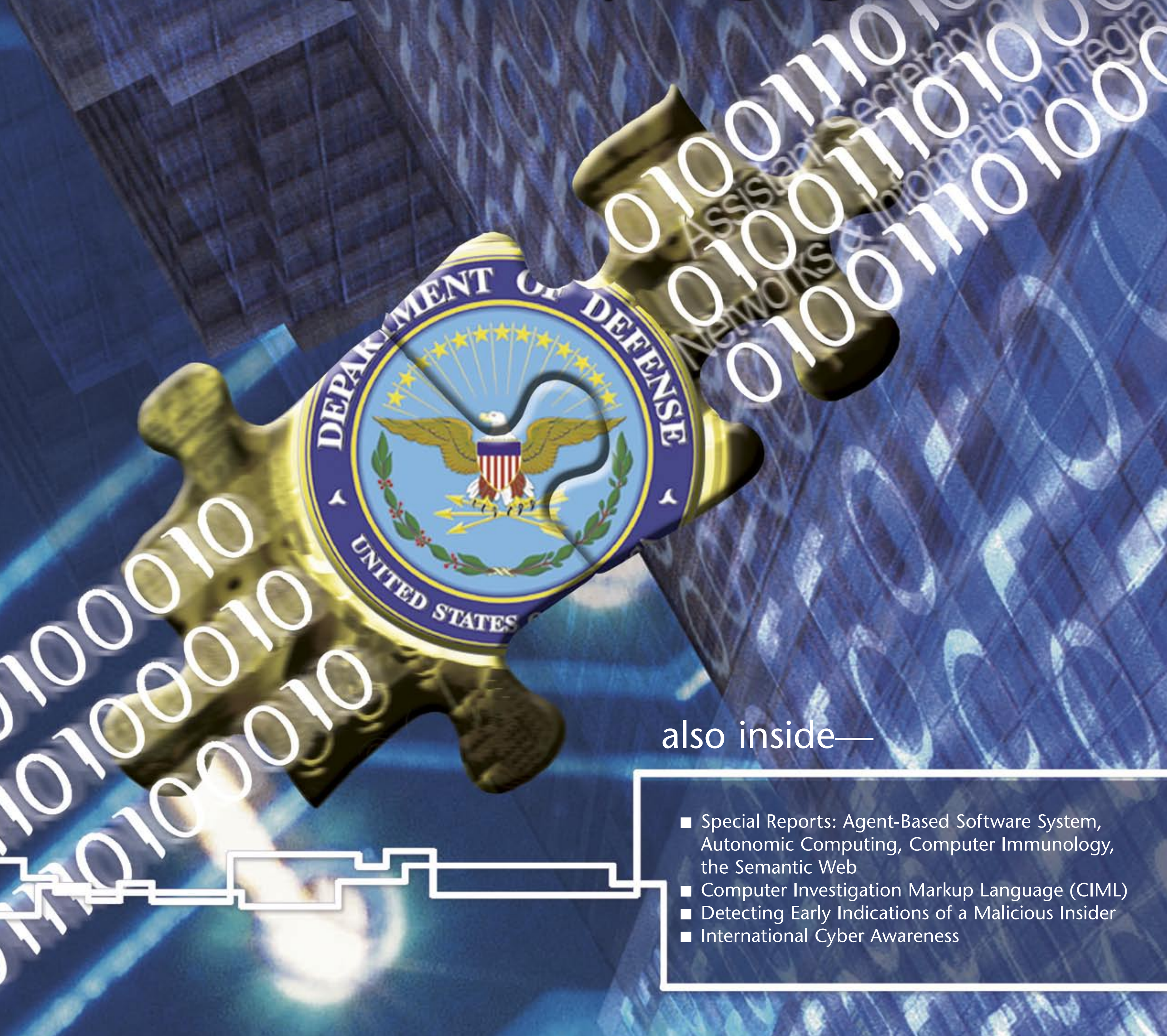
IAnewsletter

The Newsletter for Information Assurance Technology Professionals

Volume 7 Number 2 • Fall 2004

Ontology Development Challenges and Applications Using the DARPA Agent Markup Language (DAML)


DEPARTMENT OF DEFENSE
UNITED STATES
Assistant Secretary of Defense
Networks & Information Integration
0100010
0100110

also inside—

- Special Reports: Agent-Based Software System, Autonomic Computing, Computer Immunology, the Semantic Web
- Computer Investigation Markup Language (CIML)
- Detecting Early Indications of a Malicious Insider
- International Cyber Awareness

contents

feature

IA initiatives

in every issue

About IATAC & the *IAnewsletter*—

IAnewsletter is published quarterly by the Information Assurance Technology Analysis Center (IATAC). IATAC is a Department of Defense (DoD) sponsored Information Analysis Center, administratively managed by the Defense Technical Information Center (DTIC), Director, Defense Research and Engineering (DDR&E).

Contents of the *IAnewsletter* are not necessarily the official views of or endorsed by the U.S. Government, DoD, or DDR&E. The mention of commercial products and/or services does not imply endorsement by the DoD or DDR&E.

Inquiries about IATAC capabilities, products, and services may be addressed to—

IATAC Director:	Gene Tyler
Acting Deputy Director:	Jim Peña
Inquiry Services:	Peggy O'Connor

IAnewsletter Staff—

Creative Director:	Christina P. McNemar
Art Director:	Ahnie Senft
Designers:	Holly Shipley Dustin Hurt
Editorial Board:	Jim Peña Tara Shea Gene Tyler

IAnewsletter Article Submissions

To submit your articles, notices, programs, or ideas for future issues, please visit http://iac.dtic.mil/iatac/IA_newsletter.html and download an "Article Instructions" packet.

IAnewsletter Address Changes/Additions/Deletions

To change, add, or delete your mailing or E-mail address (soft-copy receipt), please contact us at—

IATAC
Attn: Peggy O'Connor
3190 Fairview Park Drive
Falls Church, VA 22042

Phone:	703/289-5454
Fax:	703/289-5467
E-mail:	iatac@dtic.mil
URL:	http://iac.dtic.mil/iatac

Deadlines for future Issues—

Winter 2004	November 1, 2004
Cover design:	Holly Shipley
Newsletter design:	Ahnie Senft

Distribution Statement A:

Approved for public release;
distribution is unlimited.

IATAC Chat

Robert J. Lamb, IATAC Director

This will be my final column as the Director of IATAC, for the IAnewsletter, and my last edition as this publication's editor.

You'll note this edition is somewhat larger than usual. In addition to our slate of articles, we had the opportunity to publish a report entitled "Ontology Development Challenges and Applications Using the DARPA Agent Markup Language (DAML)." The report was prepared under the Technical Area Task (TAT) program in support of the Air Force Research Laboratory in Rome, New York. It examines the challenges and necessary steps for developing ontologies for use by software applications and specifically looks at DAML and how it might be leveraged. It also includes an analysis of the benefits and capabilities of DAML.

This will be my final column as the Director of IATAC for the IAnewsletter and my last edition as this publication's editor. I've been with IATAC a little over five years and have served as the Director since October of 2000. I have enjoyed it immensely. It has been a true pleasure working with the IATAC staff as well as the Information Analysis Center (IAC) Program Management Office at DTIC (Ron Hale, Nancy Pfeil, and Carla Percy) and our IATAC Steering Committee, led by Richard Hale has been great. The best part of this transition is knowing that IATAC is in good hands.

I'd like to take this opportunity to introduce Gene Tyler, the newly appointed Director and Program Manager of IATAC. Gene joins us following a distinguished 35-year career in the United States Army serving in a variety of troop and staff positions overseas and in the United States. Originally from Massachusetts, Gene enlisted in the Army in June 1969. He was commissioned in the Signal Corps in October 1977 upon graduating from Officer Candidate School. Along the way he commanded C Company, 97th Signal Battalion and later was the Battalion Commander, 86th Signal Battalion, 11th Signal Brigade. His career included two tours to the Republic of Vietnam and four tours to Germany. Most recently he served as the Director of the Defense-Wide Information Assurance Program (DIAP).

Gene is uniquely qualified for this position. In addition to his broad experience in the Army, he has an Associates Degree in Law Enforcement and a Bachelors of Science (B.S.) in Management from the University of Maryland, a Master of Science from the Industrial College of the Army Forces, a Master of Science System Management (MSSM) and an MBA from Florida Institute of Technology. He has completed the National Defense University (NDU) Chief Information Officer (CIO) and Information Assurance programs.

I'd like to conclude with a special thanks to IATAC staff with whom I've worked these past five years. There are some truly outstanding professionals that are genuinely committed to making this IAC the best it can possibly be.

- Peggy O'Connor, who has been one of those unsung heroes making this IAC function day to day with the IA Digest, IO Calendar, product orders, and bibliographic inquiry support
- Christina McNemar and Ahnie Senft, who along with their staff, have made this publication the outstanding periodical it is
- Jim Pena, Tara Shea, and April Perera providing responsive, timely, multi-faceted inquiry and conference support to IATAC users
- Jack Benkert, Brad Soules, Pam Stevens, Tina Demme, and Ken Wierzbic driving the TAT program across DoD and the federal government
- …and Bob Thompson who was its first Director and laid the foundations

Warmest regards,

Bob

Ontology Development Challenges and Applications Using the DARPA Agent Markup Language (DAML)

by Alan Nawoj and Mark Gorniak

Author's Note

As the need for automated information search and retrieval as well as automated information processing grows, ontologies will continue to play a central role in the way that information is organized and represented. This article presents observations based on a research project conducted in 2003 that investigated the various steps and challenges involved in developing ontologies for use by software applications. The ontologies were developed using the DARPA Agent Markup Language (DAML), and an assessment is given of the various benefits and capabilities of DAML, in the context of ontology-based software applications. The utility of DAML for enabling semantic interoperability, querying of knowledge bases, and inferencing is also dis-cussed as it relates to an Air Force application. DAML has subsequently been subsumed by the World Wide Web Consortium (W3C) Web Ontology Language (OWL) Recommendation.

As the amount of information available on the World Wide Web (WWW) and in databases continues to rapidly grow, there is an increasing need for ways to better manage information so that it can be exploited to the fullest extent in a variety of applications. One of the main goals of the Air Force Research Laboratory, with respect to the management of information resources, is to develop—

> *"…advanced technologies and approaches to the acquisition, analysis, and timely dissemination of intelligence information for the Intelligence Community" [1]*

so that military decisions can be made in a timely and accurate manner. Efficient data and information management within large domains or across multiple domains, however, is often very challenging part of this is being able to establish consensus on the meaning and representation of the data and information that will be used by its consumers. This is especially true of applications that manipulate data from multiple sources, where different data models may come into conflict.

Additionally, the level of expressivity at which data and information are represented affects their utility with respect to information search and retrieval processes as well as decision making processes. For instance, more expressive data models provide for a more comprehensive and accurate representation of a particular domain, thus allowing intelligent software applications to make more sophisticated queries or inferences over the data. Ontologies provide one approach for information modeling that allows for domain specific knowledge to be precisely defined, and this article will provide an introduction to the basic concepts and challenges of developing and using ontologies. The DARPA Agent Markup Language (DAML), which is being developed as an extension of current Web-grounded technologies, is used as an ontology rep-resentation language. [2] Benefits and capabilities, as well as shortcomings, will be revealed as the use of DAML is investigated in an Air Force related software application. This application will serve as a case study in the process of defining an ontology, expressing an ontology in DAML, and inferencing over DAML-encoded information. Finally, issues of scalability with inferencing will be introduced, followed by a conclusion.

Overview of ontologies and inferencing

Information representation approaches

One of the key decisions that must be made up front when designing a software application that processes any type of information is the means in which the information will be stored. Often the optimal form of data or information storage and representation is closely tied to the objectives of the software application and the type of data being represented. For example, relational databases and database schemas provide a very popular and effi-

cient means of data storage for applications that deal with "simple" data, namely data elements that are essentially independent and do not have explicit structural relationships defined.

Since no machine processable relationships or constraints between data elements can be represented in a data-base, this requires applications to be developed that are closely dependent on the structure of the database, and which encode the required relationships and constraints. Queries or modifications of data stored in relational data-bases must be posed in a precise language, such as the Structured Query Language (SQL), which requires the user to specify exactly which tables and columns of data need to be accessed. All data stored in relational databases is stored according to the schema, which describes the overall structure of the tables that store information.

Information that is stored using the eXtensible Markup Language (XML) is also stored explicitly according to the XML schema and Document Type Definition (DTD) that govern the XML content. Entities are represented using descriptive tags and can contain various attributes or child tags to further describe characteristics of the named entity. In general, the designer of an XML schema and DTD assigns tag names as conceptual placeholders, and it is necessary for any user of this schema to resolve the precise meanings of the concepts that are embodied in the tags with the designer or with a reference document to avoid inaccurate interpretation.

In terms of expressivity, however, one of the main limitations of XML and relational database storage mechanisms is that they lack the ability to represent complex data relationships so that "the semantics of data can be encoded in a machine processable format." [3] The ability to represent complex data relationships is necessary to enable automated information understanding, and ontologies provide one such data model for capturing knowledge about the world. In short, an ontology is comprised of all the terms and concepts that describe a particular view of world, as well as the relationships and constraints that exist between these concepts. In fact, there are many similarities between the way that information is organized in an ontology and the way that information is organized using object-oriented programming languages. General concepts or abstractions within an ontology can be loosely equated to classes within an object-oriented program, and the properties or predicates that define the characteristics of these concepts in an ontology are somewhat analogous to the attributes of a class within an object-oriented program. However, there are some fundamental differences between ontologies and object-oriented programming which prevent us from strongly equating the two information representation formats. In "Ontology Development 101: A Guide to Creating Your First Ontology," the authors make this point very clear in the following excerpt. [4]

> *Object-oriented programming centers primarily around methods on classes—a programmer makes design decisions based on the operational properties of a class, whereas an ontology designer makes these decisions based on the structural properties of a class. As a result, a class structure and relations among classes in an ontology are different from the structure for a similar domain in an object-oriented program.*

Nonetheless, the concepts of information hierarchy and inheritance apply to ontologies in a way that is somewhat similar to that of object-oriented programming. One of the main differences between the two is that hierarchical rela-tionships between classes in the object-oriented program-ming paradigm are strict classification hierarchies, whereas for ontologies these relationships are described in terms of instance membership. Subclasses within ontologies can overlap, for example, and instances of classes can belong to multiple classes. Ontology classes, for simplicity, should be characterized as lattice of mathematical sets. [5]

In general, ontological concepts can be organized in a hierarchical style, where more general concepts are listed "above" more specific concepts so that the more specific concepts can inherit all of the properties of the more generic concepts that are directly "above" in the hierarchy. This promotes information reuse by eliminating the redundant definition of attributes that are common to more than one similar concept, thus allowing for much information to be implicitly defined. Various other relationships can also be defined between concepts in an ontology, such as the equivalence of two or more concepts, logical intersections and unions of classes, and inverse relationships, among others. In general, an ontology is a declarative means of describing a particular domain of interest and organizing information, whereas an object-oriented program is expressed as a procedural sequence of commands in source code.

The basic connected graph data structure from computer science is effective in visualizing as well as modeling an ontology or knowledge base. For instance, Figure 1 below defines a simplified radar facility ontology expressed using a graph model. The blue-colored nodes in this graph represent general concepts (classes) in the ontology, and the white-colored nodes denote the datatypes of various properties that describe the concepts. The labeled arcs that connect nodes symbolize predi-cates (properties) attached to a given concept, and represent a relationship between two different concepts. For example, the two concepts "Facility" and "RadarFacility" share a parent-child hierarchical relationship as identified by the arc labeled "subClassOf" in Figure 1.

While the graph model for ontology representation is more for visualization and design purposes, there exist various ontology representation formats and lan-guages to express ontologies in ways that are machine processable. Some of these knowledge representation lan-guages include CycL (http://www.cyc.com), Loom (http://www.isi.edu/isd/LOOM/LOOM-HOME.html), and DAML+OIL (DAML+Ontology Inference Layer). The DAML+OIL ontology rep-resentation language, or DAML as it will be referred to hereinafter, was the first to be based on the World Wide Web Consortium (W3C) standards of XML, Resource Description Framework (RDF), and RDF-Schema (RDFS), making it Web-compatible. DAML was recently transitioned to the W3C and provided the basis for the Web Ontology Language (OWL). [6]

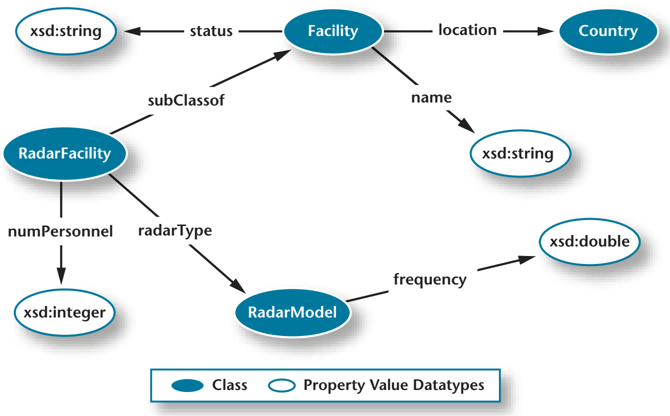


Figure 1. Sample ontology (military facilities) displayed using connected graph model

Once an ontology is designed and represented using a specific knowledge representation language, one may then begin to associate instance data with the con-cepts in the ontology and construct a knowledge base that conforms to the ontology. In short, a knowledge base is just an instantiation of an ontology—it is analogous to a database that contains actual data, whereas the ontology would correspond to the database schema. However, once a particular knowledge base exists, how does one make use of it and what are its benefits? The following section on inferencing provides some answers to this question.

Fundamentals of inferencing

The process of inferencing is one action that can exploit the structure of an ontology to provide valuable results. The term inferencing describes the process of applying reasoning techniques to a given data set in order to generate new data, and often this involves matching a given data set against a list of constraints or rules to see which rules should be executed. Software applications that are specifically designed to employ inferencing algorithms are often called "inference engines." The example shown in Figure 2 below demonstrates a basic inference given a small set of facts and a single rule, and it is based on the inference axiom called subsumption. [7]

The "if" portion of the rule that is listed in Figure 2 called the rule antecedent, is satisfied by the two given facts, since a "RadarFacility" is defined as a subclass of "Facility" and "RadarModel_B" is defined as an actual instance of a "RadarFacility." This condition causes the "then" portion of the rule, called the action, to execute and generate the conclusion, or new assertion, that "RadarModel_B" is also an instance of a "Facility." Someone who then queries the updated list of facts for any instances of the concept "Facility" would be returned "RadarModel_B" as a result. Thus, before the inference occurred, the inferred fact was an implicit piece of information that was associated with the initial two facts and rule. After the inference was complete, the inferred fact then became an explicit statement that was added to the initial list of facts.

```
Given facts: RadarFacility is subclass of Facility
     RadarModel_B is an instance of RadarFacility
Given rule:  if ((“x” is subclass of “y”)&(“z” is
an instance of “x”))
     then (“z” is an instance of “y”)
Inferred fact:      RadarModel_B is an instance
of Facility
```

Figure 2. Simple inference example using two facts and one rule

Other forms of inferencing also exist, such as automatic classification, which involves the classification of an unknown entity into one (or more) of the concepts of an existing ontology based on the known attributes of this entity. For example, if presented with the military facility ontology in Figure 1 and an unknown instance that pos-sesses the attributes of "numPersonnel" and "radarType," one may infer that this unknown entity is a "RadarFacility," since these are both characteristics that describe this concept in the given ontology. The conclusion that this instance may actually be a "RadarFacility" was arrived at by trying to automatically classify the unknown entity into an existing ontological concept whose attributes most closely matched those of the unknown entity.

In general, inferencing is a powerful mechanism that is well suited for expressive data sets where complex data relationships exist. However, the types of inferencing that can be performed over ontology based information are limited by the capabilities of the ontology representation language that is used. Such limitations will be discussed in the context of DAML as a case study in ontology creation is explored.

Ontology development challenges

The development of an ontology is an iterative and time-intensive process. To start, one must possess the necessary domain expertise to ensure that the ontology elements, as well as the element relationships, are precisely defined and capable of being mapped to an end user‘s needs. This may require the assistance of a knowledgeable person or group of experts in the particular domain of interest to help conceptualize the ontology, depending on its estimated size, scope, and application. Typically, an ontology designer will either follow a top-down or bottom-up design approach, but a hybrid of the two is also possible. The top-down approach involves mapping out the most generic concepts in the domain first and then deciding how more specific concepts can be organized under these, whereas the bottom-up method involves iden-tifying the most specific concepts first and then grouping these into more generic concepts.

Nonetheless, since many ontologies are created for a specific purpose or use in mind, the goal of the application will often have some influence on the way that an ontology is constructed. This is analogous to the process of developing the schema for a relational database, since the schema is an organization and specification of the types of data that will be accessed and processed by the applications that interact with the database. Thus, it becomes critical to precisely define the role that an ontology will play within a particular software application as well as the overall goal of the application.

Likewise, before the construction of an ontology actually begins, another important step for a designer is to determine whether similar ontologies have already been created by others so that knowledge can be reused or extended. The DAML Ontology Library is a worthwhile starting point for viewing preexisting ontologies organized by domain. [8] Yet, even if similar ontologies do exist, they may not be structured in a way that is useful to the designer in his or her application since the existing ontology may have been created for a very different application or from a very different perspective on the domain. This conclusion cannot be reached, however, until the designer invests the time to fully understand the similar ontology and its layout.

In general, the process of reusing an existing ontol-ogy by extending new concepts from the already defined concepts is desirable since it minimizes the development effort as well as promotes interoperability with other applications that use the same ontology. For instance, when developing a domain specific ontology there is always an opportunity to extend from more generic ontologies that exist within the same domain to reuse previously defined class structure and axioms. A logic axiom, in the context of an ontology, is a formal, mathematical definition of a constraint or relationship between various concepts. For example, the rule that is listed in Figure 2 (see previous page) illustrates the general structure of such an axiom in natural language. As such, axioms essentially provide meaning to a concept, and their mathematical description allows them to be defined procedurally for execution in software; i.e., machine processable.

Our research examined this particular reuse option while trying to link a superset of the ontology of military facilities in Figure 1 (see previous page) to the Suggested Upper Merged Ontology (SUMO). An upper ontology, such as SUMO, “provides definitions for general-purpose terms and acts as a foundation for more specific domain ontologies.” [9] Superficially, the concept of “Facility” in the Figure 1 ontology appears closely related to the concept “Building” in SUMO, based on the natural language definitions of the two terms along with the KIF axioms that are present in SUMO—however, we found some potential conflicts with this extension. Namely, a “Facility” can be a mobile entity that is not always fixed in location; i.e., mobile missile site, whereas the class “Building” in SUMO is a subclass of “StationaryArtifact” which, by its KIF axiom definition, gives it a fixed location. To reconcile this disagreement, our “Facility” class would instead be declared as a subclass of the “Artifact” concept in SUMO (parent class of “StationaryArtifact”), since “Artifact” is generic enough to encompass all variations of the “Facility” class. Deciding which concept or node to start extending from within an ontology is often one of the greatest challenges in trying to reuse or link multiple ontologies by extension. This mapping, nonetheless, gives us the opportunity to relate our ontology to many other ontologies through the use of SUMO.

Overall, the development of an ontology is always a complex and iterative procedure. In general, reusing components of existing ontologies or extending from existing ontologies is always desired so as to minimize the overall development effort and take advantage of the structure and axioms of the already defined ontology. However, care must be taken to avoid any potential conflicts or contradictions between con-cepts that are linked between multiple ontologies. Several approaches and tools exist for checking for such consistency. [10] In the following section, we will investigate how DAML is used as a language for expressing ontologies.

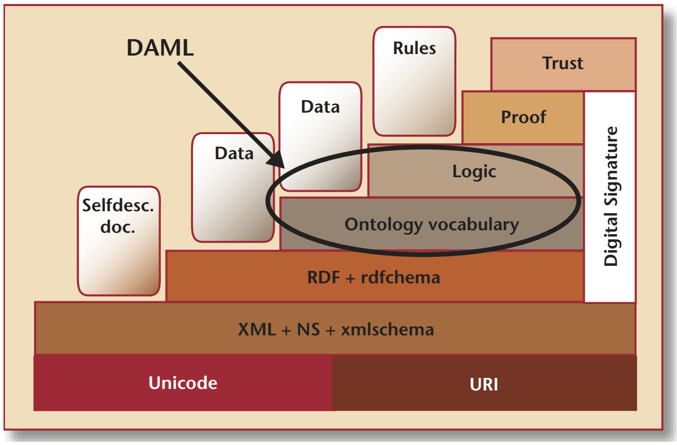


Figure 3. Layer diagram showing relation between DAML, XML, RDF, and RDFS [18]

DAML Constructors	DAML Axioms
unionOf	subClassOf
intersectionOf	sameClassAs
complementOf	subPropertyOf
oneOf	samePropertyAs
toClass	disjointWith
hasClass	sameIndividualFrom
hasValue	differentIndividualFrom
minCardinalityQ	inverseOf
maxCardinalityQ	transitiveProperty
cardinalityQ	uniqueProperty
	unambiguousProperty

Figure 4. Listing of DAML constructors and axioms (adapted from [12])

Use of DAML for knowledge representation

Fundamentals of DAML

An ontology can be represented using a variety of techniques ranging from graphical depictions, such as shown in Figure 1 (see page 6), to formal knowledge representation languages. DAML is one such knowledge representation language, based on Web standards, that is capable of expressing an ontology in a machine processable format. It is builds upon existing W3C standards, including XML, RDF, and RDFS, to provide an expressive metadata description lan-guage. As a result, valid DAML syntax is also valid XML syntax, and DAML classes and properties can be uniquely identified and referenced over the Web by a Uniform Resource Identifier (URI) just as with RDF and RDFS. This technology progression can be viewed in the layer diagram shown in Figure 3 (see page 7) where DAML constitutes the ontology vocabulary and logic layers.

