# A Distributed Multi-Agent Architecture for Computer Security Situational Awareness

**Dean Engelhardt**
Information Networks Division
Defence Science and Technology Organisation
AUSTRALIA.
dean.engelhardt@dsto.defence.gov.au

**Mark Anderson**
Information Networks Division
Defence Science and Technology Organisation
AUSTRALIA.
mark.anderson@dsto.defence.gov.au

**Abstract –** *Distributed systems for computer security analysis must perform information fusion in order to construct a cyberspace situational awareness picture. To date such fusion has been conducted in the context of a single abstraction set. As the complexity and heterogony increase, this approach becomes unwieldy. In a conceptual sense it is unscaleable. In this paper we describe an alternative approach, an architecture which supports concurrent reasoning in multiple sets of abstractions in a structured way. We present the architecture and a reasoning system for cyberspace situational awareness constructed using our approach.*

**Keywords:** Agent architectures, distributed analysis, multi-abstraction architectures, computer security.

## 1 Introduction

The task of securing computer systems and networks from unauthorized use and other forms of attack is a problem that, despite many years of research, still remains elusive. In recent years a great deal of attention has been placed upon protecting networks by locating 'intrusion sensors' at a multitude of key locations throughout the network and tasking each to perform a local analysis of its inputs (usually either network traffic or host activity). While simple intrusions such as stateless attacks may be detectable by such local analysis, the broader class of more complex intrusions is transparent to such analysis. Detection of such attacks requires some consideration of the sense inputs and/or local results from several of the sensor sites. A core challenge facing distributed security monitoring systems is how to conduct this information fusion in an efficient, sound and scalable manner so as to produce a timely common "cyberspace situational awareness" [5].

A number of commercial [7,16] and research [4] systems have appeared over the past several that attempt to provide such fusion-based intrusion detection by a variety of different approaches. The principal focus of work to date in the field has concentrated on the development of systems which are topologically scalable, that is which arrange their fusion elements in a structure which does not become congested as the number of sensors increases. A variety of effective models have appeared, ranging from purely hierarchical architectures [12,15] to fully decentralised peer-to-peer information networks [11], with a range of hybrids [16] proposed in between.

While a great deal of attention has been given to the topological structure of the networks through which fusers exchange information, little if any emphasis has been placed on the forms that these information exchanges take. In particular there has been little investigation of what set of abstractions are appropriate to the representation of both sense data from 'intrusion sensors' and intermediate fusion results. Most extant systems either define their own ad hoc representations or rely on a massive flat global taxonomy of attacks (e.g., as referenced in a proposed protocol by the Internet Engineering Task Force [8]).

We believe that for a distributed security monitoring system to provide effective situational awareness for a large and potentially heterogenous computing environment its information abstractions must be equally as scalable as its communications structures. In such a system, sensor data may be highly varied and the intermediate fusion occurring at intermediate nodes similarly diverse. Construction of a single information exchange format (or ontology) which can encapsulate both the very low level semantics of sense input and the higher level semantics of partially or fully fused results is problematic. Such an ontology would be counter to the accepted principals of ontology design, for example the clarity criteria defined in [9]. Furthermore conducting effective reasoning, including information fusion, in terms of such a diverse set of interrelated abstractions would be complex and error prone. To allow for effective fusion in such a heterogenous environment we contend that a fusion architecture must allow for multiple ontologies to coexist within the architecture in a structured fashion. This is in keeping with a similar finding reported for the field of Knowledge Integration [17].

As part of the Shapes Vector [1,2] research project, undertaken at Australia's Defence Science and Technology Organisation, novel paradigms have been developed for a variety of aspects of distributed security analysis systems. These range from consideration of new forms of deductive inferencing, to logics for reasoning about facts represented as structural possibilities [3]. In this paper we present an architecture we have designed and constructed which specifically addresses the issue of large-scale fusion systems and the requirements for multi-abstractional environments. This system, called the Shapes Vector Knowledge Architecture (SVKA), forms a core part of Shapes Vector, a distributed component-based system for security analysis and cyber situational awareness. This paper focuses on the mechanics of this architecture, describing the components and management policies it embodies.

In order to allow for large-scale, multi-abstractional fusion environments to be constructed, SVKA provides for its knowledge processing system – made up of an arbitrary number of software intelligent agents — to be conceptually partitioned into "locales". Each locale undertakes processing in terms of an ontology appropriate to the information processing being performed and can be thought of as providing a semantic (and temporal) context for that processing. Locales can be linked in a general directed acyclic graph to provide for flexible and scalable progression of knowledge from less abstract ontologies to more abstract ones.

