In Proc. of the 10th Int. Conf. on Semantic Technologies for Intelligence, Defense, and Security, Fairfax, VA, November 2015.

Towards an Operational Semantic Theory of Cyber Defense Against Advanced Persistent Threats

Steven Meckl, Gheorghe Tecuci, Mihai Boicu, Dorin Marcu
Learning Agents Center, Volgenau School of Engineering, George Mason University, Fairfax, VA 22030, USA
smeckl@masonlive.gmu.edu, teuci@gmu.edu, mboicu@gmu.edu, dmarcu@gmu.edu

Abstract— **This paper presents current work on developing an operational semantic theory of cyber defense against advanced persistent threats (APTs), which is grounded in cyber threat analytics, science of evidence, knowledge engineering, and machine learning. After introducing advanced persistent threats, it overviews a systematic APT detection framework and the corresponding APT detection models, the formal representation and learning of these models in the knowledge base of a cognitive agent, and the development and integration of such agents into a specific cyber security operation center.**

advanced persistent threat, cyber threat analytics, cognitive assistant, evidence-based reasoning, knowledge-based learning, ontology, argumentation models, symbolic probabilities.

I. Introduction

An *Advanced Persistent Threat* (APT) is an adversary that leverages superior resources, knowledge, and tactics to achieve its goals through computer network exploitation (CNE). APTs are characterized by their persistence in gaining and maintaining access to targeted networks and their ability to adapt to efforts of network defenders to identify and remediate their activity [1].

Security research companies have been tracking APT groups for years, independently giving them unique names as specific tools, techniques, and procedures (TTPs) are attributed to a group. FireEye/Mandiant has published reports on 30 APT groups since 2013, naming them simply APT1 through APT30 [2].

APT1 is the name given by Mandiant to a group of APT actors, attributed to China's People's Liberation Army unit 61398, who have lead a campaign of cyber espionage since at least 2004. APT1 is known for a regimented approach to computer intrusion activity. An APT1 intrusion typically consists of the following phases: (i) gain access to a network by sending fraudulent, malicious email messages to specific users (spearphishing); (ii) use multiple types of backdoor programs to maintain presence and provide remote connectivity to the target network; (iii) use a collection of command-and-control (C2) servers to obfuscate the source of their attacks; (iv) escalate privileges and acquire legitimate login credentials to access network resources; (v) move laterally within the target network using legitimate credentials to gain redundant points of presence and identify information of interest; and (vi) exfiltrate targeted information through their C2 infrastructure [1].

The technical appendices of Mandiant's report [1] include detailed information on their known backdoor, C2, and exfiltration tools, and a comprehensive set of indicators of compromise (IOCs). Following their report, other security researchers, including the Contagio blog [3] have released supplementary analysis on APT1 malware, techniques, and infrastructure.

APT1, among other attacker groups, practices *evolutionary development* to adapt to changes in network defense technology or simply to increase efficiency. Further analysis of APT1 by Mila at contagiodump.blogspot.com [3] shows a timeline of the attacker group's tool usage from 2004 to 2012, including information on dozens of samples of malware. The group evolved their tool set slowly over the course of at least eight years. These changes in the way malware presents itself on the network and on disk have made it difficult for signature-based intrusion detection tools to detect attacks because the attacks can change static information in their malware faster than defenders can adapt. However, the patterns of behavior change more slowly and with less variance.

Analysis of the indicators of compromise (IOCs) published in [1] shows that APT1 malware demonstrates clusters of behavior. Subsets of the programs share sets of techniques for communicating on the network, persisting through a reboot, or storing data on disk. One example of this is the cluster of malware comprised of BANGAT, SEASALT, KURTON, and AURIGA. While the specific strings used to register the malware as a service or device driver and the names of files and Registry keys differ, the only substantial difference in IOCs between those four tools is the persistence mechanism used to survive a reboot.

Clusters of malware are often called *malware families* in published research. Each member of the family incrementally builds on previous versions as they get detected and become less effective. The Sobig virus is an example of a malware family. It was used in 2003 in a widespread email phishing attack. Joe Stewart describes how the Sobig virus evolved over five different revisions [4, 5]. Over the course of those revisions, the author changed how the malware set its expiration timer, where the command-and-control servers were located, and how encryption was used in an effort to improve the effectiveness of the malware.

Modern cyber defense against APTs is currently done in a cybersecurity operations center (CSOC), which employs teams of network defense experts, analysts, system administrators, and forensics experts. CSOCs leverage a rich tool set including host-based and network-based intrusion detection systems (IDSs), data collections, analysis tools, and visualization tools. CSOCs receive incident information from high-value sources – law enforcement, user reporting, or threat intelligence from other CSOCs – and unconfirmed alerts from security infrastructure such as antivirus software, IDSs, heuristic alerts, or machine learning algorithms. The analyst's responsibility is to monitor alerts and log information from all of these information sources, each having differing levels of credibility, and use them to make a determination about the presence or absence of intrusion activity [6]. However, because a single event alone does not provide sufficient evidence that an intrusion event has occurred, and modern detection technologies are error-prone, each event must be carefully examined and investigated by a human analyst [6]. In a large enterprise, tens of thousands of alerts per day can be reported. Therefore, even sensors with a false positive rate of less than one percent can generate enough false positives to be unmanageable by even large CSOCs.

This paper presents current research on developing an operational semantic theory of cyber defense against advanced persistent threats. Grounded in cyber threat analytics, science of evidence, knowledge engineering, and machine learning, this theory provides a systematic approach to cyber defense, as well as analytical knowledge and models that can be formally represented in the knowledge bases of cognitive agents, enabling them to perform the functions of security analysts, both automatically and in collaboration with the analysts. These cognitive agents will be directly trained by security analysts to detect APTs, through specific analysis examples and explanations, acquiring and generalizing their expertise. As a result, their analyses will be very explicit, easy to understand, and easily updated by the analysts.

Such agents have the potential to radically change established practice by automating the APT analysis process, significantly increasing the CSOC's efficiency, reducing operation costs, increasing detection rates, and decreasing false positive rates. Most importantly, these agents are designed to continuously learn from security analysts and from the agents' own experience, to keep up with and even anticipate new threats. They will also facilitate sharing of evolving APT analytic expertise and training of the cyber analysts.

