Intrusion Detection System for wide Automation Network Based on the Ethernet Compatible Communication Protocols

Jaroslav Kadlec, Radimir Vrba, Radek Kuchta
Faculty of Electrical Engineering and Communication Brno University of Technology
Brno, Czech Republic
kadlecja | vrbar | kuchtar@feec.vutbr.cz

***Abstract*— This paper is focused on the description of importance, design, and implementation of the Intrusion Detection Systems for a new automation system based on the Ethernet communication protocol. Newly developed and designed automation networks for complex factory control are composed from several types of automation communication links with different communication protocols, but most of the factory middle layer and top layer communication networks are based on Ethernet communication protocol. Wide use of Ethernet communication protocol not only in IT, but also in automation field, brings not only advantages of easy implementation and interoperability between different automation communication networks, but also brings risks and vulnerabilities, well known form IT. Therefore security incidents are becoming more serious and more common not only in computer networks, but also in automation networks. Actual trends in automation networks are among others wide automation networks covering several manufacture divisions or remote controlling of automation networks through the Internet. Necessity of a remote connection to the automation networks covers all security vulnerabilities and risks, which originate from the Internet. Analogically with IT, an automation network can be secured by the conventional way through firewalls and VPN tunnels, but automation networks have several specific requirements on the QoS, against the IT networks. For this reason a new automation firewall device was defined, designed and tested. The new automation firewall includes messaging system for logging all events and alerts originates form automation network. IDMEF (Intrusion Detection Message Exchange Format) is used, as a basis for automation firewall messaging system.**

Keywords- Intrusion detection; automation network; IDMEF.

I. INTRODUCTION

When a standard security mechanism is taking some actions to prevent the system from a threat, the engineering or a local intrusion detection system might be interested in such information. For this a policy has to be defined, when and how alerts and logging messages are processed.

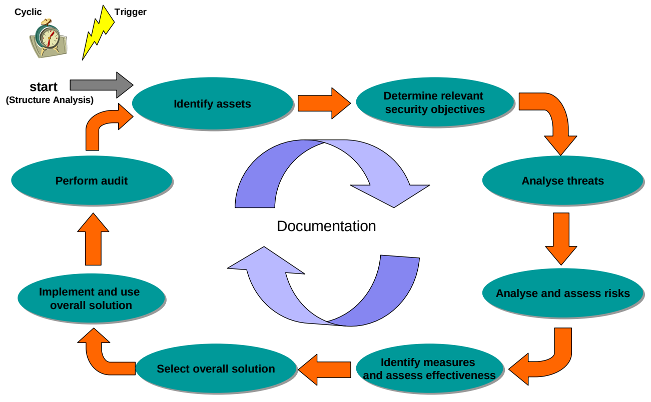


Fig. 1: Basic Security analysis procedure model [3].

The handling of messages for alerts or logging information should be in line with standardized mechanisms and methodologies, to be compatible with possible present intrusion detection systems. For this case the Intrusion Detection Message Exchange Format (IDMEF) is suitable. This format is defined in the RFC 4765 [1]. The purpose of the IDMEF is to define data formats and exchange procedures for sharing necessary information between intrusion detection and response systems and also to the management systems that may need to interact with them. Within that specification the data model is described to represent the information and the implementation in the Extensible Markup Language (XML) is presented. To realize this, a XML Document Type Definition (DTD) was developed for the specification. Beside the normative DTD a XML schema for the structure is also given in the specification, providing a definition for XML data, which is mostly used nowadays [2]. The requirements for this communication mechanism are specified in the RFC 4766 [2].

Based on this format, a new messaging system was defined. The main functions of the new messaging system are alarm creation, and logging of information. The XML structure for these messages was derived as a specialization of the general IDMEF-Message structure. The IDMEF-Message data structure definition is briefly shown in Fig. 2.

After short introduction in first section implementation section describes specification and implementation Intrusion Detection System to a heterogeneous network. Last section concludes reached results, and discusses future ways of implementation in real secured networks.

