A SECURITY MODEL FOR SYSTEM TRACK RADAR DATA

George ICRIVERZI[1], Valentin CRISTEA[2]

Informațiile radar folosite in dirijarea traficului aerian sunt date deosebit de sensibile din punct de vedere al securității. Alterarea acestora poate aduce tulburari grave activității de dirijare a traficului aerian. Formatul codificării standard a datelor radar permite introducerea de date invalide sau modificarea celor existente intr-un sistem de prelucrare a datelor radar. Sunt necesare protecția datelor reale si detecția datelor alterate. Articolul propune un model de securizare a acestor date fără a afecta structura codului folosind o infrastructură PKI. Am testat cu succes acest model pe un eşantion de date radar generate anterior şi am analizat rezultatele obținute.

Radar information used in Air Traffic Management represents very sensitive data from the security perspective. Altering this data can produce serious disturbance to the air traffic management activity. Standard radar data coding allows the input of invalid data or the modification of the existing one in a radar data processing system. It is necessary to protect real data and to detect the unauthorized modifications. This article proposes a security model for this data without altering the standard code structure using a PKI infrastructure. We successfully tested this model on a sample of generated radar data and analyzed the results.

Keywords: radar data security, system track authentication, ASTERIX, digital signature

1. Introduction

Air Traffic Management is a domain where safety is always put on the first place. Among the components of an Air Traffic Management System there are the Radar Data Processing Systems. These systems have a complex structure, distributed over large areas the size of a country and can contain up to 200 hosts. A RDP system has several distributed components. The most important are *the SSR Radar* - it's a radar system used in air traffic control which detects and measures aircraft's position and interrogates it for other information such as identity and altitude [1] and *Multi Radar Tracker (MRT)* - a component part of a radar system which combines consecutive radar reports of the same target into a

[1] PhD student, Computer Science Department, University POLITEHNICA of Bucharest, Romania, e-mail: gicriverzi@yahoo.com
[2] Prof., Computer Science Department, University POLITEHNICA of Bucharest, Romania, email: valentin.cristea@cs.pub.ro

track. It is used when the radar system reports data from different targets or when it is necessary to combine data from different radars heads or other sensors [2].

Radar plots are received by the Radar Head (RHP) and processed by the local tracking system, resulting local tracks. The tracks from several radar head sites are centralized in a MRT system which combines the local track into *system tracks* [3]. The system track is distributed towards other system components like controller consoles where the information is displayed graphically in order to be used by the ATC (Air Traffic Controller). The data transmitted from MRT to the ATC consoles represents the final information based on what the controllers' activity is performed. The information is displayed as moving targets with associated information (speed, level, direction, etc.). It is very important that this information arrives unaltered to the destination, otherwise critical situations may occur [4].

The system track data format sent from MRT to the consoles is regulated by the ASTERIX standard [5][6]. ASTERIX (All Purpose Structured Eurocontrol Surveillance Information Exchange) is an air traffic control information exchange standard. It is developed and maintained by the European organization for air traffic control EUROCONTROL. ASTERIX is also a refined data format adopted by almost all ATC community as the universal standard in the field. The ASTERIX standard fits at the presentation and application levels (six and seven) defined in the OSI standard. Defining the communications at lower levels is not part of the standard purposes.

We'll describe a few possible attacks over radar data and their effects. Among the security vulnerabilities of a RDP system one of the most dangerous and hard to detect is related to the tampering with the system track data arriving at the display consoles. A tampering attack can have several forms:

- system track data records **modification** - erroneous information will get to the controller consoles, the radar targets will be displayed in different from reality positions or with modified flight parameters (speed, prediction, direction, emergency code, etc.). The Air Traffic Controller will deal with fake targets instead of real traffic.
- system track data records **dropping** - the targets will not be displayed on the screen. The ATC is not informed about the air traffic situation and can't make the correct decisions.
- system track data **addition** - means that fake targets will be displayed. The ATC will deal with the fake targets neglecting the real ones.

All of the above attack forms result in erroneous information of the ATC which won't be able to make the correct decisions, risking the air traffic safety and the lives onboard the aircrafts. The problems above could be avoided using a system track data **authentication** method.