The next section presents the systematic APT detection framework and the corresponding APT detection models. Section III overviews the formal representation of these models in an APT ontology and APT detection patterns with ontology-based applicability conditions. Section IV discusses the learning of these models. Section V overviews the architecture of a learning agent shell, a general agent building tool that contains reasoning and learning modules for APT detection, as well as a general knowledge base. It also discusses its use in the generation of customized agents for a specific CSOC.

II. Discovery-Based APT Detection Framework

The APT detection framework is represented in Fig.1. Evidence of suspicious activity triggers alternative explanatory hypotheses, which are used to guide the collection of relevant evidence which, in turn, is used to assess the hypotheses.

As will be discussed in this paper, this is both an adaptation and an extension of the general discovery-based framework we have previously developed for intelligence analysis, and implemented in the TIACRITIS [7], Disciple-CD [8, 9] and Cogent [10] analytical tools. A major difference, however, is the significantly higher degree of automation required by the cyber defense process. While the human analysts will still be involved in this process, it is assumed that all the operations can be automatically performed by the cognitive agents.

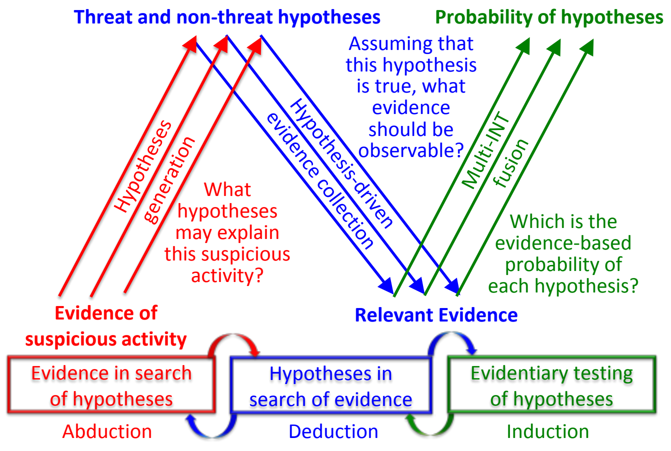
Threat and non-threat hypotheses
Probability of hypotheses
Assuming that this hypothesis is true, what evidence should be observable?
Hypotheses generation
Hypothesis-driven evidence collection
Multi-INT fusion
What hypotheses may explain this suspicious activity?
Which is the evidence-based probability of each hypothesis?
Evidence of suspicious activity
Relevant Evidence
Evidence in search of hypotheses
Hypotheses in search of evidence
Evidentiary testing of hypotheses
Abduction
Deduction
Induction

Fig. 1. Automatic APT detection framework.

APT detection is modelled as a continuous collaboration of three automated processes: *Evidence in search of hypotheses*, *Hypotheses in search of evidence*, and *Evidentiary testing of hypotheses*, each described in the following sections.

A. Evidence in Search of Hypotheses

As shown in the left hand side of Fig.1, evidence of suspicious activity was detected (e.g., by monitoring agents, by Bro [11] or Snort [12] IDSs, etc.) and the question is: *What hypotheses may explain it?* Through *abductive* (imaginative) reasoning, which shows that something is *possibly* true, the agent generates a set of alternative hypotheses, some corresponding to actual APT activity, while others corresponding to non-threat activities. Fig.2 is an illustration of this process.

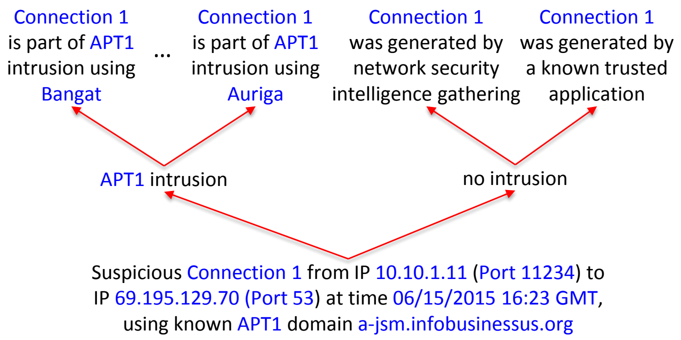
Connection 1 is part of APT1 intrusion using Bangat
...
Connection 1 is part of APT1 intrusion using Auriga
Connection 1 was generated by network security intelligence gathering
Connection 1 was generated by a known trusted application
APT1 intrusion
no intrusion
Suspicious Connection 1 from IP 10.10.1.11 (Port 11234) to IP 69.195.129.70 (Port 53) at time 06/15/2015 16:23 GMT, using known APT1 domain a-jsm.infobusinessus.org

Fig. 2. Evidence in search of hypotheses.

A suspicious connection was signaled by a Snort alert [12] from the internal IP 10.10.1.11 to 69.195.129.70, which is mapped to "a-jsm.infobusinessus.org," a known APT1 domain. It is therefore possible that there is an APT1 intrusion using one of the APT1 malware implants (Bangat, Seasalt, Kurton, Auriga, etc. [1]). But it is also possible that there is no intrusion, and the above connection is the result of network security intelligence gathering, or it was generated by a known trusted application for some legitimate purpose. Each of these hypotheses may explain the suspicious connection. The agent would need to automatically analyze each of these hypotheses to determine which of them is actually true. For this, it needs additional evidence which is obtained through the next process.

B. *Hypotheses in Search of Evidence*

As shown in the middle part of Fig.1, the agent puts each of the generated alternative hypotheses to work guiding the collection of relevant evidence. The question is: *Assuming that this hypothesis is true, what evidence should be observable?*

Fig.3 is an illustration of this process for the hypothesis "Connection 1 is part of APT1 intrusion using Bangat." This hypothesis is successively decomposed into simpler and simpler hypotheses, down to the level of elementary hypotheses for which it is clear what evidence to look for, and collection agents can be automatically invoked with specific search requests. In particular, if there is an APT1 intrusion using Bangat, then there should be activity attributable to APT1 on the Alpha network containing the computer with the IP 10.10.1.11, and the Bangat malware should be present on this host computer. Each of these sub-hypotheses is reduced, in turn, to specific indicators. The indicators for the first one are the possible detection of patterns of DNS resolution consistent with the TTPs of APT1, and the usage of other APT1 domains on the Alpha network. These, in turn, lead to the generation of specific search requests to be carried out by special collection agents, as indicated at the bottom of Fig.3.

