A Security Supervision System for Hybrid Networks

Françoise Sailhan, Julien Bourgeois, Valérie Issarny

A Security Supervision System for Hybrid Networks

Francoise Sailhan[1], Julien Bourgeois[1], and Valérie Issarny[2]

[1] LIFC, University of Franche-Comté, Centre de Développement Multimédia, 1 cours Leprince-Ringuet 25201 Montbéliard, France
sailhan@ieee.org, julien.bourgeois@univ-fcomte.fr
[2] Arles project, INRIA-Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153, Le Chesnay Cédex, France
valerie.issarny@inria.fr

Summary. The traditional way of protecting networks and applications with e.g., firewalls and encryption, is no longer sufficient to protect effectively emerging hybrid wired-cum-wireless networks including *ad hoc* networks. Intrusion detection mechanisms should be coupled with preventive measures so as to identify unauthorised abuses. To this end, we propose a novel Hybrid Distributed Security Operation Center (HDSOC) which collects logs that are generated by any application/service, layer of the protocol stack or resource (e.g., router), providing a global view of the supervised system based on which complex and distributed intrusions can be detected. Our HDSOC further (i) distributes its capabilities and (ii) provides extensive coordination capabilities for guarantying that both the networks and the HDSOC components do not constitute isolated entities largely unaware of each others.

Keywords: Distributed intrusion detection, host-based and network intrusion detection, security and protection management, event notification.

1 Introduction

Until recently, most systems were operating over wired networks and were intended to be used as part of specific applications or in localised settings (e.g., building, campus or corporation). Spurred by the emergence of Wifi technologies and the advance of generalised eCommerce and pervasive applications (e.g., rescue and military application), interest has moved towards the provision of a global solution that interconnects in a secure manner changing sets of clients, services and networks. This construction of Internet-scale applications introduces new challenges consisting in securing large-scale wired-cum-wired networks, including Wifi-enabled *ad hoc* networks, which are spanning geographically dispersed sites and distinct administrative domains. The traditional way of protecting these networks and applications with e.g., firewalls and encryption is no longer sufficient due to the following reasons. First, *ad hoc* networks introduce security holes due to their vulnerability to a variety of factors e.g., open medium, cooperative algorithms. In addition, the best protection is always vulnerable to attacks due to unknown security bugs and improper configuration. It is therefore clear that

R. Lee (Ed.): Soft. Eng., Arti. Intel., Net. & Para./Distri. Comp., SCI 149, pp. 137–149, 2008.
springerlink.com

preventive measures should be coupled with intrusion detection mechanisms so as to identify unauthorised use and abuse. The Distributed Network Intrusion Detection Systems (DNIDSs) that have been proposed in the literature [1] [2], are extremely diverse in the mechanisms they employ to gather, analyse data and identify intrusion. However, DNIDSs share in common the fact that they glean intrusion data by monitoring the traffic and intercepting the network communications. More specifically, they mostly operate on the IP and transport layer headers and packets as well as the packet content, providing in depth packet analysis. Consequently, while DNIDSs are in a very convenient position wherein it has a complete access to all traffic traversing the managed network, their perspicacities suffer from:

- the cost (in term of processing usage) associated with the in depth analysis of the intercepted traffic. Note that one class of attacks commonly launched against DNIDS, lies in letting this DNIDS in a lethal state by spamming it with a large number of spurious traffic.
- the absence of information owned by the DNIDS on resources (hosts, services, protocols and applications) that constitute the network, which renders the DNIDS impotent to detect, correlate and report a wide range of (host, service, protocol and application-specific) intrusions.

This inefficiency of actual DNIDSs engaged us to propose a novel approach to intrusion detection, which were based on a Distributed Security Operation Center (DSOC)[6, 7]. Rather than relying exclusively on a resource-consuming and prone to attack traffic monitoring system, our DSOC collects logs that are generated by any application, service, DNIDS, layer of the protocol stack or resource (e.g., router) composing the managed system. As a result, our DSOC owns a global view of the supervised system - the state of any component being reported - based on which it can detect complex intrusions that are possibly originated by any component of the protocol/application stack and any hardware resource.