The problem analyzed by this paper is the securing of ASTERIX code that describes MRT information sent to the radar consoles. Securing of this information can be made at inferior levels (network, transport) as well as application level by adding supplementary information to the ASTERIX code. The method presented here does not add information other than the ASTERIX code itself. The securing information is integrated into the code thanks to some facilities existent in the ASTERIX standard specifications. The used technique for this purpose is the digital signature generated in a PKI framework integrated in the RDP system [7]. The idea has a large application area because every state uses a complex operational radar data processing system.

The documentary research showed that direct-in-the-code ASTERIX authentication is a new approach in the ATM field. The existing commercial products [8][9] advertise external code securing methods. The challenges that this approach faces are the increased volume of the transmitted code and the extra processing resources needed to perform the security operations. The transmission of the radar information must be done in real time, so it is critical that the overhead induced by this method will not delay any data exchange.

The paper is organized as follows. Section 2 contains information about the present ATM data securing. Section 3 details the problem's solution and the implementation of the described method. Section 4 presents the applied method's results over a sample of radar data, Section 5 contains the results analysis, in Section 6 we analyze the different systems compatibility and Section 7 presents the conclusions.

2. The present ATM data security

Securing of an ATM system is made at all levels, starting from physical level (robust location, 24/7 guarding, access control, etc.) and further to the network, host, application and user levels.

Present integrated ATM systems use different mechanisms for securing data. The techniques used by THALES [8] in designing ATM systems are protected network access (Intranet / Internet / Extranet), system protection (integrity, configuration), secure data transfer networks migration, data confidentiality and backup, single sign-on, profile management.

The Comsoft's SPDE product (Surveillance Data Front End Processor) relies on IP technologies applied over ASTERIX data for securing. Integrated firewall prevents unauthorized access and security attacks from external interfaces [9]. This system can handle IPv4 and IPv6 protocols. It manages local and remote data exchange offering support for TCP and UDP services like unicast, multicast and broadcast. SPDE uses a highly customizable integrated firewall that prevents

unauthorized access. This system uses inhibition methods for DoS and flooding attacks by internal load detection and control mechanisms.

The above systems do not use the same security mechanisms as none is imposed by ICAO/EUROCONTROL standards. The ASTERIX standard also does not enforce any method for securing the data; moreover it does not specify a recommended transport protocol to use, leaving it to the users' choice.

According to ATM Infrastructure Strategy for Europe [10] the ATM systems must become compatible in order to facilitate the exchange of aeronautical data (navigation, surveillance, flight plans). The use of different security mechanisms will restrict the secure data exchange between two ATM systems. Using the proposed solution this limitation will be eliminated for ASTERIX cat.062 data. The digital signature allows facile checking of the received data. This level of security applied to certain transmitted data in a RDP system does not modify the standard code structure thus preserving the compatibility with similar systems that haven't implemented this security mechanism yet.

Another reason to sustain this model is that the security measures of a certain product are the property of respective company. By enforcing a security mechanism in the standard specification we obtain a public solution that will be implemented by all ASTERIX users. The code level authentication is also justified as a supplementary security level in case all the other mechanisms fail.

3. The method's description

The solution proposed for ASTERIX cat.062 code authentication relies on the use of a digital signature added in a field of an ASTERIX Record. This standard is extendable with a number of different categories, each corresponding to a particular type of information. The data category referred by this paper is 062 and contains *system track* data used to draw the existent air traffic at a certain moment in order to be used by the ATC.

We will describe briefly the format of the ASTERIX code. The base notions of this format are: Data Items Catalogue - it's the list of all possible data articles from every Data Category describing the articles by reference, structure, dimension and measuring unit, Data Block - it's the unit of information seen by the application as a discrete entity by content; a data block contains one or more Records of Data from the same Category, Data Category - the sorting of data in order to allow easy identification of a data item, Data Field - for communication purpose it represents the physical implementation of the Data Item, it is associated with an unique reference number (Field Reference Number) and it is the smallest transmitted unit of information, Data Item - the smallest unit of information from every data category, Record - a collection of Data Items from the same Category

preceded by a Field Specification which signals the presence/absence of different Data Fields, User Application Profile - The association mechanism between Data Items and Data Fields and the necessary information for the successfully message coding and decoding standardization [5].