DAML focuses on describing information on the Web, rather than displaying information for human readability, which is the main function of the HyperText Markup Language (HTML). Specifically, DAML describes the structure of a domain using classes and properties, and it asserts relationships between classes or properties via its unique set of constructors and axioms. The DAML constructors and axioms have a logical foundation in Description Logic (DL), where a DL is a "very carefully restricted logic that aim[s] to capture concepts and relationships between concepts." [11]

> *The basic building blocks [of a DL] are concepts, roles and individuals. Concepts describe the common properties of a collection of individuals and can be considered as unary predicates which are interpreted as sets of objects. Roles are interpreted as binary relations between objects. Each description logic defines also a number of language constructs (such as intersection, union, role quantification, etc.) that can be used to define new concepts and roles. [12]*

The classification of DAML as a DL is a key strength of the language since DLs are known to be computationally decidable and tractable, meaning that inference services performed using a DL will always reach an outcome within a reasonable amount of time.

```
<daml:Class rdf:ID=”Facility”>
    <daml:subClassOf>
      <daml:Restriction>
        <daml:onProperty rfd:resource=”#name”/>
        <daml:toClass rdf:resource=”&xsd:string”/>
      </daml:Restriction>
    </daml:subClassOf>
</daml:Class>
```

Figure 5a. Class "Facility" has "name" property restricted to datatype "string" (xsd:string)

```
<daml:Class rdf:ID=”Facility”/>

<daml:Class rdf:ID=”RadarFacility”>
    <daml:subClassOf rdf:resource=”#Facility”/>
</daml:Class>
```

Figure 5b. Class "RadarFacility" is a subclass (child) of class "Facility" (parent)

Furthermore, since DAML is built upon the existing Web standards of RDF and RDFS, any DAML ontology can ultimately be decomposed into a sequence of triples, where a triple is the fundamental unit of information representation in RDF. Essentially, a triple is a three-element set that consists of a subject, predicate, and object. The subject corresponds to a general concept (class) within an ontology, while the predicate (property) corresponds to an attribute that describes the subject. The object portion of a triple represents the value of the predicate; i.e., property value.

In general, the "expressive power of the language is determined by the class (and property) constructors supported, and by the kinds of axioms supported," [11] and DAML consists of ten different class constructors and 11 different axioms as indicated in Figure 4 above.

For example, the "toClass" constructor can be used to define the datatype of a DAML property, and the "subClassOf" axiom can be used to assert a parent-child hierarchical relationship between two different classes. A demonstration of how these two particular DAML constructs would be used and expressed in DAML syntax is displayed in Figures 5a and 5b (see above) using the concepts from the ontology defined previously in Figure 1.

The "daml" prefix that appears before each of the DAML language constructs in Figures 5a and 5b indicates a reference to the "daml" namespace, which has its own URI, where all of the DAML language constructs are defined. [13] The concept of namespace is important to DAML since it provides a mechanism for associating

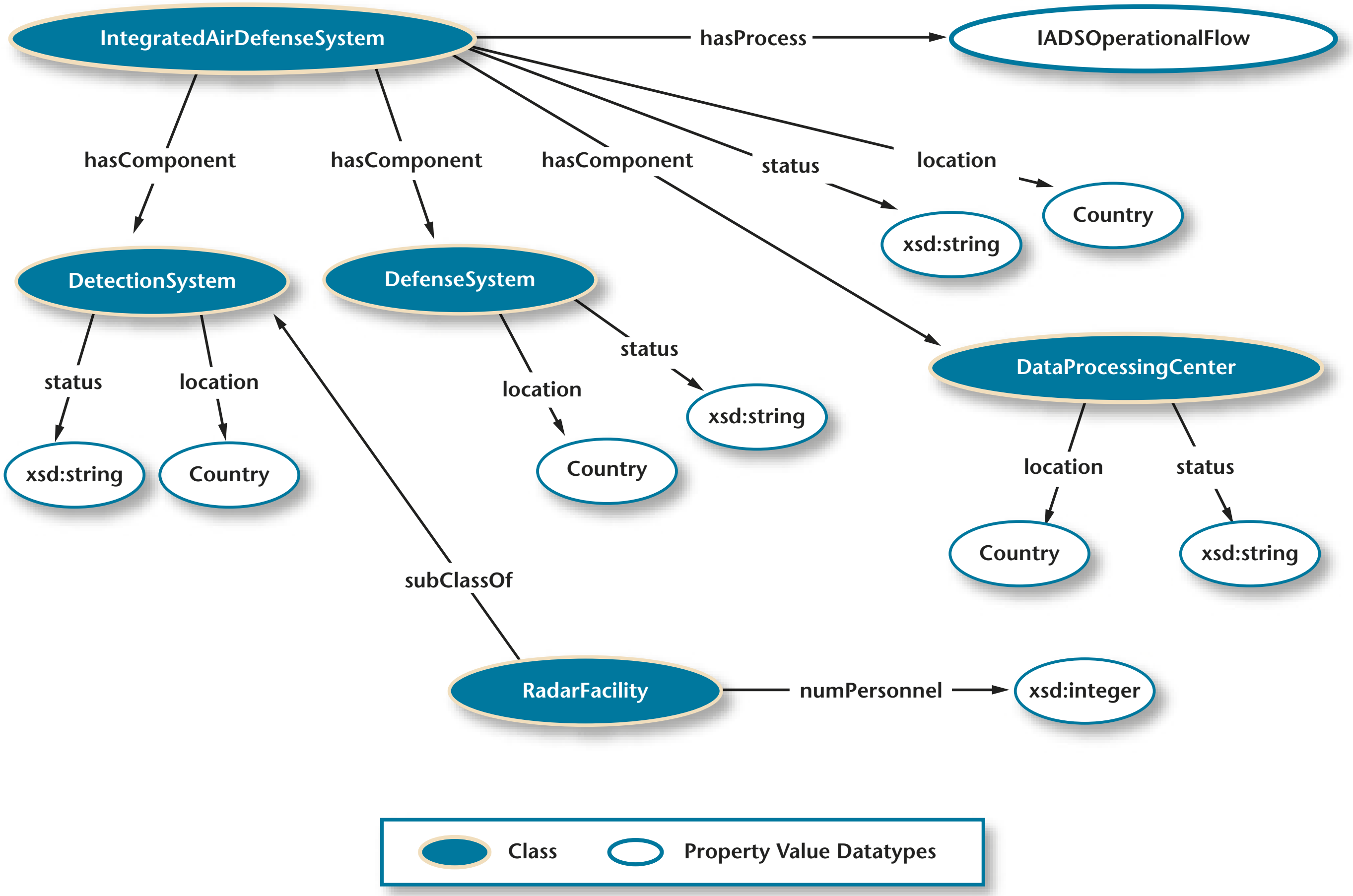


Figure 6. Graph model of top-level IADS ontology showing reuse of "DetectionSystem" class

ontologies with URIs, thus making it possible to build vocabularies of reusable terms that are accessible over the Web for discovery and reuse by software. Namespaces also make it possible for ontology designers to post multiple versions of the same ontology on the Web and be able to distinguish them by their namespace URI. For a complete description of all DAML language elements, refer to the DAML Reference Description document. [14] Moreover, to avoid any ambiguity in interpretation of the language constructs, the axiomatic semantics for DAML provides a complete mapping of all DAML language elements into a first-order predicate calculus logical theory expressed in Knowledge Interchange Format (KIF). [15]

Recapping, the finite set of constructors, axioms, and other elements that constitute the DAML language are the building blocks for expressing a given ontology in DAML. Once an ontology is designed at the conceptual level, such as a connected graph model, it is then necessary to decide how to organize this into classes and properties by using the DAML language elements. Typically, concepts correspond to classes in a DAML ontology and the attributes that describe and define a particular concept become the properties that are associated with this class. The relationships between the various classes and properties within the ontology must then be expressed (Figures 5a and 5b see page 8) using the set of axioms and constructors that DAML provides, as listed in Figure 4 (see page 8).

Case study— Expressing an ontology in DAML for an IADS

One particularly important piece of intelligence information to the United States Air Force when execut-ing airborne mission planning is the identification of an Integrated Air Defense System (IADS). An IADS is "a collection of sensors, Command, Control, Communications, and Intelligence (C3I) systems, and active defense systems used to protect an area from hostile air systems." [16] Thus, any IADS is characterized by the presence of three main physical components—

1. Detection system
2. Communications and data processing infrastructure
3. Weapons system

An IADS can further be described as a sequential process since it is characterized by distinct processes, namely the detection and tracking of a potential enemy threat, the communication of this intelligence to a data processing center, decision making activity based on the evidence gathered, the identification of the potential threat, and potential engagement with the identified target. However, this sequential process and temporal definition of an IADS will not be modeled in our example ontology. The fact that an IADS is a complex system currently requires intelligence analysts to gather and correlate information from numerous sources before conclusions can be drawn. As a result, one research objective that was explored involved the automation of this process in software using DAML-encoded intelligence data in conjunction with an inference engine in order to infer the existence of an IADS from a set of instance data.

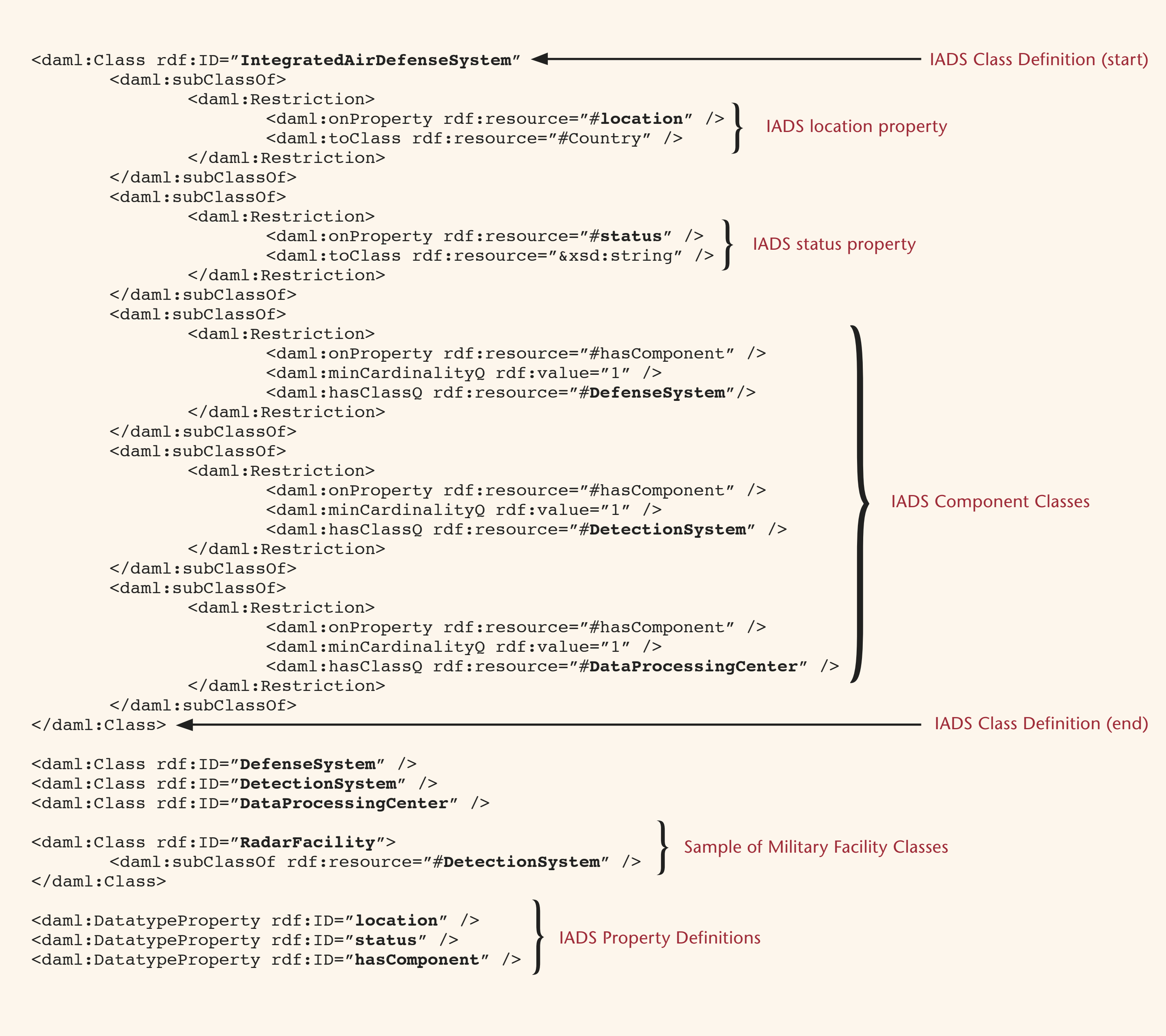
Given this knowledge of the domain, we first set out to model the concept of an IADS with an ontology, and sought to reuse and extend concepts from already defined ontologies to minimize the development effort and maximize interoperability. A graph model of the high-level IADS ontology is represented in Figure 6 (see page 9).

Notice that portions of the radar facility ontology from Figure 1 (see page 6) can be "reused" in the IADS ontology by simply declaring the "RadarFacility" class to be a subclass of "DetectionSystem," which is one of the three main components of an IADS. This enables access to radar concepts and properties that are connected to this class from the ontology in Figure 1. Care must be taken to resolve potential conflicting properties between a Detection System and the Facility concepts.

Once the class and property framework was outlined for the IADS ontology, it was then necessary to place constraints on the classes and properties to further refine the meaning of an IADS. In our simplified view of this domain, we defined an IADS to be operational only when the following two conditions were true—

1. The values of the "status" property for the three component classes of IADS (DetectionSystem, DefenseSystem, and DataProcessingCenter) all must be "Operational"

2. The values of the "location" property for the three component classes must be equal



```
<daml:Class rdf:ID=”IntegratedAirDefenseSystem”
        <daml:subClassOf>
                <daml:Restriction>
                        <daml:onProperty rdf:resource=”#location” />
                        <daml:toClass rdf:resource=”#Country” />
                </daml:Restriction>
        </daml:subClassOf>
        <daml:subClassOf>
                <daml:Restriction>
                        <daml:onProperty rdf:resource=”#status” />
                        <daml:toClass rdf:resource=”&xsd:string” />
                </daml:Restriction>
        </daml:subClassOf>
        <daml:subClassOf>
                <daml:Restriction>
                        <daml:onProperty rdf:resource=”#hasComponent” />
                        <daml:minCardinalityQ rdf:value=”1” />
                        <daml:hasClassQ rdf:resource=”#DefenseSystem”/>
                </daml:Restriction>
        </daml:subClassOf>
        <daml:subClassOf>
                <daml:Restriction>
                        <daml:onProperty rdf:resource=”#hasComponent” />
                        <daml:minCardinalityQ rdf:value=”1” />
                        <daml:hasClassQ rdf:resource=”#DetectionSystem” />
                </daml:Restriction>
        </daml:subClassOf>
        <daml:subClassOf>
                <daml:Restriction>
                        <daml:onProperty rdf:resource=”#hasComponent” />
                        <daml:minCardinalityQ rdf:value=”1” />
                        <daml:hasClassQ rdf:resource=”#DataProcessingCenter” />
                </daml:Restriction>
        </daml:subClassOf>
</daml:Class>

<daml:Class rdf:ID=”DefenseSystem” />
<daml:Class rdf:ID=”DetectionSystem” />
<daml:Class rdf:ID=”DataProcessingCenter” />

<daml:Class rdf:ID=”RadarFacility”>
        <daml:subClassOf rdf:resource=”#DetectionSystem” />
</daml:Class>

<daml:DatatypeProperty rdf:ID=”location” />
<daml:DatatypeProperty rdf:ID=”status” />
<daml:DatatypeProperty rdf:ID=”hasComponent” />
```

IADS Class Definition (start)
IADS location property
IADS status property
IADS Component Classes
IADS Class Definition (end)
Sample of Military Facility Classes
IADS Property Definitions

Figure 7. Sample portion of DAML-encoded IADS ontology

Thus, to constitute an operational IADS, according to our ontology, there must exist at least one instance of each of the three component classes with a status value of "Operational" and a common location value among them as well. A sample of the DAML-encoded IADS ontology is listed below in Figure 7 (see page 10).

Note that the class definition for "IntegratedAirDefenseSystem" in the above ontology makes use of the cardinality construct from the DAML language to constrain the "hasComponent" property. Setting a "minCardinalityQ" of one (1) for the three different component classes of an IADS ensures that at least one instance of each of these three classes must exist in order for an "IntegratedAirDefenseSystem" to exist.

Inference applications of DAML

Once an ontology is completely expressed in valid DAML syntax, instance data can then be associated with the classes and properties of the ontology to form a knowledge base, which can be accessed and manipulated by software applications for inferencing. Given the logical foundations of DAML, including its set of constructors and axioms, it is a natural fit for a variety of inference-based artificial intelligence applications. The next step is to develop the application to exploit the knowledge represented by the ontology. In our case, it was to create an inference and query application that would automatically deduce when an IADS was present based on the presence of its components. Currently, intelligence analysts must gather data from various sources and analyze the combined evidence to conclude the presence or absence of an enemy IADS, and this can be a time consuming process. The following sections will highlight the various software tools; i.e., parsers, inference engines, etc., that were incorporated in this experiment to help automate the IADS identification process as well as include an overall analysis of the inference being performed.

Tools for inferencing with DAML— DAMLJessKB and JESS

At the time, there were no reasoning (inference) engines that could process DAML content directly. As a result, creat-ing an application that applies an inference engine to a DAML ontology or knowledge base required translation of the DAML syntax into the knowledge representation language used by the core of the inference engine itself. One particular DAML tool that already captures most of the logical axioms of the DAML language for use with an inference engine is the DAMLJessKB Java Application Programmer Inference (API), which was developed by researchers at Drexel University. [16] This API essentially allows a DAML ontology to be translated into JESS syntax (see Figure 8 below). DAMLJessKB expresses many of the DAML language axioms as "rules" in the JESS (Java Expert System Shell) language so that they can be executed by the JESS inference engine. Likewise, any additional axioms that represent domain specific knowledge can also be imported by JESS, as denoted by the "domain axioms" block in Figure 8. The general structure of a rule definition in JESS is the standard "if-then" statement (conditional), which is read by the JESS inference engine and compared against its currently stored list of facts. JESS is not only a declarative programming language, but also consists of an inference engine that is based on the Rete algorithm as well as a Java API to support application development.

The DAMLJessKB API is essentially a DAML "wrapper" on the JESS architecture, since it uses JESS for its inference capabilities while providing the DAML axioms expressed in JESS to automatically generate conclusions on a given DAML ontology or knowledge base.

Case study— Generating inferences on DAML IADS ontology

Once the IADS ontology was formulated and expressed in DAML, the next step was to associate actual instance data with the IADS concepts, thus creating a knowledge base that could be manipulated by software. Our knowledge base was constructed as a set of military facility instances using MIDB as the source of instance data and the DIA Category Code listing as the guideline for mapping database tables and columns into classes and properties in the IADS ontology. The Modernized Integrated DataBase (MIDB) is a standard military database containing vast amounts of information on items such as facilities, equipment, and events. For example, MIDB contains status and location information on radar facilities, which was used as property instance data for any instances of the class "RadarFacility" in the IADS ontology. For our prototype application, we used JESS as an inference engine and DAMLJessKB for reading the IADS ontology and sample knowledge base of military facility instances. The DAMLJessKB Java API was used to parse the ontology and knowledge base, using the Jena Java API as its parser, into a sequence of triples (JESS facts) that could be ingested into the JESS inference engine. [18] JESS would then execute any of the rules that matched its list of facts derived from the IADS knowledge base and store any new inferred facts.

One of the common inferences that was performed by this software prototype was a standard subsumption (subclass) inference between the military facility classes and the three components classes of IntegratedAirDefenseSystem. For example, if an instance of a RadarFacility existed in the knowledge base, then the inference engine would also generate a new fact that there was an instance of a DetectionSystem as well, since a RadarFacility is defined to be a subclass of a DetectionSystem in the IADS ontology (see Figures 6 and 7 on pages 9 and 10). Likewise, when an instance of a DetectionSystem, DefenseSystem, and DataProcessingCenter were found to coexist in the knowledge base with equivalent location property values and status values of "Operational," the inference engine would generate the conclusion that an IADS was present.

However, this inference was not possible using the language axioms of the DAML language alone in the rule set of JESS, since DAML is actually not capable of express-ing

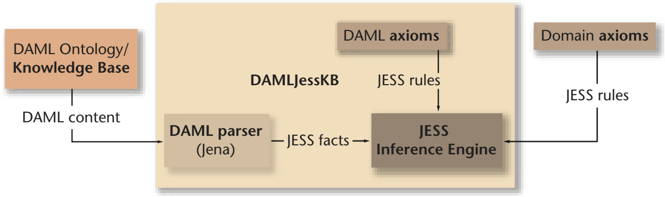


Figure 8. Block diagram of DAMLJessKB and JESS inference engine, adapted from [11]

the various constraints on the location and status predicates of the three components of an IntegratedAirDefenseSystem, as mentioned previously in our defini-tion of an IADS. Namely, the inference using DAML alone does not take into account the fact that the three IADS components (DetectionSystem, DataProcessingCenter, and DefenseSystem) must all have the same location value and must have a status value of "Operational." This is due to an expressivity limitation with the DAML language, since DAML does not allow an ontology designer to specify a constraint on property values that are not constants.

Thus, we are led to investigate other means of represent-ing these important constraints in the definition of IADS so that the correct inferences are made, and this involves the introduction of domain-specific rules that capture detailed knowledge of the domain in question. The follow-ing section will address this issue and the general need for external, domain-specific rules to supplement the axioms of DAML.

Rules as a supplement to DAML axioms

As the IADS inferencing case study has proven, there are certain limitations in the expressivity of DAML when trying to develop an ontology that accurately describes the world. For this case, in particular, it was not possible for us to express that an IADS exists only when the location values of its three component classes have the same value and the status values of these same classes have a value of "Operational." This suggests the need for a more expressive "rule language" to supplement the language constructs of DAML in order to accurately describe a particular domain. For example, the RuleML effort and DAML-Rules (DAML-R) working group were initiated to propose a solution to this problem, but there is no standard web rule lan-guage at this time. [19]

For the IADS inference application, we found that the JESS language for expressing rules was sufficient, since it allowed for property values to be expressed as variables. The solution, then, was to construct a rule in JESS syntax to capture the constraints on "location" and "status" for the three IADS component classes. The basic format of this rule, independent of a particular rule representation language, was as follows—

```
if ((DetectionSystem.location==DataProcessingCent
er.location==DefenseSystem.location==?X)&
(DetectionSystem.status==DataProcessingCenter.
status==DefenseSystem.status==“Operational”))

then (IntegratedAirDefenseSystem exists in
location ?X)
```

Once this is expressed in JESS syntax, the domain specific rule could then be imported into the JESS inference engine, along with the rules corresponding to the DAML language constructors and axioms, to enhance the inferencing capabilities of the application. In this example, the variable "?X" represents any location value that may exist in the knowledge base. JESS internally handles all variables and rules by matching facts against variables in the rules to determine when a particular rule should fire. Ultimately, the JESS inference engine was able to successfully deduce that an IADS was present when the necessary conditions existed in the IADS knowledge base due to the addition of this external rule.

Scalability

The case studies that have been presented thus far have dealt with ontology development as well as inferencing in very small scale applications using rather small data sets. For example, given that the Air Force collects and records large amounts of intelligence data on a daily basis and analysts must review, extract, and correlate data from multiple sources to identify an IADS the next step would be to see how well these technologies scale in a more computationally demanding environment. Understanding the computational performance of the underlying inference algorithm of a reasoner can provide much insight into this question even before experimentation is actually carried out.

Most inference systems involve some form of pattern matching between a set of rules (conditionals) and a set of facts, where any fact or combination of facts that satisfy the antecedent of a rule cause the rule to execute and perform an action. However, the means in which this matching between rules and facts actually occurs is defined by the inference algorithm of the reasoning system. For example, one approach is to have an algorithm that continuously cycles through the list of rules and tries to match up the antecedent of every rule against the data currently stored in the knowledge base. However, this is a very inefficient technique with an order of performance that is proportional to O(RFP), where "R" symbolizes the number of rules in the inference engine, "F" represents the number of facts in the knowledge base, and "P" represents the average number of patterns or expressions in the ante-cedent of every rule. [20] Thus, as rule antecedents become more and more complex, causing more pattern matches to be done, or the number of facts in the knowledge base increases, the performance of this inference algorithm tends to degrade exponentially.

However, JESS employs the efficient Rete algorithm for its inference services which minimizes such inefficiencies by remembering past test results across iterations of the rule loop. The end result is a major improve-ment in the computational complexity of inferencing, since the performance of Rete is on the order of O(RFP). Thus, the computational time of an inference application, such as JESS, that employs the Rete algorithm should be expected to increase linearly based on the number of facts (F) in the knowledge base, assuming that the rule set does not change often. Testing these theories in a large scale information processing environment, however, would be valuable in testing the limits and feasibility of using an ontology representation language, such as DAML, in an inference-based application.

Conclusion

Effective information management and exploitation is an increasing concern not only for the Air Force but for any communities that manage large repositories of information that provide critical input for decision making processes. As we have learned, ontologies provide a useful mechanism for representing complex data relationships so that various reasoning techniques can be applied to such information stores to make simple inferences. Such capabilities are not possible with more primitive forms of data storage, including database and XML schemas, where only simple relationships among data can be expressed.

Nonetheless, the process of developing an ontology is often more involved and requires the selection of a suitable knowledge representation language to serve as the medium for ontology expression. DAML was analyzed as one particular ontology representation language for describing a domain and uses existing Web standards, including XML, RDF, and RDFS, as its foundation. Our case studies in ontology development demonstrated how ontology reuse and extension can minimize overall development time as well as provide access to an already defined domain of knowledge. We then later showed how an already created ontology and knowledge base expressed in DAML could be leveraged for inferencing using popular software tools and APIs, such as DAMLJessKB and JESS.

However, the IADS case study proved that DAML's language elements alone are not always expressive enough to accurately describe a domain or concept. Thus, in certain circumstances, it may be necessary to import rules that are defined in more expressive rule languages to supplement the axioms and constructors of DAML. Future efforts by the DAML-R working group will dictate whether a universal standard rule language will ultimately materialize. ■

About the Authors

Alan Nawoj

Alan Nawoj has worked as an Associate and Technical Lead at Booz Allen Hamilton, supporting the Air Force Research Laboratory. He graduated from Harvard University with an M.S. degree in Computer Science and graduated summa cum laude from Cornell University with a B.S. degree in Electrical and Computer Engineering.

Mark Gorniak

Mark Gorniak is the Air Force Research Laboratory (AFRL) Program Manager for the DARPA Agent Mark-up Language (DAML) Program. He holds a M.S. degree in electrical engineering from Syracuse University, and a B.S. degree in electrical engineering from the Rochester Institute of Technology. He may be reached at Mark.Gorniak@rl.af.mil.

