UNIVERSITÀ DEGLI STUDI FIRENZE

FLORE
Repository istituzionale dell'Università degli Studi di Firenze

Security Concepts in IPv6 Based Aeronautical Communications

Questa è la Versione finale referata (Post print/Accepted manuscript) della seguente pubblicazione:

Original Citation:

Security Concepts in IPv6 Based Aeronautical Communications / Tommaso Pecorella; Romano Fantacci; Luigia Micciullo; Antonietta Stango; Neeli Prasad; Piotr Pacyna; Norbert Rapacz; Tomasz Chmielecki. - STAMPA. - (2011), pp. 101-128.

Availability:

This version is available at: 2158/510657 since:

Publisher:

Intech

Terms of use:

Open Access

La pubblicazione è resa disponibile sotto le norme e i termini della licenza di deposito, secondo quanto stabilito dalla Policy per l'accesso aperto dell'Università degli Studi di Firenze (https://www.sba.unifi.it/upload/policy-oa-2016-1.pdf)

Publisher copyright claim:

(Article begins on next page)

12 October 2024

5

Security Concepts in IPv6 Based Aeronautical Communications

Tommaso Pecorella[1], Romano Fantacci[1], Luigia Micciullo[1], Antonietta Stango[2], Neeli Prasad[2], Piotr Pacyna[3], Norbert Rapacz[3] and Tomasz Chmielecki[3]
[1]*Università di Firenze, Firenze*
[2]*Center for TeleInFrastruktur (CTIF), Aalborg*
[3]*AGH University of Science and Technology, Kraków*
[1]*Italy*
[2]*Denmark*
[3]*Poland*

1. Introduction

Although aeronautical networks rely on communications, nowadays the largest part of such communications is based on old but proven standards, many of them being analogical and voice-based.
It is widely recognized that this communication model will not be able to support the increasing complexity and worldwide spread of aeronautical communications, especially with the ever-increasing number of players and locations.
The IP communication model seems a good candidate to replace the analogical communications. On the other hand, the debate about the well-known IPv4 pools depletion made it clear that the only viable and sustainable solution is to adopt IPv6 as a common basis. Concerning IPv4, it is believed that the only real need will be for passengers communications. IPv4 traffic can be segregated so to not harm the main network architecture.
IPv6 network security is not different, from a logical point of view, from IPv4 one. The main difference is related to a simple yet hard to fully admit concept: security is not subject to the black-swan theory (Taleb, 2010), and while we do have years of experience in IPv4 networks, we do not have the same amount of case studies for IPv6.
Literature does identify security as the "sum" of a number of properties like confidentiality, integrity, availability, authenticity and non-repudiation. We do prefer to use a different approach, as those are the properties that need to be ensured, but if we look at them alone we will probably end up missing big points. Any network (or any system) can be seen as the result of two main processes: design and implementation. This is valid for a whole network and its single components. In order to achieve the goal of enforcing the five above mentioned properties, a network has to be designed and implemented with specific features. The properties can then be enforced directly on the design and implementation or being built

on top of it. On the other hand, a flawed design (or implementation) will make all the efforts vain.
Another point to keep in mind is related to the concept of security itself. The term "security" is fancy and sounds good, but it is quite generic. According to the Oxford Dictionary, security is defined as *the state of being free from danger or threat.* The problem is then to identify what is the meaning of "danger" and "threat" and to relate with their prevention.
An aeronautical system is a Critical Infrastructure (CI) whose ultimate goal is to carry passengers and goods across the sky in a secure mode. A CI is, in general term, any infrastructure that is considered very important for economical or social relevance for a country. Any CI nowadays rely for its operations on a telecommunication system and on associated informatics systems. Those are called Critical Infrastructure Information (CII). It is worth noticing that the CII itself have little or no interest on the original goals of the CI. Instead, the CII is interested in meeting its own targets about security, performance, reliability and so on. The latters have to be driven by the CI mission, and this is the main problem in defining the security of a CII. Once those security requirements have been defined, the separation of concerns is completed and the CII can be defined as an almost completely orthogonal domain although being still a fundamental part of the CI.
To set the CII priorities in a correct and traceable way, it is of paramount importance to use an enterprise model usually in form of Enterprise Architecture built on top of an Enterprise Architecture Framework.
Even if the approach seems linear, in the particular case of aeronautical systems there are some peculiarities that have to be considered and, so far, are still open points:

- Each airport serves a number of different carriers,
- Each carrier lands in a number of different airports,
- Each carrier and airport is bound to follow:
  - International rules (aviation ones, mainly),
  - Local rules (national laws).