The remainder of this paper is arranged as follows. In Section 2 we present the SVKA architecture, describing the framework it uses to construct and manage a distributed 'gestalt' of software intelligent agents. The system's mechanisms for knowledge exchange between agents are described, as are the management structures enforced across the gestalt to minimise knowledge processing instabilities. In Section 3 we describe a particular application of the SVKA to generating a near-realtime situational awareness picture for a network. We conclude with a few comments on the applicability of the architecture in Section 4.

## 2 Shapes Vector and the SVKA

Shapes Vector is a component-based system for the realtime security analysis of large-scale computer systems and networks. The system seeks to combine the deductive strengths of knowledge fusion systems with the inductive strengths of the human mind and to apply this combined expertise to detecting 'harmful' activity on protected systems and networks. The system features a comprehensive three-dimensional realtime visualisation environment in which a (potentially very large) network can be displayed and navigated. The details of this "cyberspace situational awareness picture," including its core semantic associations, are highly customisable and several different "views" are offered.

In order to produce realtime depictions of the network's composition and state, Shapes Vector components monitor network and host activity through an architecture of mobile software sensors. Observations made by these sensors is forwarded to one or more analysis sites, each of which is responsible for applying a variety of expert analyses to the individual sensor streams before conducting further analysis across multiple streams. The set of software components that collaborate to achieve this distributed fusion-based analysis is collectively termed the Shapes Vector Knowledge Architecture (or SVKA).

The task of the SVKA system is a complex one. It is called upon to provide a semantic "bridge" from very low-level concepts (facts observed about the system or network) to high-level ones (events which bear some security-significance of interest to the user). We divide the broad semantic gap between sense input and user output into a range of intermediate stages, defining for each stage a suitable set of abstractions. This abstraction set is codified into an ontology that forms the universe of discourse for knowledge processing and exchange at that 'locale' of our abstraction stack. In practice situations often arise where parallel analyses are desirable at similar levels of abstraction but within the context of a different abstraction set. The SVKA caters to such requirements by allowing for the concept of a stack of abstraction locales to be generalised into a directed acyclic graph of abstraction locales (where direction is defined as towards higher levels of abstraction).

As knowledge processing proceeds through successive locales of the SVKA, the individual processing elements deductively conclude information about the network under consideration. These discovered items of knowledge may be specific security-relevant details (e.g., alerts) but may equally well be configuration details or connectivity information which may be relevant to later analyses. A subset of the discoveries made by SVKA elements are of relevance to the Shapes Vector visualization systems: these are forwarded on as time-stamped semantic 'events' which are managed by a realtime event management system. A further (possibly disjoint) set of discoveries will be of long-term interest to users and future processing stages: these facts are stored in a persistent knowledge based as time-stamped ontological statements.

### 2.1 Architectural Overview

Shapes Vector's knowledge processing architecture is extremely flexible and highly modular. The system can manage an arbitrary number of intelligent agents which may be physically distributed across multiple hosts and architectures. The distributed 'gestalt' may be divided into an arbitrary number of 'locales' that can be chained together in a directed acyclic graph. Within a locale the agents exist in a structured layering scheme which defines a policy of intercommunication: agents resident in a layer may only receive knowledge from the next lower layer and pass deductive outputs on to those in the next higher layer. This structuring policy defines clear organizational roles for agents as well as avoiding global instabilities introduced due to cyclic passage of knowledge through the gestalt. The enforced structure also serves to encourage semantic abduction and avoid agents within an SVKA gestalt becoming context sensitive.

Figure 1 shows a simple SVKA ontology with three locales. Observations about a computer system or network are fed to the lowest level of intelligent agents in Locale A. These agents perform simple expert analyses in terms of a 'perimeter security' universe of discourse (ontology). Agents in the SVKA may adopt either a data-driven knowledge processing model (e.g., forward chaining) or a goal-driven model (e.g., backward chaining). The data driven agents in the initial layer, D1 and D2, supply the results of their analyses, expressed in the perimeter security ontology, to the next level of agents through an 'assertion router' (AR in the figure). This component matches ontology outputs from agents below it against complex expressions describing 'items of interest' which agents above it have previously supplied. Data-driven agents receiving knowledge from an assertion router proceed to analyse that input, which may have been received from multiple sources. These components, therefore, can apply a level of information fusion to that partially-processing sense input. In the case of the depicted gestalt, the data driven agent D3 receives the output of two distinct analytical processes – the two agents below it – to which it could apply fusion techniques.