Based on this preliminary work, we propose a novel Hybrid DSOC (HDSOC) which is dedicated to provide intrusion detection in a large-scale hybrid network built upon wired-cum-wireless networks, which are geographically-dispersed and include Wifi-based *ad hoc* networks. Our HDSOC takes into account the specificities of *ad hoc* networks:

- Wireless hosts may operate under severe constraints e.g., limited bandwidth. This requires defining an HDSOC that reduces the overhead caused by its usage.
- The dynamics of hybrid networks including *ad hoc* networks, diminish the resilience of the HDSOC and necessitates to increase the decoupling of the HDSOC components so as to enable this latter to react and reconfigure in a timely way to network dynamics.

All these factors circumvent the need for supporting a global low-overhead HDSOC that is adapted to the network topology and characteristics (e.g., its dynamics, organisation) as well as the medium of communication which may be

e.g., unreliable and subject to unexpected disconnection. In order to increase the resilience of the HDSOC, we propose to (i) distribute its capabilities and (ii) provide extensive coordination capabilities for guarantying that both the networks and the HDSOC components do not constitute isolated entities largely unaware of each others. The proposed security operation center collects logs for detection and correlation without consuming significant network bandwidth while addressing missing, conflicting, bogus, and overlapping data. Further support for dynamic reassign correlation, and intrusion detection management responsibilities is provided to nodes as the topology evolves. The remainder of this paper is organised as follows. We introduce the proposed Distributed Security Operation Center (§ 2) before detailing each of its constitutive component (§ 3). Then, we conclude this paper with a summary of our results along with directions for future works (§ 4).

2 Design Rational

HDSOC aims to detect intrusion in a hybrid network composed of a collection of *ad hoc* networks which either (i) operate in physically isolated geographic sites (e.g., disaster-affected areas) or (ii) extend the coverage of e.g., public hot-spots, corporate buildings or large-scale urban areas. Each of these geographically dispersed *ad hoc* networks is further connected to a (wired) wide area network thanks to some gateway nodes (see Figure 1 for an overview of the network and the HDSOC architecture).

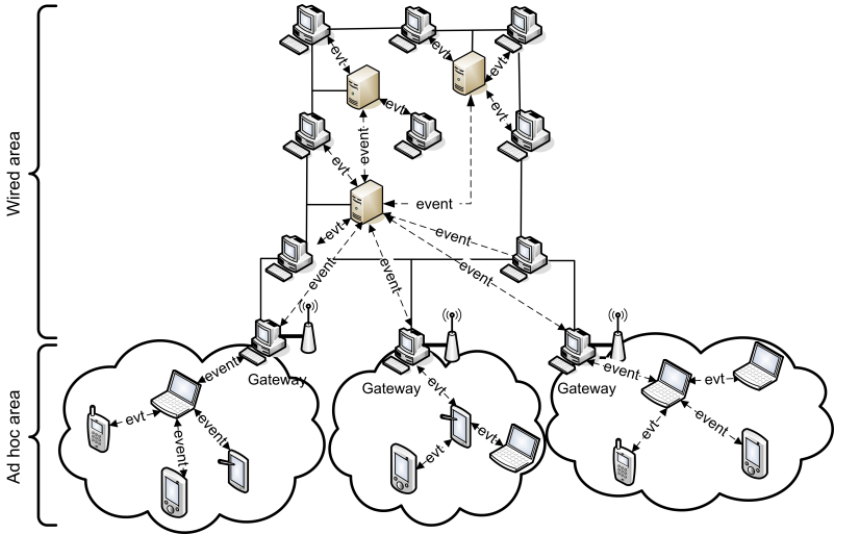

Fig. 1. HDSOC Architecture

The main challenge in collecting logs in *ad hoc* networks stems from the need to minimise the generated traffic and the computational load. This calls for:

1. parsing logs (i.e., extract only relevant data) rather than collecting raw logs that are characterised by a large size and prone to overload our system (as it is the case with attacks on the log size),
2. enabling resource-constrained devices that are incapable of parsing locally their logs, to delegate this parsing activity to a remote device which offers sufficient capabilities.
3. parsing logs as close as close as possible from the device that generates it so as to diminish the number of long distant communications.

