate several autonomous agents, each specialized for a specific phase of APT intrusion detection in a CSOC, as illustrated in the right-hand side of Figure 1.

The Alert Agent receives alerts from various sources, such as the network IDS BRO, creates a JSON representation of each alert, and sends it to the Trigger Agent. For each alert, the Trigger Agent creates a new knowledge base (stored on disk in XML) in which it represents the alert as an ontology fragment, instantiates the corresponding trigger hypothesis, and then places the new knowledge base in the hypothesis generation queue. The Hypothesis Generation Agent consumes knowledge bases from this queue. In each knowledge base it abductively generates hypotheses that connect the trigger hypothesis with intrusion hypotheses, and places these knowledge bases into the hypotheses analysis queue. Each of these knowledge bases is retrieved by an Automatic Analysis agent which decomposes the intrusion hypotheses as much as possible, generates evidence collection requests and places the knowledge base into the evidence collection queue for the Collection Manager. The Collection Manager invokes Collection Agents, receives the evidence found by them, represents it into the corresponding knowledge base and then places the knowledge base in the queue for Automatic Analysis Agents, to be used to further develop the analyses, in an "Analysis – Collection – Analysis" loop, until the analyses of all the intrusion hypotheses are completed. The cyber analyst (operator) is alerted when any intrusion hypothesis is likely and can investigate the analysis of that hypothesis by using the Mixed-initiative Analysis Assistant. All these agents are developed using Java and communicate between them through a Repository Server using custom Java RMI messages.

Our prototype system operates in a simulated network running inside of a VMWare vSphere system. For the system to work, it requires one virtual machine each to run Elasticsearch, Google Rapid Response, the Collection Manager, the Alert Agent, and the other reasoning agents. As the size of the protected network scales, additional servers can be added to support the additional volume.

To be able to perform the learning and reasoning required by APT detection, the cognitive assistants synergistically integrate several knowledge representation, reasoning, and learning methods.

The hypothesis generation step in Figure 1 involves *abductive* reasoning that shows that something is *possibly* true. The hypothesis-driven evidence discovery involves *deductive* reasoning which shows that something is *necessarily* true, and the multi-INT fusion of evidence involves *inductive* inference which shows that something is *probably* true.

The cognitive assistants are knowledge-based systems that reason using the knowledge learned from the expert cyber analysts. This knowledge is represented in an ontology and several types of rules. The ontology represents the concepts and instances from the cyber domain, together with their relationships.

ILLUSTRATION OF APT DETECTION

We will illustrate the evidence-based detection of APTs by following the three-phase process from Figure 1.

Evidence in search of hypotheses

Figure 2 illustrates the hypothesis generation process. The bottom left part shows an alert issued by the BRO intrusion detection system.[8] The question is: What hypotheses may explain this alert? We call the process of answering this question, evidence in search of hypotheses.

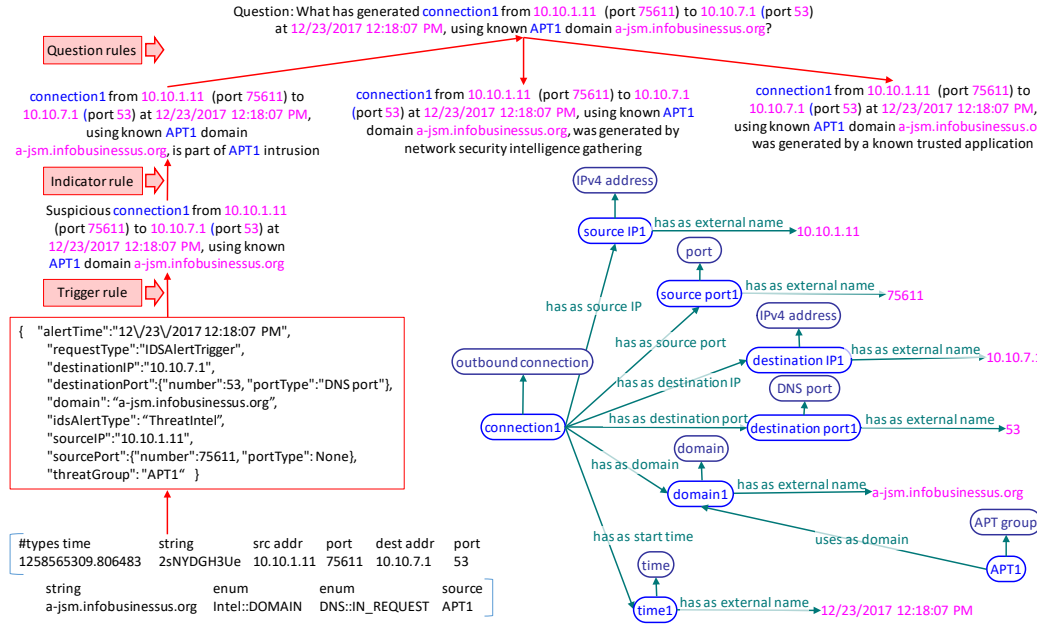

Figure 2. Automatic hypotheses generation from a BRO alert.