References

1. DAML has recently become part of the W3C Web Ontology Language (OWL) found at http://www.w3.org/TR/owl-ref/
2. Air Force Research Laboratory Information Directorate Mission Statements. "AFRL Missions Statements," April 2003. http://www.rl.af.mil/mission/missions.html
3. Network Inference. "Cerebra Inference Engine: Competitive Differentiation" 2002.
4. Noy, Natalya F. and McGuinness, Deborah L. "Ontology Development 101: A Guide to Creating Your First Ontology" Stanford Knowledge Systems Laboratory Technical Report KSL–01–05, March 2001.
5. Hayes, P "Subject: RE: some basic questions, thanks" Sep 21, 2003, Posting to www-rdf-logic@w3.org: http://lists.w3.org/Archives/Public/www-rdf-logic/
6. McGuinness, Deborah L., and van Harmelen, Frank "OWL Web Ontology Language Overview" http://www.w3.org/TR/owl-features/
7. Finin, T., Geyer-Schulz, A., and Dyer, C. "Logical Reasoning Systems (Chapter 10)" Notes for CMSC 671, University of Maryland Baltimore County, Fall 2002. http://www.csee.umbc.edu/~ypeng/F02671/lecture-notes/Ch10.ppt
8. DAML Ontology Library. Updated August 2003. http://www.daml.org/ontologies
9. Niles, I., and Pease, A. "Towards a Standard Upper Ontology". In Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), Chris Welty and Barry Smith, eds, Ogunquit, Maine, October 17–19, 2001.
10. Horrocks, I. "Description Logic: Axioms and Rules" http://www.cs.man.ac.uk/~horrocks/Slides/dag-stuhlS070202.pdf. February 2002
11. Horrocks, Ian. "DAML+OIL: a Description Logic for the Semantic Web" Department of Computer Science, University of Manchester, 2001.
12. Lambrix, P. "Description Logics" (Overview) Department of Computer and Information Science, Linkoping University, Linkoping, Sweden.
13. DAML Language Specification (March 2001). http://www.daml.org/2001/03/daml+oil. 2001
14. Connolly, D., Horrocks, I., McGuinness, D., Patel-Schneider, P., Stein, L., van Harmelen, F. "DAML+OIL (March 2001) Reference Description." http://www.w3.org/TR/daml+oil-reference
15. Fikes, R. and McGuinness, D. "An Axiomatic Semantics for RDF, RDF-S, and DAML+OIL". Stanford KSL Technical Report KSL–01–01, 2001.
16. Fruchey, D. "Applications from David B. Fruchey" http://www.ams.org/careers/dfruchey-app.html
17. Kopena, Joe and Regli, William C. "DAMLJessKB: A Tool for Reasoning with the Semantic Web" Geometric and Intelligent Computing Laboratory, Department of Computer Science, Drexel University, October 28, 2002.
18. McBride, B., Seaborne, A., and Carroll, J. "Jena Tutorial for Release 1.4.0". http://www.hpl.hp.com/semweb/doc/tutorial/index.html. April 2002
19. Boley, H. The Rule Markup Initiative (RuleML Homepage). July 2003. http://www.dfki.uni-kl.de/ruleml/
20. Friedman-Hill, E. J. "JESS, The Rule Engine for the Java Platform" Distributed Computing Systems, Sandia National Laboratories. Version 6.1p2. May 2003.

Acknowledgements

This research was funded by the Air Force Research Laboratory, Rome Research Site, Information Awareness and Understanding Branch (IFTB) under contract with Booz Allen Hamilton and was supervised by Mark Gorniak.

Agent-Based Software System

by Michael Colon

Agent-based software systems are a new and emerging technology that has attracted a great deal of attention over the last few years. Agent-based software systems provide a framework that extends distributed, open, and dynamic computing environments. The framework enables software agents to communicate with each other productively in a rapidly changing heterogeneous networked environment. Agents work best at solving complex, distributed problems that are difficult to achieve with existing non-agent based technology. There are a number of organizations conducting and applying agent-based research and development, and they have generated new opportunities in numerous application domains. Such domains include Autonomic Computing, Grid Computing, and the Semantic Web. Although the agent-based software system architecture helps to solve many complex problems, they have some significant issues that need to be addressed before they are widely and confidently accepted. Lack of mature standard software analysis and design methodologies, limited defined specification requirements, and a considerable number of conventional and new security threats are some of the most frequently considered challenges. Agent-based software systems are generating considerable momentum and the benefits of these systems are gaining noticeable attention in the academic, commercial, and government sectors.

Currently, there is not a universal, clear and succinct textbook definition of an agent, but there is a common understanding of its goals. In their article "Software Agents," Nick Jennings and Michael Wooldridge describe an agent as a self-contained program capable of controlling its own decision-making and acting, based on its perception of its environment, in pursuit of one or more objectives. [1] Agents are authorized to act on the behalf of a human, a computer program, or another software agent. Agents can communicate with other agents, and in some cases may compete with each other, as in a multi-agent system. A community of agents often works together to achieve mutual benefits. Agents take advantage of distributed computing environments by either physically replicating them-

selves onto other host computers, communicating with other agents residing on other hosts, or by accessing data contained on a remote host. Individual agents report back to agent platforms that collaborate and interpret the data gathered by the agents.

There are a number of characteristics that have not been commonly combined in standardized software systems, but which uniquely make up an agent. Agents are autonomous, reactive, proactive, adaptive, communicative, and sometimes mobile.

In Franklin and Graesser's paper "Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents," the authors observe that an autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it over time, in pursuit of its own agenda and so as to affect what it senses in the future. [2] This enables an agent to act independently in a dynamic and unpredictable environment.

According to Joanna J. Bryson and Lynn Andrea Stein, an agent's reactive characteristic enables it to sense and act on changes in its environment. [3] Reactive intelligence controls a reactive agent—one that can respond very quickly to changes in its situation.

In *Declarative and Procedural Goals in Intelligent Agent Systems*, Michael Winikoff, Lin Padgham, James Harland, and John Thangarajah state that an agent is proactive in that it organizes its behavior in order to achieve intended goals. [4] The goals have two aspects—

- **Declarative**—where a goal is a description of the state of the world, which is sought
- **Procedural**—where a goal is a set of procedures that is executed (in an attempt) to achieve the goal

Adaptive agents must be able to react to a simple stimulus, and be able to make a direct, predetermined response to a particular event or environmental action. More advanced adaptive agents (i.e., autonomic agents), can reason and have the capacity to learn and evolve.

Agents exercise communicative characteristics that enable them to interact with agent platforms, other agents, and humans. Numerous protocols are needed to accomplish different types of communication. The Foundation for Intelligent Physical Agents (FIPA) is developing specifications of the communication languages, as well as libraries that have predefined communicative act types, interactions, and protocols for agents.

Mobile agents are programs, typically written in a script language, which may be dispatched from a client computer and transported to a remote server computer for execution. According to Tim Finin, Yannis Labrou, and Yun Peng of the University of Maryland, Baltimore County Department of Computer Science and Electrical Engineering, mobile agents can benefit from standards efforts on inter-agent communication. [5] Mobile agents travel to

on displayResult displayWhich
ext = string(searchResult.rows[displayWhich][1])
_apa").text = searchResult.rows[displayWhich][8]
act").text = searchResult.rows[displayWhich][10]
ry1").text = searchResult.rows[displayWhich][11]
Result.rows[displayWhich][12]
Result.rows[displayWhich][13]
Result.rows[displayWhich][14]
-- SLIDE OPTI
Options whichDirection, speed
if whichDirection = "open" then
max = searchOptionGroup[1][2]
speed = 0 - speed
repeat while searchOptionGroup[1][1].locH >= max
ionGroup.count
].locH + speed
end repeat
end repeat
else
max = searchOptionGroup[1][3]
speed = 0 + speed
repeat while searchOptionGroup[1][1].locH <= max
on startMovie
pdfList = getPdfList()
populateList()
arcaRegister([1090,0073,3064])
gDB = new(xtra "arca")
gDB.openDB(movie.path & "data")
searchOptionGroup = []
searchOptionGroup.append([sprite(19),494,
searchOptionGroup.append([sprite(20),458,
searchOptionGroup.append([sprite(22),399,
searchOptionGroup.append([sprite(23),509,
searchOptionGroup.append([sprite(24),399,
searchOptionGroup.append([sprite(25),509,778])
searchOptionGroup.append([sprite(26),399,668])
searchOptionGroup.append([sprite(27),572,841])
searchOptionGroup.append([sprite(28),361,620])
on stopMovie
forget window "doc view"
gDB.closeDB()
gDB = 0
-- Custom functions
ET PDF LIST

Table 1. Agent-based technology challenges [6]

Challenge	Technology Example	Example Application
Increase quality of agent software to industrial standard	Agent oriented design methodologies, tools and development environments	Complex systems development
	Seamless integration with current technologies (SOAP, Web Services, etc.)	
Provide effective agreed standards to allow open systems development	Agent communication languages (FIPA)	eCommerce
	Interaction protocols	
	Multi-agent architectures	
Provide semantic infrastructures for open agent communities	Semantic Web	Agentcities
	Ontologies	
	Matchmaking and broker architectures	
	Electronic institution design	
Develop reasoning capabilities for agents in open environments	Negotiation algorithms	eCommerce eScience
	Planning and DBI architectures	
	Coalition building	
	Ontological reasoning	
Develop agent ability to understand user requirements	User profiling	Information agents
	Personalization	
	Utility modeling	
	Knowledge acquisition tools	
Develop agent ability to adapt to changes in environment	Learning	Social Simulation
	Evolutionary programming techniques	
Ensure user confidence and trust in agents	Security technologies	Medical Informatics
	Deception-proof interaction protocols	E-Government
	Models and infrastructure for trust and reputation	

host platforms and execute using the remote host's computing resources. This is useful because many agents can be distributed throughout a network and to perform calculations or data mining and report back to a centralized source. This technology is not new—Java Applets have successfully implemented mobile code for years.

Technology leaders

Commercial and government agencies that are leading the way in agent-based software initiatives include—

- **International Business Machines (IBM)**—Autonomic Computing, Grid Computing and the Semantic Web
- **Defense Advanced Research Projects Agency (DARPA)**—Control of Agent-Based Systems (CoABS); DARPA Agent Markup Language (DAML)
- **National Institute of Standards and Technology (NIST)**—Agent-based intrusion detection
- **AgentLink, Europe's Network of Excellence for Agent-Based Computing**—Agent Based Roadmap

Current and future applications

Currently, there are three significant efforts underway that utilize the agent-based software architecture. They are autonomic computing, grid computing, and the semantic Web. These technologies take advantage of the unique characteristics of agents, and they have been shaped by the agent-based architecture.

1. **Autonomic computing** is an approach to self-managed computing systems with a minimum of human interference. According to IBM, the industry "thought leader" in autonomic computing, the term "autonomic" derives from the body's autonomic nervous system, which controls key functions without conscious awareness or involvement. Autonomic computing was conceived by IBM Research (among others) to lessen the spiraling demands for skilled information technology resources, reduce complexity, and drive computing

into a new era that may better exploit its potential to support higher order thinking and decision making. Autonomic elements use agent-based software technology to continuously sense and respond to their environment.

2. **Grid computing** enables the virtualization of distributed computing and data resources such as processing, network bandwidth, and storage capacity to create a single system image, granting users and applications seamless access to vast IT capabilities. According to IBM, which is also a leader in implementing grid technologies, just as an Internet user views a unified instance of content via the Web, a grid user essentially sees a single, large virtual computer. Intelligent agents provide a useful means to achieve grid computing objectives.

3. **The Semantic Web** is an extension of the current Web in which structure and meaning is added to Web content. Currently, the Web is HTML-based and is primarily concerned with distributing information to Web browsers in a pleasant, human readable format. The semantic Web attempts to add a layer of sophistication to the existing Web by facilitating standard ways to mark data for meaning, to enable intelligent searching. The semantic Web enables software agents to roam Web sites to find the true meaning of a site's content, allowing for much more focused searches of information.

The future of agent-based computing depends on the level of success that is achieved in a number of disciplines. These areas include but are not limited to the challenges described in Table 1 (see page 16).

Standards efforts

The following are agent-based software standards efforts that are currently underway—

- **Foundation for Intelligent Physical Agents (FIPA)**—Open Standards and Open Source for Agent-Based System
- **Agent Unified Modeling Language (UML)**—Bauer 2001
- **Object Management Group (OMG) Agent Platform Special Interest Group**—Working to extend the OMG Object

Table 2. Categories of agent-based software threats [7]

Threat Vector	Threat	Description
Agent-to-Platform	Masquerading	Unauthorized agent claiming the identity of another agent in order to gain access to services and resources provided by the platform
	Denial-of-service	Mobile agent launching attacks by consuming excessive amounts of the agent platform's computing resources
	Unauthorized access	Unauthorized users or processes accessing services and resources to which they have not been granted permission
Agent-to-Agent	Masquerading	Unauthorized agent claiming the identity of another agent in order to gain access to services and resources provided by another agent
	Denial-of-service	Mobile agent launching attacks by consuming excessive amounts of another agent's computing resources
	Repudiation	Agent denies that a transaction or communication ever took place
	Unauthorized access	Agent interferes with another agent by invoking its public methods, or by accessing and modifying the victim agent's data or code
Platform-to-Agent	Masquerading	Agent platform claims to be that of another platform in an effort to deceive a mobile agent
	Denial-of-service	Agent platform maliciously and intentionally denies mobile agent's service requests
	Eavesdropping	Agent platform monitors communication and agent instructions that were executed
	Alteration	Agent platform that maliciously modifies an agent's code, state, or data
Other-to-Agent Platform	Masquerading	Agent to a remote system claims to be another agent on the platform in an effort to deceive an agent. Also, a remote platform masquerades as another platform in an effort to deceive another platform
	Unauthorized access	Conventional denial-of-service attacks aimed at the underlying operating system
	Copy and play	Agent message or instruction is maliciously captured and repeatedly executed

Management Architecture (OMA) to better support agent technology

- **World Wide Web Consortium (W3C)**—Resource Description Framework (RDF) and RDF Schema, OWL, Semantic Web, Web Ontology

Security implications

Security threats

Disclosure of information, denial of service, and corruption of data are the three classes of security threats that apply to agent-based computing systems. Most security threats to agents have corollaries in traditional client-server systems. The nature of mobile agents adds a degree of significance to the extent of harm that a malicious user could cause to an agent-based framework. Threats stemming from an agent attacking an agent platform, an agent platform attacking an agent, an agent attacking another agent on the agent platform, and other entities attacking the agent are four categories of threat that could affect an agent, as enumerated by Wayne Jansen and Tom Karygiannis of NIST's Computer Security Division in NIST Special Publication 800–19: Mobile Agent Security. Table 2 (see page 17) describes the types of threats that affect agent-based architectures.

Security requirements

The security requirements of agent-based software address conventional system security needs, including confidentiality, integrity, accountability, and availability. The agent framework must ensure that communication and data that are meant to be private remain undisclosed. Modification of code, state, and data must be protected to ensure that only authorized agents and processes can alter them. The agent framework must be able to provide continuous support for agent access to data and services. There must be mechanisms in place to uniquely identify every action that an agent makes, and every agent must be held accountable for their acts. The following sections briefly discuss how each of the security requirements applies to agent-based computing and the types of threats that are possible within each category.

- **Confidentiality**—The agent framework must be able to provide a mechanism for protecting private data and agent actions. If not properly secured private information contained in the

Table 3. Countermeasures to agent-based software threats [7]

Protecting	Countermeasure	Description
Agent Platform	Software-Based Fault Isolation	Isolation of application modules into distinct fault domains enforced by software
	Safe Code Interpretation	Commands considered harmful made safe or denied to an agent
	Signed Code	Digital signatures such as PKI used as a means of confirming authenticity of an object, its origin and its integrity
	State Appraisal	Ensuring that an agent had not been subverted due to alterations of its state information
	Path Histories	Maintenance of authenticated record of prior platforms visited by an agent
	Proof Carrying Code	Proof that an agent possesses safety properties stipulated by the consumer
Agents	Partial Result Encapsulation	Encapsulation to provide confidentiality using encryption and integrity and accountability using digital signatures
	Mutual Itinerary Recording	Tracking and recording by cooperating agent of the itinerary and path history of another agent
	Itinerary Recording with Replication and Voting	Multiple agents performing the same computation and comparing and voting on results to ensure integrity
	Execution Tracking	Each platform retaining a log of the operations performed by agents while resident on the platform that cannot be repudiated
	Environmental Key Generation	Using generated keys to unlock executable code cryptographically
	Computing with Encrypted Functions	Ensuring that mobile code can safely compute cryptographic functions even if the code is executed in an untrusted computing environment

content of messages sent between agents and agent platforms could be intercepted by eavesdroppers and maliciously used. An agent's actions on a platform, or the path on which a message flows, could also reveal sensitive information. Even if the actual message content is not intercepted, the actions that an agent executes could reveal vital confidential information about a transaction. It may be necessary for mobile agents to keep their identity and location private. An agent may not want to disclose the path that it has taken to other agents and platforms that it came into contact with along the way. An agent's audit logs that detail its activities must be protected and restricted to authorized users, and may even be unable to be completely read by one individual (i.e., enabling separation of roles and duties and implementation of two-man rule) depending on the boundaries that it has crossed and the policies of each domain.

- **Integrity**—The agent framework must be able to detect and report tampering to its code, state and data. Since mobile agents travel from host to host, it is possible for a malicious platform to modify an agent's message or action, then send the compromised agent on its way. A malicious platform may interfere with the communication between agents, and may even tamper with agent audit logs.
- **Availability**—The agent framework must be able to deal with intentional failures or failures that cause vulnerabilities. The agent platform must be able to detect and recover from software and hardware failures. The agent platform must be able to handle large amounts of agents. It must also be able to detect rogue denial of service attacks and account for them.
- **Accountability**—Agents and agent platforms must be held accountable for all of their actions. To this end, agents and agent platforms must keep audit logs that track relevant events, as defined in the security policy of the framework. The logs must be protected from unauthorized access and modification. The logs can be used if there is a breach in security, or if an agent or agent platform is behaving maliciously.

Security countermeasures

Security countermeasures in traditional computer systems have existed for many years, and they are continuously being enhanced. Agent-based systems use conventional security measures as well as a few unique ones that control mobile code. Table 3 (see page 18) describes some of the possible countermeasures that could be taken for securing agent-based software.

Conclusion

Agent-based software systems provide a framework that extends distributed, open and dynamic computing environments, enabling software agents to communicate effectively with each other in rapidly changing environments. Agent-based architecture has facilitated the growth of new applications domains, including autonomic computing, grid computing, and the semantic Web. Lack of mature standard software analysis and design methodologies, limited defined specification requirements, and a considerable number of conventional and new security threats are concerns that need to be further developed. Agent-based software systems have made a significant impact on technology. Agent-based software systems have a promising future, for they have paved the way for many new and exciting types of applications. ■

About the Author

Michael Colon

Michael Colon is an Associate with Booz Allen Hamilton, where he specializes in Secure Application Design and Development. He holds a B.S. degree in Computer Science and has over 12 years of Software Engineering experience.

References

1. "Software Agents" in IEEE Review (January 1996), Nick Jennings and Michael Wooldridge
2. Franklin and Graesser's paper "Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents"
3. Joanna J. Bryson and Lynn Andrea Stein in Modularity and Design in Reactive Intelligence, (Cambridge, MA: MIT Artificial Intelligence Laboratory, NE43–833)
4. Declarative and Procedural Goals in Intelligent Agent Systems, Michael Winikoff, Lin Padgham, James Harland, and John Thangarajah of RMIT University (Melbourne, Australia)
5. Y. Labrou, T. Finin, and Y. Peng: "Agent commutation languages: The current landscape" in IEEE Intelligent Systems (1999; 14[2], pp. 45–52)
6. Michael Luck, Peter McBurney, and Chris Priest, AgentLink: "Agent Technology: Enabling Next Generation Computing—A Roadmap for Agent Based Computing"
7. Wayne Jansen and Tom Karygiannis, Computer Security Division, National Institute of Standards and Technology: NIST Special Publication 800–19: Mobile Agent Security

Resources

B. Bauer, J.P. Muller, and J. Odell: "Agent UML: A Formalism for Specifying Multiagent Interaction", in Agent Oriented Software Engineering (P. Ciancarini and M. Wooldridge, eds. Berlin: Springer-Verlag, 2001)

C. Kesselman Foster, and S. Tuecke: "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" in International Journal of High Performance Computing Applications (2001)

N. R. Jennings: "An agent-based approach for building complex software systems" in Communications of the ACM (2001; 44[4], pp. 35–41)

G. Weiss: "Agent orientation in software engineering" in Knowledge Engineering Review (2002; 16[4], pp. 349–373)

M. J. Wooldridge: Introduction to Multiagent Systems (New York: John Wiley and Sons, 2002)

IBM Autonomic Computing. http://www.research.ibm.com/autonomic/glossary.html

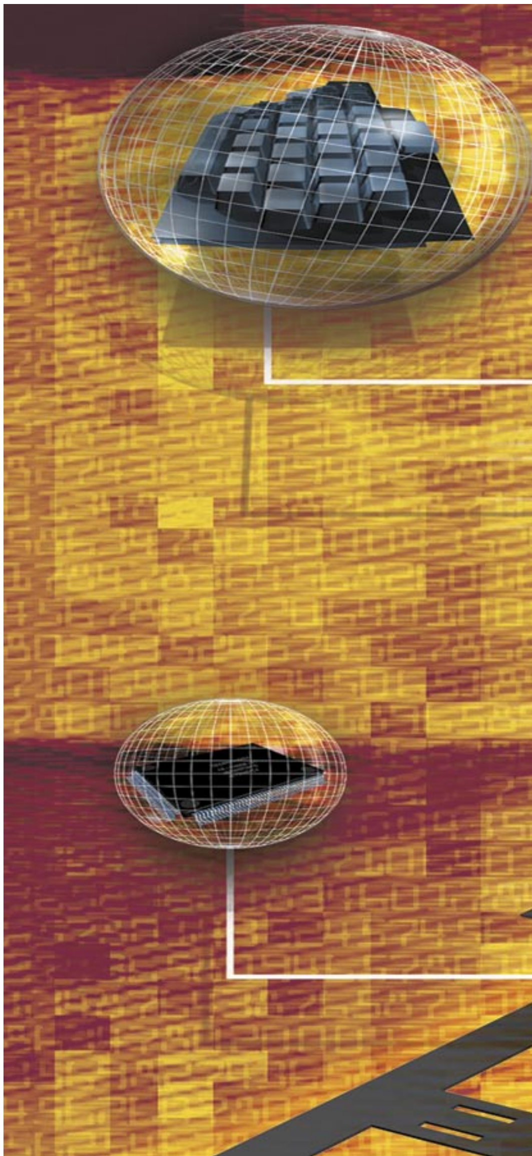
IBM Autonomic Computing. http://www.research.ibm.com/autonomic/overview/benefits.html

IBM Grid Computing. http://www-1.ibm.com/grid/about_grid/what_is.shtml

Autonomic Computing

by Karen Mercedes Goertzel, CISSP

Autonomic computing refers to a computing infrastructure that automatically adapts to care for itself, configure itself, monitor itself, repair itself, and plan for future eventualities, all with little or no human intervention. IBM, the "thought leader" in the definition and development of autonomic systems, has already "autonomic-enabled" some of its key product lines. IBM has also taken the lead in identifying a long list of existing and proposed standards, mainly in the realms of Web services, process management, and grid computing that are relevant and should be adopted by developers of autonomic systems. Autonomic computing combines concepts from service-oriented architectures/Web services, component-based computing, agent-based computing, grid computing, computer cognition, and computer immunology. The security threats against, security requirements for, and security controls that will be used in autonomic systems will, to a great extent, be derived from the security concerns and countermeasures identified for technologies in those disciplines. Autonomic computing should not be confused with "autonomous computing," as conceived by Microsoft. Microsoft's vision of autonomous computing is characterized by secure (well-guarded) islands of computing resources () operating in a hostile environment in a way that enables cooperation (i.e., negotiation by emissaries) between independent systems that do not trust each other. Autonomous computing is a design pattern for securing applications and computing resources whereas autonomic computing refers to a cooperative system of widely distributed self-configuring, self-healing, self-optimizing, and self-protecting elements. By contrast, the rationale for autonomic systems is quality of service, not security, except in so far as security can help guarantee quality of service.