Each of the above should contribute to the definition of the EAF principles and drive the EAF constraints.
All these elements make it extremely difficult to find a common ground that might be useful and good for every single entity involved in the system.
In this chapter we will briefly describe the EAF principles and how they can be successfully used to model a CI. We will then outline what are the major differences between a "classical" IP network and an IPv6 network, and how these differences should be particularly be addressed in order to not pose a security risk for the CI.

2. Enterprise Architecture frameworks

The concept of enterprise can be applied to any type of organizations, commercial, public services, governments, and in general also to CI. The aims of an enterprise are to optimize all parts of the organization in a coherent way, rather than to achieve local optimization at business unit level (Sherwood et al., 2005). "The role of 'architecture' is to provide the framework that breaks down complexity into apparent simplicity" (Sherwood et al., 2005). EA is an abstract view which requires collaborative information from both business and IT professionals (Oda et al., 2009).

2.1 Importance of using an EAF in a Critical Infrastructures

Critical Infrastructures are large and complex enterprises, which often are composed of multiple systems. Organizations seek to establish common frameworks that allow them to rule on the development and evolution of such systems. The enterprise architecture frameworks are established templates for the development of enterprise architectures that aim to describe the enterprise structure, goals, processes and organization.
The CI protection process has the following steps:

- Identifying critical infrastructures essential for mission accomplishment.
- Determining the threats against those infrastructures by looking into business processes.
- Analyzing the vulnerabilities of the critical infrastructure.
- Assessing the risks of the degradation or loss of capability to achieve a mission.
- Defining countermeasures where risk is unacceptable.
- Defining security controls.

To help fulfill the steps it is necessary to identify critical business mission and then to conduct the classical steps of business process analysis, risk assessment and risk management.

2.2 The value of Enterprise Architecture framework

Enterprise Architectures (EA) are established approaches to developing descriptions of complex systems. An EA is often used to create a description of a structure and operations of an enterprise. An EA is a set of models that depicts how various business and technical elements work together as a whole. An Enterprise Architecture identifies the enterprise structure and gives a blueprint of its operation. The descriptions include multiple components, such as business drivers, principles, strategies, assets, technology and people. They also include selective views. Furthermore, an EA often addresses both current, future and interim states of an enterprise, including a transformation roadmap, change management process, program and portfolio context. EA describes the terminology, the composition of enterprise components, and their relationships with the external environment.
Enterprise Architecture needs a plan that defines a sequence of architecture states, also known as the transition architectures, that will change the existing ("as-is") EA to a desired target architecture. The EA roadmap is typically implemented through a number of projects, each delivering a solution architecture. These projects vary widely in scope and complexity.
Enterprise Architecture Frameworks (EAF) are common templates, and methods, for the development of instances of Enterprise Architecture (EA). They capture and represent all the relevant aspects of a business-driven enterprise. The known examples of such frameworks include the Department of Defence Architecture Framework (DODAF) used by the US DoD (*The DoDAF Architecture Framework Version 2.02*, n.d.), Federal Enterprise Architecture (FEA) used by the US federal agencies(*Federal Enterprise Architecture (FEA)*, n.d.). Zachman Framework (*Zachman Framework*, n.d.) and TOGAF (*The Open Group Architecture Framework 9 (TOGAF9)*, n.d.) can be considered enterprise architecture frameworks too. Figure 1 outlines the historical development of some major architecture frameworks.

2.3 Architecture Framework components

Typically, an Architecture Framework components include: views, meta-model, techniques, tools and guidelines for architecture development. It may also include best practices and design patterns. Other components can be used by architecture developers at their will.