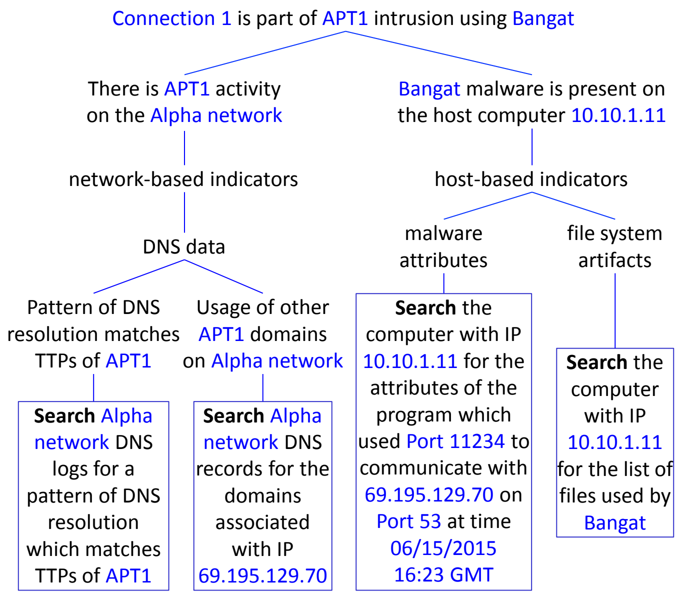
Connection 1 is part of APT1 intrusion using Bangat
There is APT1 activity on the Alpha network
Bangat malware is present on the host computer 10.10.1.11
network-based indicators
host-based indicators
DNS data
malware attributes
file system artifacts
Pattern of DNS resolution matches TTPs of APT1
Usage of other APT1 domains on Alpha network
Search the computer with IP 10.10.1.11 for the attributes of the program which used Port 11234 to communicate with 69.195.129.70 on Port 53 at time 06/15/2015 16:23 GMT
Search the computer with IP 10.10.1.11 for the list of files used by Bangat
Search Alpha network DNS logs for a pattern of DNS resolution which matches TTPs of APT1
Search Alpha network DNS records for the domains associated with IP 69.195.129.70

Fig. 3. Hypothesis in search of evidence.

1) *Context-dependent Reasoning*

Notice in the tree from Fig.3 that some words, such as APT1 and Bangat, appear in blue. This is because they are part of the agent's knowledge base, and are recognized by it.

Notice also that only some hypotheses are completely specified, while their sub-hypotheses are abstracted and understood in the context of their upper-level hypotheses. For example, "host-based indicators" is understood as "There are host-based indicators that the Bangat malware is present on the host computer 10.10.1.11." Similarly, "malware attributes" is understood as "The malware attributes are a host-based indicator that the Bangat malware is present on the host computer 10.10.1.11."

Context-dependent analysis enables a very succinct representation of the reasoning structures of the agent, helping the analyst to visualize a larger portion of it in the agent's whiteboard [10]. At the same time, the agent is aware of the complete representation of each hypothesis which is necessary for the learning and reuse of analytic expertise, as will be discussed in a follow-on section.

2) *Collection Agents*

The collection agents will return the found evidence as formal statements, each with its own *credibility* which represents the probability that the statement is correct [9]. For example, "Search the computer with IP 10.10.1.11 for the attributes of the program which used Port 11234 to communicate with 69.195.129.70 on Port 53 at time 06/15/2015 16:23 GMT" may return, among others, the following items of evidence:

[E4] ati.exe 10.10.1.11 made connection Connection 1 (credibility: certain)

[E5] ati.exe 10.10.1.11 is registered as a Windows Service with name iprip (credibility: certain)

[E6] ati.exe 10.10.1.11 has as unique Bangat string superhard corp. (credibility: certain)

In some cases, the collection agents may also return the *relevance* [9] of an item of evidence to a corresponding elementary hypothesis. For example, the result of "Search Alpha network DNS logs for a pattern of DNS resolution which matches TTPs of APT1" may return the following item of evidence:

[E1] pattern of DNS resolution partially matches TTPs of APT1 (credibility: certain, relevance: very likely)

In this case the collection agent is certain that there is a partial match, and the relevance of very likely expresses the degree of match.

This evidence is represented in the agent's knowledge base, and is used to estimate the probability of the top-level hypothesis from Fig.3, which is done through the next process.

C. *Evidentiary Testing of Hypotheses*

As shown in the right hand side of Fig.1, the agent uses the discovered evidence to test each hypothesis. Hypothesis testing is probabilistic because the evidence is always incomplete, usually inconclusive, frequently ambiguous, commonly dissonant, and with various degrees of credibility [9, 13].

As in Cogent [10], it is possible to use different assessment scales, but all are based on the same system of Baconian probabilities [14, 15] with Fuzzy qualifiers [16], where the values are on an ordered positive scale, as in the following "Probability" scale which is used in this paper:

lack of support < likely < very likely < almost certain < certain

In this case, there may be a lack of support from the available evidence to the considered hypothesis, or the evidence may indicate some level of support (e.g., likely).

Examples of other assessment scales are:

lack of belief < weak < moderate < strong < total belief

no strength < very low < low < medium < high <
< very high < full strength

As evidence is returned by the collection agents, the corresponding parts of the tree in Fig.3 are regenerated, either to assess sub-hypotheses or to generate additional search requests, ultimately resulting in an evidence-based argumentation for assessing the top-level hypothesis, as illustrated in Fig.4.

Notice that each leaf hypothesis is directly assessed based on evidence, and these assessments are automatically combined, from bottom-up, based on the structure of the argumentation, to obtain the assessment of the top hypothesis. Let us consider the item of evidence "[E5] ati.exe 10.10.1.11 is registered as a Windows service with name iprip" from the bottom of Fig.4. Notice that the statement asserted by this item of evidence is precisely the elementary hypothesis to which it is attached ("ati.exe 10.10.1.11 is registered as a Windows service with name iprip"). Therefore, its *relevance* is certain. When the collection agent returns this item of evidence, it also returns its *credibility*, which, in this case, is also certain. The credibility and the relevance are combined (through the minimum function) to obtain the inferential force of the item of evidence on the elementary hypothesis (certain, in this case), as discussed in [9, 10].