In order to answer to these commitments, our approach lies in collecting and parsing logs locally whenever possible. For this purpose, a lightweight collector and parsing agent (hereafter referred as Embedded Collection Box or simply ECBox) is embedded on the devices that show sufficient memory and computing resources. Alternatively, for resource-constrained nodes, we rely on a service discovery protocol (§3.4), which discovers dynamically ECBoxes. Among potential candidates, the closest ECBox is selected so as to keep to a minimum long distance communication. The selected ECBox is further assigned the task of collecting and parsing logs on behalf of resource-constrained devices. After parsing and extracting relevant data from logs, local/remote ECBoxes generate event notifications that are further disseminated over the network, causing a slight increase of the network traffic due to the lightweight size of event notifications. The dissemination of events is performed by an event notification service which aims to ensure that each device is delivered the information (i.e., events) relating to the distributed intrusion to which that devices participates. This information gives to the device a global view of the intrusion (i.e., intrusion state, intrusion development and its level of implication), helping it in reporting in early stage any intrusion furtherance. In order to prevent the device from flooding the network whenever an intrusion is reported while providing to the devices a global view of the intrusion attempts to which it takes part, we rely on a publish/subscribe distributed event notification service whereby:

- consumers (e.g., devices taking part in the intrusion attempt, security administrator's computer) express their demands to producers (e.g., devices taking part in the intrusion) during a subscription process,
- event producers transfer to subscribers the description of any relevant event that has been triggered locally.

This event system faces the requirements (namely scalability, autonomy and timeliness) driven by HDSOC by disseminating in a distributed way events over a self-configured delivery structure organised as a cluster-based hierarchy (Figure 3). This cluster-based hierarchy provides convenient aggregation and correlation points while rending our HDSOC more adaptable to network failures and less vulnerable to attacks due to its distributed nature.

3 Distributed Operation Center for Hybrid Networks

We propose an hybrid distributed security operation center which attempts to detect intrusions within a large-scale network including geographically-dispersed wired or wireless networks (e.g., *ad hoc* networks). Such detection necessitates to collect logs (§3.1) so as to identify intrusion attempts (§3.2) and initiate proper reactions. Rather than collecting bandwidth-consuming raw logs, our approach lies in parsing logs so as to extract only security-relevant information and generate compact event notifications and alarms that are beside disseminated at low cost (§ 3.3). Note that such parsing is either performed locally, i.e., on the device that generated the logs if its capacities are sufficient) or alternatively remotely; such delegation of the parsing task being enabled by a pervasive service discovery protocol (§ 3.4).

3.1 Local Event Collection

Prior developing mechanisms for detecting intruders, it is crucial to understand the nature of attacks[1] as well as the possible security holes that characterise *ad hoc* and hybrid networks. In *ad hoc* networks, intruders take advantage of the lack of physical protection inherent to the absence of clear physical boundary (no protection being applied inside the network by any layer 3 resources, e.g., gateways), the collaborative nature of algorithms and the resource-limited capacity of the network (devices being more likely to be exhausted). This renders network components particularly vulnerable. These components include:

- *Networking components* (e.g., MAC, zeroconf and routing protocols) included in the physical/link/network layers are subject to eavesdropping, jamming, interceptions (physical layer), identity falsification (zeroconf protocol) and finally attacks initiated on routing tables (routing protocol) with e.g., the so-called wormhole and blackhole attacks.
- Security components, including DNIDS, cryptographic facilities, motivation functionalities which recompense collaborative nodes, and reputation and exclusion mechanisms whereby nodes vote and attribute a reputation to each node.
- Transport components, applications and services which are affected by session hijacking or flooding and application/service/scenario-specific attacks.