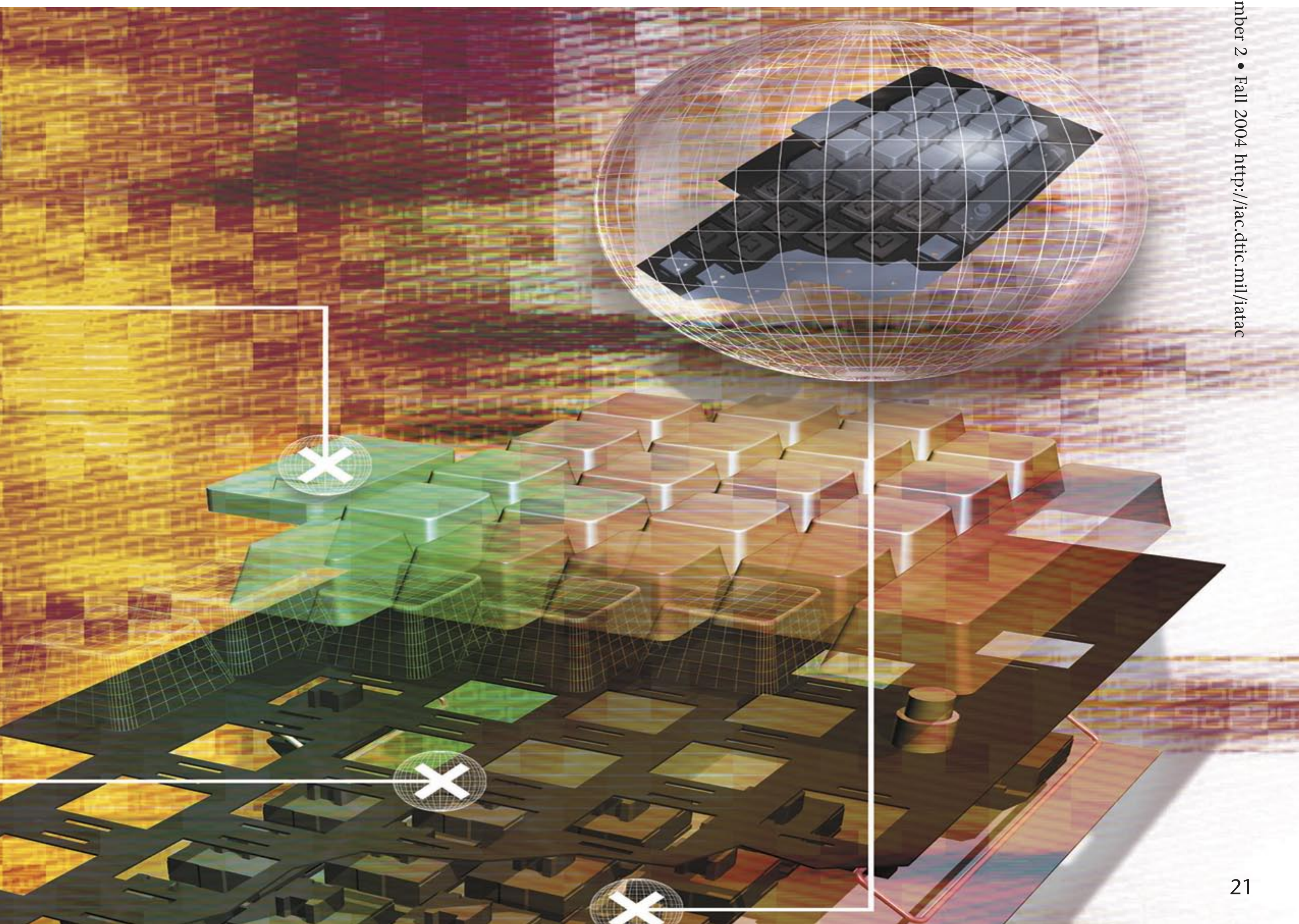
Autonomic computing is IBM's term for what has also been called self-healing technology, holistic computing, and introspective computing. Autonomic computing combines the concepts of "autonomous" and "automatic" to describe computing systems that are intelligent and able to operate, manage, and improve their own operations with minimal or better yet, no human intervention. A fully autonomic system will function as a whole to achieve self-governing operation of the entire system, not just of individual components.

Autonomic computing brings together concepts and technologies from component-based computing, object oriented computing, service-oriented architectures/Web services, agent-based computing, computer cognition, computer immunology. An autonomic software component (or element, in IBM parlance) [1] is—

> *"The fundamental atom of autonomic applications and systems—a modular unit of composition with contractually specified interfaces, explicit context dependencies, and mechanisms for self management, responsible for providing its own services, constraints (e.g., system resource requirements, performance requirements, etc.), managing its own behavior in accordance with context, rules, and policies, and interacting with other autonomic components. [1]"*

Autonomic elements also employ computer cognition techniques which enable them to learn from the events they encounter and from their observations of the results of their own reactions to those events. This self-awareness enables autonomic elements to refine their future reactions to similar events. The key characteristics of autonomic systems are/will be [1]—

- **Self-healing**—Autonomic systems detect and isolate improper operations and initiate corrective actions without disruption of processing
- **Self-optimizing**—Autonomic systems maximize resource allocation and utilization to maintain optimal quality of service while meeting user needs. Self-optimization in autonomic systems builds on existing capabilities for logical partitioning, dynamic workload management, and dynamic server clustering by extending those capabilities enterprise-wide across multiple heterogeneous systems
- **Self-protecting**—Autonomic systems defend themselves against unauthorized access, detect, and identify hostile behaviors (e.g., intrusions, viruses), and take actions to protect themselves against those behaviors, to report those actions, and to automatically perform secure backup-recovery in case of detected compromise
- **Self-configuring**—Autonomic systems define and redefine themselves whenever necessary to adapt dynamically to changes in their execution environment. They add new features "on the fly" with no disruption of service and with minimum or no human intervention
- **Self-stabilizing**—Some autonomic systems are also self-stabilizing. This means that the elements of the system tolerate arbitrary state corruption because their ordinary operation ensures that a good state will eventually be reached.



The system does not attempt to detect that it is in a bad state from which it needs to recover. Instead, all system operations eventually lead the system into a good state, even if operation is started after the system has been corrupted by a transient disruption. Error detectors are not needed in self-stabilizing systems, so there is no concern that the error detectors themselves might be corrupted or inaccurate. Most self-stabilizing systems are also forgetful—current state is continually regenerated and old state is forgotten, thus limiting the effect of any corrupted state.

In practical terms, several techniques will be combined to implement secure autonomic computing systems, such as protection from virus attacks, protection from intrusions, monitoring of data integrity, and monitoring of subnet health. To achieve this some complementary technological initiatives will provide enablers for autonomic computing [2]—

- Support for on-the-fly system change, achieved through software rejuvenation and self-stabilization [3]
- Adaptive fault-tolerance protocols, such as the SPIDER protocols developed by NASA, and those being defined by NIST's Information Technology Laboratory in its work on self-adaptive discovery mechanisms for fault-tolerant networks [4]
- Self-healing real-time schedulers, which extend the automated task scheduler concept used in mainframe computing to schedulers appropriate for use in real-time fault-tolerant systems [2]
- Enhanced detection technologies that combine data mining and collaboration with intrusion-detection and system monitoring (e.g., DARPA-funded Data Mining-Based Intrusion Detection Systems [developed by several universities]) [5]
- Application-specific monitoring (e.g., application firewalls)
- Machine learning/computer cognition
- Genetic programming [6], to evolve small software components

Technology leaders

In industry, the firms leading the way in development of autonomic systems are—

- **IBM Autonomic Computing Initiative** [7], [8]—IBM is the industry thought leader in the area of autonomic computing and appears to be committed to long-term investment in this technology. IBM has implemented autonomic capabilities in several of their current products and recently teamed with Cisco to develop an Adaptive Services Framework that will be implemented in products from both companies
- **Hewlett Packard (HP) Research Laboratory** (Bristol, England) Resilient Infrastructures Program [9]
- **Fujitsu/Siemens SysFrame** [10]
- **Cassatt Corporation** [11]

In government, the agencies most involved in research and development related to autonomic technology definition and prototyping include—

- **DARPA**—Information Processing Technology Office (IPTO) [12], Advanced Technology Office (ATO) [13], and Information Exploitation Office [14], have all anticipated autonomic computing with several of their research initiatives, including Self Regenerative Systems (SRS), Fault-Tolerant Networks and, earlier, Active Networks, Adaptive and Reflective Middleware Systems (ARMS), etc.
- **NIST** [15]—NIST has done significant research in the area of security for autonomous (autonomic) mobile agents, including addressing issues of privilege management. In addition, NIST's work in the areas of robust agent-based intrusion detection, and network security testing, while not specifically addressing autonomic agents, will be directly applicable to autonomic system security

Table 1. Key examples of current autonomic components

Vendor	Product
IBM	• DB2 relational database management system (RDBMS) Version 8.0 • WebSphere Application Server Version 5.0 • WebSphere Studio* • BlueBox policy-driven host-based IDS • IBM Tivoli Risk Manager 4.1 • eServer (formerly eLiza) • Client Rescue and Recovery for PCs • Distributed Wireless Security Auditor for PCs
Cisco Systems (with IBM)	• Adaptive Services Framework
Intel	• Itanium 2 Machine Check Architecture (MCA)
Sun Microsystems	• N1 • Netra Proxy Server
Hewlett Packard	• Utility Data Center with Utility Controller Software[1] • OpenView Continuous Access Storage Appliances[2]
Tripwire Inc.	• Tripwire
MS Research	• Aladdin Lookup Service
DEC/Compaq	• Autonet Configuration Protocols
Unisys	• ClearPath Plus system management capabilities

[1] Used in the Shipbuilding Partners and Suppliers (SPARS) component built by IBM for General Dynamics' Electric Boat system. [2] These are Secure Distributed Storage (SDS) solutions; SDS is often cited as a key application for autonomic computing.

Current and future applications

According to Yankee Group vice president Zeus Kerravala [16], while there is tremendous interest in and demand for self-healing, on-demand networks and systems, as described by IBM, there are currently no examples of large-scale deployments. This said, several vendors have released individual autonomic computing components. These will provide the building blocks for larger autonomic systems. Key examples of current autonomic components can been seen in Table 1 (see page 22).

Some key projects in academia and independent research and development (R&D) include—

- Rutgers University Center for Advanced Information Processing (CAIP), The Applied Software Systems Laboratory (TASSL): Project AutoMate, http://automate.rutgers.edu/
- University of California-Berkeley: Recovery-Oriented Computing (ROC), http://roc.cs.berkeley.edu/, and OceanStore global persistent data store, http://oceanstore.cs.berkeley.edu/
- University of Arizona High Performance Distributed Computing Laboratory: AUTONOMIA Autonomic Computing Environment, http://www.ece.arizona.edu/~hpdc/projects/AUTONOMIA/
- Imperial College, London and University of Glasgow: Autonomic Management of Ubiquitous Systems for e-Health (AMUSE), http://www.dcs.gla.ac.uk/amuse/
- Indiana University - Purdue University at Indianapolis: Discrete Optimal Network (DON), http://netlab.ece.iupui.edu/~tkam/optim.html
- Columbia University Programming Systems Lab: Kinesthetics eXtreme, http://www.psl.cs.columbia.edu/kx/index.html, and Worklets/Workflakes, http://www.psl.cs.columbia.edu/worklets/index.html
- Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory: Self Adaptive Software, http://www.csail.mit.edu/research/abstracts/abstracts03/dynamic-languages/05shrobe.pdf
- NAI Labs and Boeing Phantom Works Cooperative Intrusion Traceback and Response Architecture (CITRA) and Intruder Detection and Isolation Protocol (IDIP), http://www.mcafeesecurity.com/us/nailabs/research_projects/adaptive_network/
- Joint Information Systems Committee (JISC) (UK) and Engineering and Physical Science Research Council (EPSRC) (UK) Semantic Grid and Autonomic Computing Programme, http://www.jisc.ac.uk/index.cfm?name=programme_semantic_grid

Table 2. IBM has identified these standards as being directly relevant in IBM and HP's effort in building the "first generation" autonomic elements based on Web services

Standards Body	Relevant Standards
Organization for the Advancement of Structured Information Standards (OASIS)	Web Services (WS) Standards: WS-Distributed Management (WS-DM), WS-Security (OASIS)
Java Community Process	Java Specification Requests (JSRs): Java Management Extensions (JMX) (JSR3), Logging API Specification (JSR47), Java Agent Services (JSR87), Java Portlet Specification (JSR168)
Storage Networking Industry Association (SNIA)	Storage Management Initiative Specification (SMI-S), aka "Bluefin"
Global Grid Forum (GGF)	Open Grid Systems Architecture (OGSA), Open Grid Services Infrastructure (OGSI), Open Grid Service Common Management Model (CMM)
OpenGroup	Application Response Measurement (ARM)
World Wide Web Consortium (W3C)	XML, SOAP, Web Services specifications

Standards efforts

Possibly because autonomic computing is an application of other technologies and computing disciplines, including agent-based computing, composition of software, Web services, and grid computing, system and policy management, and workflow process definition and management, it appears that no standards specific to autonomic computing are being pursued, at least not at this point. Rather, IBM cites a long list of standards in these other technology areas and disciplines that are relevant for autonomic systems. IBM and Cisco also stated, as part of the objective of their partnership, the development of standards for autonomic computing and self-healing networks. In terms of implementation, the "first generation" autonomic elements being built by IBM and HP are based on Web services technologies, IBM has identified certain standards as being directly relevant (see Table 2 above). [17]

Potential government applications

Autonomic computing is the natural next stage in the "unattended mode of operations" and "lights out" methodologies already employed by the government in many of its computer operations. In addition, autonomic systems are most likely to be employed to enhance computer network defense. By enabling the design and implementation of networked systems that can tolerate attacks and failures by automatically changing their state or structure during execution, emerging autonomic systems will be able to render ongoing attacks ineffective through dynamic changes to state.

As autonomic computing technology evolves and matures, the incorporation of artificial immunology into autonomic systems will enable them to actually alter their own vulnerabilities by changing their state or structure in response to attacks, thereby introducing diversity that will blunt attacks. The ultimate objective will be to create systems that can self-repair by making substantive changes to their own structures in response to attacks.

Security implications

In IBM's vision [18], each autonomic element can be seen as a service with a two-part architecture—

1. **Functional unit**—Performs the basic function of the element, such as storage, database functions, Web

service, etc. The functional unit is what constitutes the application of the autonomic element.

2. **Management unit**—Oversees the operation of the functional unit, ensuring that it has adequate resources to perform its function, that it configures/reconfigures itself to adapt to changing conditions, and that it carries out negotiations with other autonomic elements, etc. Management units perform their oversight role through a series of sensors and effectors by which they monitor and control the functional units. It also exercises access control over the secure management channels it establishes between itself and its functional unit, and between itself and other autonomic elements.

From a security standpoint, the management unit will be responsible for enforcing policy and protecting the operations and data of the functional unit.

Security threats

Most of the security threats to autonomic systems are comparable to those cited for other distributed agent-based systems, since autonomic elements are, in fact, software agents. These threats include—

Malicious autonomic element attacks host—Malicious agents can steal or modify the data on the host or "hog" resources, causing denial of service to other applications or agents. Lack of sufficient authentication and access control mechanisms lead to these attacks. If resource constraints are not set, they can also commit Denial of Service (DoS) attacks by exhausting computational resources and denying platform services to other agents. Traditional countermeasures include—

- Authentication [19], authorization, and access control to establish trust of autonomic elements, and control their ability to access/affect host resources, and each other's data and computational logic.
- Virtual machine safe code interpretation (.NET) and sandboxing (Java)
- Cryptographic techniques, including digital signature of elements (for authentication and integrity assurance) and encryption of data (for access control)

In addition to these countermeasures, two novel countermeasures have been described—

- **Path histories** [20]—Each host in the element's routing path appends a signed entry to the element's path log, so that the current host can determine the route taken by the agent and determine if all previous hosts visited by the element are trustworthy.
- **State appraisal** [19]—The agent must be written to include a "maximum" function that makes a calculation, based on the current state of the agent, of the maximum set of permissions that should be granted to the agent. This calculation is compared with the results of the calculation by a "request" function that determines the privileges that a user wishes the agent to have. Whichever calculation produces the more restricted set of privileges is used. This ensures that if the element has become malicious due to alterations in state, it will not obtain the same permissions it would have been granted if its state had remained benign.

Malicious host attacks element—It is expected that host platforms will provide a safe environment in which the autonomic elements can execute. A malicious host can attack an element to steal or modify its data (which may be sensitive), corrupt or modify its logic or state, deny requested services, return false system call values, re-initialize the element, or terminate it completely. A malicious host can also delay the element until its task is no longer relevant. The host may also analyze and reverse engineer the agent to produce a malicious version. Providing countermeasures against host attacks on autonomic elements is complicated by the fact that the host needs must have full knowledge of the element's code and the state in order to execute it. Some possible countermeasures [21] include—

- **Encrypted functions** [22], [23]—The function of the element is encrypted according to a conversion algorithm that produces ASCII ciphertext. The host can read the program but won't understand its functions, thus inhibiting its capability to re-engineer the element.
- **Encrypted data**—The element's data is encrypted to prevent unauthorized disclosure to a malicious host. If, however, the data needs to be decrypted by the host to perform a computation, the element will have to carry the decryption key, which increases vulnerability of the element.
- **Code obfuscation**—A "blackbox" element is generated from the element's code specification. This obfuscation prevents the code and data from being readable or modified—only its inputs and outputs can be observed. To prevent dictionary attacks the algorithm that converts the code specification into an element uses random parameters that enable the algorithm to create a number of different agents from the same specification—these agents differ in code and data representation, but produce the same results.
- **Secure routing**—The element must be programmed to include a routing policy that limits its migration to only trusted hosts. Replication and voting can be used to achieve fault tolerance in case a malicious host manages to tamper with the element's itinerary or computation results. By replicating the agent at various stages, then voting (i.e., comparing and selecting) the computation results produced by the different element-copies, a correct result will be produced as output.
- **Detecting attack using dummy data** [2]—Dummy data objects are stored in the element's database. The element should not modify these data objects when it performs its functions. After the element finishes processing, the dummy objects are checked. If they have remained unchanged, one can be reasonably sure the legitimate data in the database have also remained uncorrupted. For this technique to work, the dummy data must be of a nature that does not adversely affect the results of any query.
- **Trusted hardware**—By encapsulating the entire element and its execution environment in a tamper proof trusted devices, such as a PC card or smartcard, the element remains isolated from the malicious host, interacting with it only through messages.

Malicious element attacks another element—A malicious element may invoke the public methods of another element to interfere with its work. Authentication, authorization, and access control between agents are possible countermeasures

Other entity attacks element—Entities not directly involved with the element may attempt to manipulate or eavesdrop on the element's communications. Encryption of the element's messages and data is the obvious countermeasure

Security challenges

In addition to the threats cited above, some security challenges to autonomic elements introduced by their autonomous, mobile nature include [18]—

- Autonomic elements traverse multiple hosts that may be trusted to different extents. This increases the difficulty of defining security policies and countermeasures that will be effective throughout the autonomic computing system
- As they evolve, autonomic systems will use new techniques and architectures whose security implications may not be well understood
- Autonomic systems are designed to reduce the amount of human interaction with the system, and thus cannot rely on human detection and analysis of, or response to anomalous behavior caused by security compromises

Security requirements

Some key security requirements for autonomic systems include [18]—

Autonomic intrusion detection (ID)—To be self-protecting and self-healing autonomic systems must be able to detect and react independently to intrusions to eliminate the threat and restore the system to an uncompromised state. By definition, an autonomic system will perform these functions independently with very little or no human intervention. Autonomic elements that have been compromised can be terminated or isolated through changes to high-level policies. If this stops the attack, the system can be restored to an uncompromised state by first restoring the stateless part of the system through secure backup (e.g., using the original distribution set for the system) performed automatically (without human intervention). However, to ensure that the "state" segment of the system can be automatically restored, the system state must be maintained in a way that enables detection and elimination of corruption. Techniques for doing this include storing redundant distributed copies of the state of the system in an encrypted form, so that even if one copy of the state is corrupted, the correct state can be determined from another copy.

Mutual authentication of autonomic elements and hosts, users, services, and other elements.

Establishment and maintenance of trust between autonomic elements—Elements that comprise an autonomic system must have a good reason to trust other elements that they discover and with which they interoperate. There are a number of existing mechanisms for establishing and reasoning about trust and trustworthiness, including X.509 certificate hierarchies. The security and trust policies that govern an autonomic element will determine how demanding it is when making trust decisions, how readily it trusts other elements, and how much corroboration it seeks before relying on information obtained from those elements. These policies will also constrain some of the actions that an element can take. For instance, if one of an element's suppliers becomes unavailable or stops performing acceptably, the element will need to decide which of the potential replacement suppliers can be trusted to take over the function of the failed supplier. The objective is to allow computing systems to make trust decisions in consistent and reliable ways, thus enabling extremely adaptive and dynamic operations without compromising security. Techniques must be provided that enable elements to represent and reason about the trust relationships between themselves and other elements. Techniques must be defined for constructing individual autonomic elements so that their collective behavior is both trustworthy and trusted.

Authorization and context-aware fine-grained access control (e.g., RBAC)—In a heterogeneous, dynamic autonomic computing environment, the large number of distributed elements must be able to delegate privileges to each another. No autonomic element or other entity may be allowed to provide a resource to another element, or obtain any service from another, without first obtaining permission from that element's management unit, as negotiated and obtained through the element's protected management channels.

Security policy definition and reasoning—To enable self-protection based on security policy that may differ across the autonomic environment both security policies and security-related tasks and states within the system must be represented. Based on these representations, autonomic elements will be able to automatically handle a wide range of security issues that are currently addressed by human intervention or by comparatively ad hoc programmed solutions. Elements must carry (or have access to) policies that govern and constrain their behaviors, and task and state representations that functionally describe their current mission, strategy, and status.

These policies will either be directly specified by a human, implicitly specified (as by a human accepting a default), or derived from higher-level policies by the rules of the appropriate policy calculus and distributed to all the elements of the autonomic system. These security policies will describe the level of protection needs to be applied to the various information resources that the element contains or controls, rules that determine how much trust the element places in other elements with which it communicates, the cryptographic protocols the element should use in various situations, the circumstances under which the element should apply or accept security-related patches or other updates to its own software, the strategies an element uses to recover when one of its suppliers fails to provide an expected resource, and which of its commitments to give the highest priority when not all can be fully met.

The task and state representations that a management unit holds to describe the element's current status and activities represent the other elements upon which this element currently depends, and how much it trusts each of them—

The current life-cycle state of the software that the element is running and whether or not there are any security updates available for it—

- The list of contact information for one or more other autonomic elements or human administrators who should be notified when certain suspicious circumstances are observed
- The agreements with one or more other elements to provide it with security-relevant information, such as log file analyses or secure time-stamping
- The list of previously vetted resource suppliers, used to quickly

verify the digital signatures on the resources they provide.

All of this will require the ability to compose policies, resolve policy conflicts, and negotiate policies between elements. Unlike conventional software programs, which behave as they are explicitly programmed to, an autonomic element may have a wide range of possible strategies available for fulfilling the policies that govern it, and must be able to provide an explicit representation of the current state of its efforts to carry out those policies. Techniques are needed that enable the elements to represent and reason about security policies. Common languages, taxonomies, ontologies, and standards must be defined that are expressive enough to allow the specification and negotiation of security and privacy policies between elements.

There is also a critical need to be able to prevent compromise to the security policy of an element. The policies that govern an autonomic element's behavior, and the task and state representations that allow it to reason about its own activities, represent high-value targets to a potential attacker. If an attacker can compromise an autonomic element and add to its policy database a policy that requires it to provide important information at a certain interval, the autonomic element would by its very nature attempt to use every resource at its disposal to ensure that the information was delivered. In this way, an attacker could exploit the element's own ability to adapt to changing conditions to create new strategies for stealing the desired information.

Preventing this sort of subversion is critical. The security policies that govern an autonomic element must be secured against tampering. Then, even if the attacker can implant a backdoor in (or otherwise alter) the programmatic logic of the functional unit to cause the unit to leak information to the attacker, the element's management unit will block the backdoor code's attempt to implement that leak because the element's security policies will not allow the transmission. Because they contain explicit computer-readable representations of the security policies under which they operate, autonomic systems are potentially more resistant to attack and subversion than systems that contain only functional code whose behavior may or may not conform to policy.

Confidentiality/privacy—Confidential data held/transmitted by an element must be protected from unauthorized disclosure and must be capable of being processed without violating any relevant confidentiality/privacy policies. Autonomic elements must be able to maintain representations of data protection policies and the privacy status of the various kinds of information that they process. The autonomic element must be able to reliably and automatically determine the data protection class of each unit of data it handles, and to securely and automatically retrieve the correct policy to apply to that class of data. It must also be possible to make routine changes in policies without changing the underlying programming or architecture of the autonomic systems. The element must be able to trust the security/privacy of other elements. Complex political and geographical situations must also be recognized and handled (i.e., there must be a way to enable elements to recognize when they have crossed physical/geographic boundaries that cause the policy governing releasability of their data to change).

Integrity of element logic and data

Non-repudiation by and accountability of elements and the users or services on whose behalf they operate—An autonomic system that spans domain boundaries must gather and securely exchange the information required to verify compliance to policy and satisfy audit requirements. This will require technologies and standards for record keeping and auditing of behavior relevant to security policy enforcement. Where geographical constraints are imposed by security and privacy policies (i.e., if an autonomic element has data that, by law, may not be released to a certain country (or other entity), the elements must be able to assure), and later perhaps prove for auditing purposes, that they did not export data to other elements in that restricted region. In addition, each element to which they do provide data must be able to assure and later prove that it does not physically reside in that region. In some applications, it may be necessary for one party to monitor in real time the operations performed on sensitive data provided to the system.

Cryptography and key management—In a widely distributed environment in which little or no human monitoring or intervention can be expected, new cryptographic and key management techniques will be needed to enable data to be anonymized or aggregated where required, and to be encrypted and signed against threats of disclosure and tampering.

Continuity of security (in terms of confidentiality, integrity, and availability)—This continuity must be maintained in every configuration into which autonomic elements put themselves and in every state into which they optimize themselves. Common languages and taxonomies must be defined for communicating and negotiating security and privacy states among elements. Techniques must be provided that enable elements to represent and reason about their security states. Criteria and methods must be defined that enable elements to differentiate between normal system failures and failures caused by denial-of-service attacks.

Resistance to fraud and persuasion (i.e., by attacker attempts to exploit an element's inherent capabilities in order to subvert the system)—Elements must be robust against attempts to provide them with false or misleading information that might lead them to configure or optimize themselves insecurely, to enter into an unjustified trust relationship, or to fail to protect adequately against a malicious attack. Autonomic elements depend on accurate information from other elements and from their execution environment. Autonomic elements must be protected from attackers who provide them with inaccurate or biased information.

Policies and algorithms are needed to make autonomic elements resistant to spoofing and subversion. Protection may be accomplished in the short term by implementing policies that instruct the elements to rely only on information derived from sources declared trustworthy by humans (with proof of trustworthiness possibly implemented through digital signature). In the longer term, autonomic element security policies should be able to become more complex and flexible, enabling them to make their own trust decisions. Elements' ability to make their own trust decisions will become a necessity as the complexity of the autonomic system increases.

Security initiatives

Autonomic computing is still in its infancy. Many of the security requirements above are being addressed through human intervention, as the application of autonomic techniques to the security of autonomic elements and systems is not yet possible give the current state of the art. Since, in the near term at least, autonomic elements are most likely to be implemented using Web services technologies, it is expected that security techniques used for Web services will be employed for autonomic elements. As security methods and technologies emerge for grid computing elements and software agents, these may also be applied to autonomic elements.

Conclusion

Like the software agent and computer cognition technologies that are their key enablers, autonomic systems have, as their predominant characteristic (and, indeed, their raison d'être) the ability to sense, reason, and learn about their environment and their own operation in it to the extent that they can function almost wholly independently of human intervention. The security implications for systems that have that much control over their own operation are significant. Autonomic systems are by their very nature, intended to enable human users and administrators to be far less directly involved with controlling how their computer systems operate, and concentrate instead on the results of those operations. But the danger of this reduced involvement is that users and administrators will be lulled into a false sense of safety—because their computer systems manage themselves and thus require far less monitoring of normal operational activities, they will erroneously translate this into a belief that autonomic systems also need less security monitoring. If anything, the opposite is true. A number of the threats against autonomic systems will have the objective of stealthy subversion. Successful exploits of this type are often not being recognized until significant damage is done.