The transmitted data contains one or more consecutive Data Blocks [5][6]. A Data Block contains a one octet field Data Category (CAT) that indicates the Data Category this transmission belongs to, a two octet field (LEN) indicating the total length (in octets) of the Data Block, including CAT and LEN fields and one or more Records containing data from the same category

There is a special facility offered by a special field (**Special Purpose Field**) that allows a subgroup of users to transmit a variable length field which will be transparent to non-interested users. This completes the mechanism of the ASTERIX Record and intents to offer an alternative for exchanging exceptional data. When this facility is used, a special indicator (Special Purpose Indicator - SP bit) will be set in the FSPEC field. The first octet will contain the explicit length of the field in octets that includes the indicator's length. The next field contains the special information that needs to be transmitted.

The model described in this paper uses this special field (SPF) to add digital signature information to a Record. The digital signature will be generated within a PKI framework [11]. The PKI modules can be hosted on dedicated machines or can be installed on existing hosts from the system [12]: CA (Certification Authority) and RA (Registration Authority) can be installed on the Flight Data Processing Server, KGS (Key Generation System) and CSP (Crypto Service Provider) can be installed as CA sub modules. CSP must also be present on the machines that will digitally sign the Records transmitted over the network (MRT server).

The process of transmitting ASTERIX Cat.062 packets in a RDP system integrated with PKI will contain the following steps:

- MRT is receiving the *local track* data from the RHP radars in Cat.001 and Cat.002 format
- the *system track* is generated based on the *local tracks*
- MRT server digitally signs the Cat.062 Records and build the Data Blocks, putting the corresponding SPF bit in FSPEC on 1
- the data is transmitted over the network for the interested parties (radar displays, flight plans server, etc.)
- when the signed data is received the digital signature is verified using the signer server's digital certificate available through a public directory (LDAP)

The receiving Cat.062 packets display consoles and servers from the network have the possibility to verify the digital signature present in the SP field. An option must be added to skip checking this field to use the system track data

without origin authentication in order to keep the compatibility with another system track sources.

If the signatures verification occurs then the data originator must be identified. In order to avoid confusions every system will have a unique identifier in the area where this code for data exchange is used [6]. The format of the ASTERIX system identifier contains two subfields: SAC - System Area Code (8 bits) and SIC - System Identification Code (8 bits). For the radar data processing system defined, the SAC can be the country code and the SIC will identify the MRT server.

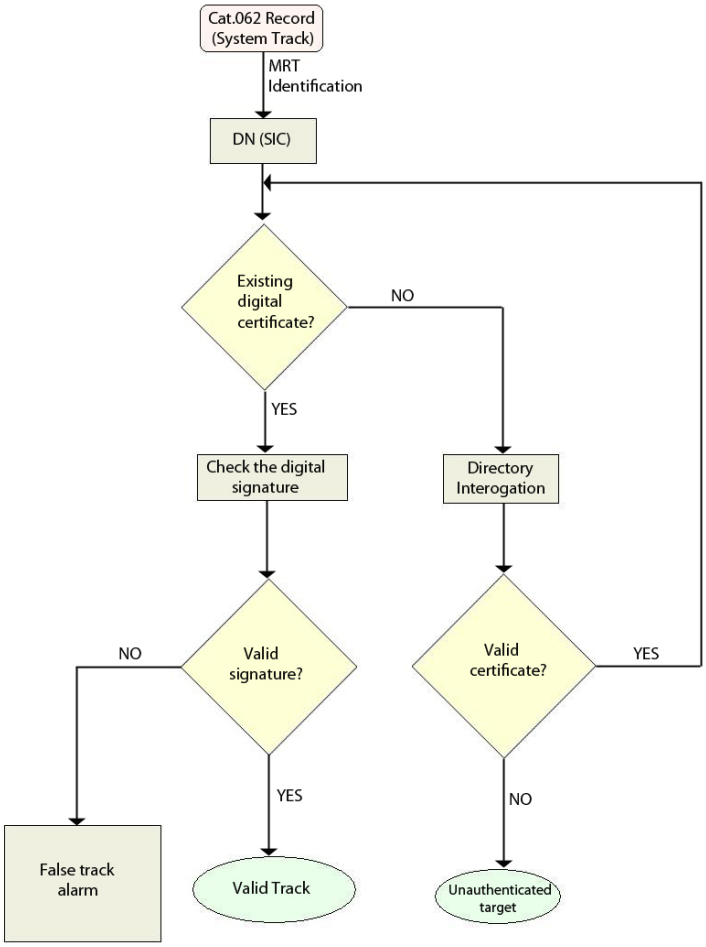

Fig. 1. System Track Cat.062 message digital signature checking

The steps for using the digital signature are highlighted below.