Each of these software components generates logs expressed in different formats including standard formats (e.g., syslog, MIB, HTML) or proprietary/application-specific formats. These logs can be easily collected relying on standard Xmit protocols (e.g., SNMP, SMTP, HTTP) as it is the case in traditional monitoring architectures. For instance, OLSR [5, 4] logs can be collected using MIB format that is defined and used in [8, 9]. The pervasiveness of these protocols ensures a significant level of interoperability to our HDSOC despite the heterogeneity of hardware

[1] Interest reader can refer to [12] for a comprehensive survey on attacks and countermeasures in *ad hoc* networks.

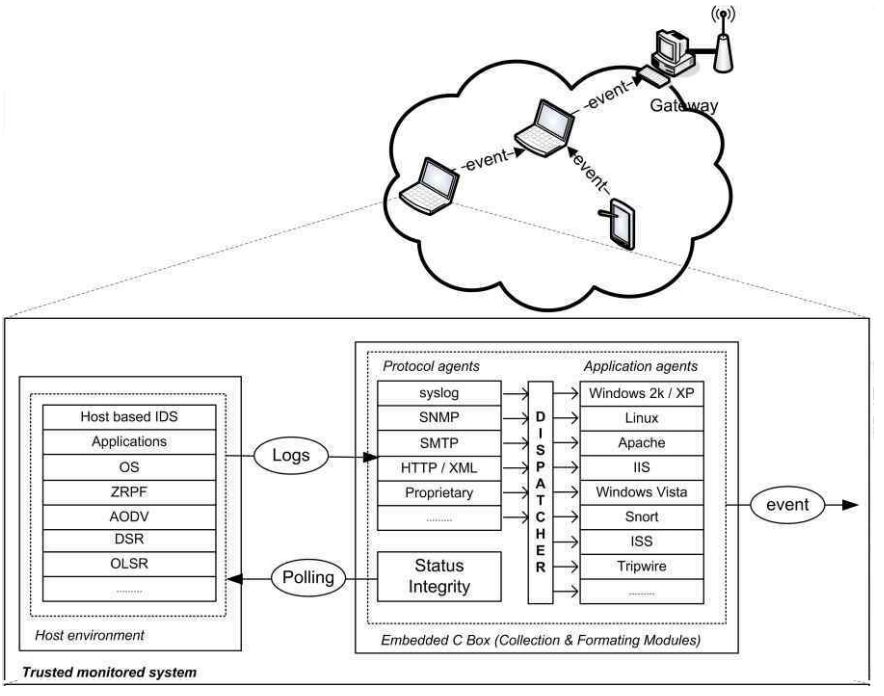

Fig. 2. Device Architecture

and software platforms. We therefore consider a HDSOC in which each device generates logs whose collection is enabled thanks to a protocol agent. In practice, this protocol agent corresponds to a collection of clients which implement standard Xmit protocols (Figure 2). Collecting logs from heterogeneous sources implies setting up a dispatcher and an application agent. The *dispatcher* determines the source-type of an incoming event and then forward it to the appropriate application agent which formats it in a common format (i.e., a format understandable by any HDSOC module). In practice, this dispatcher performs the following tasks:

- listening for incoming message transmitted by a protocol agent through an particular channel (e.g., as socket, named pipe, system V message queue),
- identifying the message source and the Xmit protocol used. More precisely, a patterns database is pre-loaded in the device memory (for performance considerations) and is used to find patterns matching the message.
- redirecting the original message to the application agent responsible for managing the messages generated by that type of source and Xmit protocol.

This application agent parses the message and expresses it in a common format that is transmitted to the event notification agent.