While a number of the security protections identified for autonomic systems are applications of "traditional" countermeasures, like strong authentication, digital signature, and encryption, the ability of autonomic elements to "run themselves" without human intervention means that new classes of security mechanisms must be implemented, to protect against the types of exploits that would not be likely in more traditional systems, in which human vigilance was the rule rather than the exception. ■

About the Author

Karen Mercedes Goertzel

Karen Mercedes Goertzel is manager and lead contributor of a DISA application security initiative that creates detailed technical guidance to help developers, analysts, and engineers in the specification, design, and composition of secure web applications and web services using provably secure COTS, open source, and custom-developed software components. Ms. Goertzel also supports the ASD NII Software Assurance initiative. She is also a subject-matter expert in multilevel security and cross-domain solutions, and is involved with specifying architectures and mechanisms to enable multi-domain Service Oriented Architectures.

References:

1. S. Hariri and M. Parashar: Autonomic Computing: Research Issues, Challenges and Opportunities (AICCSA 2003 Autonomic Computing Tutorial, July 2003). http://automate.rutgers.edu/tutorials/aiccsa03-tutorial-slides/s2-ac-issues.pdf
2. Fred B. Schneider: Research to Support Robust Cyber Defense (Report to DARPA Information Technology Office, 21 January 2002). http://www.cs.cornell.edu/fbs/darpa.RobustCyberDefense.ppt
3. Mark Cathcart, IBM Distinguished Engineer: Towards Autonomic Computing (z/OS Expo and Performance Conference, October 2002). http://www-1.ibm.com/servers/corner/zosexpom07.pdf
4. Kevin Mills, Doug Montgomery, Scott Rose, Stephen Quirolgico, and Chris Dabrowski, NIST; Mackenzie Britton and Kevin Bowers, NSF Summer University Research Fellows: Self-Adaptive Discovery Mechanisms for Improved Performance in Fault-Tolerant Networks (DARPA FTN PI Meeting on Self-Organizing Systems for Hostile & Volatile Environments, 22 July 2003). http://www.antd.nist.gov/~mills/presentations/NIST-FTN072203bis.pdf
5. ACM SIGMOD Record, Volume 30 , Issue 4 (December 2001 - ISSN: 0163-5808): Special section on data mining for intrusion detection and threat analysis
6. The GP Tutorial. http://www.geneticprogramming.com/Tutorial/index.html
7. IBM Autonomic Computing. http://www-306.ibm.com/autonomic/index.shtml
8. IBM Research: Autonomic Computing. http://www.research.ibm.com/autonomic/
9. Matthew M. Williamson, Information Infrastructure Laboratory, Hewlett Packard Laboratories – Bristol: Resilient Infrastructure for Network Security (HPL-2002-273, 11 October 2002.) http://www.hpl.hp.com/techreports/2002/HPL-2002-273.pdf
10. Fujitsu Siemens: Toward the Grid – and Beyond. http://www.fujitsu-siemens.com/Resources/230/45177742.pdf
11. Cassatt Corporation. http://www.cassatt.com/home.shtml
12. DARPA Information Processing Technology Office: Programs. http://www.darpa.mil/ipto/programs/programs.htm
13. DARPA Advanced Technology Office: Programs. http://www.darpa.mil/ato/programs.htm
14. DARPA Information Exploitation Office: programs. http://dtsn.darpa.mil/ixo/programs.asp
15. NIST: Mobile Agent Projects. http://csrc.nist.gov/mobileagents/projects.html
16. Jay Lyman: IBM, Cisco Team on Standard for Self-Healing Networks (TechNewsWorld, 10 October 2003). http://technewsworld.com/perl/story/31826.html
17 IBM Autonomic Computing: Open standards driving the development of autonomic technologies http://www-306.ibm.com/autonomic/open-standards.shtml
18. D. M. Chess, C. C. Palmer, S. R. White: Security in an Autonomic Computing Environment (IBM Systems Journal, Vol. 42, No. 1, 2003). http://www.research.ibm.com/journal/sj/421/chess.pdf
19. William M. Farmer, Joshua D. Guttmann, Vipin Swarup: Security for Mobile Agents: Authentication and State Appraisal (Proceedings of the Fourth European Symposium on Research in Computer Security, 1996). http://citeseer.ist.psu.edu/farmer96security.html
20. Gerald Knoll, Niranjan Suri, and Jeffrey M. Bradshaw, Institute for Human & Machine Cognition, University of West Florida: Path-based Security for Mobile Agents (Elsevier Electronic Notes in Theoretical Computer Science 58 No. 2, 2002).

continued on page 41…

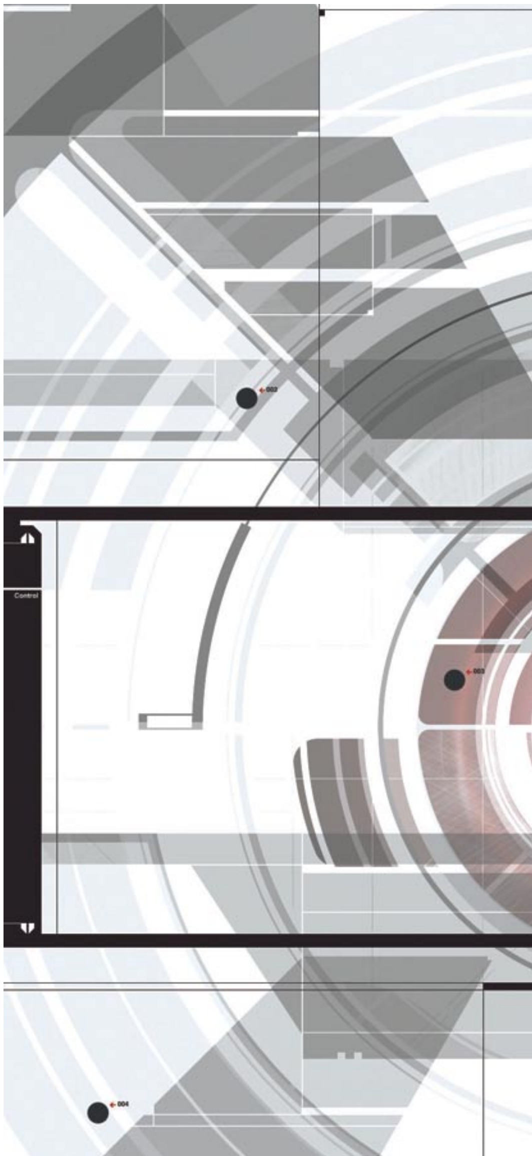
Computer Immunology

by Karen Mercedes Goertzel, CISSP

Computer immunology, also called artificial immunology, provides computers with sophisticated immune systems modeled after biological immune systems, especially the human immune system. Just as the body's immune response depends on its ability to quickly detect, recognize, and defeat intruding pathogens, computers need to be able to detect and recognize and proactively affect changes in a system's behavior. Computer immunology is definitely still in the realm of research and development, where computer immunology has been addressed solely as an enabling technology, including an enabler for security solutions such as antivirus protections and intrusion detection. None of the literature to date addresses the security implications of artificial immune systems themselves. This is understandable as the technology is still very much in its embryonic stages. Researchers agree that wide-spread practical applications are unlikely before 2020. However, a few actual deployments of prototype computer immune systems have yielded promising results. To the extent that these prototypes have been implemented using software agents, computing grids, Web services, etc., they will inherit the security issues, and benefits from the security standards and solutions, relevant to those technologies.

Over the past fifteen years, computer scientists have expended considerable effort devising ways to model the complex, adaptive biological systems such as the human immune system, and have developed a variety of new computational techniques that draw inspiration from those biological systems. Biologically-inspired computer systems are based on the notion that computations of all sorts are performed by natural systems that aren't actually computers. [1]

Biologically-inspired models are often used by computer scientists when a problem is too difficult to solve by more conventional means, and part of that difficulty lies in determining an appropriate initial state for a large complex system. Much of the natural world can be perceived as doing some form of search, optimization, or pattern recognition (*i.e.*, "non-symbolic" computations that explore a vast phase space but consider only an extremely small part of it). They provide no guarantees of optimal results, but are very good at "satisfic-



ing", and tend to be robust in the presence of the noise, error, and change that characterizes the real world.* [8]

Mimicking the characteristics of biological systems, however, is not intuitive to computers, which lack the inherent self-awareness they need to continually monitor themselves while at the same time performing their normal tasks.

Examples of computer-based systems that are modeled on natural biological systems include [1] [9] [10]—

- Artificial neural networks
- Artificial autonomous adaptive agents for colony-based systems
- Artificial trading agents for e-commerce, business modeling, and market-based control
- Evolutionary computation, which includes genetic algorithms, genetic programming, and computer immune systems.

Computer immunology is not, as sometimes misunderstood, a branch of bioinformatics, which is the application of information technology to biology, such as sequencing the human genome. Indeed, it is the converse—the application of biological models to information technology, and specifically the artificial simulation by computers of the human immune system. In this, a computer immune system may be said to be a form of artificial life.

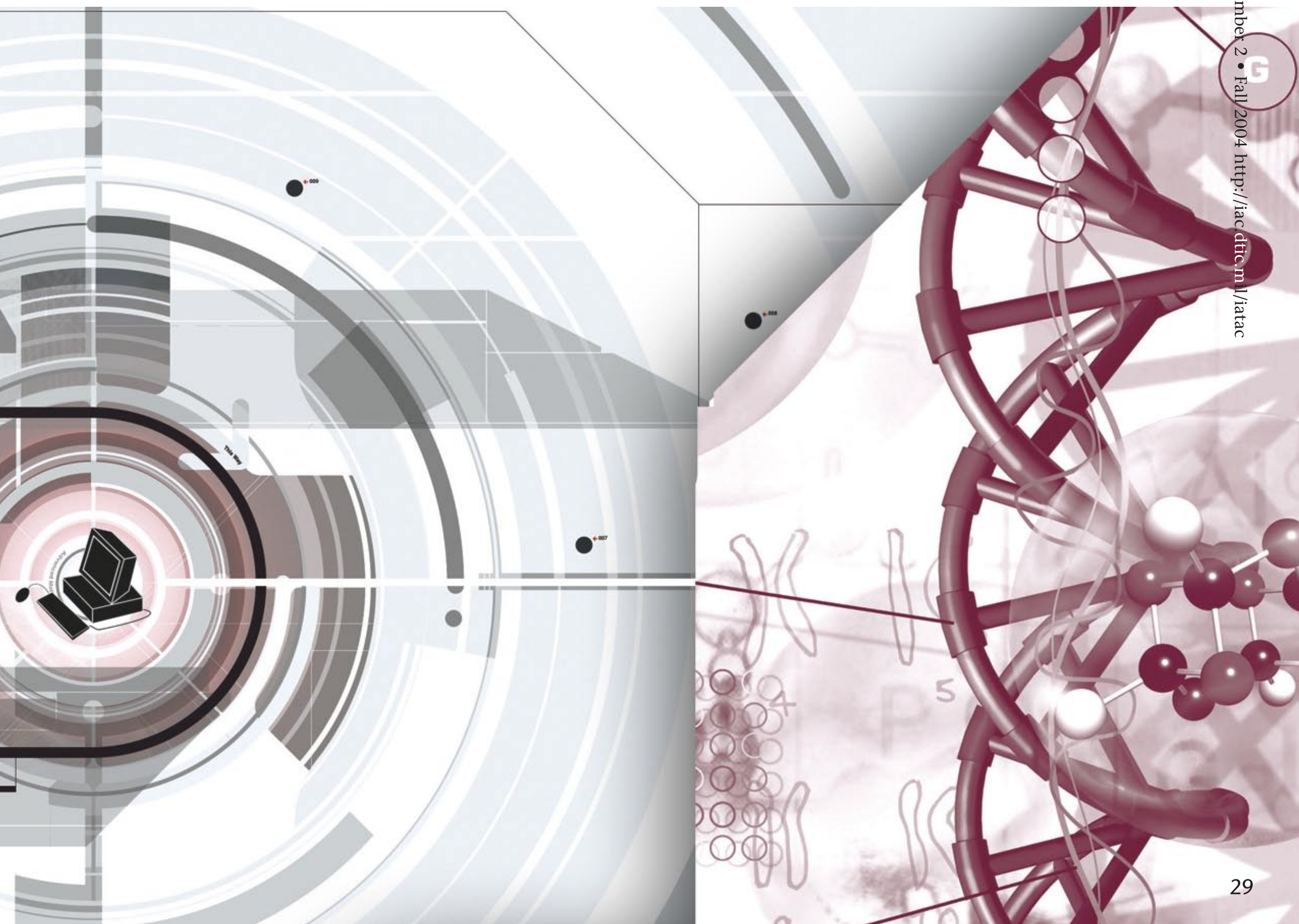
One of the characteristics that computer immunology strives to achieve is the resilience often seen in nature. Computer immune systems use techniques that are adaptive—that respond in pseudo-natural ways to the behavior of the system, such as polling of participating cache servers to agree/disagree on the integrity of the data they hold—and analysis of packet traffic within a network, looking for signatures of "normal" use and responding when abnormal behavior is seen. [2]

Biological and other robustness metaphors work at two levels [11]—

1. **Time scale**—lifetime of an entire species versus that of a single member organism
2. **Structure**—cell versus organism versus ecosystem

Robustness at one level often translates into robustness at a different level. Natural robustness also makes extensive use of diversity, adaptation, evolution, and use of disposable components—capabilities not automated in most computer systems.

Natural robustness also benefits from the study of "systemic effects." Research already exists that characterizes the temporal relation between attacks on computer networks and corresponding patch deployment, and that characterizes the events that indicate and track the spread of computer viruses, *etc*. Meanwhile, study and modeling of how biological epidemics spread is forming the basis for understanding computer virus dissemination patterns and for defining new, reliable information dissemination algorithms. Other theories that are contributing to the understanding and modeling of biological



systems for computer immunology include percolation theory, random-graph theory, and small-world theory. Together, research into these theories is offering insights into how reliable the Internet can actually be. [3]

In the current state of computing, there is very little diversity. Due to the need for economies of scale and the emphasis on standardization, there are millions of identical copies of the same software packages, for example. But diversity is a key strategy employed by biological systems to ensure, for example, that a disease that makes one person very sick has only moderate effects on others, and affects still others not at all. Computer immune systems attempt to duplicate diversity by using techniques for dynamic software composition and adaptation in order to avoid common, known failures and vulnerabilities. Software components are rewritten or passed through specially-coded filters to create different specialized instantiations of those components in ways that render their code different while ensuring that their computational results are the same. Another approach to software diversification applies pseudo-genetic algorithms to enable executable code components, once installed, to gradually change in composition within the constraints of a defined set of acceptable bounds of their functional specifications.

Increasingly close interactions between biologists and computer scientists are creating trends in computer science that have biological interpretations, such as "software rejuvenation" (implemented in the Apache Web server). New results of biological and genetic research are being interpreted in ways that are relevant to the problem of robustness. These include software evolution and phylogenetic trees for predicting vulnerabilities, intra-cellular signaling and cascades (chemostaxis), inter-cellular signaling networks (e.g., immune systems), genetic buffering, individual gene repairs, revolutionary mechanisms (genotype/phenotype mappings), and ecosystem modeling (diversity, keystone species, patch models, allometry, resource flows), all of which can be applied to create nature-based models of computing/network systems. [4]

Technology leaders

In U.S. academia, the "thought leaders" in computer immunology, both in terms of specifying techniques for implementing artificial immune systems and prototyping technologies based on those specifications, are University of New Mexico (UNM) with its Computer Immune Systems research, and University of Memphis' Intelligent Security Systems Research Lab. Outside the United States, significant computer immunology research is being performed by Oslo University College (Norway), developers of the Cfengine Autonomous Agent; University of Wales-Aberystwyth, with its ISYS, which applies the immune system metaphor to the problems of data analysis and machine learning; and University of West England and initially University of Bradford, succeeded by University of Nottingham, which have jointly undertaken noteworthy computer immune system prototypes.

In the U.S. Government, DARPA, with its funding of research into Immunity-Based Intrusion Detection Systems and Self-Regenerative Systems (SRS), leads the way in computer immune system research for military applications. DARPA researchers often work in concert with U.S. Air Force Research Laboratory, whose Information Directorate's Defensive Information Warfare Branch (AFRL/IFGB) is implementing a prototype Computer Defense Immune System (CDIS), and the Office of Naval Research (ONR), which is investigating an Anomaly Detection Using a Technique Inspired by the Immune System. In the civilian sector, the National Science Foundation has completed Preliminary Research on Immunity-Based Computational Techniques. [5]

Not surprisingly, the industry leaders in computer immunology research are IBM Research, with its Digital Immune System for Cyberspace, and Hewlett Packard Research Labs with its Biologically Inspired Complex Adaptive Systems (BICAS)—we say "unsuprisingly" because IBM and HP are also industry thought leaders in the area of autonomic computing, and computer immunology is being approached by them as an enabler for their autonomic systems.

Other smaller commercial applications include the Bit 9, which received a $2M Advanced Technology Program (ATP) grant from NIST to develop a Computer Immune System (CIS) designed to shield computers and networks from previously unknown virus attacks. The Bit 9 CIS will not work through pattern recognition and heuristic monitoring of code behavior, techniques employed by current antivirus software and IDSs. Instead, it will continuously check the system's state to ensure that it conforms to CIS's knowledge of what is the correct state for that system.

At least one commercial product has already reached the marketplace—Sana Security Inc.'s Primary Response is an application-level intrusion prevention system modeled after the human immune system. Primary Response's computer cognition capabilities enable it to undergo an initial "learning period" during which it observes and learns to recognize normal operation of the application it is to protect. Then, whenever it detects anomalous behavior by the application that deviates from its idea of "normal," it can be configured to block all access paths to the misbehaving application until it is patched. Primary Response's learning is incremental, so that it can continuously adjust its understanding of normal application behavior, and adjust its own recognition of what is "anomalous" accordingly.

Current and future technologies

Computer immunology is definitely still in the realm of research and development. Those involved in developing prototypes tend to agree that widespread practical applications are unlikely before 2020. However, a few actual deployments of computer immune systems have yielded promising results, providing interesting new approaches in the areas of cybersecurity, robotics, and Semantic Web/data mining. IBM, among others, is developing intrusion detection and antivirus systems that use artificial immunology. UNM is developing host-based and network-based intrusion-detection methods that mimic biological immune systems, using randomly-generated then selected (for anomalous behavior patterns) "antibodies"—detectors—to autonomously detect (through behavior pattern matching) and control or counteract foreign invaders (intrusive packets) [7] they haven't seen before. IBM's Digital Immune System for Cyberspace includes computer immunology-based antivirus capabilities.

Outside the U.S., The Post Office (UK), Kings College London, and Anite Government Systems have jointly undertaken to develop a

Computational Immunology Fraud Detection (CFID) system, in which immunological capabilities will be used to detect and identify patterns of computer activity that indicate fraudulent behavior by users.

Both University of Memphis in the U.S. and the partnership of University of West England and University of Bradford/University of Nottingham have independent projects applying artificial immune system concepts, and especially the characteristics of B-cell paratopes in biological immune system response, to the problems of complex document classification for the Semantic Web and fuzzy logic applied to data mining. Other initiatives were referred to earlier in the section on Technology Leaders.

In addition to the key applications in the areas of intrusion detection, anomaly detection, antivirus, and data mining, applications of computer immunology that have been proposed and/or prototyped include [1] [6] [10]—

- Robot controllers/intelligent robotic systems (e.g., robots used for mine detection and error detection in aerospace systems)
- Fault detection in mechanical systems
- Non-security-related anomaly detection in software and hardware systems
- Imagery enhancement and optimization
- Training systems
- Spam detection
- Development of multi-objective optimization algorithms for economics
- Job scheduling
- Development of cognitive systems

Standards efforts

As noted earlier, computer immunology still lies very much in the realm of research and development. No standards have been proposed governing the implementation of immunology-based applications. To the extent that computer immune systems will be implemented using software agents and computer cognition, any standards specified in those disciplines will be relevant for computer immune system technology. [1]

Potential government applications

The bulk of government-funded work in computer immunology to date has been in the areas of anomaly and intrusion detection and response, control of robotic systems, and fraud detection. Given the emphasis on computer network defense and net-centric operations in the Department of Defense (DoD), the Intelligence Community, and the civilian agencies, these applications are likely to continue to drive the majority of government research and investment into computer immunology in the foreseeable future. [5]

Computer immunology techniques are also being incorporated by industry into autonomic computing solutions. Eventual adoption of those solutions by government will represent a back door entry of computer immune systems into the government application space. [1]

Security implications

Computer immunology is being addressed entirely as an enabling technology, including an enabler for security solutions. None of the literature addresses the security implications of artificial immune systems themselves. This is understandable because the technology is still very much in its embryonic stages, as noted earlier, with the target date for real world implementations as distant as the year 2020. Currently researchers are striving to simply move the technology from "thesis-ware" to demonstrably functional prototypes.

To the extent that these computer immune prototypes are implemented using software agents, computing grids, Web services, etc., they will inherit the security issues, and benefit from the security standards and solutions relevant to those technologies. As computer immunology moves through technology insertion into real world applications, the security issues unique to immunology will certainly have to be addressed. At that time, standards proposals will no doubt begin emerge and security solutions will be defined.

Security initiatives

Computer immunology is being applied to intrusion detection and antivirus capabilities. [3] This approach combines anomaly/change detection algorithms based on natural selection processes, such as those found in DNA negative selection with the creation of software "antibodies," detector agents that are able to respond unrecognized anomalies such as unrecognized packets that indicate possible intrusion or computer viruses. The antibodies work on the principle of recognizing self and distinguishing it from non-self. Any entity encountered that is not recognized as "self" (i.e., a valid entity belonging to the system—is designated "non-self"). Each antibody is programmed to ignore, block, remove, or destroy any non-self entity it encounters, and also to issue warning signals to other entities in the system of the presence of the non-self entity, so that those entities can also work to detect and act upon any additional non-self entities. [6]

The computer immune system's ability to learn about its environment and to learn from the results of its own responses to that environment enables the system to increasingly anticipate and to grow immunity to the "pathogens" (e.g., attack packets, malware) in its environment. This enables the system to "evolve," over time reducing the probability of processing interruption caused by those pathogens. Because they are self-learning and adaptive, computer immunology based antivirus, and intrusion detections systems continually reduce the number of false reports (either positive or negative) as they learn about and adapt to changes in their execution environment.

Conclusion

Computer immune systems have long been envisioned as a key enabler for next-generation systems to enable them to better protect themselves from attack, operate more reliably, and perform complex data and state discovery, classification, and reasoning more effectively. Research in computer immunology has gone on for over a decade, with a significant amount of promising prototyping having been performed to prove computer immunology concepts, but real-world applications are still very few and the general consensus among experts in the field is that widespread use of computer immunology probably will not occur before 2020.

This said, the prototypes that have been undertaken have translated purely theoretical advantages of applying the human immune system as a model to computers to demonstrable applications of computer immunology to solve various real-world computing problems, most notably the need for better protection against cyber attacks and malicious code. With Sana Security's Primary Response, computer immunology can even be said to have finally reached the marketplace, even if a limited form. It is not unreasonable to speculate that other "immunology-enabled" products may soon follow, and that perhaps the projected "ready for prime time" estimate of 2020 is too conservative. ■

About the Author

Karen Mercedes Goertzel

Karen Mercedes Goertzel is manager and lead contributor of a DISA application security initiative that creates detailed technical guidance to help developers, analysts, and engineers in the specification, design, and composition of secure web applications and web services using provably secure COTS, open source, and custom-developed software components. Ms. Goertzel also supports the ASD NII Software Assurance initiative. She is also a subject-matter expert in multilevel security and cross-domain solutions, and is involved with specifying architectures and mechanisms to enable multi-domain Service Oriented Architectures.

**When confronted with too many options and not enough time and/or resources to evaluate them all, it can be impossible to make a "best" choice. "Satisficing" is a form of reasoning that accepts lower-than-optimal standards in order to generate and recognize results that are "good enough."*

References

1. Jim Austin, Dave Cliff, Robert Ghanea-Hercock, Andy Wright: Large-Scale, Small-Scale Systems (Foresight Cognitive Systems Project Research Review, Foresight Directorate Office of Science and Technology)
2. Mark Burgess, Oslo College: Computer Immunology (Proceedings of the 12th Systems Administration Conference, LISA_98–Boston, Massachusetts, 6–11 December 1998)
3. A. Somayaji, S. Hofmeyr, and S. Forrest: "Computer Immunology" (Communications of the Association for Computing Machinery, 21 March 1996Y). http://www.cs.unm.edu/~immsec/publications/cacm96.pdf
4. Robert E. Marmelstein, David A. Van Veldhuizen, Gary B. Lamont, U.S. Air Force Institute of Technology Graduate School of Engineering: Distributed Architecture for an Adaptive Computer Virus Immune System.
5. Robert H. Anderson, Phillip M. Feldman, Scott Gerwehr, Brian K. Houghton, Richard Mesic, John Pinder, Jeff Rothenberg, James R. Chiesa , RAND Corporation: Securing the U.S. Defense Information Infrastructure: A Proposed Approach (MR-993-OSD/NSA/DARPA, 1999). http://www.rand.org/publications/MR/MR993/
6. D. Dasgupta, Z. Ji, University of Memphis (TN); F. González, Universidad Nacional de Colombia: Artificial Immune System (AIS) Research in the Last Five Years (proceedings of the Congress on Evolutionary Computation Conference (CEC), Canberra, Australia, 8-12 December 2003). http://ais.cs.memphis.edu/papers/ais/2003/CEC-03.pdf
7. Wei Lu and Issa Traore, University of Victoria: A Novel Framework for Network Intrusion Detection Using Learning Techniques (IEEE, 2003). http://www.ece.uvic.ca/~wlu/T2.pdf&e=8092
8. Susan Stepney, Professor of Computer Science, University of York, UK: Non-Standard Computation: an Overview. http://www-users.cs.york.ac.uk/~susan/complex/nstdcomp.htm
9. L.N. de Castro and F.J. Von Zuben: "Artificial Immune Systems: Part I—Basic Theory and Applications" (RT DCA 01/99, 1999). ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/lnunes/trdca0199.pdf
10. L.N. de Castro and F.J. Von Zuben: "Artificial Immune Systems: Part II—A Survey of Applications" (RT DCA 02/00, 2000). ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/lnunes/rtdca0200.pdf
11. A. Somayaji, S. Hofmeyr, and S. Forrest: Principles of a Computer Immune System (Proceedings of the Association for Computer Machinery 1997 New Security Paradigms Workshop, pp75–82). http://www.cs.unm.edu/~immsec/publications/nspw-97.pdf

Additional Resources

Stephen Andrew Hofmeyer's home page http://www.cs.unm.edu/~steveah/

Computational Immunology for Fraud Detection http://www.icsa.ac.uk/CIFD/

University of Memphis Intelligent Security Systems Research Lab: Artificial Immune Systems. http://ais.cs.memphis.edu/

Workshop on Bio-Inspired Computing and Enabling Technologies (31 January 2001) (downloadable presentations). http://www7.nationalacademies.org/compbio_wrkshps/Workshop_on_Bio-Inspired_Computing.html

Computer Immunology @ Oslo University College. http://www.iu.hio.no/~mark/research/immune/

IBM Research: Antivirus Research. http://www.research.ibm.com/antivirus/

Harvard University Molecular Technology Group & Lipper Center for Computational Genetics Physiome Bio-System Models. http://arep.med.harvard.edu/ome.html

Lyanne Wai-Yin Lau, Department of Information Technology and Electrical Engineering, University of Queensland (Australia): Computer Immunology (Bachelor's Thesis, October 2002). http://innovexpo.itee.uq.edu.au/2002/projects/s358604/thesis.pdf

Leandro Nunes de Castro: Artificial Immune Systems. http://www.dca.fee.unicamp.br/~lnunes/immune.html

International Conferences on Artificial Immune Systems (ICARIS). http://www.artificial-immune-systems.org/

Leandro N. de Castro and Jonathan Timmis: Artificial Immune Systems: A New Computational Intelligence Approach (Heidelberg, Germany: Springer Verlag, 2002).