ISBN:978-1-61208-114-4

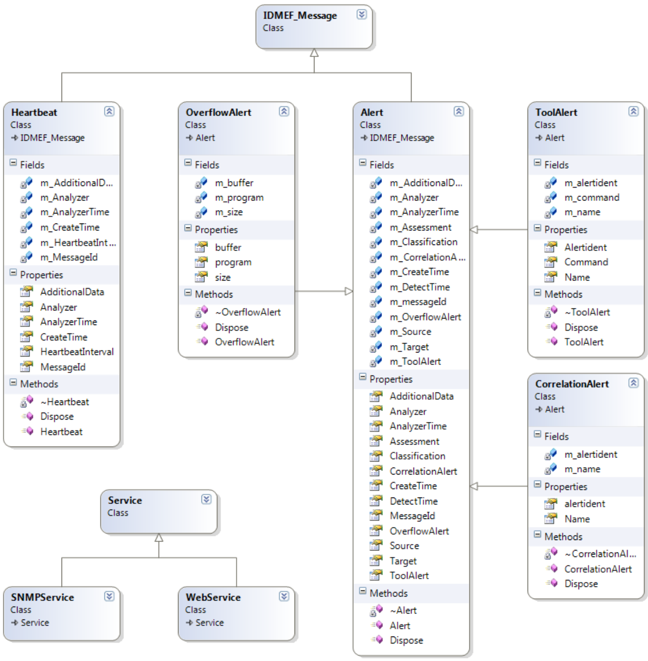


Fig. 2: Basic IDMEF-Message class diagram

II. IMPLEMENTATION

Implementation of Intrusion Detection System can be split into two implementation areas. The first area is implementation of IDS into firewall firmware and the second is developing of IDS message logger software. Both areas are interconnected by alert messages according to IDMEF-Message standard.

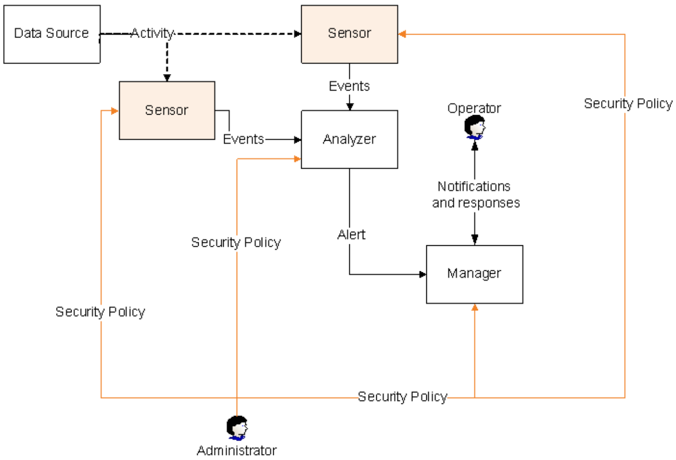
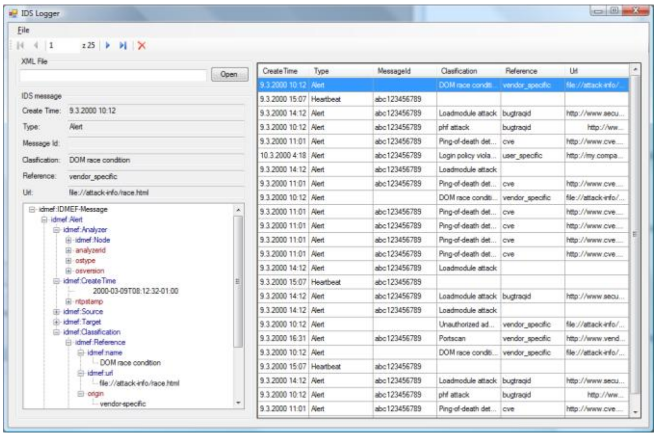


Fig. 3: Block diagram of the Intrusion Detection System [3]

The new automation firewall generates alert messages in XML file type according to RFC 4766 [2] and sending it to IDS message logger. For processing and logging of created messages by the Intrusion Detection System a software tool was developed. The IDS Logger offers GUI (Graphical User Interface) for displaying IDMEF messages and processing these messages to database for later evaluation. It also offers possibility to evaluate stored messages from database and export stored alerts back to the XML file.

Main window of the IDS logging software tool is shown in Fig. 4.