1. Based on the SAC/SIC values from the Record Data the Distinguished Name (DN) is built. This value is necessary in retrieving the corresponding digital certificate from the directory.

2. If the digital certificate corresponding to the DN is not already imported, or if a determined time from the last certificate check expired, a request is issued to the directory in order to retrieve the certificate.

3. If the certificate is revoked or suspended this information must forwarded to the end users: an alarm must be raised on the radar displays for the ATC and one at the monitoring station for the technical staff.

4. If the certificate is valid, the digital signature check can be performed:

- the digital signature is extracted from field 16 (SPF)
- check that the hash value is identical with the one generated by the verifier

5. If the verification passed, the information may be used by the host.

6. If the verification fails an alarm is raised as in step 3.

There may be more than one MRT sources in a RDP system [4]. Each source will be identified by the SAC/SIC field from the Cat.062 Record header. The application that decodes the Cat.062 information will request the digital certificate for every MRT source.

If one of the sources hasn't implemented the system track packets authentication yet (the source can be from a neighbor state) then the useful information will be used as it is but the end user will be informed about this. The end-users are the ATC and the technical staff that administrates the system hosts.

FSPEC (SPF bit = 1)	Field 1	Field 2	...	Field 16 - SPF contains the digital signature	...

Fig. 2. Data Record with the digital signature in SPF (Field 16)

We studied the effect of adding digital signature information to cat.062 code. For this purpose we generated a radar data sample in accordance with the ASTERIX format. The data from every Record were digitally signed and the signature was added to the Record in the SPF field. The generated test data category was 062, the one that contains system track data. The transmission of system track data requires the transmission of the target position and identification. The data from this category are especially important, any tampering with these data can lead to critical situations.

In order to generate ASTERIX cat.062 code we used the following UAP derived from the cat.062 standard [6]:

Table 1

System Track UAP

Field Reference Number	Status in F-Spec	Data Item	Name	Length in bytes
1	1	I062/030	Data Source Identifier	2
2	1	I062/044	Time Of Track Information	3
3	1	I062/056	Track Number	2
4	1	I062/100	Calculated Track Position (Cartesian)	4
5	1 / 0	I062/160	Track Mode 3/A Code	2
6	1 / 0	I062/235	Calculated Track Flight Level	2
7	1 / 0	I062/336	Measured Flight Level	2
FX	1	-	Field extension indicator	-
8	1	I062/480	Calculated Track Velocity	4
9	1	I062/500	Mode Of Flight	1
10	1 / 0	I062/620	Calculated Rate Of Climb/Descent	2
11	1	I062/780	Track Status	1 / 4
12	1	I062/840	Measured Information	2
13	1	I062/890	Flight Plan Related Data	3 / 29
14	1 / 0	I062/902	Track Mode 1 Code	1
FX	0	-	Field extension indicator	-
15	1 / 0	I062/903	Track Mode 2 Code	2
16	1 / 0	SP	Reserved For Special Purpose Indicator	1 / n

The digital signature is applied to every Record: the bit array starting from Field1 to Field15 is extracted and digitally signed. The result is stored in Field16: the first two octets indicate the signature length (sLEN), the next sLEN octets contain the digital signature of the Record data.

The testing of this method was made in a few steps. We generated a file with ASTERIX cat.062 data without digital signature content. After that the file was parsed in order to add a digital signature to every encountered Record and the data were saved in a new file.

4. Experimental results

The application was written in Java 1.5 under Eclipse 3.2. For digital signature generation we used the default security provider (SunRsaSign version

1.5) and the BouncyCastle distribution for comparison. The used algorithms were *SHA256withRSA* and *SHA512withRSA* and the RSA private key length were 512, 1024 and 2048 bits.

The traffic generation was made based on estimations from Miami Air Route Air Traffic Control Center [13] and Federal Aviation Administration [14]. With an average of 6500 flights per day for a region corresponding to a MRT system the one hour load is 270 units. For 15 minutes we have 67 units. Considering the incoming and outgoing traffic the value doubles to 135 units. We considered that 150 units for 15 minutes is a realistic estimation for simulating the real traffic. The update interval of MRT information (system track information frequency) for track data is approximately 5 seconds (once every antenna rotation [2]).