Notice also "[E7] ati.exe 10.10.1.11 does not have as MD5 hash Bangat MD5 hash." This is disfavoring evidence for the hypothesis "ati.exe 10.10.1.11 has as MD5 hash Bangat MD5 hash," and therefore there is a lack of support for this hypothesis.

As indicated, the probabilities of the elementary hypotheses are combined, from bottom-up, based on the structure of the argumentation. For example, there are two favoring arguments for the hypothesis "ati.exe 10.10.1.11 has Bangat attributes":

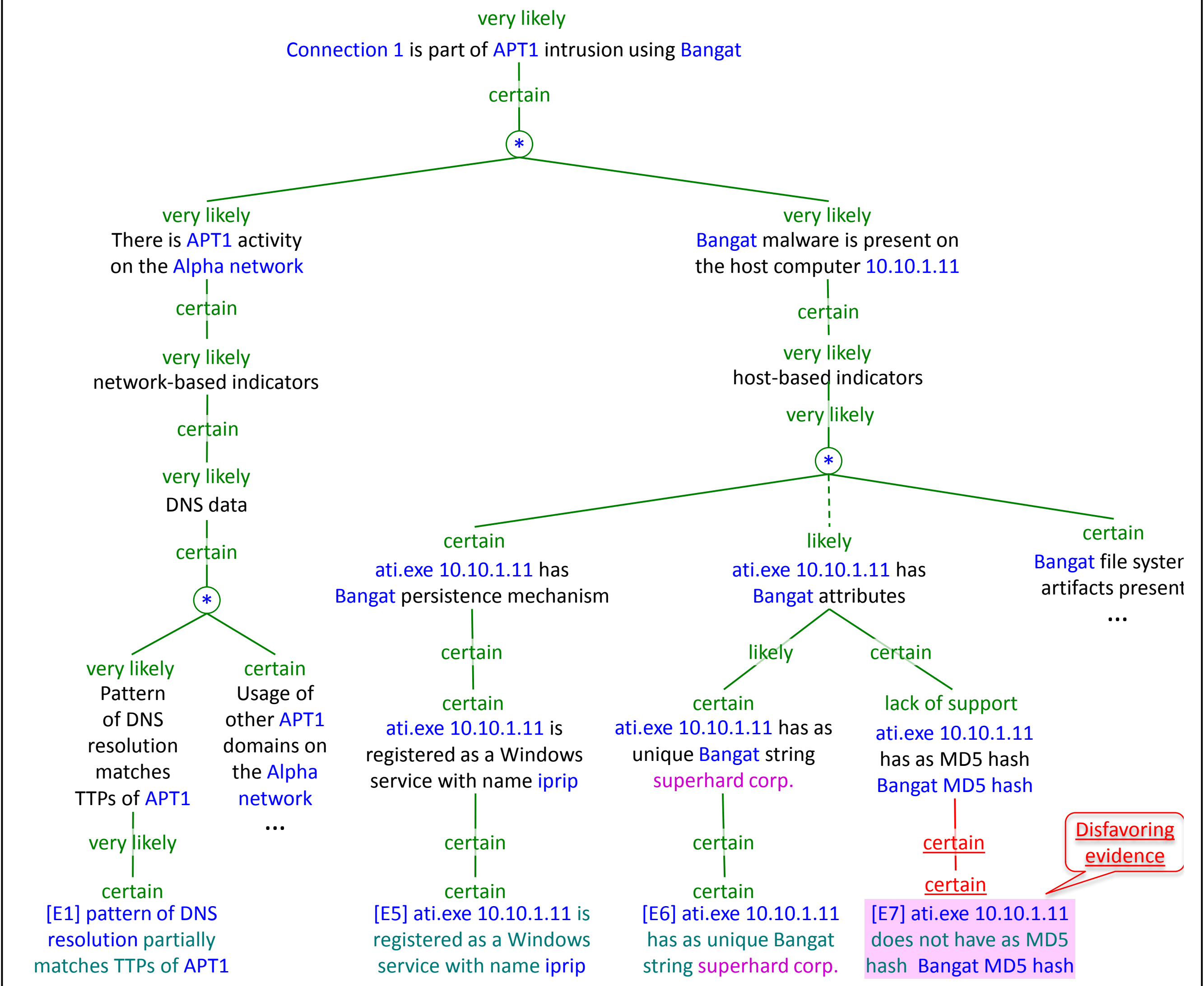
very likely
Connection 1 is part of APT1 intrusion using Bangat
certain
*
very likely
There is APT1 activity on the Alpha network
certain
very likely
network-based indicators
certain
very likely
DNS data
certain
*
very likely
Pattern of DNS resolution matches TTPs of APT1
very likely
certain
[E1] pattern of DNS resolution partially matches TTPs of APT1
certain
Usage of other APT1 domains on the Alpha network
...
very likely
Bangat malware is present on the host computer 10.10.1.11
certain
very likely
host-based indicators
very likely
*
certain
ati.exe 10.10.1.11 has Bangat persistence mechanism
certain
certain
ati.exe 10.10.1.11 is registered as a Windows service with name iprip
certain
certain
[E5] ati.exe 10.10.1.11 is registered as a Windows service with name iprip
likely
ati.exe 10.10.1.11 has Bangat attributes
likely
certain
ati.exe 10.10.1.11 has as unique Bangat string superhard corp.
certain
certain
[E6] ati.exe 10.10.1.11 has as unique Bangat string superhard corp.
certain
lack of support
ati.exe 10.10.1.11 has as MD5 hash Bangat MD5 hash
certain
certain
[E7] ati.exe 10.10.1.11 does not have as MD5 hash Bangat MD5 hash
Disfavoring evidence
certain
Bangat file syster
artifacts present
...

Fig. 4. Evidentiary testing of hypothesis.

IF “ati.exe 10.10.1.11 has as unique Bangat string superhard corp” THEN “ati.exe 10.10.1.11 has Bangat attributes” is likely

IF “ati.exe 10.10.1.11 has as MD5 hash Bangat MD5 hash” THEN “ati.exe 10.10.1.11 has Bangat attributes” is certain

The relevance of each argument (i.e., likely and certain, respectively) is combined with the probability of the corresponding sub-hypothesis (i.e., certain and lack of support, respectively), and the results are again combined, to produce an assessment of the probability of “ati.exe 10.10.1.11 has Bangat attributes”: max(min(certain, likely), min(lack of support, certain)) = likely.