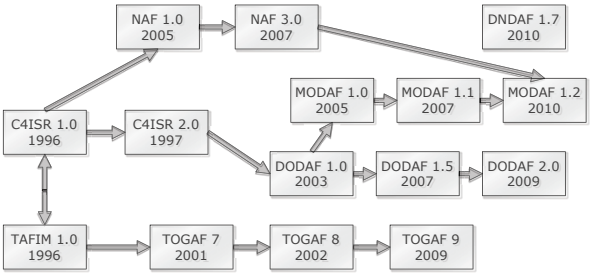

Fig. 1. Historical development of major Enterprise Architecture Frameworks

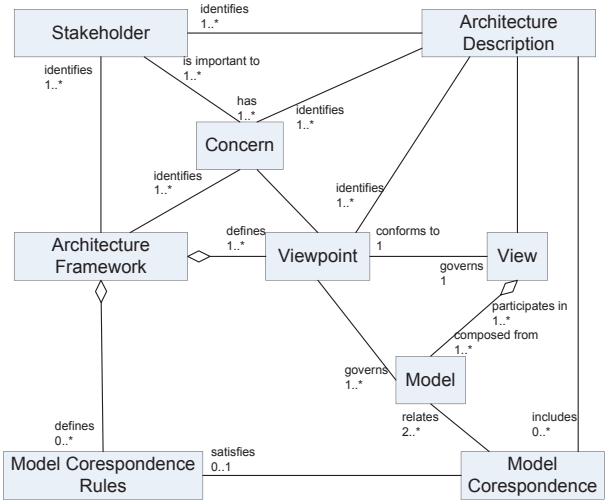

Fig. 2. Meta concepts used to describe architecture frameworks and their relationships

2.3.1 Viewpoints

Viewpoints are collections of useful views on the architecture data. They are selective distillations of complex architecture descriptions intended for the purpose of architecture presentation for different groups of architecture stakeholders. Viewpoints enable people to comprehend very complex systems. They limit the scope of presentations of various solutions to domain-specific issues, brought up by stakeholders involved in the architecture development. The views are concrete instances of the architecture data captured in a model. The aggregated architecture data that is represented in all the views from all the viewpoints could be considered a full composite model of the enterprise. This coherent set of architecture data is not always directly accessible as an artifact but is often scattered across different views. The model correspondence rules keep the composite model coherent and minimal.

2.3.2 Meta-model

Meta-model is a repository of terms that are used to represent the domain of interest. These are used during the AF development process to refer all the important aspects of the architecture. The meta-model serves, to a large extent, as common vocabulary. As such, it needs to be

complete, consistent, and minimal. Examples of meta-models are DM2 of DODAF2 and Content Framework from TOGAF.
A Meta-model is a critical element in an Enterprise Architecture Framework but it is even more important to also have a domain-specific Meta-model, which is understandable and widely accepted to all the stakeholders involved in the development and use of the Architecture Framework. Such an enhanced dictionary allows for a compact presentation of recommendations, procedures, methods and techniques used in a specific enterprise domain. There are some restrictions imposed on the integrity between the concepts. At various views (viewpoints) concepts are used to present some specific technique or procedure.

2.3.3 Techniques, guidelines, best practices and design patterns

The availability of viewpoints, consisting of multiple selective models and selective views of an enterprise, enables the incorporation of the domain specific knowledge into the framework. This is carried out with the use of the domain-specific concepts from the meta-model. During that activity, the set of models with predefined processes, enterprise design patterns (Wolthusen, 2004), and structures library are created. Security patterns are also reflected here. The typical scenarios such as security procedures are analysed and described with the use of framework concepts originating from meta-model and presented on the viewpoints defined in the framework.

2.3.4 The methodology and specific sub-methodologies

A generalized methodology is required for developing an enterprise architecture for critical infrastructures. It is required to produce such a description of a critical infrastructure architecture, so that different desctiptions be compatible and comparable.