IDS message

Create Time: 9.3.2000 10:12

Type: Alert

Message Id:

Clasfication: DOM race condition

Reference: vendor_specific

Url: file://attack-info/race.html

```
idmef:IDMEF-Message
  idmef:Alert
    idmef:Analyzer
      idmef:Node
      analyzerid
      ostype
      osversion
    idmef:CreateTime
        2000-03-09T08:12:32-01:00
      ntpstamp
    idmef:Source
    idmef:Target
    idmef:Classification
      idmef:Reference
        idmef:name
          DOM race condition
        idmef:url
          file://attack-info/race.html
        origin
          vendor-specific
```

CreateTime	Type	MessageId	Clasfication	Reference	Url
9.3.2000 10:12	Alert		DOM race conditi...	vendor_specific	file://attack-info/...
9.3.2000 15:07	Heartbeat	abc123456789			
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack	bugtraqid	http://www.secu...
9.3.2000 10:12	Alert	abc123456789	phf attack	bugtraqid	http://ww...
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
10.3.2000 4:18	Alert	abc123456789	Login policy viola...	user_specific	http://my.compa...
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack		
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
9.3.2000 10:12	Alert		DOM race conditi...	vendor_specific	file://attack-info/...
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack		
9.3.2000 15:07	Heartbeat	abc123456789			
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack	bugtraqid	http://www.secu...
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack		
9.3.2000 10:12	Alert		Unauthorized ad...	vendor_specific	file://attack-info/...
9.3.2000 16:31	Alert	abc123456789	Portscan	vendor_specific	http://www.vend...
9.3.2000 10:12	Alert		DOM race conditi...	vendor_specific	file://attack-info/...
9.3.2000 15:07	Heartbeat	abc123456789			
9.3.2000 14:12	Alert	abc123456789	Loadmodule attack	bugtraqid	http://www.secu...
9.3.2000 10:12	Alert	abc123456789	phf attack	bugtraqid	http://ww...
9.3.2000 11:01	Alert	abc123456789	Ping-of-death det...	cve	http://www.cve....

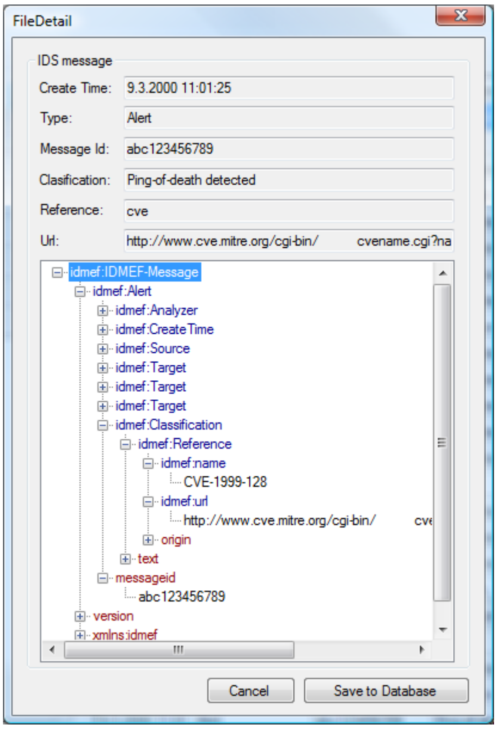
Fig. 4: Main window of IDS logging software [4]

An IDMEF-Message can be either an alert or a heartbeat message. In case of security device or application it won't send any regular status information, then the *Heartbeat* is not necessary and only the *Alert* message is used. For keeping backward compatibility with full IDMEF-Message specification we implemented also *Heartbeat* message. Each IDMEF-Message element has a *version* attribute that must contain string "1.0" for the value according to the specification.

The *Alert* message is generated every time an event occurs. In this case it means, a packet filter rule is fired with a higher severity level, the access to a security relevant function is denied or a similar event occurred. This element has an attribute called *messageid* that contains a unique identifier for this message.

The *Alert* message contains exactly one *Analyzer* element, holding the information about the analyzer, from which the alert originates. The most statements within this element are optional. Because there can be several security measures in an automation network, it is recommended to provide some basic information. The element contains the attribute *analyzerid*, which specifies the unique identifier for this application. The alert message also contains exactly one *CreateTime* element. This attribute specifies the time when this message was created. This element is the only time related element within IDMEF which is required. The value of this entry is the time, a regulative rule of an automation network security system is fired, which caused the creation of this message. Because there is no difference between the occurrence of the event and the creation of the message the optional element *DetectTime* is not used. The values of those time elements are the date and the time according to the dateTime data format and attribute *ntpstamp* which contains the time in the NTPSTAMP data type.

The next element within the alert message is the *Source* element, which contains the information about the sources of the event leading to the alert. This element is an array of *Source* elements for defining unlimited number of alert sources. It contains the *Node* element, which holds the address information about the source of the packet causing the filter rule to fire. To notify of the port number and the protocol of the source, the *Service* element can be used. The *Service* node is also part of *Source* attribute.