First, the Alert Agent (shown in the middle right of Figure 1) generates a JSON representation of the alert (shown above the alert in Figure 2). This is a "trigger" for the Trigger Agent that uses a

trigger rule to represent it into the agent's ontology and generates a basic hypothesis, both shown in Figure 2. The basic hypothesis (shown above the JSON representation) is the following one:

> Suspicious connection1 from 10.10.1.11 (port 75611) to 10.10.7.1 (port 53) at 12/23/2017 12:18:07 PM, using known APT1 domain a-jsm.infobusinessus.org

The Hypotheses Generation Agent employs an indicator rule to abductively generate the following intrusion hypothesis from the basic hypothesis:

> connection1 from 10.10.1.11 (port 75611) to 10.10.7.1 (port 53) at 12/23/2017 12:18:07 PM, using known APT1 domain a-jsm.infobusinessus.org, is part of APT1 intrusion

Then it uses a question rule to generate the question which has the above hypothesis as a possible answer:

> What has generated connection1 from 10.10.1.11 (port 75611) to 10.10.7.1 (port 53) at 12/23/2017 12:18:07 PM, using known APT1 domain a-jsm.infobusinessus.org?

After that the agent uses two other question rules to generate the other possible answers of the above question:

> connection1 from 10.10.1.11 (port 11234) to 8.8.8.8 (port 53) at 05/15/2017 16:23 GMT, using known APT1 domain a-jsm.infobusinessus.org, was generated by network security intelligence gathering

> connection1 from 10.10.1.11 (port 11234) to 8.8.8.8 (port 53) at 05/15/2017 16:23 GMT, using known APT1 domain a-jsm.infobusinessus.org, was generated by a known trusted application

These are false positive hypotheses that also may explain the BRO alert.

Hypotheses in search of evidence

As illustrated in the previous section, three hypotheses may explain the alert generated by the BRO IDS, an intrusion hypothesis and two false positive hypotheses (representing legitimate activities). One would need additional evidence to assess the probability of each of these hypotheses, and thus determine whether there is an intrusion or not. The strategy is to put each of the generated alternative hypotheses to work guiding the collection of relevant evidence. The question is: Assuming this hypothesis is true, what evidence should be observable? We call this process, shown in the middle of Figure 1, hypothesis in search of evidence.

The automatic Analysis Agent employs hypothesis analysis rules to decompose the three hypotheses from the top part of Figure 2, as much as possible, down to the level of specific evidence collection requests. Figure 3 illustrates the decomposition of the first hypothesis which is the intrusion hypothesis. There are two main indicators of this hypothesis. The left branch investigates the first indicator, whether connection1 involves an APT1 command and control server. In order to make this determination, it needs more information about the APT1 domain a-jsm.infobusinessus.org. Its information needs are expressed through specific search requests to be processed by the Collection Manager, such as the following one:

> **Search** for the IP address mapped to domain a-jsm.infobusinessus.org at time 12/23/2017 12:18:07 PM

The right branch of the decomposition in Figure 3 investigates the other indicator of the top hypothesis, whether the program that made connection1 is an APT1 malware. To investigate this further, however, the agent needs to identify this program, and therefore it formulates another information request for the Collection Manager:

> **Search** the computer 10.10.1.11 for the program that made connection1 using port 75611 to communicate with 10.10.7.1 on port 53 at 12/23/2017 12:18:07 PM