Further up in the argumentation are three possible host-based indicators of the presence of the Bangat malware on the host computer 10.10.1.11: persistence mechanism (P), malware attributes (M), and file system artifacts (F). Each indicator may be present with a certain probability, or it may not be present. The more indicators are present, the more relevant they are, collectively, to the presence of the Bangat malware. Fig.5 shows the relevance of each subset of these indicators.

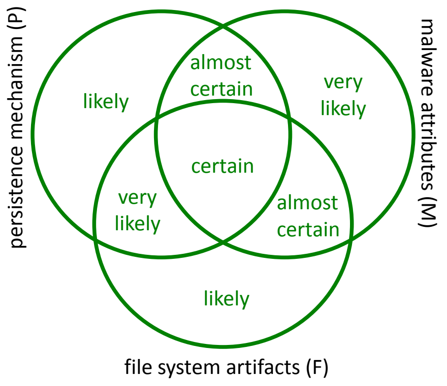

Fig. 5. Relevance of different subsets of indicators.

If all of them are present (i.e., the intersection of the three discs), then their relevance is certain. However, if only the persistence mechanism (P) is present, then its relevance is only likely. The actual probability of the “host-based indicator” hypothesis based on the three indicators (i.e., P, M, F) can be assessed by computing the inferential force for each possible combination of present indicators, and then selecting the maximum value. In this particular example, since all the three indicators are present, their relevance is certain, but the minimum of their probabilities is likely, which produces an overall assessment of likely. However, one obtains a higher assessment by ignoring the “malware attributes” indicator which the evidence indicates to be only likely. In such a case, the relevance of the other two indicators is lower (very likely), but they are both certain, giving an assessment of very likely to the “host-based indicators” hypothesis. In Fig.4, this “combined indicators” operator is marked with “*”. Fig.4 also shows that the “Bangat attributes” indicator is ignored in this assessment by using an interrupted relevance link.

There are additional argument structures that are not illustrated by the argumentation from Fig.4. For example, a hypothesis may have an argument that consists of a conjunction of sub-hypotheses, in which case its probability is obtained as the minimum between the relevance of the argument and the probabilities of the sub-hypotheses.

For a hypothesis there may be both favoring and disfavoring arguments, and a hypothesis may have both favoring and disfavoring evidence. In such cases, the probability of the hypothesis is obtained by using an on-balance function which is initialized to a default set of values and automatically updated based on the values provided by the analyst when analyzing hypotheses.

III. Representation of the APT Detection Models

The previous section presented a systematic process of APT detection through evidence in search of hypotheses, hypotheses in search of evidence, and evidentiary testing of hypotheses (see Fig.1). To enable an agent to automatically perform this kind of reasoning, the knowledge of the APT detection models has to be formally represented.

We employ a learnable hybrid knowledge representation consisting of an APT ontology and reasoning tree patterns with ontology-based applicability conditions. The ontology language is an extension of RDFS [17, 18] with additional features to facilitate learning and evidence representation [19, 20, 21]. A fragment of the APT ontology, corresponding to the reasoning discussed in the previous section, is illustrated in Fig.6.

The middle left side shows the (partial) representation of Connection 1, the suspicious connection that triggered the APT1 detection process. The upper-right side represents some malware instances and concepts (e.g., APT1, Bangat, Seasalt, APT group). Under this fragment there is a partial representation of the network structure, showing the Alpha network and Corp_wkst_1, the computer used as host by the Bangat malware. The upper-left side shows some general concepts related to cybersecurity. The bottom right shows two features together with their domains and ranges. The bottom left of Fig.6 shows some of the evidence items returned by the collection agents as a result of executing the searches from the bottom part of Fig.3. Each evidence about a fact (e.g., “ati.exe 10.10.1.11 has as unique Bangat string superhard corp.”) is labelled with its evidence name (i.e., [E6]), and has a certain credibility (not shown in Fig.6). Such facts are used to generate argumentation structures such as that from Fig.4, as will be discussed next.

Another component of the hybrid knowledge representation are the general tree patterns with ontology-based applicability conditions, such as that shown in Fig.7. The condition represents the semantics of the tree pattern and the context in which it can be applied to automatically generate specific tree structures. In particular, the tree pattern from Fig.7 will generate the search tree from Fig.3 if its condition is satisfied in the current situation, where V1 is instantiated to Connection 1, V2 is instantiated to APT1, and V3 to Bangat. The current situation is represented by the facts in the ontology from Fig.6.

IV. Mixed-Initiative Learning of APT Detection Models

Tree patterns, such as that in Fig.7, can be learned from specific examples of trees defined by cyber security experts, by employing a multi-strategy learning approach that integrates