The generation of cat.062 data for 15 minutes and the conditions above produced a 718KB file. There were generated 2296 Blocks and 20406 Records with an average of 8.88 Records/Block.

The file parsing was made in 2172 ms on a system with P4/2.8MHz processor and 1.5GB RAM. Adding digital signature to every Record using *SHA256withRSA* using a private key of 512 bits and the default security provider of JDK 1.5 distribution produced a 2033KB file in 60.98 sec. Using *SHA512withRSA* and a 1024 key generated a 3308KB file in 351.282 sec. For *SHA512withRSA* and a 2048 key the time was 2662.125 sec. and the file was 5859KB long. Below we have a table with the time values needed for generating the signature.

Table 2

The time needed to generate the digital signature (seconds)

Provider/Algorithm-Key	SHA256withRSA / 512bits	SHA512withRSA / 1024bits	SHA512withRSA / 2048bits
SunRsaSign version 1.5	60.980	351.282	2666.110
BouncyCastle version 1.46	79.485	502.500	2662.125

The time values for verifying the signature are given below.

Table 3

The time needed to verify the digital signature (seconds)

Provider/Algorithm-Key	SHA256withRSA / 512bits	SHA512withRSA / 1024bits	SHA512withRSA / 2048bits
SunRsaSign version 1.5	9.969	26.313	82.781
BouncyCastle version 1.46	10.687	27.500	84.531

The private keys were previously generated and stored on the disk. The application loads the private key from the file and then starts the data parsing.

The methods security power is given by the RSA private key length. In 2010 a 768 length RSA number was factored [15]. The paper recommends that in a few years to consider upgrading to keys longer than 1024bits.

SHA256 and SHA512 are part of the SHA-2 set of cryptographic functions. Security flaws were found in SHA-1 indicating the need for stronger functions. These flaws were not successful on SHA-2 [16]. Although SHA-3 is under development for the time being these hash functions are the strongest in SHA family.

5. Results analysis

The data files sizes allowed us to compute the transfer speed needed to transfer cat.062 code. For the Records that were not digitally signed we have 0.79KB/sec and for those digitally signed the necessary speeds are 2.25KB/sec, 3.67KB/sec and 6.51KB/sec respectively. The growth from 0.79 to 6.51 is significant but the absolute values of the necessary traffic are extremely low compared to today's resources (122070KB/sec for Gigabit Ethernet). We do not have a significant overhead on the network traffic by adding the digital signature.

For the processing resources analysis the results showed that for *SHA256withRSA* and 512bits keys the raw parsing of the collected data in 15 minutes took 60.98 seconds using the SunRsaSign 1.5 security provider. For 20406 Records the necessary resources needed for 15 minutes are 22.9Records/sec. The recorded raw speed for signing with the mentioned algorithm is 334.63 Records/sec according to the table. For *SHA512withRSA* and 1024bits key the speed is 58.09Records/sec and for SHA512withRSA with 2048bits key we have 7.66 Records/sec.

We noticed that for 512bits and 1024bits RSA keys the processing power is adequate (the raw speed is greater than the needed speed) and for 2048bits keys the needed resources must be stronger than P4/2.8MHz for real time signing.

The necessary resources to produce the digital signature are significantly greater that those needed for verifying the signature. From the used security providers we saw that the default one generally produced far better results, the only exception being the 2048bits key signature which yielded similar results.

Signing data with RSA keys of maximum 1024 bits induces insignificant overhead for network traffic and processing resources and 2048 bits keys need a stronger processing resource than the one used in the test. For a system with little processing power the use of digital signature with maximum 1024 length keys is justified but a system with stronger processing unit is needed for 2048bits key real time digital signing. The achievements of this model are the authenticated cat.062 data that can be exchanged with other hosts or systems.

6. Compatibility analysis

Two or more different RDP systems must keep the system track messages exchange compatibility even after one of them implemented the presented security model.

Let's assume that we have two ATM centers that have different implementations for ASTERIX coding: ATM1 and ATM2 [16]. ATM1 uses a PKI framework to obtain Cat.062 data authentication and ATM2 does not use this mechanism.

Case I

ATM1 sends system track data to ATM2.