Data-driven processing proceeds through the subsequent assertion router and agents within Locale A. The output of the top-most data-driven agent, D4, is made available for further processing in Locale C. Such inter-locale knowledge exchanges are made via a 'bridge,' a component which maps between a pair of knowledge processing contexts. This can incorporate translating knowledge from one universe of discourse (ontology) to another. It can also involve buffering knowledge to cater to differences in the temporal contexts of two locales. In the depicted gestalt, the bridge AC accepts ontological statements in the perimeter security ontology and emits

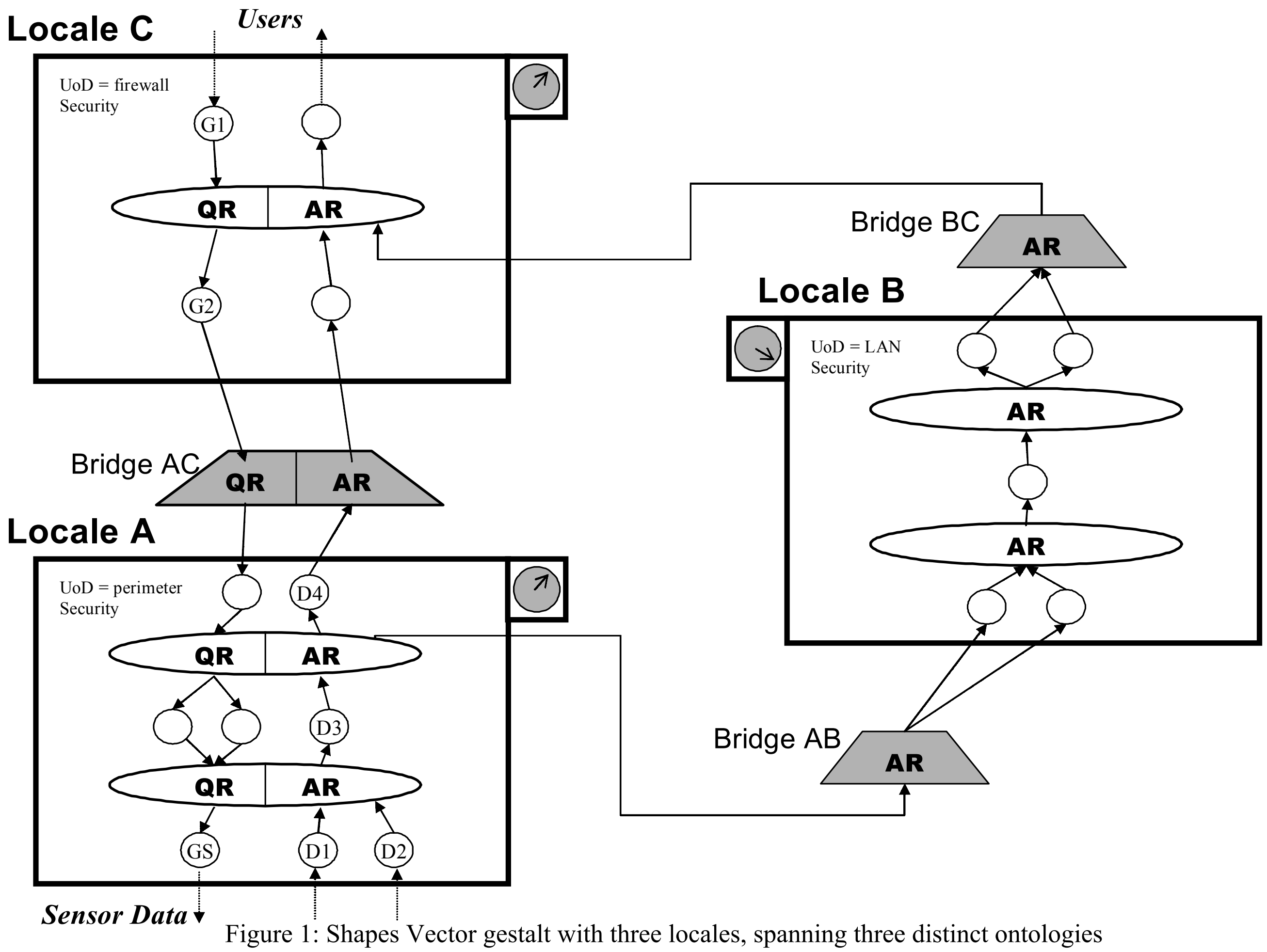

Figure 1: Shapes Vector gestalt with three locales, spanning three distinct ontologies

(to Locale C) translations of those statements in terms of the firewall security ontology. This bridge provides input to the data-driven agents in the lowest level of Locale C via an embedded assertion router. These agents and those in the second layer of Locale C perform analyses in terms of the firewall security ontology, ultimately reporting results to the user.