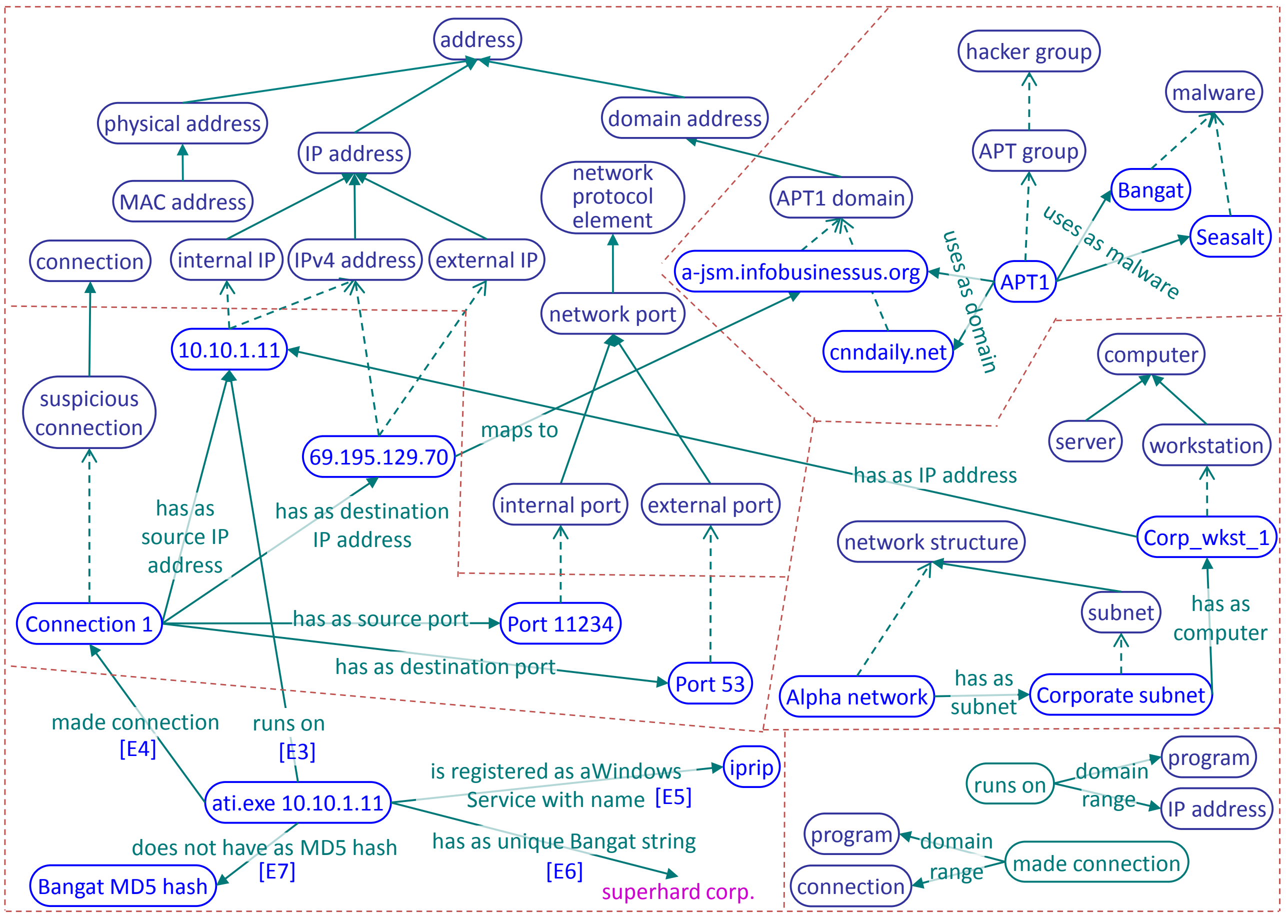

Fig.6. Ontology fragment.

several learning strategies including *learning from examples*, *learning from explanations*, and *learning by analogy and experimentation*, in a mixed-initiative interaction with the analyst, as presented in [20 - 24].

In essence, the expert will direct the agent to learn a tree pattern from a fragment of an argumentation, such as that shown at the bottom left of Fig.8. Then the agent interacts with the expert to determine the important features of the instances from the argument fragment. They include those that link the instances appearing in the top-level hypothesis (i.e., Bangat and 10.10.1.11) with the instances that appear only in the sub-hypotheses (i.e., ati.exe 10.10.1.11). Such a feature is "[E3] ati.exe 10.10.1.11 runs on 10.10.1.11." These and other potentially relevant features, such as "[E4] ati.exe 10.10.1.11 made connection Connection 1," are proposed by the agent and the relevant ones are selected by the expert. The upper left side of Fig.8 shows the identified relevant features together with the superconcepts of the instances in the APT ontology.

Next the agent automatically generates the tree pattern from the bottom right side of Fig.8. The tree pattern is obtained by simply replacing each instance (e.g., Bangat) with a variable (i.e., V1), and by removing the probabilities of the hypotheses. Direct relevance links, such as certain of "host-based indicator", are preserved in the pattern. The relevance of the combined indicators, such as those under the "host-based indicator" hypothesis, is generalized to a function.

The agent also automatically generates the applicability condition of the learned pattern, shown in the upper right side of Fig.8. Notice however that, instead of a single applicability condition (such as the one in Fig.7), there is an upper bound condition and a lower bound condition. They are obtained as maximal and, respectively, minimal generalizations of the instances and their important relationships, in the context of the APT ontology which is used as a generalization hierarchy.

As the agent learns new tree patterns from the cybersecurity expert, their interaction evolves from a teacher-student interaction, toward an interaction where they both collaborate on APT detection. In this case the agent automatically generates argumentation structures by applying the partially learned patterns and the expert critiques the reasoning, guiding the agent in refining its patterns. Correct argument structures generated by the agent lead to automatic generalization of the lower bound conditions of the used patterns. Any mistake identified by the expert leads either to the specialization of the upper bound condition of the responsible pattern, or to the addition of an except-when condition (with both an upper bound and a lower bound). The except-when conditions should not be satisfied in order for the pattern to be applicable. In time, the lower and the upper bound conditions of a pattern converge toward one another and to an exact applicability condition. The goal is to improve the applicability condition of the pattern so that it only generates correct argumentation fragments.

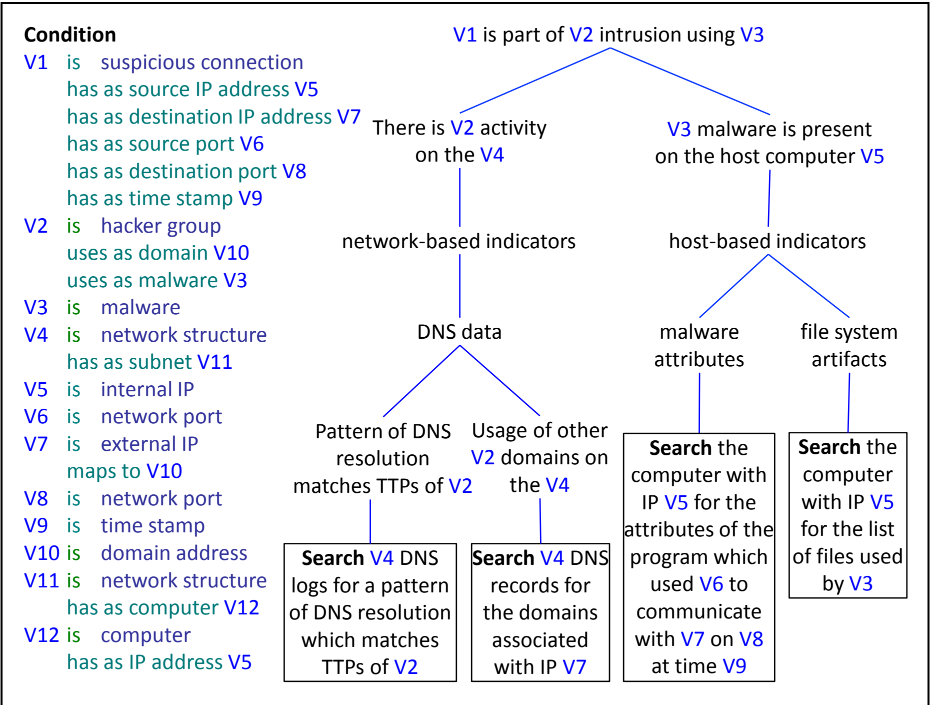
Condition
V1 is suspicious connection
has as source IP address V5
has as destination IP address V7
has as source port V6
has as destination port V8
has as time stamp V9
V2 is hacker group
uses as domain V10
uses as malware V3
V3 is malware
V4 is network structure
has as subnet V11
V5 is internal IP
V6 is network port
V7 is external IP
maps to V10
V8 is network port
V9 is time stamp
V10 is domain address
V11 is network structure
has as computer V12
V12 is computer
has as IP address V5
V1 is part of V2 intrusion using V3
There is V2 activity on the V4
V3 malware is present on the host computer V5
network-based indicators
host-based indicators
DNS data
malware attributes
file system artifacts
Pattern of DNS resolution matches TTPs of V2
Usage of other V2 domains on the V4
Search V4 DNS logs for a pattern of DNS resolution which matches TTPs of V2
Search V4 DNS records for the domains associated with IP V7
Search the computer with IP V5 for the attributes of the program which used V6 to communicate with V7 on V8 at time V9
Search the computer with IP V5 for the list of files used by V3