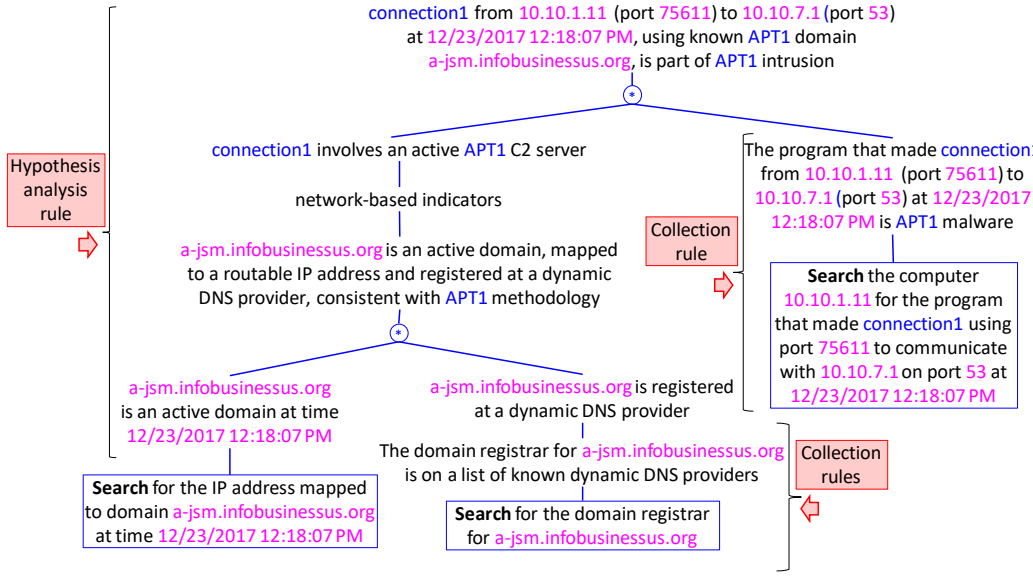

Figure 3. Automatic hypothesis analysis and evidence search.

The Collection Manager invokes specialized collection agents to search for this information on the network and on the host computer.

The Collection Manager is the main integration point between the analysis agents and the CSOC infrastructure. The analysis agents know what information is needed to expand their analyses, but the search requests are in abstract form. They are not tied to specific data sources. The primary function of the Collection Manager is translating high-level (abstract) search requests into specific API calls to host and network agents, determining to which such agent to send the search request on behalf of the analysis agents, and wrapping calls to specific search agents with a JSON API. Results returned from a specific search agent to the Collection Manager are then converted into evidence and added to the knowledge bases of the analysis agents.

To illustrate the Collection Manger's operation, consider the Search from the bottom-left side of Figure 3. This is first translated into a JSON representation. Processing this representation leads to the invocation of the ***GetDomainIPResolutionRequest*** search function**.** This is processed by a collection agent that returns the result also as a JSON representation. This representation is translated into ontology fragments that are added to the system's ontology. One fragment is the found evidence item, E1 evidence, with credibility L11 (certain). This replaces the search request in the analysis tree, confirming the hypothesis above it:

> It is certain (L11) that a-jsm.infobusinessus.org is an active domain at time 12/23/2017 12:18:07 PM

Evidence-based assessment of hypotheses

Figure 4 shows how the analysis in Figure 3 was refined after the automatic analysis agent has received the results of the three search requests from the bottom of Figure 3. Notice that the two searches from the left branch of the tree returned evidence items that enabled the analysis agent to assess the probabilities of all the hypotheses on that branch. The search on the right branch returned the name of the program that made connection1. This enabled the analysis agent to further decompose the hypothesis on that branch, leading to other evidence collection requests, including the following one:

> **Check** whether Auriga Registry key HKEY_LOCAL_MACHINE\SOFTWARE\riodrv32\TEMP is present on the host computer 10.10.1.11

Let us now briefly review the process of assessing a hypothesis based on evidence, by explaining the assessments made on the left branch of Figure 4.