In parallel with the progression of knowledge from Locale A to Locale C, the depicted gestalt also shows knowledge passing to a locale for LAN security analysis, Locale B. This locale receives input from a bridge which translates the output of the top-most assertion router in Locale A. The results of Locale B's processing are fed back into Locale C via the intermediate assertion router in that locale. Thus the data-driven agents in the top-most layer of Locale C are fusing the outputs of processing which has occurred in terms of two parallel sets of abstraction.

In addition to the network of data-driven agents, the gestalt in Figure 1 also includes goal-driven agents in Locales C and A. Goal-driven processing is initiated by a direct interrogation from the user. In the sample gestalt this would be manifested in terms of an expression-based query in terms of the firewall security ontology. The query is initially analysed by goal-driven agent G1. If that agent can satisfy the goal directly by knowledge it has of the network or computer system it may provide an answer directly. In the case that G1 cannot immediately answer the user's request, the request is decomposed into a set of queries which, if answered, would allow that agent to answer. These derived sub-goals are supplied to the next layer of goal-driven agents (i.e., the one below G1). The task of distributing sub-goals is managed by a component called a 'query router' (QR in the figure).

The downward propagation of sub-goals continues (if required) between Locales by virtue of an embedded query router in Bridge AC. As with the bridge embedded assertion routers, the query routers inside these components may be called upon to map between two universes of discourse (ontologies). Ultimately, if no intermediate goal-driven agents can supply solution to the full set of sub-goals at one of the processing phases, the propagation will reach the bottom of Locale A. The bottom-most goal-driven agent, GS, will then directly interrogate sensors to discover information relevant to the set of sub-goals. As soon as knowledge becomes available (e.g., an answer being unearthed to a sub-query), that knowledge is propagated upwards by the same paths and routers.

## 2.2 Intelligent Agents and Ontologies

Intelligent Agents form the basic computational units which perform the knowledge processing and information fusion within the SVKA. As mentioned previously, the architecture caters to both data-driven and goal-driven execution models. Agents may be present in the architecture in both goal-driven and data-driven versions simultaneously, and multiple instances of any agent are allowed. The set of intelligent agents operating within the system is dynamic: agents can be instantiated while the SVKA is operational.

Each agent conducts its knowledge manipulation in terms of whatever internal abstractions are appropriate to its analysis task. The agent, however, is constrained in that it may only send and receive messages representing facts described in terms of the formally-defined ontology of the locale which contains it. An agent may only reside in a particular locale if it knows how to interpret facts represented in the locale's ontology. Its interaction with routers is always in terms of abstractions drawn from the ontology. For example, data-driven agents interact with lower-level assertion routers by specifying, in terms of abstractions from the ontology, which types of ontology facts they are interested in receiving.

Beyond the requirement that an agent's input and output must be expressed in terms of an ontology, the SVKA places no limitations on the nature of the agents. This freedom allows for the possibility of placing third-party agents or analytic systems into the SVKA gestalt by means of an appropriate 'wrapper.' This wrapper acts to translate between the knowledge (and/or data) formats handled by the third-party system and a recognised ontology. In our current implementation of the SVKA wrappers have been constructed for several third-party security analysis tools, including Snort [13].

Agents within the SVKA are autonomous entities but they may also be externally controlled, either by a user or another agent. Thus it is possible for an agent to dynamically change its priority, the types of inputs it considers, the time aperture it works across (see Section 2.6) and other details of its processing. Since agents are self-contained data processors (with encapsulated state) it is also possible to migrate them between computers. This allows the system, for example, to perform load balancing or to perform more 'forward processing' of sensor input.