3.2 Distributed Intrusion Detection

After translating events into a common format that is understandable by any DSOC component, events are analysed and correlated so as to avoid transmitting all the events across the network. The main objective of correlation lies in producing a succinct overview of security-related activities. This necessitates to (i) filter events for the purpose of extracting only relevant events, and (ii) aggregate events so as to generate a compact representation of those events that eases the intrusion detection. Broadly sketched, event filtering aims to eliminate events that are not relevant, i.e.,

- duplicate events that do not provide additional information while consuming significant bandwidth.
- events that match policy criteria e.g., administrator login, identification, authorisation or restriction processes.
- events that are not critical to the supervised system, excluding events that relate to some vulnerabilities whose system is not exposed to. For this purpose, the device stores and maintains (structural, functional and topology-related) information about security breaches and insecure behaviour that either impact the overall security level or that can be exploited by an attacker.

Relevant events (i.e., events that went thought the filtering pipe) are further aggregated so as to provide a more concise view of what is happening in the system. This actual system view called *context* is stored locally with the previously generated contexts, before being transmitted by the event notification service. Based on the collection of contexts owned by the device, intrusion detection may take place. Intrusion detection consists in analysing a sequence of events so as to identify event sequence patterns characterising intrusion attempt. Note that during this process, time considerations are taken into account so as to take into account slow intrusions. In practice, such intrusion detection consists in matching a sequence of events (a context) against a set of attack signatures whose structure is described bellow.

Attack signature

A conquering attack can be broken down into a collection of successive steps that are successfully completed. This renders an attack characterisable as an attack signature, which corresponds to a labelled tree rooted by a node representing the goal, and intermediate nodes representing an attack step (i.e., an observable event) with a succession of children defining a way of achieving it. An attack scenario (i.e., the overall set of attacks that can threat the supervised system) is then represented as a forest of trees, with some part of trees being shared when a subset of steps involved in two distinct attacks is similar. Such attack scenarios are defined by the security administrator based on vulnerabilities specific to the network and the past attacks. Such tree-based representation of attack signatures renders intrusion detection easy to carry out. Indeed, an attack attempt is easily identifiable by matching attack signatures against a context, i.e., against a

succession of (possibly distributed) events occurring on a specific set of systems (e.g. devices, collection of devices, network segments). From a practical point of view, an attack identification therefore consists in matching an attack signature on the instance of a particular context. Central to intrusion identification is therefore the context accuracy. This accuracy is maintained by the event notification service, which updates the context of each device (its system view) with the most up-to-date events arising in the network.

3.3 Distributed Event Notification

Our event notification service aims to deliver events to devices so as to enable them to update their context, giving that device a global view of the intrusion attempts and hence rending the detection of intrusions more accurate. In order to prevent the device from blindly flooding the network whenever an intrusion is reported, our event model derives from the asynchronous publish/subscribe paradigm. From a communication perspective, our distributed event notification consists in exchanging notifications and control messages (i.e., subscriptions and un-subscriptions) between producers and subscribers through a collection of intermediate event agents (Figure 1). Note that a potential *event agent* (hereafter simply called agent) designates a device which holds our notification service. In practice, this collection of intermediate agents is organised into a cluster-based structure wherein each agent corresponds is a cluster leader and maintains information and connectivity with its cluster members and its clusterhead[2]. This underlying structure is then used for delivering control messages (subscriptions and un-subscriptions) to producers, as well as notifications to consumers. When delivering a notification, the main objective pursued by agents lies in forwarding that notification to an agent only if, toward this direction, there exists a consumer interested in receiving it. For the purpose of forwarding selectively notifications, each agent holds a subscription repository that includes each received subscription along with the respective neighbouring agent which forwarded it. Note that a neighbouring router constitutes the potential candidate for forwarding notifications. This repository is used to define if there exists a consumer along the direction of the considered router that subscribed for this notification. Based on this event notification, security information can be efficiently disseminated to the HDSOC.

Event notification Delivery Structure

In order to support an efficient event dissemination over a hybrid network, we propose a distributed event system, which distinguishes itself by:

- providing seamless integrated event notification over a network composed of different types of networking technologies (wired *versus* wireless, infrastructure-less *versus* infrastructure-based networks) and possibly spanning geographically dispersed domains/sites.

[2] The root of the delivery structure maintains information restricted to its cluster member.

- addressing the problem relating to the support of extensive control, security and autonomy by mean of a distributed event notification system based on a overlay infrastructure, which organise the nodes for delivering event notifications.