2.3.5 Architecture framework's taxonomy

The architecture frameworks differ in type and scope. To understand better a given architecture framework, a hierarchical categorization is shown in Figure 3. The five levels position the domain-specific Architecture Framework between an organization-specific AF and the generic AF. Hierarchical layout inherits all the features of a generic Architecture Framework, but it also accommodates the methods that allow addressing systematically domain-specific concerns. 'Domain specific' means that terminology is typical of the particular domain interest - here: critical infrastructures. Typical processes and procedures found in CI and CII are identified, and the catalogue of good practices and guidelines is compiled together with common questions and their possible solutions.
There are five levels of AFs. The higher level the more abstract the AF is:

- Product Architecture (PA), often referred to as a solution Architecture, defines the architecture of a single product. It is usually developed for a single solution only and it is a natural way of product development.
- Organization-specific Architecture Framework (OAF) standardizes architecture description within a single enterprise, which allows it to have harmonic approach to development of multiple products and to maintain all the process in a uniform way.
- A Domain-Specific Architecture Framework (DAF) adds concepts which are common across similar organizations of a domain, and serves as a reference for OEM-specific architecture frameworks (e.g. PSAF). This approach specifically allows multiple organizations to cooperate easier together.

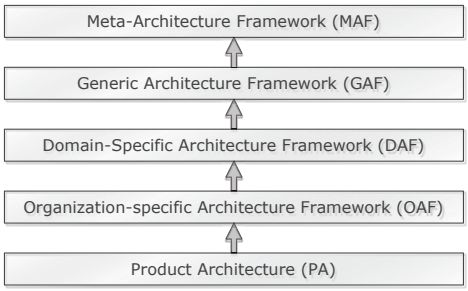

Fig. 3. Architecture Frameworks hierarchy

- The Generic Architecture Framework (GAF) defines concepts that are domain-independent (for example DODAF and TOGAF) and are universal enough to cover all the typical issues related to architecture development.
- The Meta-Architecture Framework (MAF) is independent of any type of system development. It provides a conceptual infrastructure to define and to reason about architecture frameworks. The major effort in this area is ISO standard 42010:2007 with extensions under development related to Architecture Frameworks.

The architecture framework for securing aeronautic communication can be considered as a domain specific architecture framework. It is not related with any particular product or organization, nor it is a general purpose framework or a meta-framework. The scope of this AF is the security in an aeronautical environment, and the purpose of building the architecture with this AF is to study and to reason on the protection of the communication infrastructure.

2.4 Existing Enterprise Architecture frameworks

2.4.1 Department of Defence Architecture Framework

The Department of Defense Architecture Framework (DoDAF) has been defined to serve as the structure for organizing architecture concepts, principles, assumptions, and terminology about operations and technology into meaningful patterns to satisfy specific DoD purposes. DoDAF version 2.0 focuses on architectural data, rather than on developing individual products as described in previous versions (DODAF 1.0, DODAF 1.5). The data-centric approach for architecture description allows for data re-use within and across different projects and over project life-cycles. It also supports flexible reporting and integration with other enterprise information systems. Finally strict representation of architecture data and their physical representation supports data sharing among multiple vendor tools for architecture development.

2.4.2 DoDAF data model

The architecture data in a described architecture is defined according to the DoDAF Meta-model (DM2) concepts, associations, and attributes. The terminology is used in similar way as defined in ISO 42010 and extensions regarding the architecture frameworks which are currently under development in ISO. The structure of architecture data is defined and constrained by DM2 data model. Figure 4 presents some selected example concepts from the DM2 Conceptual Data Model and their relationships. The concepts such as the Information, the Activity and the Performer are shown along with relationships between them.

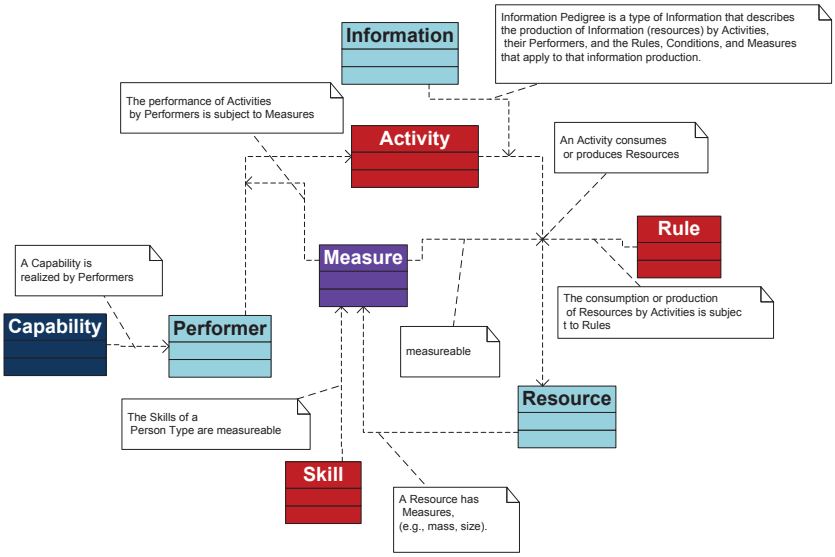

Fig. 4. Sample concepts and their relationships originating from DoDAF Data Model DM2

CDM consists of 26 concepts with their relationships.