## 2.3 Query Routers and Assertion Routers

As shown in Figure 1, agents within the SVKA are grouped into layers by the presence of query and assertion routers. The basic operation of an assertion router is to receive facts issued by the data-driven agents in the layer directly below. These facts are matched against the set of criteria that define the 'areas of interest' of the data-driven agents in the layer above. An agent's interests are expressed as an arbitrary set of expressions using abstractions drawn from the ontology of the agent's locale (i.e., the ontology in which its input will be represented). Whenever a new fact matches one of an agent's defined interest areas that fact is forwarded to the agent. The assertion router effectively implements a publish-subscribe notification as advocated in [6].

Query routers perform a simpler management function. Queries received from a higher-level goal-driven agent are broadcast to all goal-driven agents in the layer below. When one answer has been received, the router forwards it on to the initial higher-level initiator of the query.

## 2.4 Locales

A locale is a purely abstract entity that groups together multiple layers of agents with a common universe of discourse (ontology). There is no computational overhead in running multiple locales beyond any additional work required in translating between abstraction systems (performed in the bridges). There is no practical limit on the number of locales that may be present in the SVKA or their ontological bases.

Each locale may be divided into as many 'layers' as required (by providing it with the appropriate number of routers). The layer structure within a locale is driven primarily by the sets of analyses across which fusion we wish to perform fusion. If we wish for a fuser to correlate or otherwise fuse the results of several agents residing at a particular level of the locale, that fuser must live at the next higher level of the same locale.

## 2.5 Managing the Gestalt

As depicted in the example gestalt (Figure 1) the SVKA structures both the sets of agents that may directly communicate and the ways in which locales can be chained together. In the case of the former, the system mandates that an agent may only ever exchange knowledge directly with agents directly above and below in terms of the locale layer structure. In practice this means that the agent communicates with exactly one router (assertion router for data-driven agents, query router for goal-driven agents) above it and one below it. Intra-level communication is strictly prohibited. With respect to locale interconnections, the architecture requires that the resulting locale structure contain no cycles.

These restrictions on the patterns of knowledge exchange derive from a basic requirement to eliminate the possibility of knowledge 'cycles' (agents receiving output they have themselves generated previously). The rationale behind the elimination of knowledge cycles is that such cycles have the potential to introduce ambiguities and instabilities into the knowledge system. Statements and sub-goals generated by an agent are contextualized by the 'world-view' of that agent (as encoded in its expert knowledge of the real world). If an agent performs subsequent processing of that same knowledge this bias is amplified. After multiple cycles around the knowledge loop the statements or sub-goals being generated may have become sufficiently context-sensitive as to be meaningless (or even worse ambiguous) to another agent.

In our implementation of the SVKA the management policy of knowledge cycle avoidance is enforced by a gestalt manager. This component is responsible for notifying agents upon their initialization which router or bridge they should request input from and which they should deliver output to. Each time locales are added or modified this manager performs a cycle check before approving the change. The gestalt manager is also responsible for explicitly assigning each agent into an appropriate locale, based on the abstractions it reasons in terms of. The manager also assigns the agent into a 'layer' of the locale based on the sets of analyses it will be required to perform fusion across.

## 2.6 Temporal Contexts

One of the fundamental difficulties with performing any complex knowledge-based analysis of real-world systems is that 'facts' are only true in the context of the time they were observed by a sensor. This contextualisation has the potential to greatly complicate the internal knowledge processing undertaken by an intelligent. In order to minimise these effects, we provide each agent with an explicit 'time aperture' over which it is currently processing. The time aperture mechanism allows SVKA agents to avoid or minimise problems such as contradictions in facts over time, as well as providing a finite data set for the agent to process across

In practice making unique clock settings for every agent in the gestalt is often unnecessary. Typically, entire regions of the gestalt will be configured to analyse data from a particular time period – all agents collaborating in such an analysis should have synchronised clocks. To facilitate this type of coordination of time the SVKA provides each locale with a temporal context in the form of a synthetic 'gestalt clock.'

## 3 Experience with the Architecture

The SVKA architecture as presented in the previous section has been implemented and used to construct several multi-abstractional distributed computer security applications. In this section we describe a simple SVKA gestalt for multi-protocol mapping and security analysis. We further comment on the viability of constructing such a gestalt under our architecture as compared to a traditional monolithic abstraction-set approach.