FileDetail

IDS message

Create Time: 9.3.2000 11:01:25

Type: Alert

Message Id: abc123456789

Clasification: Ping-of-death detected

Reference: cve

Url: http://www.cve.mitre.org/cgi-bin/   cvename.cgi?na

```
⊟ idmef:IDMEF-Message
   ⊟ idmef:Alert
      ⊞ idmef:Analyzer
      ⊞ idmef:CreateTime
      ⊞ idmef:Source
      ⊞ idmef:Target
      ⊞ idmef:Target
      ⊞ idmef:Target
      ⊟ idmef:Classification
         ⊟ idmef:Reference
            ⊟ idmef:name
                  CVE-1999-128
            ⊟ idmef:url
                  http://www.cve.mitre.org/cgi-bin/      cve
            ⊞ origin
         ⊞ text
      ⊟ messageid
            abc123456789
   ⊞ version
   ⊞ xmlns:idmef
```

Fig. 5: Detail view of the IDMEF message

To send the information about the intended targets of the event that caused the alert the element *Target* is used. It has almost the same structure as *Node* and *Service* attributes.

The *Node* element is used to represent information about a host or a device. This element can hold a *Name* and unlimited *Address* elements. In the new automation network firewall application, this node holds address information.

The *Address* element is used to represent any form of address of hardware or an application. It has an attribute *Category* to represent the type of address. The following keywords are used:

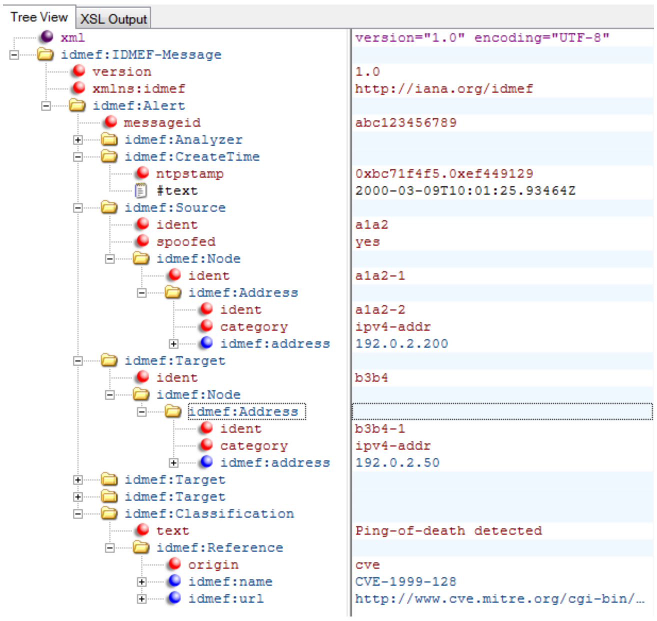
- *unknown* - for a not specified address type,
- *mac* – for the hardware Media Access Control address,
- *ipv4-addr* – for an IP address in the dotted-decimal address,
- *ipv4-net* – for an IP network address range in dotted-decimal with slash and significant bits (e.g. 123.123.123.123/24).

Beside this attribute it contains the string type element *Address*, holding the actual address.

The *Service* element is used to identify services for the source or the target by name, port, and protocol. This element has the optional attributes *iana_protocol_number* and *iana_protocol_name*.

The alert message also has exactly one *Classification* element, to provide a naming for the alert. It contains a text attribute with the name which illustrates type of the alert. This element can be used to distinguish a remote logging from a real alert.

Another elements are not required and not useful for the new automation network security messaging system but to keep IDMEF standard, IDS logger is fully compatible with other IDMEF messages and can process all attributes and nodes defined in RFC 4765 [1].



```
xml                                  version="1.0" encoding="UTF-8"
⊟ idmef:IDMEF-Message
    version                          1.0
    xmlns:idmef                      http://iana.org/idmef
  ⊟ idmef:Alert
      messageid                      abc123456789
    ⊞ idmef:Analyzer
    ⊟ idmef:CreateTime
        ntpstamp                     0xbc71f4f5.0xef449129
        #text                        2000-03-09T10:01:25.93464Z
    ⊟ idmef:Source
        ident                        a1a2
        spoofed                      yes
      ⊟ idmef:Node
          ident                      a1a2-1
        ⊟ idmef:Address
            ident                    a1a2-2
            category                 ipv4-addr
          ⊞ idmef:address            192.0.2.200
    ⊟ idmef:Target
        ident                        b3b4
      ⊟ idmef:Node
        ⊟ idmef:Address
            ident                    b3b4-1
            category                 ipv4-addr
          ⊞ idmef:address            192.0.2.50
    ⊞ idmef:Target
    ⊞ idmef:Target
    ⊟ idmef:Classification
        text                         Ping-of-death detected
      ⊟ idmef:Reference
          origin                     cve
        ⊞ idmef:name                 CVE-1999-128
        ⊞ idmef:url                  http://www.cve.mitre.org/cgi-bin/…
```