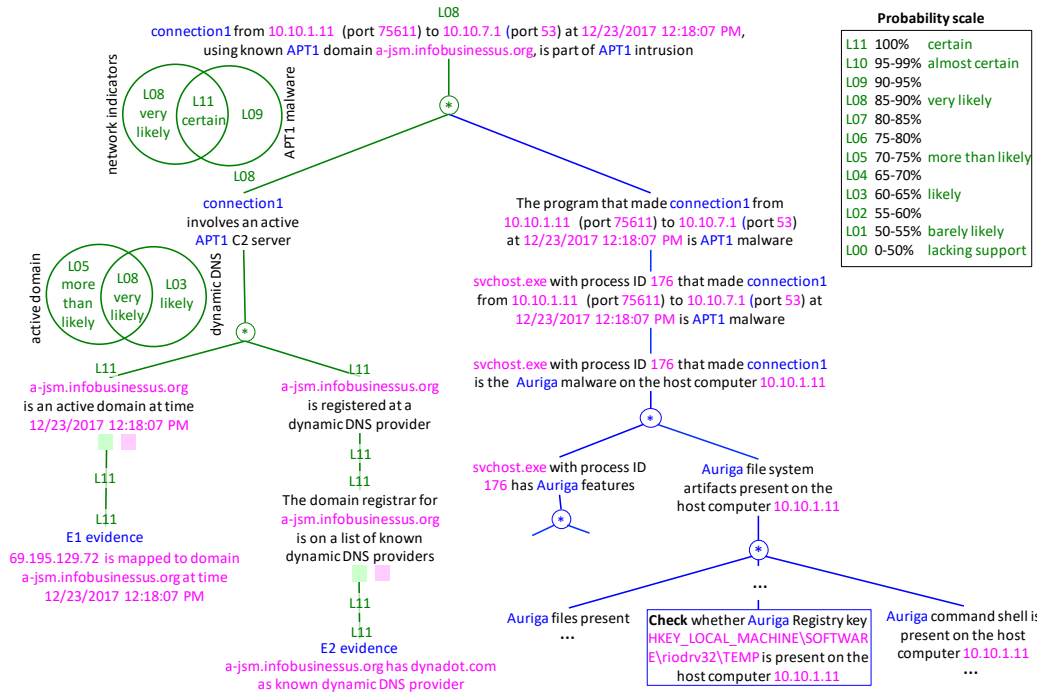

Figure 4. Integrated analysis, collection, and assessment.

Notice that the two search nodes in Figure 3 were replaced with the evidence items E1 and E2 in Figure 4. Each of these evidence items favors the truthfulness of the corresponding hypothesis and therefore appears under the left (green) box. If an evidence item disfavors the truthfulness of the hypothesis it is relevant to, it is represented under the right (red) square.

In general, an evidence item E is characterized by its credibility (i.e., the probability that E is true), and by its relevance to the corresponding hypothesis H (i.e., the probability that the hypothesis H is true, assuming that the evidence E is true).[6]

The agents employ an intuitive system of Baconian probabilities [6,9,10] with Fuzzy qualifiers [6,10,11] that are shown in the table from the top-right of Figure 4. Notice that this is a symbolic probability scale, from L00 (corresponding to the interval 0-50%) to L11 (corresponding to 100%). Each of the intermediary values corresponds to a 5-point interval, such as L02 that corresponds to the probability interval 55-60%. Notice also that some of the probability intervals are associated with familiar phrases, such as L03 (60-65% likely) and L10 (95-99% almost certain).

Once the evidence items relevant to a hypothesis are found, the probability of the hypothesis is determined based on the credibility and relevance of these evidence items. Then the probability of each of the upper level hypothesis is determined based on the probabilities of its sub-hypotheses, based on the structure of the argumentation: OR, AND, or * (combined indicator). In Figure 4 there are only combined indicator structures.

Let's follow the assessment of the probabilities in the left branch of Figure 4, from bottom to top. The credibility of E1 is L11 (100%, certain) and its relevance is also L11. Therefore, the probability of the hypothesis above it is also L11 (the minimum of E1's credibility and relevance). Let's now consider the hypothesis above this hypothesis: connection1 involves an active APT1 C2 server. The two sub-hypotheses of this hypothesis are indicators for it. According to the combined indicator (represented as two overlapping circles in the middle left of Figure 4), if only the left sub-hypothesis is true, the hypothesis above it is L05 (70-75% more than likely). Similarly, if only the right sub-hypothesis is true, the hypothesis above it is L03 (60-65% likely).

In this example, however, both sub-hypotheses are true and therefore the hypothesis above them is L08 (85-90% very likely).

Further up in the argumentation, the two sub-hypotheses of the top hypothesis are also indicators for it. In this case, however, it has only been determined that the probability of the left indicator is L08. The automatic analysis cannot infer the presence of the second indicator for the top-level hypothesis in Figure 4. Therefore, it has to determine the probability of this top hypothesis based only on the left indicator. The relevance of this indicator alone is L08 (see the description of the combined indicator in the top left of Figure 4). Therefore, at this point, the analysis agent assesses the probability of the top hypothesis as L08 (the minimum between the probability of the left sub-hypothesis and its relevance).

After the results of the new evidence collection requests are returned by the Collection Manager, the automatic analysis agent will refine the analysis, and so on, until no further refinements are possible. Based on further evidence collection and analysis, the analysis agent determines that the probability of the right indicator of the top hypothesis is L08. Now the relevance of the combined indicator is L11 (certain). However, the probability of the top hypothesis is still L08 (the minimum between the relevance of the combined indicator and the probabilities of the two indicators).