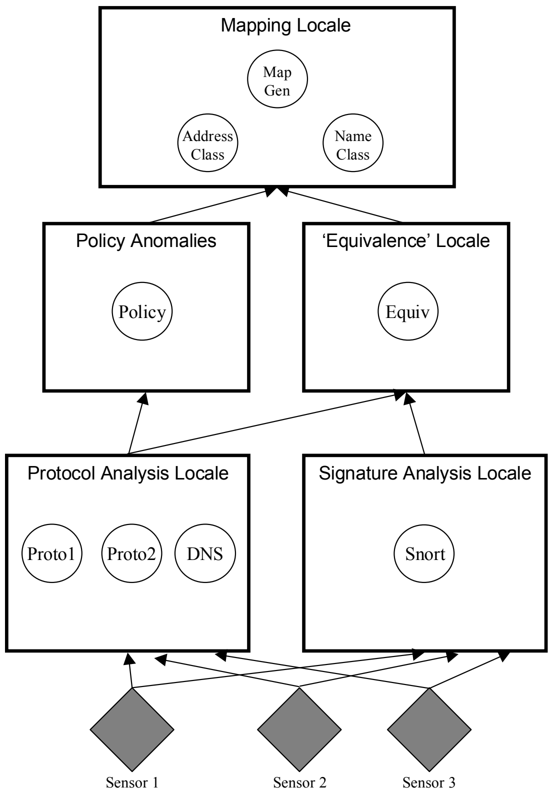

Figure 2: Gestalt for Cyberspace Situational Awareness

## 3.1 A Gestalt for Situational Awareness

Figure 2 shows a SVKA gestalt for multi-protocol mapping with security event overlay. The configuration uses nine intelligent agents / fusers partitioned into five locales to construct the realtime situational awareness picture. Sensor input from three network-based sensors is fed into two locales, one for protocol analysis and another for signature analysis. The latter locale is populated by a SVKA agent which wrappers the third party signature-based intrusion detection system Snort [13]. This agent applies signature-based analysis to the sensor data, generating an ontological statement recording each alert discovered by Snort. Ontology statements also record the existence of certain computers. In addition to this ontology output, the agent emits an event to signal the visualisation system of Snort's report. Ultimately this causes the display to register an animated visual alert.

Sensor inputs from all three sensors are received by the protocol analysis locale and distributed to three agents. Two of these agents apply protocol reassembly and parsing techniques to extract details of computers participating in protocol traffic and the means by which they are connected. This information forms the basis for the realtime map generation function. The two protocol reassembly agents produce their depiction of partial network maps as a collection of statements in the protocol analysis ontology. A third agent in this locale scans the sensor input for DNS traffic, extracting name-address pairs from successful name resolutions. These findings are reported as equivalency relationships between pairs of computers (one named according to a domain name, another according to the IP address).

Knowledge discovered during the protocol analysis processing includes details of which protocol accesses occurred when. This output is routed (via assertion router) to a policy anomaly locale. This locale contains a single agent that assesses the profile of protocol traffic against valid organisational and/or security policies. This analysis includes detection of situations where users have misused valid privileges to perform activities that are against network use policy. Upon detecting such breaches, the agent issues events to the visualisation system (which ultimately trigger an animated visual alert) and also forwards an ontological description.

A fourth locale, termed the 'equivalence locale' in the figure, receives input from both the protocol analysis and signature analysis locales. The single agent within this locale has the job of deducing structural equivalencies within the network under consideration. There are many instances where individual protocol and signature analyses deduce the existence of a computer on the network. Such analysis engines generate an identity for that computer. Protocol engines have only information available within the protocol packets from which to derive this name. Thus for some protocols, computers can be named according to fully qualified domain names, for others only IP addresses are available. Without subsequent analysis to detect equivalencies between computers, our network visualisation will contain multiple entities representing the same real-world computer. The equivalence agent correlates between each of the protocol agents, the DNS agent (a rich source of name-address equivalencies) and the signature agent in order to deduce as many equivalencies as possible. As such equivalencies are discovered, the visualisation is notified (which causes the fusion of visual entities) and the relationship is also forwarded as an ontology statement.

The top-most locale in our cyberspace situational awareness gestalt is the 'mapping locale' that accepts inputs from a variety of sources to produce a rich map of the protected network. Important to this phase of the processing is the classification of the various computers and networking elements whose existence was deduced by earlier agents. Two classifiers act across the range of reported data, attempting to divide the resulting map into meaningful sub-regions. These classifiers forward ontological outputs of their classifications to a map fuser that performs the deductive work required for synthesizing the map itself. This map generator produces a range of events notifying the visualisation system of the map's constitution and configuration.