Ray Paton, Hamid Bolouri, Mike Holcombe, Howard Parish, Richard Tateson, editors: Computation in Cells and Tissues: Perspectives and Tools of Thought (Heidelberg, Germany: Springer-Verlag, 2003).

Gary Anthes: Future Watch: Immune Computer Systems (Computerworld, 9 December 2002)

University of Wales-Aberystwyth ISYS. http://www.aber.ac.uk/compsci/Research/mbsg/isys/

Fred B. Schneider: Research to Support Robust Cyber Defense (Study commissioned for Dr. Jay Lala, DARPA Information Technology Office). http://www.cs.cornell.edu/fbs/darpa.RobustCyberDefense.ppt

Martin Thorsen Ranang: An Artificial Immune System Approach to Preserving Security in Computer Networks. http://www.idi.ntnu.no/grupper/ai/eval/students/immune/immune.html

Efraín Torres Mejía, Department of Systems Engineering, Pontificia Javeriana University (Colombia): Immune System for the Detection of Intrusions at HTTP Protocol Level (May 2003). http://pwp.007mundo.com/etorres1/paperIDSHTTPen.pdf

Alexander O. Tarakanov, Russian Academy of Sciences, St. Petersburg (Russia) Institute for Informatics and Automation: Information Security with Formal Immune Networks. http://www.auth.gr/chi/PROJECTSIMCOM/T2.doc

An Immuno-Architecture for Managing Software Evolution. http://www.yy.cs.keio.ac.jp/~suzuki/project/immunity/index.html

NIST ATP Project Brief: Computer Immune System. http://jazz.nist.gov/atpcf/prjbriefs/prjbrief.cfm?ProjectNumber=00-00-5460

Sana Security, Inc. http://www.sanasecurity.com/

Analytical System Administration 3 http://www.iu.hio.no/SystemAdmin/res3.html

Kyrre Matthias Begnum's home page http://www.iu.hio.no/~kyrre/

Computer Investigation Markup Language (CIML)

An Information Sharing Method for Investigative Information

by Keith Hasselstrom and Peter Tran

The Department of Defense's (DoD) effort to fight cyber crime requires a well-orchestrated effort between the law enforcement components of DoD and US civilian agencies, as well as with allied nations. A critical component to this effort is the seamless and automated flow of information between each organization to facilitate coordination, de-confliction and computer network defense efforts. The Computer Incident Markup Language (CIML) creates a framework for the description and sharing of computer crime information.

As terabytes of unstructured information flow throughout organizations, an increased need for extensibility, structure and validation for information storage and migration methodologies has developed. This need has been the basis for the evolution of Extensible Markup Language (XML) and other similarly derived "Markup Languages." XML, like HTML, is based upon Standard Generalized Markup Language (SGML), which allows documents to be self-describing, through the specification of tag sets and the structural relationships between the tags. HTML is a small, specifically defined set of elements and attributes that enables users to bypass the self-describing aspect for a document. XML, while retaining the key SGML advantage of self-description, avoids the complexity of SGML. A focus area within the XML-based unified information migration is the aggregation and management of Cyber Crime Investigative Information for the Law Enforcement and Counterintelligence community (LE/CI). XML is not new; however, concrete application of the technology, such as with CIML, reinforces the importance and practicality of the technology.

The main thrust for this methodology is to provide stakeholders and those interested with an overview of LECI investigative information migration. It also provides a background for the methods, technical framework, computer investigation definition, implementation, and benefits for the LE/CI community. By applying this methodology as a template to other government domains, information sharing barriers can be lowered.

Background

In early 2002, the then-Joint Task Force Computer Network Operation Law Enforcement and Counterintelligence Center (JTF-CNO LE/CIC) stood up two systems (the "Systems"): one for storing law enforcement (LE) investigative information and the other for storing counterintelligence (CI) investigative information. The Systems store similar information with the specific nation source being the system discriminator. For example, United States (U.S.) sources are stored in the LE system and non-US sources are stored in the CI system.

The Systems exist for the purpose of coordinating and de-conflicting DoD LE and CI investigations. The Systems include automating the identification of similar investigations within LE and/or CI; cross-correlation of investigative information; and system-wide, text based searching and investigation tracking.

The challenge with the Systems was the migration of information in and out, seamlessly and electronically. Information originators have their specific structures, if any structure at all.

Methodology framework

The main challenge that exists with aggregating information from disparate sources is that the information is stored in varying formats that are unique to a particular source's business practices. These practices may not be conducive to sharing and distributing computer investigations outside the organization. In these cases, Investigative information needs to be extracted, translated, and placed in a template that is understood by the aggregating database.

Utilizing XML, unstructured information such as a new article can have structure applied to identify people, places, and things. The final outcome is a document that has its information broken down into "marked up" elements that are comprehendible by an automated information system. At this point, automation tools and databases can identify apples from oranges but still understand that both are fruit.

To leverage the full power of XML, there must be a template that defines the elements and attributes of a document. This template in markup language terminology is known as "XML Schema." Automation systems can ensure that a document is properly marked by examining the document against its XML Schema.

The XML Schema of CIML is defined to express the use-specific investigative components such as victim-system IP addresses, victim-system organization, victim-system attack date, etc. By implementing existing XML tools, any computer investigation that is marked up can be matched against the investigation XML Schema (template). If everything is compliant, the document is said to be valid.

The use of the XML Schema is critical to the successful distribution of a document to an automated system that relies on the data to exist in a certain format. Both the sender and recipient have access to the XML Schema, enabling both parties to develop automated document authoring and decomposing tools.

The extension of common document markup techniques is manifested in XML. The XML specification, which is controlled by the World Wide Web Consortium (W3C), allows for business and government bodies to develop their own markup language, satisfying a specific domain of information. For example, CIML. By adhering to the XML specification, automation products for documents can be built reliably.

Computer investigation defined

A CIML document stores computer investigative information including, but not limited to, case numbers, agencies, agency contact information, source computers involved, target computers involved, as well as subjects involved. Figure 1 (see below) shows most, but not all, of the information contained in a CIML document. The parsing engine that reads CIML documents and places the information within the LE and CI systems can receive investigative information from any source that provides a valid CIML document.

Scope

To complete the methodology, there is an effort required on sources that feed investigative information to the Systems. This means it is the responsibility of the source to place their information in a CIML document, validate the document, and deliver the document to the aggregating Systems. The delivery can be facilitated by electronic mail (E-mail), file transfer protocol (ftp) and/or any other file transfer method. Often this involves pulling information from an existing database and writing out a CIML document as a flat file. Other methods include hand-typing a document or creating a template using a Microsoft Word format. Authoring a CIML document from various sources can be difficult. Issues can occur such as differences between CIML elements and the source of investigative information, centralization of investigation information, having sufficient resources to perform the automated authoring of a CIML, and overcoming information obfuscation and inter-service/agency political boundaries.

Many benefits are gained by this implementation. As investigative information comes from the field into the LE and CI database, correlated and conflicting pieces of information will be identified, aiding law enforcement

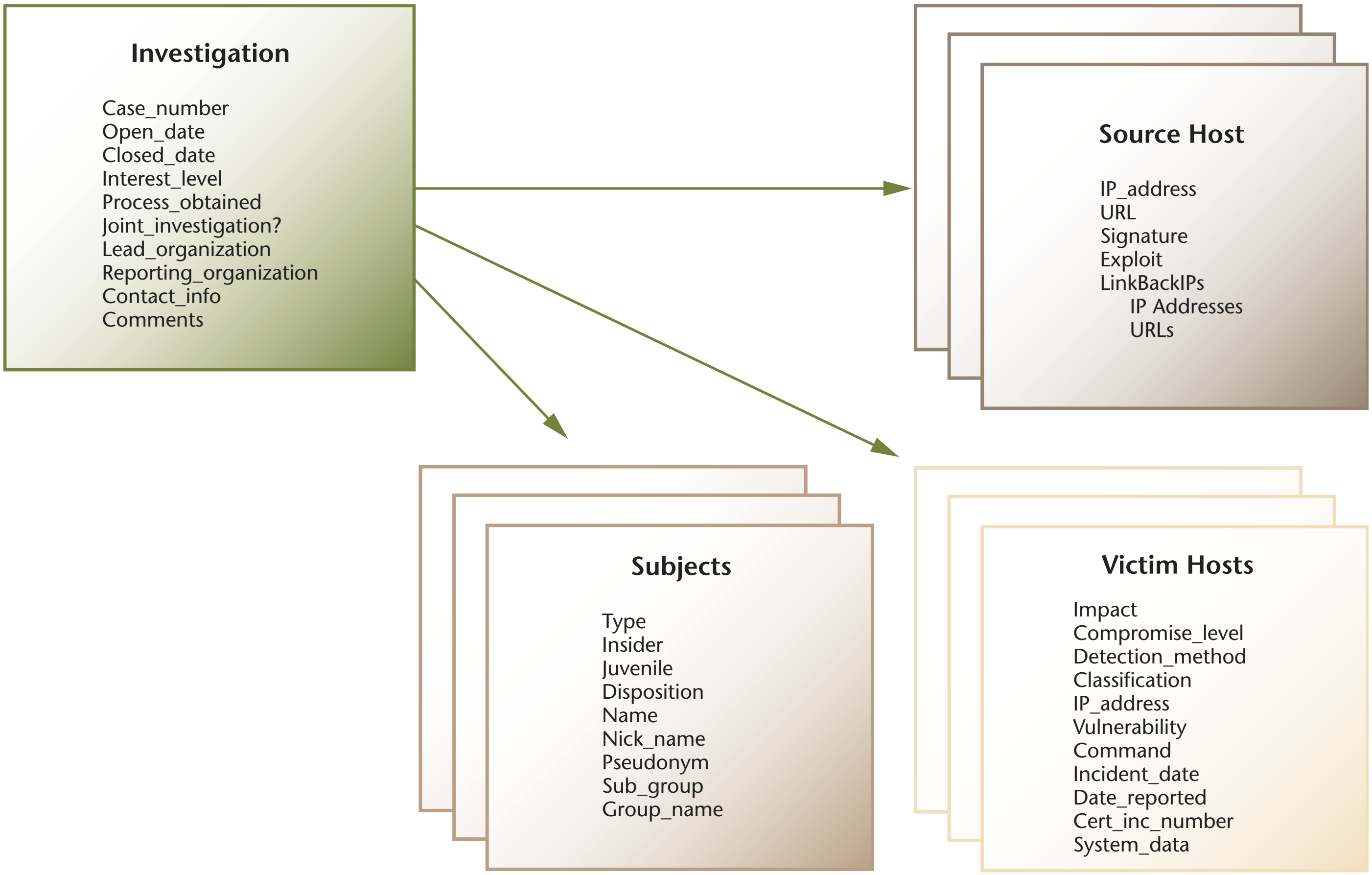


Figure 1. This model shows most of the information contained in a CIML document

and counterintelligence agents who will work within their service component or across services. De-conflicting case information is the cornerstone of the CIML LECI database. In the end, the benefit to all LE and CI communities will be information dominance and optimal resource utilization. ■

About the Authors

Keith Hasselstrom

Keith Hasselstrom supports the Joint Task Force-Global Network Operations Law Enforcement Counter Intelligence Center (JTF-GNO LE/CIC) as the principle Software Engineer/Knowledge Manager. He has over seven years of experience in the field of computer network defense (CND) systems architecture design and engineering. Hasselstrom holds a BS in Information Technology from the Hawaii Pacific University and a MS in Object Oriented Programming and Database Technologies from Regis University. Keith Hasselstrom can be contacted directly at the JTF-GNO LE/CIC at 703/607-6364 or hasselsk@jtfcno.ia.mil.

Peter Tran

Peter Tran supports the Joint Task Force-Global Network Operations Law Enforcement Counter Intelligence Center (JTF-GNO LE/CIC). His main area of expertise encompasses designing and deploying proactive information system protections, cyber attack detection and performing information assurance (IA) threat analysis and operations. Mr. Tran served as a Special Agent in the Computer Investigations Division of the U.S. Naval Criminal Investigative Service (NCIS). Mr. Tran holds numerous software patents in biometric data authentication over TCP/IP and has a broad research background in automated comparative forensic analysis of genomes over networks from the Harvard University where he was a research fellow in 1995. Mr. Tran holds a Master of Forensic Sciences from the George Washington University, a Bachelor of Arts from the University of California at Santa Barbara and is an MBA candidate at the Johns Hopkins University. Peter Tran can be contacted directly at the JTF-GNO LE/CIC at 703/607-7255 or tranp@jtfcno.ia.mil.

Secure Configuration Compliance Validation Tool (SCCVT) Selection

An acquisition effort led by the Defense Information Systems Agency (DISA) Chief Information Assurance Executive (CIAE) Office on behalf of the DoD Enterprise-wide Information Assurance (IA) and Computer Network Defense (CND) Solutions Steering Group (ESSG) resulted in a competitive contract award for eEye Digital Security's Retina® vulnerability scanner product. The Retina® tool will provide network administrators and security personnel with the ability to verify compliance with DoD CERT Information Assurance Vulnerability Management (IAVM) notices (IAVA, B, TA) as well as vulnerability notices published by the Service CERTS/CIRTS. The tool will undergo a 60-day rapid deployment beginning in mid-July with installations at JFCOM and test sites from each Service. With the acquisition of the SCCVT tool completed, preparations for the companion remediation tool selection are underway. ■

The Semantic Web

by Dina Dywan

This paper describes Tim Berners-Lee's futuristic vision known as the Semantic Web. The work of leaders in academia, the government and the commercial industry are discussed along with the steps they are taking in moving the Semantic Web technologies forward. A review of current standards efforts led by the World Wide Web Consortium (W3C) is given. This paper also discusses several current applications of the Semantic Web, the anticipated threats and vulnerabilities along with suggested countermeasures, and finally addresses an ongoing Semantic Web security initiative.

In a June 2002 interview on Meet the Press, FBI director Robert Mueller discussed the information management dilemma intelligence agencies have had in the past decade. Relating this problem to the events of September 11, 2001, Director Mueller stated—

> *"It would be nice if we had the computers in the FBI that were tied in to the CIA that you could go in and do flight schools, and any report relating to flight schools. What would be even better is if you had the artificial intelligence so that you don't even have to make the query, but to look at patterns like that in reports".*

What Director Mueller was describing is a Semantic Web, which allows not only users, but also powerful computing agents, to find hidden relationships between data in government, corporate, and personal databases that already exist. [1]

On the Semantic Web, not just web pages, but databases, programs, sensors, and even household appliances will be able to present data, multimedia and status information in ways that will permit a Web user to search, filter and prepare information in new and exciting ways. To fully understand the potential impact of the Semantic Web, imagine going to a search engine and typing the query "How many people live in the United States?" There will be many documents found, but few, if any, will actually contain the answer to the question being asked. There are a few reasons why this answer is difficult

to find. First, current language processing and search technologies are nowhere near good enough to find the correct documents. Second, there are web-accessible databases and programs that could be accessed to provide the answer, but word-based text matching is not sufficient to pull them out. Third, a complicated program could be written to find the pages containing population information, identify the data, and retrieve the total count, but this is a massive undertaking and requires more effort than users would be willing to make. It is possible to imagine a day when a Semantic Web query tool could provide answers like—

- http://www.census.gov/main/www/popclock.html says the United States population is 292,547,805
- There is a database that can provide that number, but an authorization number is needed
- There is a web service that can compute the number, but it will cost $100

The Semantic Web will be an extension of the current Web, providing a better way for computers and people to work together. It will drastically improve our current methods of finding, sorting, and classifying information by using agents to perform sophisticated tasks for users. [2]

Description

In order to create the Semantic Web, smarter, machine-processable data must be created. The initial four stages of the "smart data" continuum are—

- **Text and databases (pre-XML)**—The initial stage where most data is proprietary to an application. The "smarts" are in the application and not in the data
- **XML documents for a single domain**—The stage where documents and data achieve application independence within a specific domain. XML has a standard syntax for metadata along with a standard structure for both documents and data. By using an open, standard syntax and verbose descriptions of the meaning of the data, XML is readable and understandable by everyone and not just the application and those that produced it. The data is now considered "smart" and can move between applications in a single domain. [1]
- **Taxonomies and documents with mixed vocabularies**—Data can now be composed from multiple domains and accurately classified in a hierarchical taxonomy. Relationships between categories can be used to relate and combine data. This classification capability makes data smart enough to be easily discovered and sensibly combined with other data
- **Ontologies and inference rules**—An ontology is a document or file that formally defines the relationships among terms. The most common type of ontology for the Web has a taxonomy and a set of inference rules. The taxonomy defines classes of objects and rela-

> tions among them. An ontology may be used to express a rule "If a city code is associated with a state code, and an address uses that city code, then that address has the associated state code." The computer doesn't really understand this information, but it can now utilize the terms more effectively in ways that are useful and meaningful to a human user. [3]

The evolution of smart data will be accompanied by the development of new web languages to translate Hypertext Markup Language (HTML) documents to metadata languages such as Extensible Markup Language (XML) and Resource Description Framework (RDF). Languages will also be created to allow ontologies, rules, proofs, and logics to be realized at a web-wide scale. Tim Berners-Lee, the inventor of HTML, HyperText Transfer Protocol (HTTP), Uniform Resource Identifiers (URIs), and creator of the World Wide Web's first browser, also created "the layer cake" to depict how these new web languages are related to one another. Figure 1 (on this page) displays this model.

The Uniform Resource Identifier (URI) is a fundamental component of the current Web and is the foundation of the Semantic Web. One form of URI is the Uniform Resource Locator (URL), most commonly known as the address that lets a user visit a Web page such as http://www.yahoo.com. URIs provide users with the ability to uniquely identify resources as well as relationships among the resources. XML allows users to add arbitrary structure to their documents through tags. These tags are essentially hidden labels that annotate web pages or sections of text on a page. RDF expresses the meaning of the structure of these tags. An RDF statement is encoded in a set of triples, which are a lot like a simple sentence with three parts—a subject, a verb, and an object. In an RDF triple, almost all the words are composed of URIs and XML tags. These triples make the statement machine-processable. However, writing these RDF statements becomes very time-consuming. It is therefore expected that much of the RDF information will come from databases. A 'semantic link' between similar terms such as a database with a 'zip-code' column and a form with a 'zip' field that means the same thing will need to be made. Ontologies are ways to describe the meaning and relationships of terms. Web Ontology Language (OWL), and Defense Advanced Research Projects Agency (DARPA) Agent Markup Language with Ontology Inference Layer (DAML+OIL) have been developed to create the 'semantic link' between terms by providing information on how to convert between them.

The remaining three layers of the cake are conceptual. They are being explored in research labs, but are not yet available on the Semantic Web. While having a system that understands basic concepts is very helpful, it would be even better to make logical statements that allow the computer to make inferences and deductions. Once systems are built following logic, they can be used in proofs. Different people around the world can write logical statements and software agents can follow the Semantic "links" to begin to prove facts. These software agents are programs that collect Web content from diverse sources, process the information and exchange the results with other programs. The software agents will use encrypted blocks of data known as digital signatures to verify that the information is true. Digital signatures can provide proof that a certain person wrote (or agrees with) a document or statement. A user can then designate whose signatures to trust and whose not to trust by setting their own levels of trust. [5]

Implementation

In August 2002, the Gartner Group reported—

> *"By 2005, lightweight ontologies will be part of 75 percent of application integration projects."*

The World Wide Web Consortium (W3C), the Internet Engineering Task Force (IETF), and the Organization for the Advancement of Structured Information Standards (OASIS) have had widespread support from corporations and academic institutions alike for interoperability. The support of XML has spawned support of XML-based technologies, such as Simple Object Access Protocol (SOAP) based Web services that provide interoperable interfaces into applications over the Internet. Using XML as a serialization syntax, RDF is the foundation of other ontology-based languages of the Semantic Web. Ontology languages such as OWL and DAML+OIL are already being used by many organizations to add semantics to their corporate knowledge bases. Adobe, for example, is reorganizing its software metadata around RDF, and they are using Web ontology-level power for managing documents. IBM is also performing significant research at its Institute of Search and Text Analysis in California. Other companies, such as Germany's Ontoprise, are making a business out of ontologies, creating tools for knowledge modeling, knowledge retrieval, and knowledge integration. In the

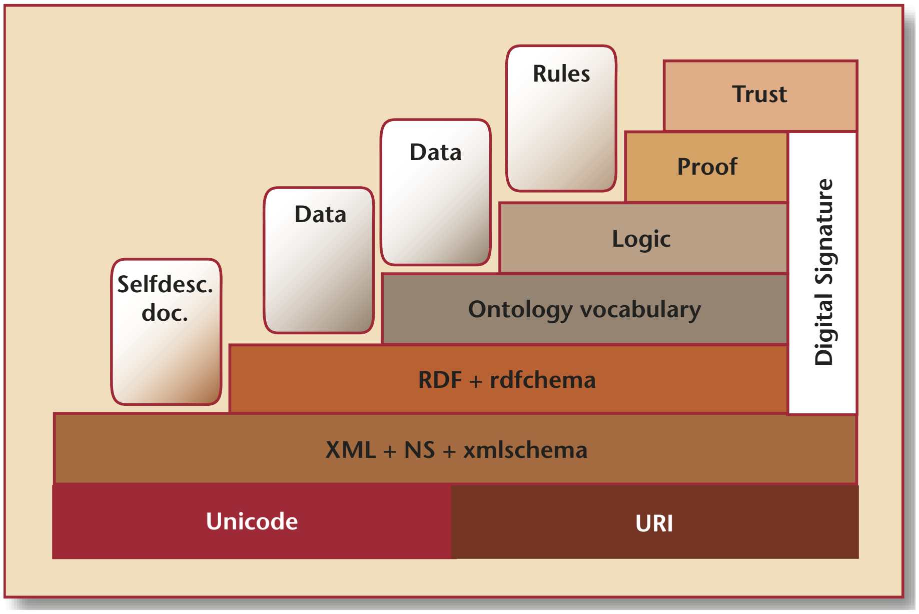


Figure 1. The Layer Cake [4]

same Gartner Group report, it was also recommended that enterprises should begin to develop the need for semantic modeling and information management skills within their integration competence centers. [6]

Leaders

The Semantic Web is progressing in two places—in the efforts coordinated by the W3C, and by those promoting the Semantic Web technologies through demonstrations, applications, and products. The W3C has organized four initiatives that are designed to work in collaboration with a large number of researchers and industrial partners to stimulate development and facilitate the deployment and future standards work associated with the Semantic Web. These Semantic Web Advanced Development (SWAD) initiatives include—

- **SWAD DAML**—This initiative is intended to build on the DAML infrastructure in an effort to demonstrate how it can be used by two or more working, user-oriented applications
- **SWAD-Europe**—This initiative will highlight practical examples of where real value can be added to the Web through Semantic Web technologies. The focus will be on demonstrating how the Semantic Web can address problems in areas such as site maps, calendaring, scheduling, quality ratings, Web service description and discovery, trust and rights management and how to integrate the technologies together.
- **SWAD Simile**—Being developed through a collaboration by Hewlett Packard, the Massachusetts Institute of Technology (MIT) Libraries, and MIT's Lab for Computer Science (LCS), this initiative is intended to enhance interoperability among digital assets, schemas, metadata, and services across distributed individual, community, and institutional stores and across value chains that provide useful end-user services by drawing upon the assets, schemas, and metadata held in such stores.
- **SWAD Oxygen**—MIT/LCS has formed Project Oxygen to enable pervasive, human-centered computing through various user and system technologies. The Semantic Web is intended to provide similar computing abilities. SWAD Oxygen is the joint effort between the W3C and Project Oxygen in working towards these goals. [7]

The Hewlett Packard (HP) Labs Semantic Web research group believes that the Semantic Web represents a huge potential technology disrupter, enabling new and more flexible approaches to data integration, web services, and knowledge discovery. The group is committed to furthering the Semantic Web vision through—

- Standards work via the W3C to support the growth of the Semantic Web
- Tools development to encourage the exploration and exploitation of the Semantic Web by developers
- Core research to help develop the field
- Applications research to demonstrate the value of the Semantic Web
- Consultancy within HP to promote its use within the company [8]

HP has already developed two open source software packages called Jena and Joseki. Jena is a Java framework for writing Semantic Web applications. Joseki is the Jena RDF server. Jena features an RDF Application Programming Interface (API) for manipulating an RDF model as a set of RDF triples, an RDF/XML parser that is fully compliant with the RDF Core Working Group, a reasoning subsystem to construct inference models, and an ontology API. [9]

Standards

The W3C Semantic Web Activity ("the Activity") has been established to serve a leadership role in both the design of specifications and the open, collaborative development of technology. The Activity is focused on the design and development of enabling standards. Specifically, the RDF Core and Web Ontology Working Groups have been tasked to provide a set of base level standards for supporting the Semantic Web. The RDF Core Working Group is co-chaired by Brian McBride of HP Labs and Dan Brickley of the W3C. This standards effort is designed to revise the RDF Model and Syntax Recommendation, complete work on the RDF Schema Specification, and provide a means to support tighter integration with the XML Schema Part 2: Datatypes Recommendation. Jim Hendler of the University of Maryland chairs the Web Ontology Working Group. This effort is designed to build a language upon the RDF Core for defining structured, Web-based ontologies, which will provide richer integration and interoperability of data among descriptive communities. [10]

Applications

McDonald Bradley, Inc. (MBI) has worked with the Department of Defense (DoD) to complete a multimillion-dollar contract on the department's intelligence Virtual Knowledge Base (VKB) project. For this project, MBI focused on the high-level Internet architecture and high-end requirements, especially security applications. It also looked at how to collect information from different systems and get the most relevant data to the war fighter in the field quickly, safely, and securely. [11] By exposing all information sources as Web services, abstracting the details into knowledge objects, providing an ontology for mining associations between data elements, and providing a registry for the discovery of information sources, the VKB is utilizing key Semantic Web concepts and technologies to solve the information management quandary that every organization today faces. [1]

MBI is currently working on a one-year, $7.8 million contract to help unify the DoD back-end intelligence systems. The contract is for the Defense Information Systems Agency's (DISA) new Net-Centric Enterprise Services (NCES) project, which will provide enterprise services in support of the Global Information Grid (GIG). The project will allow authorized users to quickly draw information from many intelligence, surveillance, and reconnaissance resources and configure them into unique combinations via a Web browser. It will use emerging standards and the experience gained on the VKB project to develop an open-standard solution that is compliant with Sun Microsystems Inc. Java 2 Enterprise Edition.11 MBI will explore the use of metadata tagging, data-content markup, taxonomies, ontologies, and other XML and Semantic Web techniques to help systems index and provide information. Users may include military intelligence, as well as agents from the Central Intelligence Agency, Defense Intelligence Agency, the National Security Agency, and the Department of State. Access based on individual users' security clearance will be built into the design.[12]

In his February 2002 budget submission to Congress, President Bush outlined a management agenda for making government more focused on citizens and results, which includes expanding Electronic Government (E-Government). E-Government uses improved Internet-based technology to make it easy for citizens and businesses to interact with the government, save taxpayer dollars, and streamline citizen-to-government communications. The Federal Enterprise Architecture (FEA), a business and performance-based framework to support cross-agency collaboration, transformation, and government-wide improvement, is using semantic technologies to meet the goals of E-Government. These technologies are being used because they offer the power of managed relationships for discovering knowledge about the FEA models, about government agencies and bureaus, and about partnerships and programs of work. They are also being used for their power of models based on RDF for federated navigation and inference of distributed models. [13]

Security Implications

To succeed in the process of information retrieval, the Semantic Web will be required to ensure security, privacy and trust. As information is used more and more by agents and not human users, security becomes increasingly important, and it is crucial for security mechanisms to be embedded into the fabric of the Semantic Web and be as automated as possible. The obvious solution is to extend security mechanisms applicable to distributed systems (e.g. Kerberos, Public Key Infrastructure, etc.) to the Semantic Web. However, this may be difficult given the completely decentralized nature of the web and the extremely large number of agents and users. Along with this is the problem of semantic meaning of the security information; it is not feasible to expect all entities to use the same terminology to represent security protocols and information. A security framework for the Web must be flexible, semantically rich, impervious to common network problems like network partitioning, and simple enough to automate.