Our prototype has been tested in a simple virtual network. The scale required to detect an APT is dependent on available threat intelligence. For a well-known APT group such as APT1, with 17 stage 1 malware programs, 27 stage 2, and small number of additional tools, it is estimated a rule set numbering less than 100 would be sufficient to detect all known APT1 malware as well as unknown but predictable variants of APT1 malware.

ONTOLOGY FOR APT DETECTION

The ontology is a main component of the knowledge representation of this system. During the process of teaching the learning agent shell, illustrated in Figure 1, the knowledge engineer and the cyber analyst need also to develop its ontology. When they model the detection process for a specific malware, they also identify the ontological knowledge needed by the agent to automatically perform the same analysis. This results in an ontology specification. An initial ontology is then developed based on this ontology specification, existing cyber ontologies[13], CSOC's network configuration, and threat intelligence. It will contain general network concepts, concepts and instances describing the local network, general threat knowledge, and alert knowledge.

The ontology language is an extension of RDFS[13,14] with additional features to facilitate learning and evidence representation.[7] The bottom right part of Figure 2 shows the ontological representation of a trigger.

The learning agent shell contains several tools for ontology development. For example, the bottom left of figure 5 shows the interface of the Association Browser that displays an entity (the Auriga malware) and its features. At its right is the interface of the Object Browser that can be used to define concepts, instances, and their features. There is also a Feature Browser and a Hierarchical Browser. The top part of Figure 5 shows the interfaces of two agents that have been presented previously, the Hypotheses Generation Agent and the Hypotheses Analysis Agent.

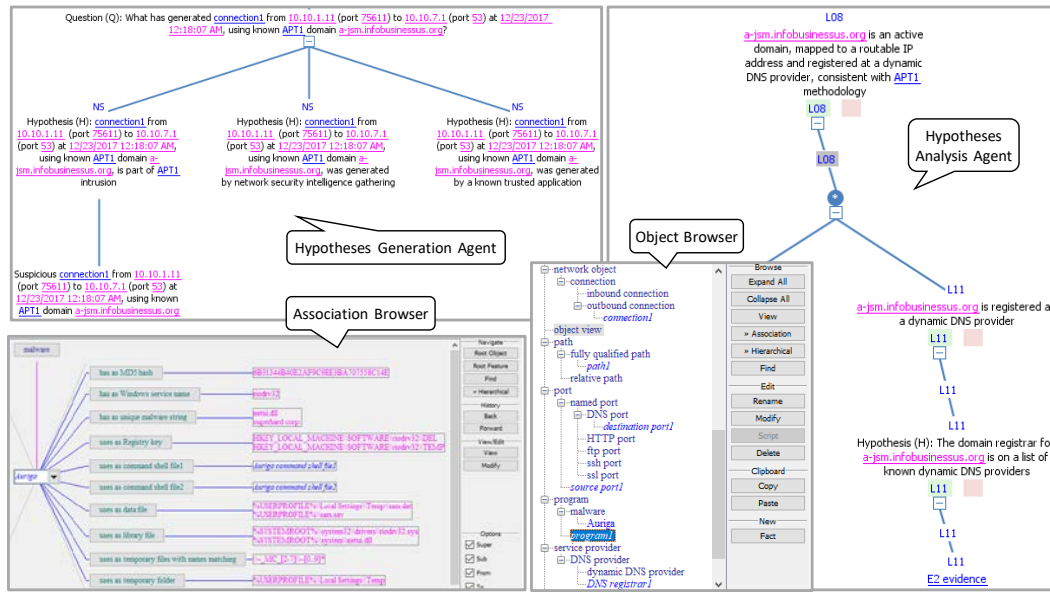

Figure 5. Sample interfaces of the system.

MULTISTRATEGY RULE LEARNING

A critical feature of the presented approach is the ability to rapidly acquire cybersecurity expertise directly from subject matter experts.

The learning agent shell is taught directly by a cybersecurity expert, with limited knowledge engineering assistance, in a way that is similar to how one would teach a student or a novice cybersecurity analyst, through specific examples of reasoning and explanations, and through the supervision and correction of the agent's reasoning, when it applies the learned rules for APT detection. The learning strategy is an extension and generalization of the Disciple multistrategy learning approach, which integrates *learning from examples*, *learning from explanations*, and *learning by analogy and experimentation*, in a mixed-initiative interaction with the analyst.[7,15-20]

Consider the reasoning illustrated in Figure 3. This time, however, assume that we are in the phase of teaching the learning agent shell from Figure 1. The cyber security expert will show the agent the reasoning in Figure 3 and the agent will learn one hypothesis analysis rule and three collection rules from it. The learned hypothesis analysis rule is shown in Figure 6.

