# MODEL OF SOFTWARE AGENT FOR NETWORK SECURITY ANALYSIS

**Hristo Emilov Froloshki**
Department of telecommunications, Technical University of Sofia, 8 Kliment Ohridski st., 1000,
phone: +359 2 965 2134, e-mail: hef@tu-sofia.bg

**Keywords:** Software agents, IP network security, Intrusion detection systems, Distributed architecture, Telecommunication middleware

*Network operators have realized the need of protection against bandwidth theft and denial of service attacks, which come as downsides of the growing mass of subscribers. The article proposes a model that takes advantage of the freedom in implementation of computational objects, given by telecommunication middleware framework. The result is a viable approach in implementing intrusion detection system functionality at the level of resource management. The model creates communication channels within the protected network and uses them for transfer of information related to network security. The proposed agent is also able to scan streaming information and search for predefined sets of strings. The implementation is based on the client-server model of data communication. It was necessary to include both a client side and a server side within a single application with respect to the given task – namely to initiate or respond to a communication request.*

## 1. Introduction

It is expected that IP technology will become the one technology for transport of voice, data, and multimedia over next generation networks. Services will be provided through application programming interfaces by the use of a common platform that contains the service capabilities. More and more clients have gained experience in the usage of different networks and realize the importance of security and safety, which forms their expectation of new network architecture and service provisioning, based on reliable tools and improved security methods. The distributed telecommunications middleware is a technology that allows the integration of security components into services as well as definition of value-added security services. The main idea behind the telecommunications middleware is to eliminate the distinction between telecommunication services and computer applications.

## 2. Telecommunications middleware

Middleware is software that runs between a machine's operating system and the applications. |It offers facilities that are common to many applications, such as managing the communication between applications on different machines. TINA (telecommunication and information networking architecture) provides a telecommunication middleware framework. TINA architecture is divided into two parts:

- *Service architecture:* the part of TINA in which services are executed;
- *Resource management architecture:* the part that contains the switching and transport equipment and logic.

The resource management architecture encapsulates the technology of the individual network elements and offers a generic API to the service architecture. By doing this so, TINA separates service logic from connection control.

The impact of that separation is that services and network resources may follow their evolutionary paths at different speeds, with a side effect of stimulating each other's advance.

Table 1 gives an informal overview of the computational objects and their functions, defined in the service architecture.

**Table 1** Computational objects in TINA service architecture

| | | |
|---|---|---|
| IA | Initial agent | Represents the first point of contact for the consumer in a foreign network. Takes care of authentication, finding the user agent for the consumer, and starting the access session. |
| UA | User agent | Represents the consumer within the retailer domain. Contains subscription data and user profiles. Runs the access session. |
| PA | Provider agent | Represents the retailer within the consumer domain. Can be seen as a proxy through which the retailer makes service offers to the consumer. |
| PeerA | Peer agent | Establishes connections between two retailer domains, if the service implies users in different domains (for example, a video conference between users in different networks). |
| SF | Service factory | Creates service instances as the result of a service request, and configures the service. |
| SSM | Service session manager | Runs the actual service session. Keeps track of the parties involved in a service and their relations. Requests connectivity from the communication session. |
| ssUAP | Service session user application part | Represents the service control interface in the terminal. |

The computational objects defined in the resource architecture, are briefly described in Table 2.

**Table 2** Computational objects in TINA resource architecture

| | | |
|---|---|---|
| CSM | Communications session manager | Responsible for negotiating and selecting the terminal and network capabilities, needed for the requested connection. Maintains the state of the communication session. |
| CC | Connection coordinator | Sets up the physical connection. Determines which layer networks to involve in the connectivity request. |
| LNC | Layer network coordinator | Contains knowledge about the transport and switching technology used in the layer network, end points within its domain. LNC is network and technology dependent. |
| TCSM | Terminal communication session manager | Counterpart of the CSM. Negotiates the terminal capabilities and maintains the state of the communication session. |
| TLA | Terminal layer adapter | Assigns communication end points in the terminal and negotiates technology-specific settings with the LNC on the network side. |

The entire TINA architecture is object-oriented and assumes the existence of a distributed processing environment that takes care of the communication between distributed objects. TINA does not define any protocols. In TINA, signaling is replaced by communications between distributed objects. The TINA distributed processing environment is based on OMG's CORBA (Common Object Request Broker Architecture). The unit of deployment and reuse in TINA is called *computational object*, which is defined as a component that consists of one or more CORBA interfaces.