In order to secure the Semantic Web two main components are required: a semantic policy language for defining security requirements and a distributed trust management approach. As the Semantic Web is composed of web resources (e.g., web pages, web services, agents, and human users), security should be uniformly applied to, and be applicable to, all. Decisions about who to believe and who to serve must be based on an entity's credentials, delegation assertions, and the appropriate security policy. It is important for Web entities to be able to express their security and belief information in a clear and concise manner so that its meaning is unambiguous. The e-Biquity group at the University of Maryland Baltimore County suggests the use of a policy language based on a semantic language like RDF or OWL to markup security information. This policy language will provide constructs for users, agents, and web resources to clearly define their security requirements. [14]

In 1996, the term "distributed trust management" was introduced and defined as creating policies, assigning credentials to users, and checking if the credentials of the requester conform to the policy before granting access to it. When combining several RDF statements together, there comes the issue of whether or not they can be trusted. RDF builds a "Web of Trust," taking in to account the user's credentials, the requester's credentials, whom the user trusts and how much they trust them. The metadata will then be filtered based on this level of trust. The following could be used to filter the metadata—

- **E-mail**—The metadata could be filtered based on the E-mail address of the requester. For example, one could trust members of the Yahoo! group "sw-discuss"
- **Digital signatures**—Digital signatures provide proof that a certain person wrote (or agrees with) a document or statement. RDF uses these signatures to ensure that the triples were actually said by who they're claimed to be said by. It is up to the user to determine whose signatures to trust and whose not to. For example, a user could give all requesters with verified digital certificates a trust level of six and all others a trust level of zero
- **Ratings**—A user could browse to a web page and find it completely useless. The browser would tell the user that ten percent of people visiting the page thought it was excellent. The user could then tell the browser to assign a trust rating of zero to all people that thought it was a good page
- **Affinity**—If the user's main goal is to filter out all metadata except that which is most likely to match their interests and opinions, there are data mining techniques that can cluster data based on patterns of similarity. This is the same technique used by http://www.amazon.com to let a user know that "other people who liked this book also liked these other books." The metadata layer could automatically detect clusters of similarity and group users based on those patterns [15]

Security initiatives

A "Web of Trust" is one of the ultimate goals of the Semantic Web. The "Semantic Web of Trust" requires that users describe their beliefs about others. The Friend of a Friend (FOAF) project is one that allows users to create and interlink statements about who they know, building a web of acquaintances. It is managed as a collaborative effort among Semantic Web developers on the FOAF (rdfweb-dev@vapours.rdfweborg) mailing list. FOAF is about creating a Web of machine-readable homepages describing people, the links between them and the things they create and do. Users can sign these files so information will be attributed to either a known source, or an explicitly anonymous source. In this project, a schema is introduced to allow users to indicate a level of trust for people and subjects. [16] The FOAF vocabulary, originated by Dan Brickley and Libby Miller, allows the expression of personal information and relationships, and is a useful building block for creating information systems that support online communities. The initial focus of FOAF has been on the description of people, since people are the things that link together most of the things described on the Web: people make documents, attend meetings, are depicted in photos, and so on.

The FOAF vocabulary definitions are written using RDF/OWL to make it easy for software to process some basic facts about the terms in the FOAF vocabulary, and consequently about the things described in FOAF documents. If people publish information in the FOAF document format, machines will be able to make use of that information. If those files contain “see also” references to other such documents in the Web, we will have a machine-friendly version of today’s hypertext Web. Computer programs will be able to search a Web of documents designed for machines rather than humans, storing the information they find, keeping a list of “see also” pointers to other documents, checking digital signatures and building Web pages and question-answering services based on the harvested documents. [17]

Conclusion

Had Semantic Web technology existed in 2002, a fingerprint found on a catalog in Montgomery, Alabama, could have been linked to one in an INS database in Washington to more quickly assist investigators in the Washington, DC sniper shooting case. By using XML tags to make data “smarter”, users are allowed to perform much more intelligent and sophisticated searches than they can with today’s limited keyword searches. The benefits of these complex searches are immeasurable, and if properly designed, the Semantic Web has the potential to improve the evolution of human knowledge. ■

About the Author

Dina Dywan

Dina Dywan received a Bachelor of Science in Electrical Engineering from Purdue University. Her focus is on database security and secure application development.

References

1. Michael C. Daconta, Leo J. Obrst, Kevin T. Smith: The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management (Indianapolis, IN: Wiley Publishing, Inc.—Copyright © 2003 by Michael C. Daconta, Leo J. Obrst, and Kevin T. Smith—ISBN 0–471–43257–1)
2. http://blogspace.com/rdf/SwartzHendler
3. http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
4. http://www.w3.org/2001/09/06-ecdl/slide17-0.html
5. http://blogspace.com/rdf/SwartzHendler
6. Jacobs, A. Linden, Gartner Group, Gartner Research Note T–17–5338, 20.August 2002.
7. http://www.ercim.org/publication/Ercim_News/enw51/berners-lee.html
8. http://www.hpl.hp.com/semweb/
9. http://www.hpl.hp.com/semweb/jena.htm
10. http://www.w3c.org/2001/sw/Activity
11. http://www.fcw.com/fcw/articles/2003/0317/web-nces-03-18-03.asp
12. http://www.washingtontechnology.com/cgi-bin/udt/im.display.printable?client.id=wtdaily-test&story.id=20307
13. http://www.web-services.gov/TQ%20-%20eGOV%20Pilot%20for%20SBIR.ppt
14. http://www.acm.org/sigmod/record/issues/0212/SPECIAL/5.Finin.pdf
15. http://www.netcrucible.com/semantic.html
16. http://www.mindswap.org/papers/Trust.pdf
17. http://xmlns.com/foaf/0.1/

…continued from page 27

“Autonomic Computing”

http://www.elsevier.nl/locate/entcs/volume58.html 16 pages

21 Padma Kapse: Security in Mobile Agents. http://www.cse.fau.edu/~security/public/Security_in_Mobile_Agents.ppt

22. Mobile Code Security and Computing with Encrypted Functions. http://www.zurich.ibm.com/security/mobile/

23. Hyungjick Lee, Samsung Electronics Ltd., Jim Alves-Foss and Scott Harrison, University of Idaho Center for Secure and Dependable Systems: The Use of Encrypted Functions for Mobile Agent Security (Proceedings of the 37th Hawaii International Conference on System Sciences, 2004). http://csdl.computer.org/comp/proceedings/hicss/2004/2056/09/205690297b.pdf

For further reading (not specifically cited in this article)

Autonomic Computing Event at Woodrow Wilson International Center for Scholars (presentations). http://www.thefutureofcomputing.org/autonomic-computing.html

Patrick Hinnelund, School of Computer Science and Engineering, Swedish Royal Institute of Technology: Autonomic Computing for Volatile Systems (Master’s Thesis, Draft, March 2004). http://www.student.nada.kth.se/~d99-phi/mthesis.pdf

Richard Murch: Autonomic Computing (IBM Press/Prentice Hall PTR/Pearson Education, 2004 - ISBN 0–13–144025–X)

Ahmar Abbas, Grid Technology Partners: Special Report on Autonomic Computing: Characteristics of Self-Managing IT Systems

Detecting Early Indications of a Malicious Insider

by Sara Matzner and Tom Hetherington

While significant effort has been put into building defenses against network-based attacks, little attention has been paid to a more insidious threat, that of a trusted employee who is inappropriately accessing data. Most organizations depend heavily on traditional network-based security systems such as firewalls and intrusion detection systems to protect them from the "external threat" emanating from the Internet, and the potential damage from an external hacker or a virus remains the most significant concern for those in charge of the security of networks. However, the "insider threat"—potential sabotage of crucial systems or unauthorized release of sensitive information by an insider—is by some measures the far greater worry.

In 1998, the Computer Security Institute and the FBI reported that the average cost of computer attacks from external threats was $56,000, while malicious acts by insiders are estimated to cost $2.7 million per occurrence. A study by the United Nations Commission on Crime and Criminal Justice, which surveyed 3,000 sites in Canada, Europe, and the United States, found that the "greatest security threat came from their own employees or other people with access to their computers." [1] The exact impact of the insider threat is difficult to ascertain because misdeeds by trusted employees often go unreported, since organizations don't want the negative publicity that could result from a court case (and, because policies are often not in-place or are unclear, the outcome from a court case is not certain anyway). Thus, public data on these insider events primarily describes a minimum level of threat, but that level is enough to draw attention to the problem as a whole.

Cyber indications and warnings

Many branches of government, including the National Science Foundation, DARPA, the U.S. Secret Service, and the Computer Science and Telecommunications Board of the National Academies, have funded research on the insider threat problem. As part of their Advanced Information Assurance Challenge Problems, the Advanced Research and Development Activity (ARDA) recently sponsored a Cyber Indications and Warnings Workshop to develop information assurance tools that reliably predict the presence of a malicious insider and provide some clues to his identity. Personnel from the Applied Research Laboratories, The University of Texas at Austin, participated in this workshop, comprising a major portion of the Data Fusion team, as discussed in more detail below.

The focus of the ARDA effort was on sophisticated attacks to secure facilities. The concern is with an attack by a malicious insider sponsored internally by a nation-state, where the insiders would have access to almost unlimited resources and would employ advanced techniques to hide their presence on a compromised system. [2]

The Cyber Indications and Warnings workshop included subject-area experts from diverse disciplines and organizations, including universities and private companies working together as a team, with representatives from several government agencies. The government representatives directed and mentored the investigation to ensure that the outcome was both realistic and representative of the government's working environment. Workshop participants were to design and develop a proof-of-concept system for early indication and warning of malicious insiders. The emphasis was on timeliness of detection, taking a preventive approach, rather than concentrating on after-the-fact (forensic) analysis. The goal was to detect the intrusion and the intruder much earlier than has been the case in recent history of insider discovery. The historical cases have taken years. For instance, Robert Hanssen operated undetected as a spy within the FBI for over 20 years—Aldrich Ames spied on the CIA for nine years. Thus, ARDA wanted approaches to the problem that would shorten the time between defection and detection. Moreover, the insider threat problem has become even more complex since the Hanssen and Ames incidents. Physical security remains a significant factor. However, as the workforce becomes more technologically savvy over time, malicious actions will increasingly involve insiders' cyber activities. Thus, the workshop participants concentrated on finding early indications and warnings of malicious cyber actions.

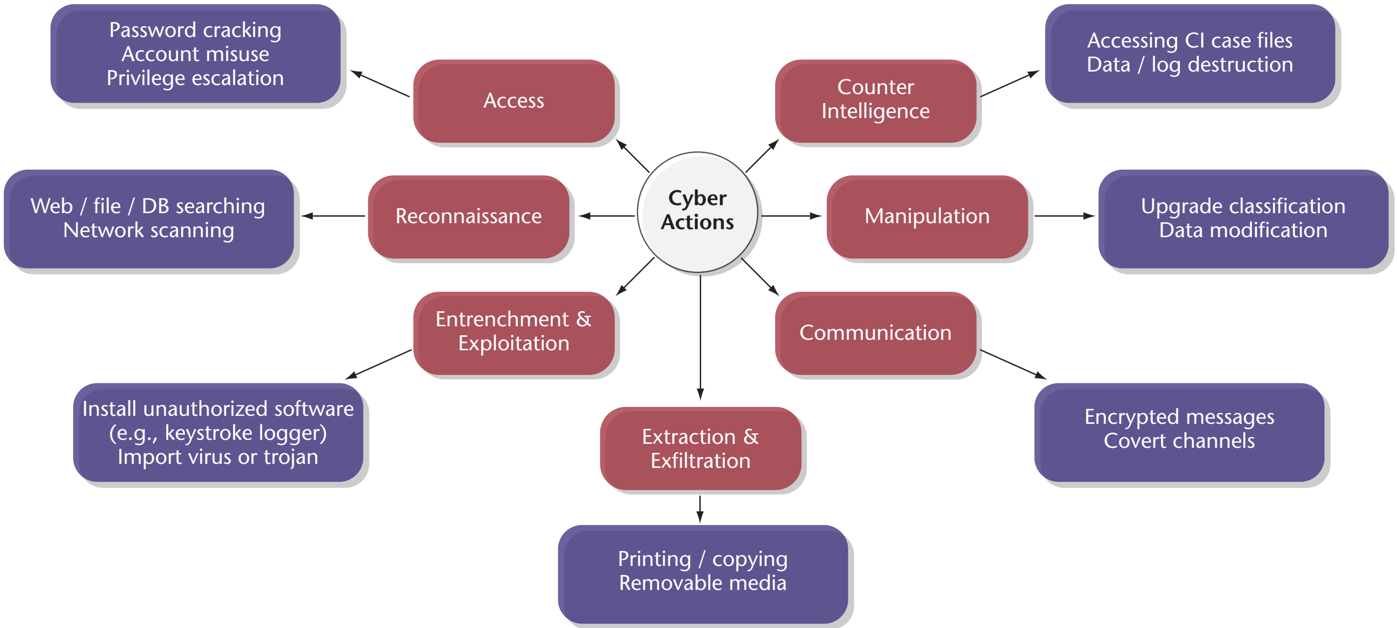


Figure 1. Taxonomy of observable cyber actions

Technical approach

The first step in the workshop study was to better understand the threat. We began with an extensive investigation of the case histories and affidavits of known insiders. From this study, we built a series of taxonomies that characterized the insiders' physical and cyber activities, their motivations, their roles within the organization, and the assets they targeted. Going beyond the known cases, we postulated other possible activities, motives and targets. Since we had made the assumption that future cases would involve more on-line activity, the next step was to build a taxonomy of cyber actions that could be observed by monitoring their behavior. Figure 1 (see above) shows this taxonomy (represented in red) with examples of actual observable behavior (represented in blue).

Following this general study of the insider problem, teams of researchers were formed from the various represented organizations based on the technical approach to be taken. One group of researchers investigated the use of honey nets in uncovering methods and targets. [3] Another group took a top-down structured analysis approach to the problem. They modeled the insider based on known behaviors and motivations and matched observed behaviors to the models. The Data Fusion team, in contrast, took a bottom-up approach. Input from multiple data sources (such as records of computer system, network and application activity, and physical facility access) were combined to augment the information provided by any single source.

Data fusion

The Data Fusion team took a data-centric approach to the problem, discovering evidence of malicious activity by correlating and analyzing a variety of low-level data. The result of this data fusion would be a "watch list" of names—individuals whose activities were suspicious and therefore might require further scrutiny. This approach is based on these hypotheses—

- While it is possible that evidence of malicious insider behavior might come from a single action, other examples of malicious behavior can be detected from an accumulation of low-level data from a number of diverse sources

- Fusing information from a variety of data sources will provide more accurate and timely indications and warning of insiders

Within any organization, there are many sources of electronic access data that could provide significant indications and warnings of malicious insider activity. However, many of these data sources are not being collected or, if they are collected, they are not being correlated and examined. Currently, the data fusion process across these logs is disjointed and manually intensive. A human must access each log separately and draw a conclusion. During our discussions with government representatives, the comment was made that output from many of the audit logs is not kept because no one knows what to do with the information. So, the Data Fusion team set out to prove that audit log data combined with other data sources is valuable, and can provide significant insight into an insider's cyber activities.

The data sources included observables of both cyber and physical access. System, application, and network intrusion detection logs were matched with physical access logs and compared with the user's role in the organization and the user's access permissions. A range of sources of information was considered (such as logs from printers, authentication mechanisms, card readers, telephone calls, and travel records). Fusion occurred along multiple dimensions—for example, according to the type of indicator (such as card reader or printer) or by the level of the IP stack (such as network or application). Indicators from these sources were then mapped to the taxonomy of observable cyber activities, as shown in Figure 1 (see page 43).

A simple yet very powerful example of the application of the data fusion process is the comparison of computer login information with badge-reader access logs. The correlation of these two data sources could indicate possible misuse of a user's computer account by showing that the user was logged on to a secure network but was not physically in the building at the time. This could be an instance of another person masquerading as the legitimate user of the account. Another example is based on a user's role within the organization and his digital access. Different roles have distinctly different patterns of computer usage. Print logs show significant printer usage for secretaries but not for system administrators. Overt actions such as scanning the network, information-level reconnaissance such as searching news groups or content on Web servers or performing bulk downloading of news groups may be unusual activity for certain users. Perhaps in isolation no one of these examples would warrant special scrutiny of an individual's behavior. But several such actions, especially outside the user's normal pattern or beyond his assigned role or tasking, would be worth further investigation.

The data fusion effort resulted in a number of significant accomplishments. A proof-of-concept data fusion engine and new sensors (to monitor specific forms of insider behavior) were developed and tested. An assessment was made of needed future research efforts.

Information was accumulated about both the physical and cyber activities of a test network. Logs from network intrusion detection systems were used to generate indications of unusual protocol activity such as ftp and http data uploads. Printer logs were analyzed and compared to the user's role within the organization as well as with the user's normal habits. Badge logs were used to determine unusual physical access. A novel sensor was developed that performed E-mail consistency checking. The contents of attachments were analyzed to determine if they contained masqueraded data. The E-mail sensor also performed consistency checks that determined what content types might be hiding under the generic binary data tag created by some E-mail client/servers. The E-mail sensor checked for unusual use of encryption for communication as a possible source of inappropriate release of information. As part of a simple experiment, the basic bottom-up approach was evaluated against several trial scenario-based attacks.

The data fusion engine correlated multiple indicators that were related by IP, user name, etc. into a single indicator. The grouping of all indicators for a particular user was designated as a *case*. Each case had an associated *weight factor* that was the sum of the weight for each of the corresponding indicators and a *breadth factor* that was a count of the distinct insider observable taxonomy elements represented by the corresponding indicators. Any case that reached a cumulative weight and breadth that was above a certain threshold was added to a "watch list"—a list of individuals requiring additional scrutiny. Certain features were selectable by the user, such as the criteria for generating an alert and for submission of names to a watch list based on the value of the breadth and weight of the indicators. Any case that reached a higher cumulative weight and breadth threshold generated an alert. Figure 2 (see below) shows the data fusion system architecture.

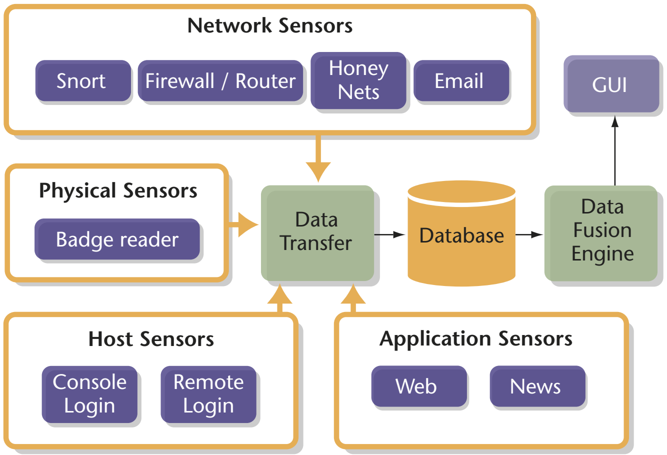


Figure 2. Data fusion system architecture

Conclusions

The final report of the Cyber Indications and Warnings workshop is being prepared by the Mitre Corporation. [4] For the Data Fusion effort, the results of the experiment supported the initial hypothesis—multiple and diverse cyber observables can provide early evidence of malicious activity. However, these results are preliminary—the Cyber Indications and Warnings workshop was a short-term effort to provide a proof of concept. So, while our data fusion engine provides a framework for incorporating additional data sources, the underlying system must be made more complete and more robust. The next steps should include inputs from host intrusion detection systems (IDS) and from other cyber and non-cyber observables. Future work to improve early evidence of malicious activity should include covering components of the observable taxonomy in greater depth and breadth. An important discovery was that standard audit logs currently are not designed with security in mind. In the course of our study, we found several ways in which these logs could be significantly improved for security purposes.

Many research questions remain as yet unexplored. For example, what is the minimal number of sensor inputs required to obtain reliable inferences? How do you acquire a normal pattern when employees change positions and roles within an organization? A longer-term consideration is the addition of a self-improving feedback loop to the overall process.

In the end, we have made a significant start to a very difficult problem. The technology has been advanced to allow earlier indications of malicious insider behavior. Another Robert Hanssen-like insider should not require 22 years to uncover. However, technological solutions cannot prevent certain actions. A malicious insider can still leak information by either photocopying documents or memorizing classified data. Moreover, while the technology to monitor activities of workers has improved significantly, there are many non-technical issues that have to be considered. So, on the one hand, we are cognizant of the fact that the technology needs to be used wisely so that innocent employees are not subjected to unwarranted investigations. On the other, we can now begin to address one of Robert Hanssen's comments when he was captured: "If I thought the risk of detection was great, I never would have done it." [5] ■

About the Authors

Sara Matzner

Sara Matzner is the Program Manager for the Cyber Information Assurance branch of the Signal and Information Sciences Laboratory of the Applied Research Laboratories, The University of Texas at Austin (ARL: UT). She received a B.A. from the University of Maryland and an M.A. from The University of Texas at Austin. She directs a group of scientists performing basic research and developing prototype systems for a number of federal government agencies. Ms. Matzner is a subject matter expert in the areas of information assurance and insider threat detection and as such publishes numerous technical papers on these subjects and acts as a trusted advisor to government. She has worked for ARL:UT for 18 years with a one year research assignment at the Los Alamos National Laboratory. Ms. Matzner may be reached at matzner@arlut.utexas.edu or at 512/835-3176.

Tom Hetherington

Tom Hetherington is a Project Manager in the Signal and Information Sciences Laboratory (SISL) at the University of Texas at Austin Applied Research Laboratories (ARL:UT). He received his B.S. in Computer and Electrical Engineering in 1990 and an M.A. in Computer Science (CS) from UT Austin in 1993. He has over ten years of experience in database applications and expert systems and is a domain expert in information assurance and computer network defense.

References

1. U.N. Commission on Crime and Criminal Justice, United Nations Manual on the Prevention and Control of Computer-related Crime, United Nations, New York, 1995).
2. See http://www.ic-arda.org/Advanced_IC/challenge.html
3. Spitzer, Lance, "Honeypots: Catching the Insider Threat," Proceedings of the 19th Annual Computer Security Applications Conference, IEEE, New York, NY, 2003.
4. To be published by the Mitre Corporation as MTR 04B0000014.
5. Newsday, Aug. 22, 2003.

International Cyber Awareness

by Robert Lentz

Editor's Note

This was delivered by Robert Lentz, Director of Information Assurance (IA), Office of the Assistant Secretary of Defense for Networks and Information Integration (OASD/NII), in Berlin, May 2004.

Last year we initiated serious discussion on the information security challenges to achieving netcentricity. I am very honored to address this prestigious group again on this critical subject. Thank you, Roger, for your leadership. To put things in perspective—*What are the security trends by 2010?*

It is estimated one cyberbug will hit the Internet every five minutes of every hour of every day. The number of security incidents will swell to 400,000 a year or 8,000 per week. Windows will approach 100 million lines of code. The average PC, which will cost $99, will contain nearly 200 million lines of code. Within that code, it is estimated there will exist two million bugs. Unimaginable computer power will challenge some of our best encryption. Sophisticated hacker tools will be widely available and will add another half-billion Internet users.