Fig.7. Tree pattern for evidence collection.

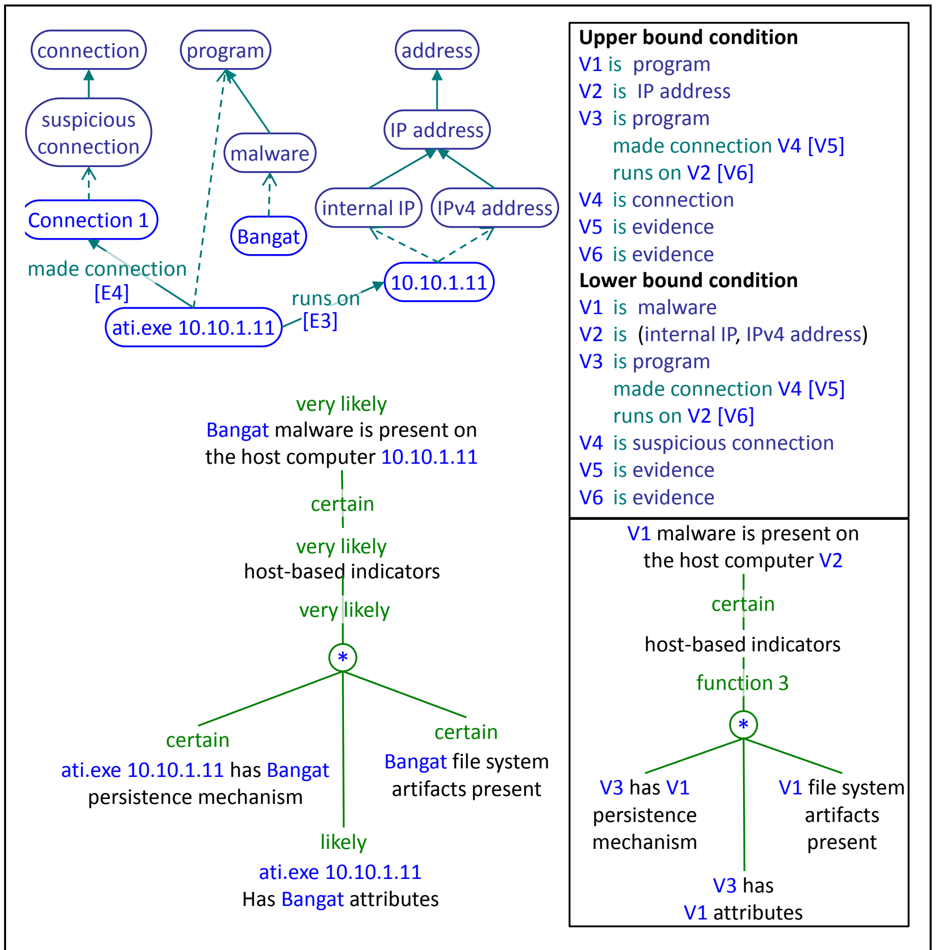

Fig.8. Illustration of tree pattern learning.

V. Learning Agent Shell

The developed APT detection theory and associated methods for knowledge representation, reasoning, and learning are being used to develop a prototype *learning agent shell* whose overall architecture is presented in Fig.9. This is a general agent development tool that contains general reasoning modules for the three APT detection processes from Fig.1, modules for development and refinement of the agent's ontology, modules for learning and refining the detection patterns, as well as modules for management of evidence, knowledge bases in the knowledge repository, and knowledge in knowledge bases. The learning agent shell will also incorporate a significant amount of general cybersecurity expertise for automatic APT detection into its general knowledge base, which is applicable in any CSOC. This includes an ontology, as well as tree patterns with ontology-based applicability conditions, for detection of a wide range of APT activities. The learning agent shell is used to rapidly generate a set of agents and their knowledge bases, all customized to a specific CSOC. They include a Hypotheses Generation Agent, several Automatic Analysis Agents, and several Mixed-Initiative Analysis Assistants.

The main customization of the knowledge base consists of populating it with a representation of CSOC's network, including its layout (see the middle right hand side of Fig.6), available sensors, operating systems used, and network security tools used.

The Hypotheses Generation Agent is a customization of the learning agent shell centered around the Hypotheses Generation module. It will generate hypotheses, such as those from the top part of Fig.2, from a variety of logs, network capture sensors, and intrusion detection devices and systems such as Bro [11] and Snort [12].

The Automatic Analysis Agents are all instantiations of the learning agent shell centered around the Automatic Analysis module. For each new generated hypothesis (e.g., "Connection 1 is part of APT1 intrusion using Bangat"), an instantiation of such an agent is automatically created to generate analysis trees such as those in Fig.3 and Fig.4.

Search requests generated by the Automatic Analysis Agents (see the bottom of Fig.3) are sent to a Collection Manager which manages and forwards them to corresponding Local Collection Agents. It also returns evidence (and its credibility) found by the collection agents to the corresponding automatic analysis agents.