Figure 3 shows a screenshot of the 3D visualisation that results from applying a gestalt of the type described above to a sensor input from a test network. Computers are depicted as double-pyramid shapes connected by lines (depicting network connectivity). Animated objects overlaying the map represent activity of interest, including detected signatures and policy violations.

It should be noted that the depicted gestalt is a very simple application of the SVKA architecture. Specifically, the various ontologies being reasoned across, while each is distinct, do not vary drastically in the forms of knowledge they represent. The architecture is, however, capable of fusing across vastly different styles of knowledge and abstraction. For example an SVKA gestalt could easily span simultaneous analysis and cross-fusion of cyberspace security events, movements of people in the real world, and littoral modeling.

### 3.2 Discussion

In constructing the SVKA gestalt for cyberspace situational awareness the ability to partition the system into abstraction regions (locales) has offered numerous conceptual and practical simplifications. To illustrate these simplifications, consider the policy anomaly analysis phase. It is worthwhile noting that while this analysis is dependant on the results of the various protocol analysis agents below it, the policy agent does not need to understand any of the basic abstractions those agents used in their reasoning. Thus, in constructing the policy agent, no consideration need be given to interpreting statements about raw protocol packets, their headers, options and error states. This greatly simplifies the logic of the agent, making it easier to construct, debug and maintain. Limiting the domain the agent reasons across also improves its performance as the number of semantic combinations considered is reduced.

In contrast, a system constructed to perform similar analyses as the gestalt depicted in Figure 3 but using a single global abstraction set would contain processing elements of greater internal complexity. Such elements must cope with receiving knowledge represented in a very broad range of diverse abstractions, ranging from primitive to conceptual. Where a particular real-world situation could be represented in multiple forms in terms of these abstractions, every element would need to understand the set of semantic 'rules' which define the different representations of input states of interest. Our informal experimentation suggests that this latter factor alone has the potential to add as much as an order of magnitude to the logical complexity of the individual processing elements. Increased complexity makes it considerably more difficult to construct processing elements which are error-free and logically complete. Also the performance impacts of this complexity can be

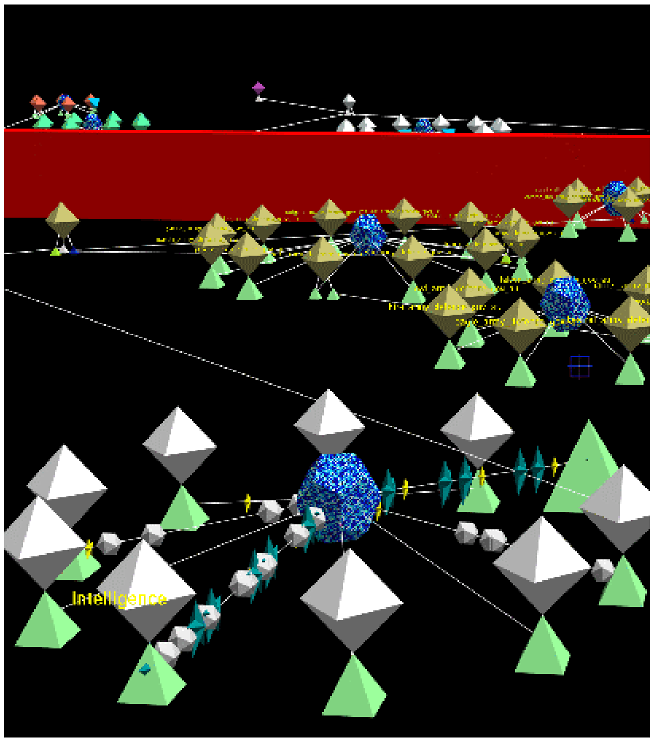

Figure 3: A 3D Cyberspace situational awareness display

severe, particularly given the fact that many knowledge processing paradigms scale worse than linearly with the complexity of the internal 'rule set.'

## 4 Conclusions

We have proposed a novel architecture for distributed security analysis that permits the fusion system to scale both in terms of the number of deployed sensors but also in terms of the abstractions the reasoning is conducted in terms of. The SVKA is an extremely flexible and highly modular environment. It permits an arbitrary number of fusion elements (in the form of software intelligent agents) to collaborate in the production of a common situational awareness picture for a network under protection. This gestalt of intelligent agents can be arbitrarily partitioned along the lines of the concepts and abstractions that are appropriate to different parts of the computation. These abstraction regions, called locales, define the semantic context for the computation both in terms of the universe of discourse (and associated ontology) for the analysis and also the temporal context. The gestalt as a whole is made up of an arbitrary directed acyclic graph of these regions, permitting a general flow of knowledge from less abstract to more abstract regions in a scalable fashion.