2.4.3 DoDAF views, fit-to-purpose views

Views in DODAF v. 2.0 can be regarded as queries to the underlying architecture data. This is contrary to legacy versions of DODAF (1.5, 1.0) that were product (view) oriented. In DODAF2.0 the focal point is the underlying data - not the views. All relationships between objects are already contained in the architecture data, while views are only kind of queries. The core of DoDAF v. 2.0 is a data-centric approach where the creation of architectures to support decision-making is secondary to the collection, storage, and maintenance of data needed for efficient and effective decisions. The architect and stakeholders select views to ensure that architectures will explain current and future states of the process or activity under review. Selecting architectural views carefully ensures that the views adequately explain the requirements and proposed solution in ways that will enhance audience understanding. DODAF, similarly to MODAF and NAF, defines a collection of viewpoints:

- All Viewpoint (AV) - describes the overarching aspects of architecture context that relate to all viewpoints,
- Capability Viewpoint (CV) - articulates the capability requirements, the delivery timing, and the deployed capability,
- Data and Information Viewpoint (DIV) - shows data relationships and alignment structures in the architecture content,
- Operational Viewpoint (OV) - includes the operational scenarios, activities, and requirements that support capabilities,
- Project Viewpoint (PV) - describes the relationships between operational and capability requirements and the various projects being implemented,

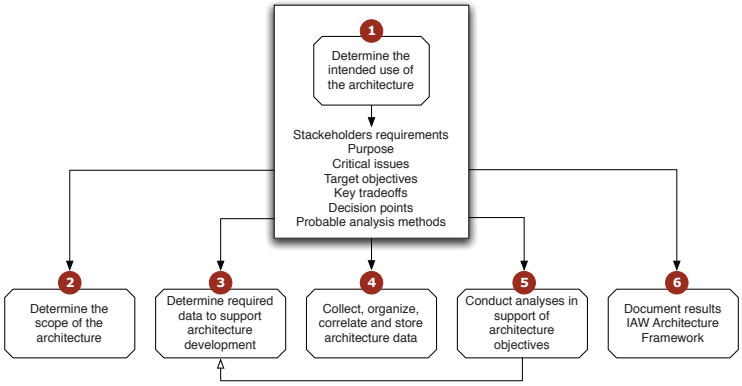

Fig. 5. DoDAF 2.0 Six-Step Process of Building an Architecture Description (Source: DoD Architecture Framework, Version 2.0, Volume 1, Section 2.1)

- Services Viewpoint (SvcV) - is the design for solutions articulating the Performers, Activities, Services, and their exchanges, providing for or supporting operational and capability functions,
- Standard Viewpoint (StdV) - addresses the policies, standards and sector recommendations across architecture,
- Systems Viewpoint (SV) - the design for solutions articulating the systems, their composition, interconnectivity, and context providing for or supporting operational and capability functions.

2.4.4 DoDAF methodology

DODAF 2.0 introduces a 6-step general methodology for architects to follow. The major steps are presented in Fig. 5.

2.4.5 The advantages of DODAF 2.0

- DM2 is based on the experience and the analysis of existing frameworks and methodologies. It is compliant with ISO/IEC 42010:2007.
- DODAF 2.0 is data-centric as contrary to product (artefact) centric. Accent is put on architecture data: methods for collecting, managing and exchanging all data related to created architecture. DM2 meta-model provides a common factor for all the products created while process. Consistent meta-model (DM2) is defined down to physical layer.