> *"So netcentricity is a dual-edged sword. We've heard about the tremendous operational advantages. But the security realities leave us at a critical crossroads."*

One consistent theme from last year's workshop is the economic advantages to pursuing netcentricity. No doubt the force-multiplying effect of netting the force and the awesome operational opportunities can't be passed by. But this is "Fool's Gold" if one does not address security up front, aggressively (not on the cheap), and globally. Don't *assume* it will get done; it's a tough job.

The costs to add security are significant if, after the architecture is locked in and the code written, the weapons platform enters acquisition on operational test and evaluation, or worse, a satellite system is launched.

In addition, the cost and operational damage to degraded critical infrastructures like the power system are another security reality. We've only seen a glimpse of the problem.

Software is the only modern technology that ignores quality until tested. The norm is one defect per every seven to ten lines of code.

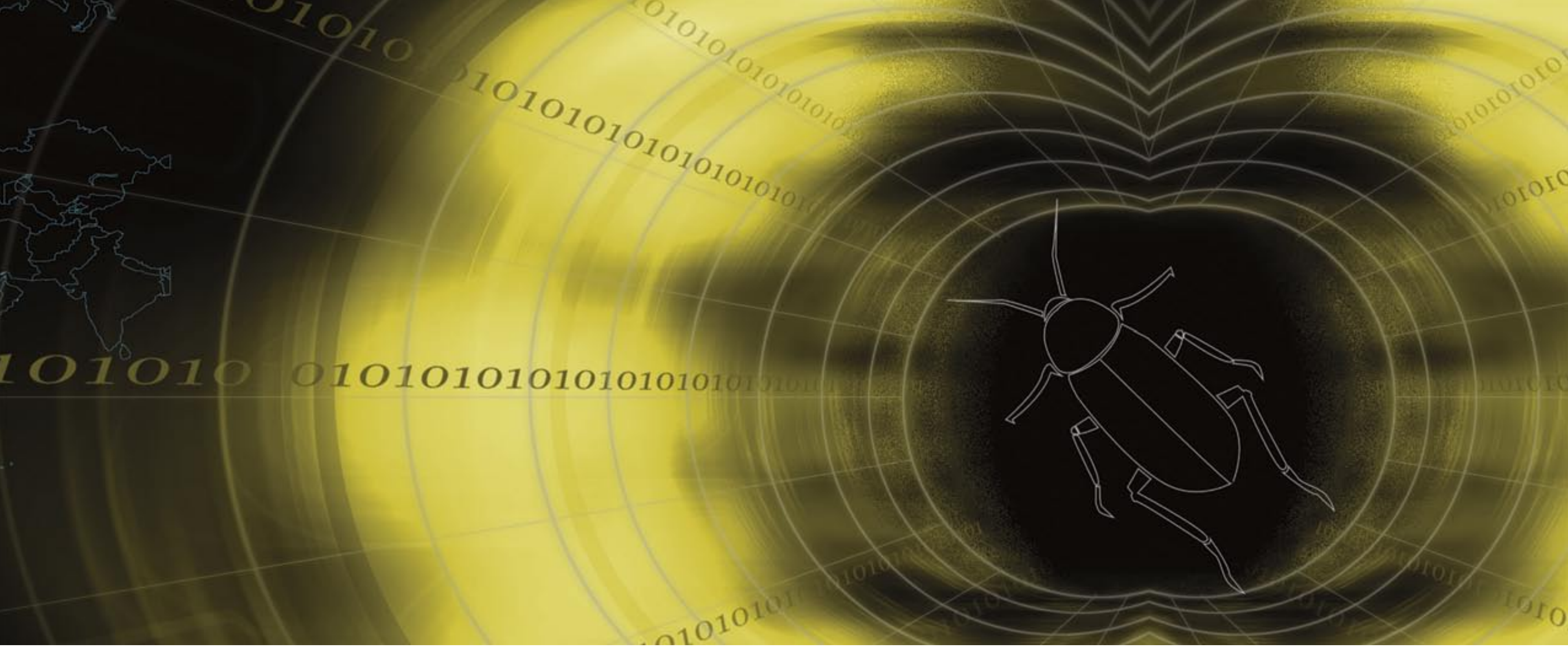
Today, we react to cyber events instead of anticipating them. Ninety-percent of successful cyber attacks are against known vulnerabilities. They are preventable. Seventy-percent of these vulnerabilities are the result of design defects. And each effort to react to a cyber attack and to patch a system costs from $1–4 million per patch.

Timing and speed are everything in this business, not only for the conduct of netcentric operations but also for defending our networks. Implementing layers of defenses will do no good without breaking down stovepipes and establishing horizontal relationships domestically and internationally.

As we evolve our IA capabilities to enable netcentricity, we need to provide several key services. First, we must ensure authentication through development of reliable cyber-identification credentials that will ensure that any person or computer requiring the identification of another person or computer can do so without worrying about an adversary trying to masquerade as someone legitimate. In addition, we must have interoperable standards that bridge all sectors including international partners.

Another key IA service will be "automation of privilege management." This service will provide end-user access to necessary information resources easily, regardless of where these are located on the network, and, conversely, will help the owners of these information sources to manage the accesses of all new end users that the netcentric world will bring.

Cyber attacks occur often and with great stealth. Critical operational processes must continue to function effectively while under cyber attack. Our IA strategy is based on the idea that appropriate defenses must stop most attacks. These protection mechanisms include physical, electronic, and procedural components and capabili-



ties to alert and warn us of attacks. The defenses must then be kept current during rapid evolutions of technology, attack strategies, and organizational change—all of which take a significant technical and operational effort. Should an adversary breach these protections, we must have the capability to detect, contain, and then respond to an attack.

IA entails high levels of situational awareness, significant analytical capabilities to characterize the nature and extent of an attack, the formulation and coordination of effective courses of action, and the ability to rapidly execute approved courses of action across a global infrastructure. The tools and procedures required to accomplish protection and reaction to attack in a highly technical, complex, joint multi-organizational environment are correspondingly sophisticated.

We cannot afford to let the wireless revolution sneak up on us again. What do you think our security confidence is on wireless systems? Not much. And worse, the cost to address wireless trust in a user-friendly way is very high.

It's certainly a challenging time for the IA community. The pace of real-world operations remains high. We have forces in harm's way in many places throughout the world, fighting the Global War on Terrorism. This is an international team fight.

> *"Not only do our military operations depend on our success, but, in a broader perspective, the computer infrastructures that drive our economies and services to our people also depend on our collective success. We must become organizationally agile and operationally adaptive."*

We must focus more on our processes and capabilities that provide products and services. Culture change is one of our biggest hurdles. We must re-think and re-implement accountability because of the paradigm shift to Network Centric Operations. Law enforcement must prosecute cyber attackers. We need vigorous IA discovery and invention, both scientific and technical, to complement and keep pace with IT innovations. Finally, we must do this work together for the greatest effectiveness and affordability. All of us have to trust the network and software. Recently, one security expert told Congress that software assurance is the next Manhattan Project, at a global level.

We need to know information can't be exploited or modified or that the wrong people have access or privileges on our systems. We need to know the net will be available when a decision maker needs it. Lastly, when we look the commander, the CEO, or policy maker in the eye, and we tell them to trust the network, we had better know what we are talking about. ■

About the Speaker

Robert Lentz

Robert Lentz is currently the Director, Information Assurance (IA), Office of the Assistant Secretary of Defense, Networks & Information Integration (NII) at the Pentagon. Mr. Lentz earned a Bachelor of Science Degree with a double major in History and Political Science from Saint Mary's College of Maryland and a Masters Degree in National Security Strategy from the National War College. While attending the National War College in 1999, Mr. Lentz's primary focus was on Homeland Security. Mr. Lentz is a graduate of the National Senior Cryptologic Course at the Cryptologic School, Federal Executive Institute (FEI), and the Resource Management Course at the Naval Postgraduate School.

Proceedings from the 2nd Semi-Annual Conference on Intelligence Support to Computer Network Defense

February 25–27, 2004
McLean, Virginia

by Timothy Madden

The U.S. Strategic Command (USSTRATCOM) and the Joint Task Force—Computer Network Operations (JTF–CNO) (now the Joint Task Force-Global Network Operations) hosted more than 140 registrants representing five nations, six departments, four agencies, the four U.S. military services, and 18 individual commands for the 2nd Semi-Annual Conference on Intelligence Support to Computer Network Defense (CND) in McLean, Virginia, February 25–27, 2004.

The conference theme of "Defending the networks, Defending the nation,"—

> *"Speaks to the increased importance of—and emphasis on—computer network operations in the current national security environment and the Global War on Terrorism (GWOT)," noted CDR Michael Greenwood (USN), JTF–CNO J2/Directorate of Intelligence in introducing the conference's keynote speaker, MG J. David Bryan, Commander, JTF–CNO (CJTF). "The brain trust at this conference is impressive, and we cannot waste this opportunity to enhance our understanding of CND. I can think of no one better qualified to open this conference than General Bryan. He has been in the trenches for a long time, and laid the foundation of cyber security for the Department of Defense and the nation."*

In remarks accompanying his presentation on the State of the Networks, MG Bryan said—

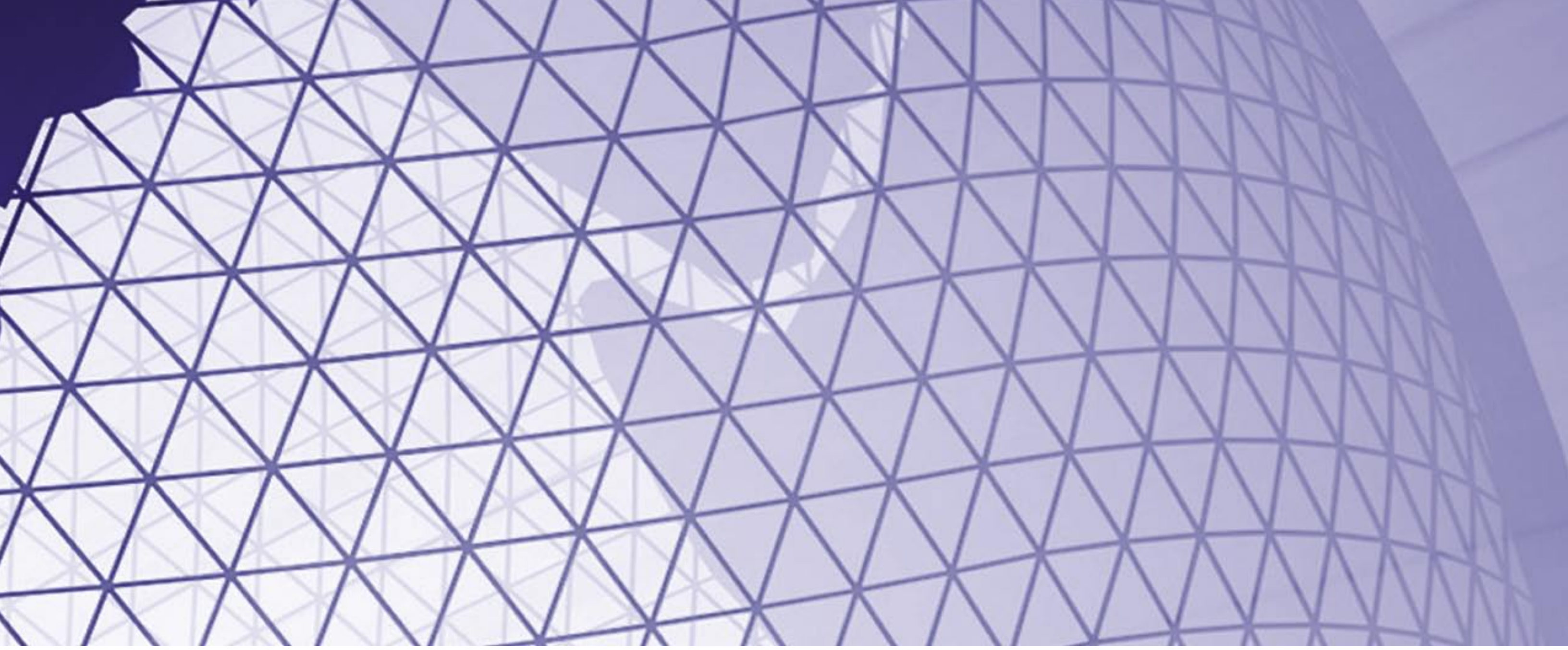
> *"We have been at war—in the Cyber World—since 1998. The range hasn't changed, but the capabilities have gotten far better—making computer network defense (CND) even more important because the stakes are higher. If we fail at CND, we risk our economy, our diplomacy, our expressions of power. Computer networks are the underpinnings of all advanced nations.*
>
> *When the Twin Towers went down, the priorities were, first, save lives; second, recover the casualties; third, get Wall Street—the nexus of the world's economy—back up and running. I am now assisting with the transformation of the Army, and our networks are basic to that effort. Making networks happen is not magic—it takes lots of very professional people, and the same is true of the intelligence supporting CND. This is not a pick-up game—it's hard, it's expensive, and it takes a committed cadre to force change through information technology.*
>
> *We now work in real time and are in front of attacks on our networks; the next natural evolution is to stop attacks before they happen. Intelligence is the key. If that support is not there for CND, we will fail. Exploits now are fast, and getting faster. They are virulent, and getting worse. We have nearly 50,000 root-level intrusion attempts every year in the DoD Global Information Grid (GIG), and these attempts aren't just noise—these are serious."*

MG Bryan discussed several major world events in the context of threat complexities, and issued a challenge to the attendees—

> *"Don't let old rules rule you in the future. Risk aversion can actually increase risk. You need to challenge existing doctrine. The evolution of the JTF from Computer Network Operations (CNO) to Global Network Operations (GNO) will give us arms around the entire GIG, and the secret to it all is you—the pro on the line."*

Following the keynote address, COL David C. Grohoski, Chief of the Information Operations (IO) Capabilities Division, Joint Force Headquarters-IO, USSTRATCOM, gave a presentation on the JFHQ-IO's mission, task organization, and CND requirements for intelligence. He said this was an "Infantry perspective" on the Secretary of Defense's vision for IO.

> *"There should be an integration and synchronization of the capabilities and functions discussed in the IO Roadmap," he said in comments accompanying his briefing, and that the "resulting synergy should be leveraged to support CND. We need to provide timely, accu-*



rate, relevant, and actionable intelligence. Intel support to CND means providing situational awareness and a common operational picture with which to develop proactive defensive plans. Our goal is to keep the network open for the warfighter."

He maintained that—

"Information sharing is broken, and it needs to be fixed. We need to leverage our relationships with industry and academia, and establish a common database of the threat—to fingerprint the hackers and adversaries. We need consistency in our product, and we need to develop the doctrine. We need to plan, prepare, and execute defense in depth, and find, fix, and finish the threat in the deep fight."

CDR Greenwood followed with his briefing on Intelligence Required in Support of CND, stating—

"Our business is operational; we do it because we support the warfighter, and the expectation is that we do our jobs." The intelligence support spectrum goes from understanding the threat to attribution of the action, he said, adding that "we have to examine our product in terms of our collection and its relation to CND needs."

In an exchange with the audience, CDR Greenwood agreed with the notion that integration, not collection, is the problem, but noted that "we are not vacuuming up everything we need to get, and much of what is collected isn't being reported to those who can best use it." He agreed that collaboration among allies is critical, indicating that the global unity of effort cannot be allowed to slide in the name of risk aversion.

In other briefings during the first day's morning session—

- Timothy D. Bloechl of the Office of the Assistant Secretary of Defense for Networks and Information Integration (OASD/NII), emphasized the integration theme, saying—

  "Commanders generally understand the problem of cyber security. A decade ago, no one thought about cyber security; now, almost everybody in the military knows we have an issue. The networks must be thought of as weapons systems and treated accordingly. When commanders and staffs at all levels understand this, not just network operators and cyber warriors, we will be well on our way to improving our defensive posture." He outlined the vision of "Dynamic Information Assurance for the Global Information Grid" and invited conference participants to examine that strategy.

- LTC Ed Sbrocco, Chief of Future Operations, JTF–CNO J3, defined and discussed CND Response Actions (CND RA), noting that "everyone has equities in this area, but deconfliction and cooperation need to occur without compromising investigations. There is a need for rapid attribution and rapid response, and that means forensics."

- Maj Douglas Kearsey, representing the CND Intelligence Branch at USSTRATCOM, discussed how the transfer of the IO/CND missions to USSTRATCOM was organized, noting that the command "needs partnerships with CND Intelligence stakeholders to get the job done."

The morning session on the second day began with the "National Security Council (NSC) Perspective on CND," presented by Col (S) Greg Rattray, Director for Cyberspace Security at the White House. The concern, he said, was one of survivability, noting that—

"The work and the effort have gotten harder as technology has advanced. We have to keep up and get ahead, responsibilities need to be defined and clarified, training and awareness levels need to increase, and we need to consider the full spectrum of the threat."

He characterized the recent Livewire Exercise as "the first baby steps to deal with the threat," and said the major issues are shaping the battlefield, dealing with insider risk, preparing for cyber espionage, improving attribution, and planning permissible defensive responses. He acknowledged that resource issues will compete, and challenged the cyber intelligence community to "step up with findings and make its case."

> *"The intelligence process is to define what the fight looks like and determine the difference between an incident and a campaign," he said. "We need to improve our counterintelligence capabilities and reach out to our international partners and global leaders, because what threatens them threatens us."*

Other presenters were—

- John Wolf of USSTRATCOM briefed attendees on evaluations of specific threats, which have led to attribution and insights into how certain hacker groups think.

- Ruby Huggins of the National Security Incident Response Center (NSIRC) outlined its relevance to CND, indicating that the intent is to share information and to collaborate at the lowest (most inclusive) level possible.

- Irene Schwarting of the Department of Energy (DoE) provided insights into the department's capabilities and concerns, noting that DoE is a "target-rich environment" because of its responsibilities as custodian of the nuclear stockpile, guardian of the nation's national power grid, and home of extensive research and development projects into critical technologies.

- Gil Trenum of the Law Enforcement/Counterintelligence (LE/CI) Center at JTF–CNO described the LE/CI's mission as one of support, coordination, and deconfliction. He noted that visibility of the Center's efforts and its tasking authority are issues that need to be resolved.

- Don Lewis of Defense Intelligence Agency (DIA) discussed the changing nature and role of the Military Infrastructure Office's Information Operations Threat Analysis Division (MIO-9). His main issues were the need to refine and narrow the descriptions and definitions of threat warning assessments; technical vulnerabilities within organizations, such as software anomalies and weaknesses; and threats posed by commercial off-the-shelf technology, noting that most assembly and fabrication of IT products is being done off-shore.

- Georgianna Hasselstrom briefed a U.S.-only audience on the objectives, focus, and key issues of current CND operations.

- Ken San Nicholas of JTF-CNO's J2 reported on the International CND Cooperation Working Group (ICCWG), a five-eyes (USA, GBR, CAN, AUS, NZL) information-sharing vehicle. He discussed the goals and initiatives of the organization's sub-working groups (SWGs), and elaborated on specific connectivity efforts. "What we do must include allies," he concluded, "because this is an international problem. It's not just the U.S., it's all of us, all of the time."

- Doug Gardner of the Applied Technology Unit (ATU), JTF-CNO, concluded the conference's final plenary session with a report on an ongoing effort to help the intelligence effort "by creating a single view of the problem, a way to see all events being traced, and gauging them by the likelihood and level of danger. That gives us visibility into the process of creating a threat matrix." The effort, he concluded, needs a collaborative approach, using information from all sources, and the results would be shared with all.

The third day of the conference was highlighted by a visit from Lt Gen Thomas B. Goslin, Jr., Deputy Commander, USSTRATCOM, and Commander, Joint Force Headquarters-Information Operations (JFHQ–IO). He commended the attendees for—

> *"The importance of the task," and added, "information is the coin of the realm. All tactics are based on information, good or bad. Today, all things are complex because everything is connected, but there are a lot of stovepipes. We need to make sure things work right; we need integrated information, regardless of who we are, and the need will only get greater."*

> *"Once 9/11 happened," he said, "the continental United States became a battleground, an Area of Responsibility with a Commander. We had never been there before. We need integrated information—good, credible, workable—from the cop on the beat to the very top. That's when we will be most effective in battling the threats against us, whether criminal or terrorist."*

He concluded by directing the attendees to—

> *"Keep talking to us, let us know who we need to keep talking to. We need you, you know the details, you do the work. You are going to provide ideas. Innovation starts with you."*

Attendees participated in Working Groups during the afternoons each day. Those sessions were—

- Senior Forum, chaired by Maj Kersey of USSTRATCOM

- Collection and Analysis, chaired by Kim Wood, JTF–CNO/J2 Senior Analyst

- Knowledge Management, chaired by Scott Atwood, JTF–CNO/J25 (Programs and Integration) Senior Analyst

The 3rd Semi-Annual Intel Support to CND Conference was hosted by USSTRATCOM at Offutt Air Force Base, Omaha, Nebraska, August 4–6, 2004. Proceedings from the conference will be available in the next issue of the *IAnewsletter.* ■

About the Author

Timothy Madden

Timothy Madden is the public affairs and protocol officer for the Joint Task Force–Computer Network Operations (now the Joint Task Force–Global Network Operations), U.S. Strategic Command. He holds B.A. degrees in journalism and literature and may be reached at maddent@jtfcno.ia.mil.

product order form

Instructions: All IATAC **LIMITED DISTRIBUTION** reports are distributed through DTIC. If you are not a registered DTIC user, you must do so **prior** to ordering any IATAC products (unless you are DoD or Government personnel). **To register On-line: http://www.dtic.mil/dtic/registration.**
The *IAnewsletter* is **UNLIMITED DISTRIBUTION** and may be requested directly from IATAC.

Name ____________________ DTIC User Code ____________________

Organization ____________________ Ofc. Symbol ____________________

Address ____________________ Phone ____________________

____________________ E-mail ____________________

____________________ Fax ____________________

Please check one: ❑ USA ❑ USMC ❑ USN ❑ USAF ❑ DoD
❑ Industry ❑ Academia ❑ Gov't ❑ Other

Please list the Government Program(s)/Project(s) that the product(s) will be used to support: ____________________

____

LIMITED DISTRIBUTION

IA Tools Reports (softcopy only)

❑ Firewalls ❑ Intrusion Detection ❑ Vulnerability Analysis

Critical Review and Technology Assessment (CR/TA) Reports

❑ Biometrics (soft copy only)
❑ Defense in Depth (soft copy only)
❑ IA Metrics (soft copy only)
❑ Wireless Wide Area Network (WWAN) Security
❑ Computer Forensics* (soft copy only)
❑ Data Mining
❑ Network Centric Warfare
❑ Configuration Management
❑ Exploring Biotechnology

State-of-the-Art Reports (SOARs)

❑ Data Embedding for IA (soft copy only)
❑ Modeling & Simulation for IA
❑ IO/IA Visualization Technologies
❑ Malicious Code

* You MUST supply your DTIC user code before these reports will be shipped to you.

UNLIMITED DISTRIBUTION

Hardcopy *IAnewsletters* available

Volumes 4		❑ No. 2	❑ No. 3	❑ No. 4
Volumes 5	❑ No. 1	❑ No. 2	❑ No. 3	❑ No. 4
Volumes 6	❑ No. 1	❑ No. 2	❑ No. 3	❑ No. 4
Volumes 7	❑ No. 1	❑ No. 2		

Softcopy *IAnewsletters* back issues are available for download at http://iac.dtic.mil/iatac/IA_newsletter.html

Fax completed form to IATAC at 703/289-5467

September

High Technology Crime Investigation Association International Conference and Expo 2004

September 13–15, 2004
Washington, D.C.
http://www.htcia2004.com

2004 Defense WW Combating Terrorism Conference

September 13–16, 2004
Hilton Alexandria Mark Center, Alexandria, Virginia
703/697-1854

Do It Yourself Vulnerability Assessment Program (DITYVAP) Certification Training

September 21–23, 2004
Cobb Hall, Fort Gordon, Georgia
http://www.rcert-s.army.mil

Electronic Warfare 2004

September 28–29, 2004
London, United Kingdom
http://www.idga.org/cgi-bin/templates/document.html?topic=228&event=5134&document=42304

DoD Architectures—Framework, Development & Implementation

September 28–29, 2004
http://www.idga.org/cgi-bin/templates/singlecell.html?topic=233&event=5158

October

Strategic Space 2004

October 5–7, 2004
Omaha, Nebraska
http://www.stratspace.org/information/imdex.cfm

TechNet Europe 2004

Rome, Italy
October 14–15, 2004
http://www.technet-europe.com/

InfoTech 2004

October 18, 2004
Dayton Convention Center, Dayton, Ohio
http://www.afcea-infotech.org/pages/over.html

Federal Information Assurance Conference (FIAC) 2004

October 26–27, 2004
Maryland University College Inn and Conference Center, Adelphi, Maryland
www.fbcinc.com/fiac

2004 Biometrics Summit

October 26–28, 2004
New York City, New York
http://www.aliconferences.com/conferences/biometricssummit/1004.html

November

MILCOM 2004 (Military Communications Conference)

October 31–November 3, 2004
Monterey Conference Center, Monterey, California
http://www.milcom.org/index.htm

Computer Security Institute's 31st Annual Conference and Exhibition

November 8–10, 2004
Marriott Wardman Park, Washington, D.C.
https://www.cmpevents.com/csi31/a.asp?optioni=N

TechNet Asia Pacific–C4 and Coalition Warfare

November 8–11, 2004
Honolulu, Hawaii
http://www.afcea.org/asiapacific/info.asp

Government CIO Summit

November 14–16, 2004
Boca Raton, Florida
http://www.fcw.com/events/cio/

Homeland Security (co-located with Information Assurance) Conference and Exhibition

November 30–December 2, 2004
Ronald Reagan Building and International Trade Center, Washington, D.C.
http://www.e-gov.com/

IATAC

Information Assurance Technology Analysis Center
3190 Fairview Park Drive
Falls Church, VA 22042

To change, add, or delete your mailing or E-mail address (soft-copy receipt), please contact us at the address above or call us at: 703/289-5454, fax us at: 703/289-5467, or send us a message at: iatac@dtic.mil