We have demonstrated a simple configuration of our architecture that delivers realtime cyberspace situational awareness for real-world computer networks. The system collects together a number of different areas of network sensor analysis – from protocol modelling and analysis to packet-based signature detection. Also incorporated are identity-fusion operators as well as a range of network classifier elements. The gestalt for this example is partitioned into several locales, each defining a different set of concepts under consideration. Experience with the construction of the sample gestalt has provided some validation of the multi-abstraction approach to distributed security analysis. The specification of the local analysis and fusion computations within each agent is clean, concise, and easily comprehensible. These properties would be harder (sometimes impossible) to achieve in the case that the agent were forced to deal with the full set of concepts that is required for the computation as a whole to complete.

Our approach to the management of heterogenous sets of concepts in the distributed knowledge processing computation is in stark contrast to that found in extant security analysis systems. Such systems typically define a flat global ontology to which all fusion elements must subscribe. While this may be sufficient for limited and conceptually simple distributed analysis of sensor inputs, experience with other AI disciplines [17] has shown that such systems cannot easily scale to provide information fusion in terms of a richer set of base concepts. For such fusion to be possible an architecture such as ours, which caters to multiple shared ontologies, would be a practical requirement.

## References

[1] Anderson, M., Engelhardt D., Marriott, D., North, C., Rajaratnam, D., *Shapes Vector: An Overview*, DSTO technical report DSTO-GD-0205, 1997.

[2] Anderson, M., Engelhardt, D., Marriott, D. and Randhawa, S., *Data Processing Architecture,* PCT patent application PCT/AU02/00529, 2002.

[3] Anderson, M. and Engelhardt, D., *Shapes Vector: An Overview of the Intelligent Agent Architecture*, in preparation

[4] Balasubramaniyan, J., Garcia-Fernandez, J., Isacoff, D., Spafford, E., and Zamboni, D., *An Architecture for Intrusion Detection using Autonomous Agents*, 14th Annual Computer Security Applications Conference (ACSAC98), December 1998.

[5] Bass, T., *Intrusion Detection Systems and Multisensor Data Fusion*, Communications of the ACM, 43(4), April 2000.

[6] Bass, T., *The Federation of Critical Infrastructure Information via Publish-Subscribe Enabled Multisensor Data Fusion,* Fifth International Conference on Information Fusion: Fusion 2002, July 2002.

[7] Cure, S.J., *ISS SiteProtector, Security Indepth,* White Paper from Red Bull Technologies, Inc. URL: http://www.itsecurityhelpdesk.com/ISS%20SiteProtector01.pdf

[8] Curry, D. and Debar, H., *Intrusion Detection Message Exchange Format*, Internet Draft, URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-10.txt

[9] Gruber, T., *Towards Principles for the Design of Ontologies Used for Knowledge Sharing,* International Workshop on Formal Ontology, March 1993.

[10] Helmer, G., Wong, J., Honavar, V., Miller, L. and Wang, Y., *Lightweight Agents for Intrusion Detection*, Journal of Systems and Software, to appear.

[11] Krügel, C. and Toth, T., *Distributed Pattern Detection for Intrusion Detection*, Network and Distributed System Security Symposium Conference, 2002.

[12] Porras, P. and Neumann, P., *EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances*, 1997 National Information Systems Security Conference, October 1997.

[13] Roesch, M., *Snort: Lightweight Intrusion Detection for Networks*, LISA99, November 1999.

[14] Symantec Corporation, *ManHunt: Reducing the Risk of Compromise*, White Paper available from http://enterprisesecurity.symantec.com

[15] Staniford-Chen, S., Cheung, S., Crawford, R., Dilger, M., Frank, J., Hoagland, J., Levitt, K, Wee, C., Yip, R. and Zerkle, D., *GrIDS – A Graph Based Intrusion Detection System for Large Networks,* 20th National Information Systems Security Conference, October 1996.

[16] Undercoffer, J., Perich, F. and Nicholas, C., *SHOMAR: An Open Architecture for Distributed Intrusion Detection Services*, Technical Report TR-CS-02-14, Dept of Computer Science and Electrical Engineering, University of Maryland Baltimore County, 2002.

[17] Visser, P. and Tamma, V., *An Experience with Ontology-based Agent Clustering*, IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5), August 1999.