Each Mixed-Initiative Analysis Assistant is an instantiation of the learning agent shell centered around the Mixed-Initiative Analysis module. It enables a specific CSOC analyst or operator to access the current state of the detection process in the form of a list of hypotheses of interest and their partial analyses. It also collaborates with the analyst who may update any of the analyses, provide additional evidence, or make several assumptions when no evidence is found. In the case of a new type of attack, the analyst and the mixed-initiative assistant can develop together an analysis tree from which new detection patterns are learned.

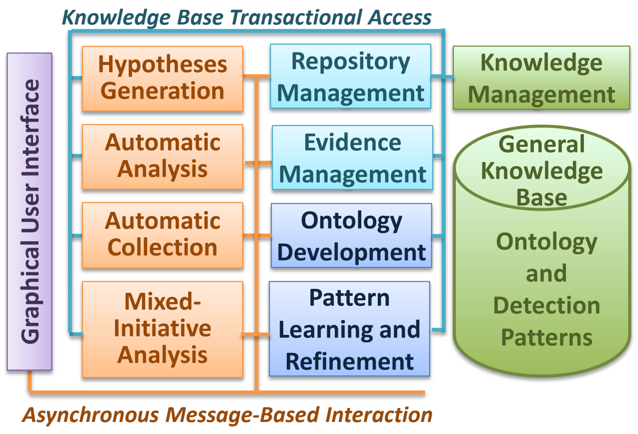

Fig.9. Learning agent shell for APT detection.

VI. Conclusions

We have presented current work on developing a semantic theory of cyber defense against advanced persistent threats, and using it to develop collaborative cognitive agents that generate defensible and persuasive analyses which show very clearly the argumentation logic, what evidence was used, and how. Such agents are integrated in cybersecurity operations centers to improve both the speed and the quality of the performed analyses, and to significantly increase the probability of accurately detecting intrusion activity while drastically reducing the workload of the CSOC operators. Moreover, this approach is developed to enable continuous learning from cybersecurity experts, based on evolving threat intelligence, to cope with new threats, thus providing high defense agility to the CSOC.

References

[1] Mandiant Intelligence. 2013. APT1: Exposing one of China's cyber espionage units, *Mandiant.com*.

[2] FireEye. 2015. APT30 and the Mechanics of a Long-running Cyber Espionage Operation, *FireEye*, April.

[3] Mila. 2013. Mandiant APT1 samples categorized by malware families, *contagio*, 03 March.

[4] Stewart J. 2003a. Sobig.a and the Spam You Received Today, *joestewart.org*, 08 Jul. http://www.joestewart.org/sobig.html, Accessed: 15 May 2015.

[5] Stewart J. 2003b. Sobig.e - Evolution of the Worm, *joestewart.org*, 08-Jul-2003. http://www.joestewart.org/sobig-e.html, Accessed: 15 May 2015.

[6] Zimmerman C. 2014. *Ten Strategies of a World-Class Cybersecurity Operations Center*. MITRE Corporation.

[7] Tecuci G., Marcu D., Boicu M., Schum D.A., Russell K. 2011. Computational Theory and Cognitive Assistant for Intelligence Analysis, in *Proceedings of the Sixth International Conference on Semantic Technologies for Intelligence, Defense, and Security – STIDS*, pp. 68-75, Fairfax, VA, 16-18 November.

[8] Tecuci G., Schum D.A., Marcu D., Boicu M. 2014. Computational Approach and Cognitive Assistant for Evidence-Based Reasoning in Intelligence Analysis, *International Journal of Intelligent Defence Support Systems*, 5(2):146–172.

[9] Tecuci G., Schum D.A., Marcu D., Boicu M. 2015. *Intelligence Analysis as Discovery of Evidence, Hypotheses, and Arguments: Connecting the Dots*, Cambridge University Press, to appear.

[10] Tecuci G., Marcu D., Boicu M., Schum D. 2015. COGENT: Cognitive Agent for Cogent Analysis. In the *Proceedings of the 2015 AAAI Fall Symposium "Cognitive Assistance in Government and Public Sector Applications"*, Arlington, VA, November.

[11] Bro home page http://www.bro-ids.org .

[12] Snort homepage https://www.snort.org/

[13] Schum D. A. 2001. *The Evidential Foundations of Probabilistic Reasoning*, Northwestern University Press.

[14] Cohen L. J. 1977. *The Probable and the Provable*, Clarendon Press, Oxford.

[15] Cohen L. J. 1989. *An Introduction to the Philosophy of Induction and Probability*, Clarendon Press, Oxford.

[16] Zadeh L. 1983. The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems. *Fuzzy Sets and Systems*, 11:199-227.

[17] W3C. 2004. http://www.w3.org/TR/rdf-schema/

[18] Allemang D. and Hendler J. 2011. *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and Owl*, Morgan Kaufmann Publishers.

[19] Tecuci G., Boicu M., Ayers C., Cammons D. 2005. 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, 2-6 May.

[20] Tecuci G., Boicu M., Marcu D., Stanescu B., Boicu C., Comello J., Lopez A., Donlon J., Cleckner W. 2002. 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.

[21] Tecuci G. Boicu M. Boicu C. Marcu D. Stanescu B. Barbulescu M. 2005. The Disciple-RKF Learning and Reasoning Agent, *Computational Intelligence*, 21(4):462-479.

[22] Tecuci G., Boicu M., Marcu D., Boicu C., Barbulescu M., Ayers C., Cammons D. 2007. Cognitive Assistants for Analysts, *Journal of Intelligence Community Research and Development*. Also in Auger J. and Wimbish W. eds. *Proteus Futures Digest*, pp.303-329, Joint publication of National Intelligence University, Office of the Director of National Intelligence, and US Army War College Center for Strategic Leadership.

[23] Tecuci, G., Boicu, M., Cox, M. T. 2007. Seven Aspects of Mixed-Initiative Reasoning: An Introduction to the Special Issue on Mixed-Initiative Assistants, *AI Magazine*, 28(2):11-18, Summer.

[24] Tecuci G., Marcu D., Boicu M., Schum D.A. 2015. *Knowledge Engineering: Building Personal Learning Assistants for Evidence-based Reasoning*, Cambridge Univ Press (to appear).