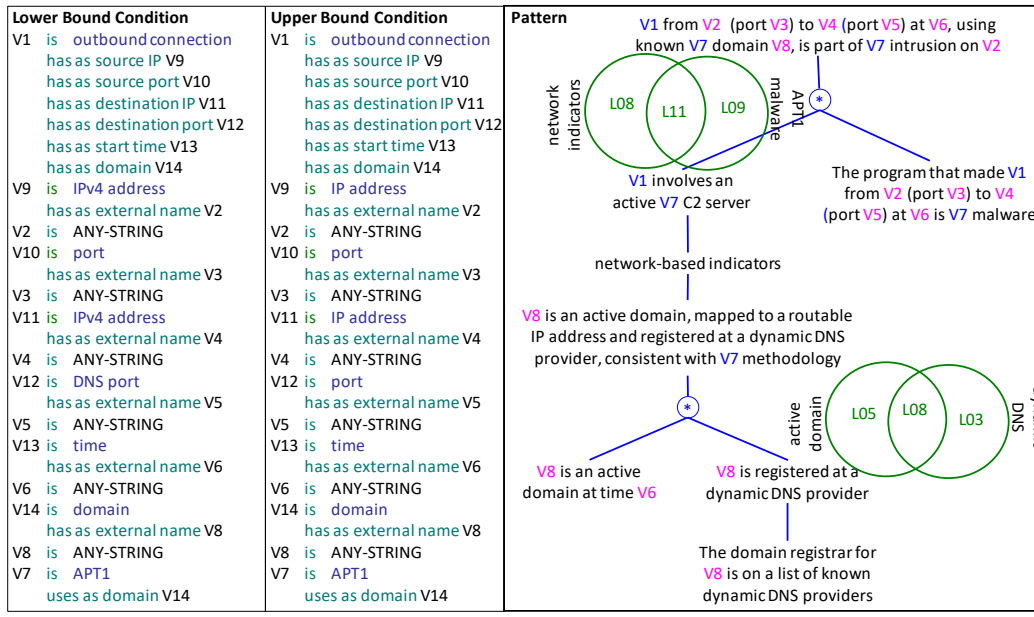

Figure 6. Partially learned hypothesis analysis rule.

The agent interacts with the expert to determine the important features of the instances from the argument fragment that assures this argumentation fragment is correct. These are called the explanations of the example and happen to be the features appearing in the description of the trigger shown in the bottom right of Figure 2.

Next the agent automatically generates the tree pattern from the right-hand side of Figure 6. The tree pattern is obtained by replacing each instance from the decomposition tree (e.g., connection1) with a variable (i.e., V1). The values for the combined indicators are specified by the cyber analyst. The agent also automatically generates the applicability condition of the learned pattern, shown in left side of Figure 6. Notice however that instead of a single applicability condition there is a lower bound condition and an upper bound condition. The lower bound condition corresponds to the minimal generalization of the example and its explanation, in the context of the current APT ontology that is used as a generalization hierarchy. The upper bound condition corresponds to the maximal generalization.

The learning agent uses the partially learned rules in reasoning. The reasoning fragments accepted by the expert as correct represent new positive examples of the rule. Those that are rejected represent negative examples. These examples and their explanations are used to refine the rule. In particular, positive examples lead to the generalization of the lower bound condition until it covers them. A negative example may either lead to the specialization of the upper bound condition until it no longer covers it, or to the learning of an except when condition, also with an upper bound and a lower bound. The except when condition should not be satisfied for the rule to be applicable. In time, the lower bound and the upper bound conditions converge toward one another and, possibly, to an exact applicability condition. The goal is to improve the applicability condition of the pattern so that it only generates correct argumentation fragments.

It is expected such a mixed-initiative learning method will lead to learning patterns with complex applicability conditions that accurately represent the knowledge of the cybersecurity expert, through a natural interaction with the expert.

At the same time with learning a hypothesis analysis rule, the agent may also extend the ontology with new concepts and features. For example, to explain to the agent why a generated argument is wrong, the expert may use a new concept or feature. As a result, the agent will add the concept or the feature definition in its APT ontology. Ontology refinement requires an adaptation of the previously learned patterns since the generalization hierarchy used to learn them has

changed. To cope with this issue, the agent will automatically regenerate the patterns in the context of the changed ontology. Notice that this is a powerful form of learning with an evolving representation language.

The other types of rules (trigger, indicator, question, and collection) are learned in a similar way.