In practice, in order to distribute the events management, we based our event notification system on a distributed grouping communication which organises nodes into a self-organised delivery structure (Figure 3) deployed of the hybrid network, as defined in [3]. This structure corresponds to a cluster-based hierarchy of n_L layers (n_L=$log(n_n)$, with n_n designating the number of nodes that are expected to join the event system), each layer being portioned into a set of bounded-size clusters (let k be that size) controlled by a cluster head. The reason for setting bounds on the number of layers and on the cluster size is twofold. First, it ensures a control overhead ranging about $log(n_n)$ at each node. Second, the length of the path used for delivering notifications, and hence the related delay, is bounded by $o\ log(n_n)$. In order to prevent a cluster of size k from changing continuously its configuration whenever it gains or looses a member, the admitted bounded-size of a cluster ranges from k up to $2k$ cluster members. In practice, to warrant a loop-free structure, each node belongs to the lowest layer (L_0) and only the leader of cluster located in a layer L_i belongs to the upper layer L_{i+1}. This delivery structure is created and maintained by a grouping solution. Considering the fact that there exists no unique grouping protocol that is optimal for any kind of network, we use a specialised grouping solutions for each type of network along with a mechanism for integrating them. We

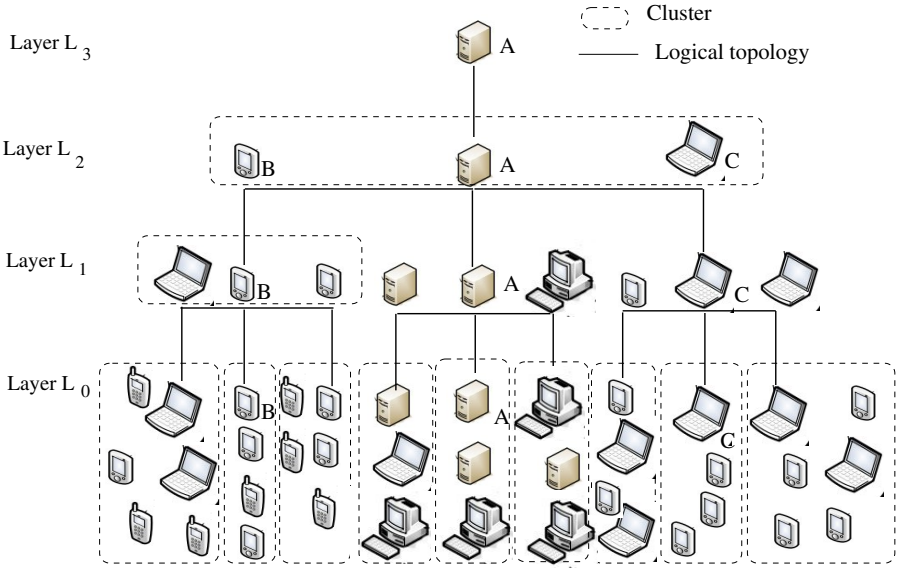

Fig. 3. Event Delivery Structure

rely on the Nice protocol [3] which has been specifically designed for operating in infrastructure-based large-scale network (e.g., the Internet) and the Madeira protocol that comes from our previous research on network management [10] and is customised to operating over *ad hoc* networks. The reason that motivates our choice for the Nice protocol, is twofold. First, this application-level protocol can operate over a large-scale network spanning different administrative domains. In addition, it were originally developed to support video streaming and hence meets the requirements driven by real-time delivery.

For scalability reasons, a node does not manage information concerning the overall group. Instead, a member or cluster head maintains information restricted to its cluster(s); each member sending periodically a *keep-alive* message. This limited knowledge permits to keep to a minimum the number of control messages exchanged for maintaining up-to-date membership information.