Nevertheless TINA provides a framework for service and resource architecture, it says nothing about the implementation of computational objects. A promising technology in the field of implementation is the one of software agents. They are appropriate for distributed systems, such as IDS (Intrusion Detection Systems). The need of security is obvious at both layers – at service architecture it is necessary to provide an access control; at the network resource layer the network operators need it to ensure that their resources are properly used. Provisioning of security and safety features is a prerequisite for successful services of the future.

## 3. Intrusion detection systems.

IDS technology is one of many software means used for the protection of electronic information. Its purpose is to track a predefined set of network or host parameters, and upon reaching a certain threshold to issue alerts to the system administrator or take countermeasures, which may limit the intrusion. There are two main approaches in intrusion detection:

*Host-Based IDS (HIDS)* **-** host-based systems were the first type of IDS to be developed and implemented. These systems collect and analyze data that originate on a computer that hosts a service, such as a Web server. Once this data is aggregated for a given computer, it can either be analyzed locally or sent to a separate/central analysis machine. In addition to detecting unauthorized insider activity, host-based systems are also effective at detecting unauthorized file modification.

*Network-Based IDS (NIDS)* - network-based intrusion detection analyzes data packets that travel over the actual network. These packets are examined and sometimes compared with empirical data to verify their nature: malicious or benign. Because they are responsible for monitoring a network, rather than a single host, Network-based intrusion detection systems (NIDS) tend to be more distributed than host-based IDS. In general, network-based systems are best at detecting the following activities:

- Unauthorized outsider access: When an unauthorized user logs in successfully, or attempts to log in, they are best tracked with host-based IDS. However, detecting the unauthorized user before their log on attempt is best accomplished with network-based IDS.

- Bandwidth theft/denial of service: These attacks from outside the network single out network resources for abuse or overload. The packets that initiate/carry these attacks can best be noticed with use of network-based IDS.

*HIDS and NIDS Used in Combination* - the two types of intrusion detection systems differ significantly from each other, but complement one another well. One implementation of IDS may include intelligent software agents (or computational objects if we use TINA terminology), which provide functions for monitoring and data collection (at resource management level). The use of these agents for solving problems in distributed systems is not uncommon. They are primarily used for extraction, filtering and analysis of information. Open Agent Systems are environments where the software agents are able to communicate in an unstructured fashion. Agents may also have goals and use available means for achieving these goals.

## 4. Computational objects modeled by IDS technology

Let us consider a possible model of an IDS agent. The agent may be used for implementing computational objects in distributed architecture of the telecommunication middleware. The agent incorporates security functions. The agent's goal should be detection of forbidden (dangerous) words and appropriate reaction. The knowledge of the agent is based on several list structures, containing the recognized alphabet, forbidden words, safe words, and cache with words which don't belong to either of previous lists. First we should point a stream for analysis – the delivery of the stream is beyond the scope of this article. The analysis is done on per word basis. The first check is for new symbols in the analyzed word – its purpose to extend the known alphabet. This check may also be used for allowing/rejecting words, based on their syntax. Next step is to match the word against the list of forbidden words – if positive an alerting message is displayed to the administrator and the word is no longer processed. If the check is negative, the word is passed to the next step – check against the list of safe words – the purpose is to ensure that the word is typical for the environment, where the agent runs. If this check is negative, i.e. the word is neither forbidden, nor safe it is considered unknown and is matched against a cache of such words – each with an associated counter. The counters indicate how many times the word has been met in the stream. The administrator (user) may predefine a threshold value for the counter. The word is matched against the cache – if not found – it is stored with initial counter value of one, otherwise a check of the counter follows – if the threshold isn't reached the counter is incremented by one and next word is extracted for analysis, otherwise – the administrator is asked to classify the current processed word as safe or forbidden. If the word is recognized as forbidden – it is added to the appropriate list and an update is issued to an adjacent agent. If the word is safe it is added to the list of safe words and next word is being processed. The algorithm is shown in Figure 1.

## 5. Conclusion

The paper examines the applicability of IDS technology for implementation of distributed telecommunications middleware. IDS technology may be used both at service architecture level and at resource management architecture level. At service architecture level software agents may be used to implement computational object for access control. At resource management level software agent may include function for network monitoring against intrusions. The paper present a model of IDS agent used to implement security functions at resource management level.