The learning of the rules is much faster than their manual definition. Indeed, the cybersecurity expert only needs to provide an example of the rule (such as the one from the top part of Figure 3, without the Search instructions), and to select the correct explanations from those proposed by the agent. Based on this example and the selected explanations the leaning agent will generate the rule from Figure 6, by automatically determining the maximal and the minimal generalizations of the example, using the ontology as the generalization language. Refinement of a learned rule is also easier than manual correction because the expert cyber analyst needs only to indicate that an example generated by the agent is correct or, if it is not correct, to select the right explanation among those proposed by the agent.

Learning the rules using classical methods (such as decision trees, neural networks, or support vector machines) is not possible because they require a large number of labelled examples. Since APT intrusions are rare compared to less-sophisticated attacks, a sufficient number of labeled examples does not exist to train a classifier.

The manual definition of the rules is very complex, as the cybersecurity expert would need to identify both the applicability condition of the rule in terms of the concepts and features of the ontology (that is, the condition from the left hand side of Figure 6).

AUTOMATED APT DETECTION

Currently there is no automatic solution to the defense against APT threats. This is a complex and time consuming process performed by experienced analysts. However, given the large and increasing number of alerts to be investigated, this manual approach is non-sustainable.

We have described an automated approach and its prototype implementation that is characterized by several innovations.

One innovation is a systematic approach to APT detection based on the scientific method of hypotheses generation, evidence collection, and hypotheses testing. The corresponding workflow is a natural way for a cybersecurity analyst to conduct an investigation and teach a learning agent to automatically perform such an investigation.

Another innovation is the learning of different types of APT detection rules (trigger rules, indicator rules, question rules, collection rules, and analysis/synthesis rules), based on a single example of APT investigation, by adapting the Disciple approach to learning.[7] A typical inductive learning from examples approach cannot be applied in this case because it would require a large number of labelled examples which are not available.

Yet another innovation is the generation of specialized agents (trigger agent, hypotheses generation agent, hypotheses analysis agent) that can collaborate autonomously in performing an APT investigation of an alert, simulating the investigation that would have been performed by a cybersecurity analyst.

Also innovative is the employment of hypothesis-driven evidence collection performed by collection agents that are invoked when evidence for a certain elementary hypothesis is needed. This is important because, in cybersecurity, capturing evidence all the time is not feasible, due to the available storage capacity.

BEYOND APT DETECTION

The presented approach to the automation of the Cybersecurity Operations Centers is, in fact, an example of a more general process, automated evidence-based reasoning that may be applied to other domains.

Consider, for example, the national security area of Intelligence, Surveillance and Reconnaissance (ISR) where one objective is to continuously monitor and understand the situation in a certain area, predict the behavior and intent of the entities of interest, and identify threats. As a concrete example, consider the continuous monitoring of the ships through the Automatic Identification System (AIS), a tracking system which is extensively used in the maritime world for the exchange of navigational information between AIS-equipped terminals. In such a case, losing the AIS coverage for a certain ship would be an event of interest, or trigger. A hypothesis generation agent, like the one described in this paper, may automatically generate explanatory hypotheses such as, the ship's AIS equipment was turned-off intentionally to perform covert goods transfer with another ship, or to perform illegal phishing, or to avoid tracking by the pirates. Then, analysis agents would analyze each hypothesis, with the help of automated collection agents.

Another potential application of this approach would be for the automatic monitoring of the functioning of an industrial installation, such as a nuclear plant.

One could even imagine the application of this approach to the monitoring of patients in a hospital, or even the monitoring of the health status of each person.

What is common to all these potential applications is the need to automatically apply a scientific-based approach to events of interest that may or may not be caused by significant threats to the monitored entity.

CONCLUSION

We have presented an approach to the automation of Cybersecurity Operations Centers with cognitive assistants that capture and automatically apply the expertise employed by cybersecurity analysts when they investigate Advanced Persistent Threats. A first version of this approach is implemented and is under further development, particularly to improve its learning capabilities.

ACKNOWLEDGEMENTS

This research was performed in the Learning Agents Center and was sponsored by the Air Force Research Laboratory (AFRL) under contract number FA8750-17-C-0002, by other agencies of the U.S. Government, and by George Mason University.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

REFERENCES

1. Zimmerman C., *Ten Strategies of a World-Class Cybersecurity Operations Center*, MITRE Corporation, 2014.
2. Ponemon Institute, *The Cost of Insecure Endpoints*, June 2017. https://datasecurity.dell.com/wp-content/uploads/2017/09/ponemon-cost-of-insecure-endpoints.pdf
3. Meckl S., Tecuci G., Marcu D., Boicu M., Zaman A.B., "Collaborative Cognitive Assistants for Advanced Persistent Threat Detection," in *Proceedings of the 2017 AAAI Fall Symposium Series*, Technical Reports FS-17-01-FS-17-05, pp.171-178, AAAI Press, Palo Alto, CA.
4. FireEye, *APT30 and the Mechanics of a Long-running Cyber Espionage Operation*, FireEye, April, 2015.
5. Mandiant Intelligence, *APT1: Exposing one of China's cyber espionage units*, Mandiant.com, 2013.