3.4 Distributed Service Discovery

The resource constraints of networked devices (e.g., routers, or devices belonging to *ad hoc* networks) coupled with the financial cost inherent to the deployment of additional functionalities on devices, circumvents the need for enabling HDSOC to delegate to remote devices a part of the functionalities relating to intrusion detection, e.g., log parsing and signature matching. The effective delegation of these functionalities, which are traditionally performed by ECBoxes, necessitates to discover on the fly the service(s) offered in the network that best match(es) these functionalities requirements. The following introduces a service discovery protocol [11] that meets this requirement. This protocol is aimed at hybrid networks including *ad hoc* networks. In the *ad hoc* network, our primary goal is to keep to a minimum the traffic generated by the service discovery process, so as to minimise consumption of resources and in particular energy. Specifically, our discovery architecture is structured around a subset of HDSOC nodes, called lookup agents or simply agents, that are responsible for discovering ECBox functionalities and capabilities (see Figure 4). These lookup agents are deployed so that at least one lookup agent is reachable in at most a fixed number of hops, $\mathcal{H}$, whose value is dependent upon the nodes density. Agents cache the descriptions of ECBoxes' functionalities (services) available in their vicinity which is defined by $\mathcal{H}$. Hence, HDSOC nodes (excluding lookup agents) do not have to maintain a cache of service descriptions, and the network is not flooded by service advertisements. A resource-constrained HDSOC device looking for a service (i.e., an ECBox or one of its embedded functionality), simply sends a query to the lookup agent for local service discovery. If the description of the requested service is not cached by the local agent, this agent selectively forwards the query to other lookup agents so as to perform global discovery. The selection of the lookup agents toward which service queries are forwarded, is based on the exchange of profiles among agents. The agent profile provides a compact summary of the agent's content and a characterisation of the host capacity. Agent profiles

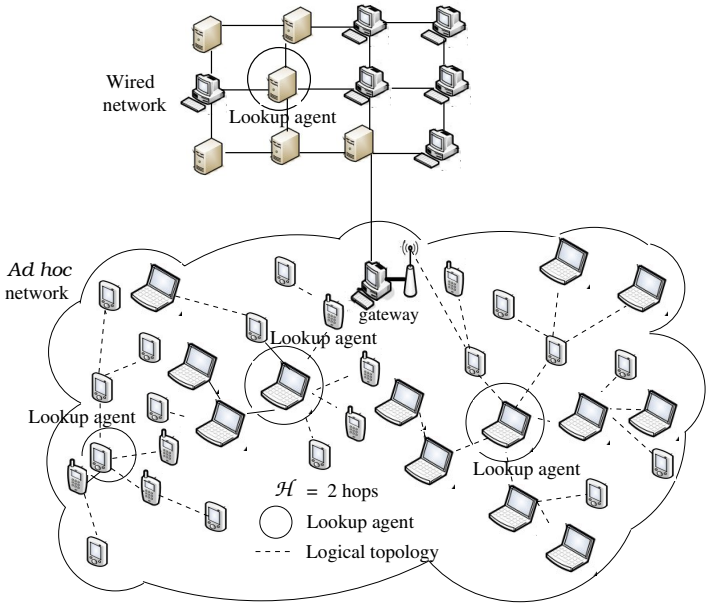

Fig. 4. Discovery Structure

allow both guaranteeing that service queries are issued to agents that are likely to cache the description of the requested service and to keep to a minimum the generated traffic. Another critical issue lies in providing convenient features so as to enable discovery of services over the hybrid network. This is supported through gateways that (i) advertise their capabilities (e.g., with a related gateway protocol, a zeroconf protocol, or using our protocol), and (ii) are assigned a local or remote lookup agent, implementing the cooperative behaviour as discussed above. A gateway lookup agent then holds the description of services available in all the networks composing the hybrid network, and advertises itself to the networks it bridges composing the hybrid network. Furthermore, to support service discovery in an infrastructure-based network in which a network administrator deploys agents, clients and service providers behave differently. Since clients and service providers do not need to elect a agents, they can listen for agents announcements or rely on the DHCP protocol.