**References**

[1] Zuidweg J. Next Generation Intelligent Networks, ArtechHouse, 2003.
[2] Paul Innella and Oba McMillan, Tetrad Digital Integrity, LLC An Introduction to Intrusion Detection Systems, http://www.securityfocus.com/infocus/1520

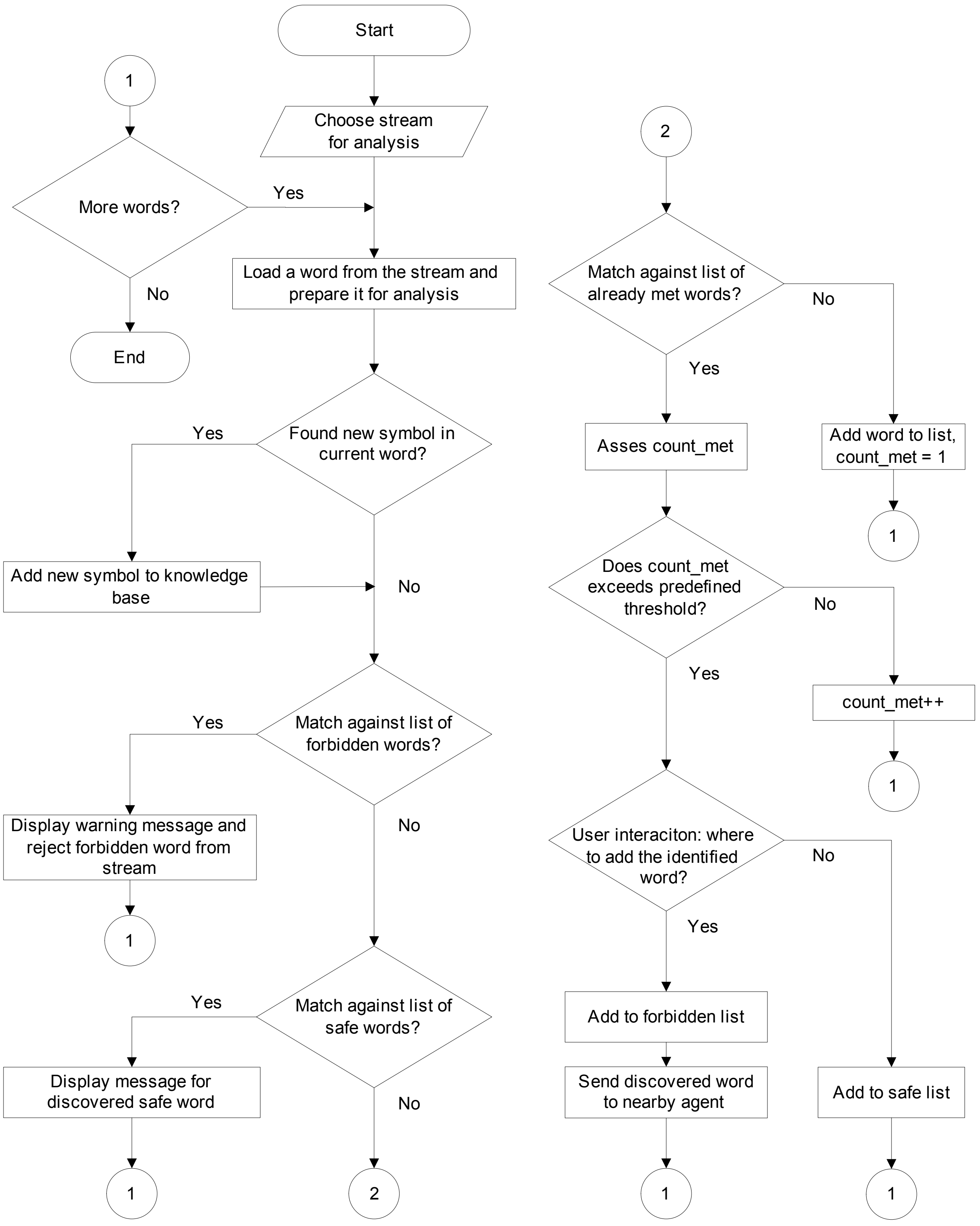
Start
Choose stream for analysis
1
More words?
Yes
No
End
Load a word from the stream and prepare it for analysis
Found new symbol in current word?
Yes
Add new symbol to knowledge base
No
Match against list of forbidden words?
Yes
Display warning message and reject forbidden word from stream
1
No
Match against list of safe words?
Yes
Display message for discovered safe word
1
No
2
2
Match against list of already met words?
No
Yes
Asses count_met
Add word to list, count_met = 1
1
Does count_met exceeds predefined threshold?
No
count_met++
1
Yes
User interaciton: where to add the identified word?
No
Yes
Add to forbidden list
Send discovered word to nearby agent
1
Add to safe list
1

Figure 1 Algorithm of the proposed model