Fig. 6: Example of alert message according to IDMEF-Message definition

All captured messages from the IDS are stored in the MS SQL database. Due to the complexity of IDMEF are all alert messages stored in one database table instead of relational database tables. In this database table are stored most important parameters along with whole XML alert string. This solution saves processing time and allows us potential recreation of original alert message.

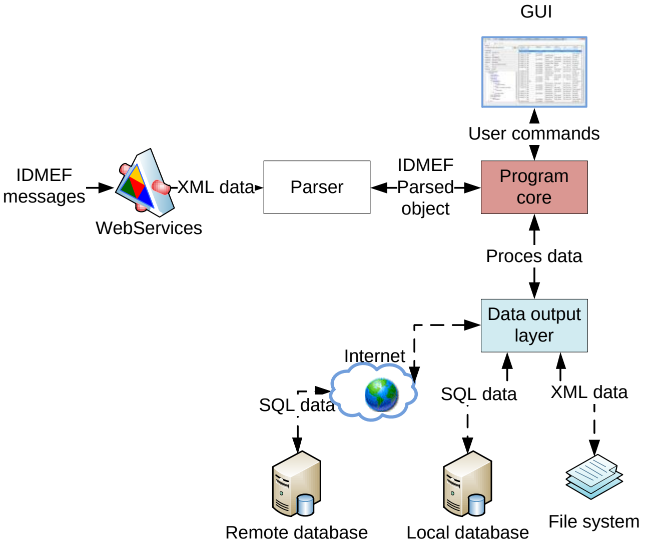


Fig. 7: Block structure of developed IDS

III. CONCLUSION

A complete implementation of IDMEF into the new automation messaging system was described. Messaging system based on the IDMEF is perfectly suitable for a new automation network firewall due to its compatibility with other similar devices. Identified and captured alert message is processed and stored by IDS Logger tool to MS SQL database. This solution allows controlling of all used firewalls in large automation network, over one database and in one easily managed place. Stored alerts are the most important sources of security information. User can monitor all attempts to un-allowed connections and intrusions to the internal secure area of the automation network. Based on the logged alerts, user can have perfect knowledge about security risks and can effectively react and protect all automation devices within automation network.

ACKNOWLEDGMENT

This research has been supported by the Czech Ministry of Education, Youth and Sports in the frame of MSM 0021630503 *MIKROSYN New Trends in Microelectronic Systems and Nanotechnologies* Research Project, partly supported by ARTEMIS JU in Project No. 100205 Process Oriented Electronic Control Units for Electric Vehicles Developed on a multi-system real-time embedded platform and by ENIAC JU in Project No. 120001 Nanoelectronics for an Energy Efficient Electrical Car, partly by the Czech Ministry of Industry and Trade in projects FR-TI1/057Automatic stocktaking system and FR-TI1/057 Intelligent house-open platform.

REFERENCES

[1] H. Debar, D. Curry, and B. Feinstein: *The Intrusion Detection Message Exchange Format IDMEF*; RFC 4765; IETF;1. March 2007, http://tools.ietf.org/html/rfc4765.

[2] M. Wood and M. Erlinger: *Intrusion Detection Message Exchange Requirements*; RFC 4766; IETF;1. March 2007; http://tools.ietf.org/html/rfc4766.

[3] ANSI/ISA, ANSI/ISA-99.02.01-2009 *Security for Industrial Automation and Control Systems*, Establishing an Industrial Automation and Control Systems Security Program, 2009.

[4] VAN – *Virtual Automation Network, Real Time for Embedded Automation Systems including Status and Analysis and closed loop Real time control*, Real-time for Embedded Automation Systems deliverable, 6th Framework Program, 1.8. 2008, http://www.van-eu.org/sites/van/pages/files/D04.1-1_FinalV1_2_060702.pdf