Using this service discovery protocol, a resource-constrained DHSOC device can delegate to a remote device the resource-consuming functionalities that implement intrusion detection. For this purpose, it discovers dynamically the ECBoxes, located within the overall hybrid network, which offer the functionalities that best match its requirements. Then, it cooperates with the selected ECBoxe's service, utilising the provided service description, so as to contribute to the global effort for detecting intrusions.

4 Conclusion

In this paper, we propose a Hybrid Distributed Security Operation Center (HDSOC) which collects logs that are generated by any application/service, layer of the protocol stack or resource (e.g., router), providing a global view of the supervised system based on which it complex and distributed intrusions can be detected. Rather than directly transmitting these logs over the network, causing its overload, logs are parsed in an early stage so as to easily extract intrusion-related information and distribute it by the mean of compact event notifications and alarms. This HDSOC couples a lightweight distributed intrusion detection components with a distributed event system and a distributed service discovery protocol for an efficient delegation of resource-consuming tasks and a bandwidth-saving cluster-based collection of the events across the hybrid network. Our Hybrid Distributed Security Operation Center further addresses the main commitments of hybrid networks: scalability, flexibility, autonomy, and fault-tolerance. More precisely,

- *Scalability* comes from the distribution of load relating to the log parsing and the intrusion detection, on the devices
- *Autonomy* is the consequence of using a group based event notification service and a discovery protocol that are (i) automatically deployed without requiring human intervention and (ii) adapt dynamically to network changes (e.g., topology changes).
- *Fault-tolerance* is attributed to (i) a loosely-distributed event delivery that adapts dynamically to any permanent or transient network failure and a service discovery protocol that permits to discover dynamically an alternative to a faulty service.

Acknowledgements

Authors would like to acknowledge the implementation work carried by R. Bidou (University of Franche Comté), R. Chibout (INRIA), E. Cuadrado-Salamanca (EIRC, LM Ericsson Ltd), P. Farrell (EIRC, LM Ericsson Ltd) and A.K. Ganame (University of Franche Comté).

References

1. Anantvalee, T., Wu, J.: A survey on intrusion detection in mobile ad hoc networks. In: Wireless/Mobile Network Security. Springer, Heidelberg (2008)
2. Axelsson, S.: Intrusion detection systems: A survey and taxonomy. Technical Report 99-15, Chalmers University (2000)
3. Banerjee, S., Bhattacharje, B., Kommareddy, C.: Scalable application layer multicast. In: ACM SIGCOMM (2002)
4. CLausen, T., Jacquet, P.: Optimized link state routing protocol (olsr), rfc 3626 (October 2003), `http://www.ietf.org`

5. Clausen, T., Jacquet, P., Laouti, A., Muhlethaler, P., Quayyum, A., Viennot, L.: Optimized link state routing protocol. In: IEEE INMIC (2001)
6. Ganame, A.K., Bidoud, R., Bourgeois, J., Pies, F.: A high performance system for intrusion detection and reaction management. Journal of Information Assurance and Security (2006)
7. Ganame, A.K., Bourgeois, J., Bidou, R., et al.: A global security architecture for intrusion detection on computer networks. In: IEEE IPDPS (2007)
8. Pacheco, V., Puttini, R.: An administration structure for the olsr protocol. In: Gervasi, O., Gavrilova, M.L. (eds.) ICCSA 2007, Part III. LNCS, vol. 4707. Springer, Heidelberg (2007)
9. Albers, P., Camp, O., Percher, J.M., et al.: Security in ad hoc networks: a general intrusion detection architecture enhancing trust based approaches. In: Wireless Information Systems (2002)
10. Sailhan, F., Fallon, L., Quinn, K., et al.: Wireless mesh network monitoring: Design, implementation and experiments. In: IEEE GLOBECOM-DANMS (2007)
11. Sailhan, F., Issarny, V.: Scalable service discovery for manets. In: IEEE PERCOM (2005)
12. Wu, B., Chen, J., Wu, J., Cardei, M.: A survey on attacks and countermeasures in mobile ad hoc networks. In: Wireless/Mobile Network Security. Springer, Heidelberg (2008)