The data sent by ATM1 contain in SPF field the digital signature of the content. The hosts from ATM2 system will not verify the digital signature so they will ignore the SPF content. The other data that is sent are unmodified so they can be used in the system without compatibility issues.

Case II

ATM2 sends system track data to ATM1

The data received by ATM1 does not contain in SPF field information related to the digital signature. ATM1 hosts will try to probe the MRT data authentication but will fail the LDAP interrogation step for obtaining the MRT source digital certificate. In order to avoid repeated interrogations a list with MRT sources that will not be tested for authenticity must be created. This situation however has the known risks presented before so the end users must be informed and offered the option of filtering these data.

This proves the compatibility in exchanging data between two different ATM systems from which one uses the digital signature model presented above.

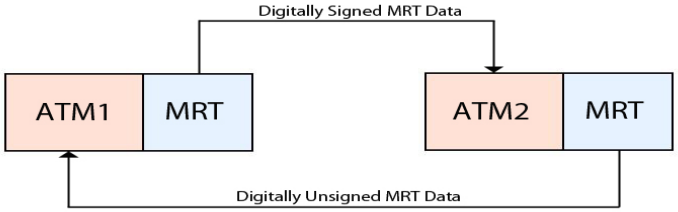

Fig. System track data exchange compatibility between different systems

7. Conclusions

By adding a digital signature in the field 16 of Cat.062 Records we obtain the system track ASTERIX code authentication. The authentication is necessary as a supplementary security measure over those already implemented by a specific producer and can be integrated in the ASTERIX standard coding in order to achieve compatibility in securing the radar data from different systems. Using a

PKI model gives a large choice area for a specific product, private or open-source. The resources needed to perform the necessary operations are not very demanding, an ordinary PC can manage the simpler signature generation cases.

A very important aspect of this model is that it keeps the compatibility with other ASTERIX based systems that didn't implement this model. The compatibility is necessary when the system track data is exchanged between different states where different producers RDP implementations operate and some may not use the system track authentication model [17].

In order to reduce the processing overhead, for future developments an alternative to adding digital signature to every Record is the use of only one signature within a data Block.

The presented method opens the perspective of securing the other ASTERIX categories in a similar way.

R E F E R E N C E S

[1] *M. C. Stevens*, "Secondary Surveillance Radar", Artech House, 2003
[2] *Merrill I Skolnik*, "Introduction to Radar Systems", McGraw-Hill, 2001
[3] *S. Kelly, J. McIntyre*, "Radar Surveillance", vol. 2, Eurocontrol, 2003
[4] *J. Still, S. Bunjevac*, "Data Processing Chain", Luxembourg, 2005
[5] *D. Doukas, J. Berends, M. Rees, G. Kerkhofs*, "Surveillance Data Exchange", Part 1, ASTERIX, Eurocontrol, 2007
[6] *D. Doukas, J.-M Duflot, B. Redeborn*, "Surveillance Data Exchange", Part 9 : Category 062, SDPS Track Messages, ASTERIX, Eurocontrol, 2007
[7] *R.L. Rivest, A. Shamir, L. Adleman*, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Volume 21 Issue 2, Feb. 1978
[8] *THALES Group*, http://www.thalesgroup.com/Portfolio/Security/D3S_Product_dev_integr_it_sol_services/?pid=1481, 2011
[9] *COMSOFT*, http://www.comsoft.aero/html/atc/products/sdp/sdfep.htm, 2011
[10] *Eurocontrol*, "ATM Infrastructure Strategy for Europe", 2008
[11] *R. Housley, T. Polk*, "Planning for PKI", Ed. John Wiley & Sons, 2001
[12] *C. Adams, S. Lloyd*, "Understanding Public Key Infrastructure", Macmillan Technical Publishing, 1999
[13] *Miami Air Route Air Traffic Control Center,* http://aspm.faa.gov/opsnet/sys/Tracon.asp, 2011
[14] *Federal Aviation Administration,* http://faa.travel/miamicenter.html, 2011
[15] *Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Joppe W. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, Paul Zimmermann,* "Factorization of a 768-bit RSA modulus", 2010
[16] *Bruce Schneier*, "Cryptanalysis of SHA-1", 2005
[17] *European Organisation for the Safety of Air Navigation*, "Infrastructure Strategy for Europe", Eurocontrol, 2008