6. Tecuci, G., Schum, D. A., Marcu, D., and Boicu, M., *Intelligence Analysis as Discovery of Evidence, Hypotheses, and Arguments: Connecting the Dots*, Cambridge University Press, 2016.
7. Tecuci G., Marcu D., Boicu M., Schum D.A., *Knowledge Engineering: Building Cognitive Assistants for Evidence-based Reasoning*, Cambridge University Press, 2016.
8. BRO, http://www.bro-ids.org, 2018.
9. Cohen L. J., *The Probable and the Provable*, Clarendon Press, Oxford, 1977.
10. Schum, D. A., *The Evidential Foundations of Probabilistic Reasoning*, Northwestern University Press, 2001.
11. Zadeh, L., "The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems," *Fuzzy Sets and Systems*, Vol. 11, pp. 199 – 227, 1983.
12. Obrst L., Chase P., and Markeloff R., Developing an Ontology of the Cyber Security Domain, *Proceedings of the 7th International Conference on Semantic Technologies for Intelligence, Defense, and Security*, George Mason University: CEUR, 2012. http://ceur-ws.org/Vol-966/
13. Allemang D. and Hendler J., *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and Owl*, Morgan Kaufmann Publishers, 2011.
14. W3C, http://www.w3.org/TR/rdf-schema/, 2004.
15. Tecuci G., Kodratoff Y. (eds.), *Machine Learning and Knowledge Acquisition: Integrated Approaches*, Academic Press, 1995.
16. Tecuci G., B*uilding Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies*, San Diego: Academic Press, 1998.
17. Boicu, M., Tecuci, G., Marcu, D., Bowman, M., Shyr, P., Ciucu, F., and Levcovici, C., "Disciple-COA: From Agent Programming to Agent Teaching," *Proceedings of the 27th International Conference on Machine Learning* (ICML), Stanford, California, Morgan Kaufman, http://lac.gmu.edu/publications/data/2000/2000_il-final.pdf
18. Tecuci G., Boicu M., Marcu D., Stanescu B., Boicu C., Comello J., Lopez A., Donlon J., Cleckner W., "Development and Deployment of a Disciple Agent for Center of Gravity Analysis," in *Proceedings of the Eighteenth National Conference of Artificial Intelligence and the Fourteenth Conference on Innovative Applications of Artificial Intelligence*, AAAI-02/IAAI-02, pp. 853-860, Edmonton, Alberta, Canada, AAAI Press/The MIT Press, 2002. http://lac.gmu.edu/publications/data/2002/dddacga.pdf
19. Tecuci, G., Boicu, M., Ayers, C., Cammons D., Personal Cognitive Assistants for Military Intelligence Analysis: Mixed-Initiative Learning, Tutoring, and Problem Solving. In *Proceedings of the 1st International Conference on Intelligence Analysis*, McLean, VA, 2005. http://lac.gmu.edu/publications/data/2005/Tecuci-Disciple-LTA.pdf
20. Tecuci G., Boicu M., Marcu D., Boicu C., and Barbulescu M., Disciple-LTA: Learning, Tutoring and Analytic Assistance, *Journal of Intelligence Community Research and Development*, 2008. http://lac.gmu.edu/publications/2008/Disciple-LTA08.pdf

ABOUT THE AUTHORS

Gheorghe Tecuci is Professor of Computer Science and Director of the Learning Agents Center at George Mason University, member of the Romanian Academy, and former chair of Artificial Intelligence at the U.S. Army War College. Email: tecuci@gmu.edu

Dorin Marcu is Associate Research Professor in the Learning Agents Center at George Mason University. Email: dmarcu@gmu.edu

Steve Meckl is a Director of Operations for Symantec Managed Security Services, Researcher in the Learning Agents Center, and Ph.D. student at George Mason University. Email: smeckl@gmu.edu

Mihai Boicu is Associate Professor of Information Sciences and Technology and Associate Director of the Learning Agents Center at George Mason University. Email: mboicu@gmu.edu