2.5 Generic EAFs - The Open Group Architecture Framework (TOGAF)

The Open Group Architecture Framework (TOGAF) is an example of a Generic Architecture Framework (GAF). It provides a user with a comprehensive approach to design, planning, implementation and governance of an enterprise architecture. TOGAF is developed and maintained by the Architecture Forum of The Open Group. It was originally developed in the mid-1990's, and has continuously evolved since then. The recent revision, TOGAF 9, can be downloaded from the web site of The Open Group. This section is descended from the

TOGAF specification and in many cases it's required to refer it for more detailed description of some particular issue.
TOGAF is a high level and holistic approach to design, which is typically modelled at four domains:

- Business Architecture - defines the business strategy, governance, organization, and key business processes.
- Data Architecture - describes the structure of an organization's logical and physical data assets and data management resources.
- Application Architecture - provides a blueprint for the individual application systems to be deployed, their interactions and their relationships to the core business processes of the organization.
- Technology Architecture - describes the logical software and hardware capabilities required to support the deployment of business, data and application services. This includes IT infrastructure, middleware, networks, communications, processing, standards, and so on.

TOGAF is comprised of six parts:

- Architecture Development Method (ADM) - an iterative sequence of steps to develop an enterprise-wide architecture.
- ADM Guidelines & Techniques - guidelines and techniques to support the application of the ADM.
- Architecture Content Framework - a detailed model of architectural work products, including deliverables, artifacts within deliverables and the Architecture Building Blocks (ABBs) that deliverables represent.
- Enterprise Continuum - a model for structuring a virtual repository and methods for classifying architecture and solution artifacts.
- Reference Models - the TOGAF Technical Reference Model and the Integrated Information Infrastructure Model.
- Architecture Capability Framework - a structured definition of the organizations, skills, roles and responsibilities to establish and operate an Enterprise Architecture.

2.5.1 TOGAF Architecture Development Method

The Architecture Development Method (ADM) is central to TOGAF. The ADM explains how to derive an organization-specific enterprise architecture that addresses business requirements. It includes establishing an architecture framework, developing architecture content, transitioning, and governing the realization of architectures. All of these activities are carried out within an iterative cycle of continuous architecture definition and realization that allows organizations to transform their enterprises in a controlled manner in response to business goals and opportunities. Structured as a series of nine phases plus requirements management phase that interacts with each of the nine phases throughout the ADM lifecycle, the ADM becomes an iterative method, over the whole process, between phases and within phases. Additionally, TOGAF, like most of the Architecture Frameworks, may introduce some new paradigms in the run of the architecture development process.
The phases within the ADM are:

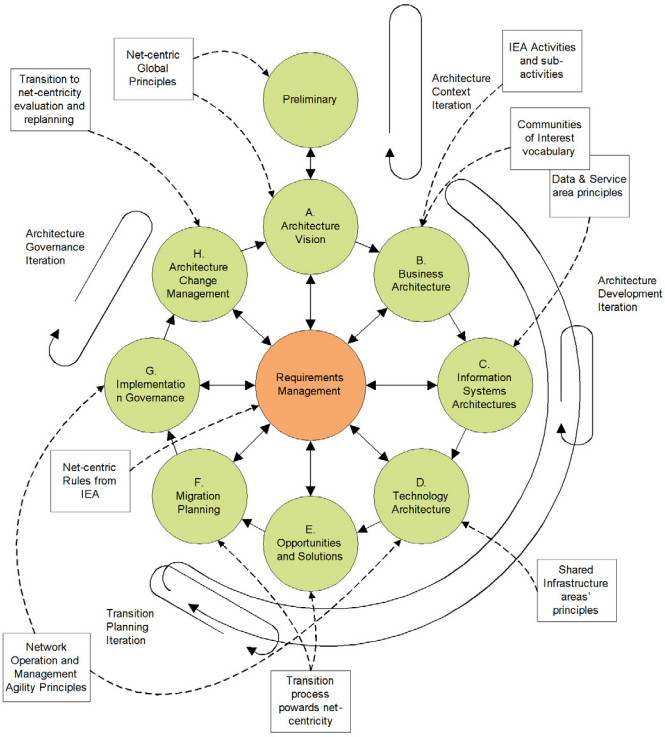

Fig. 6. The Architecture Development Method cycle

- The Preliminary Phase describes the preparation and initiation activities required to meet the business directive for a new enterprise architecture, including the definition of an Organization-Specific Architecture framework and the definition of principles.
- Phase A: Architecture Vision describes the initial phase of an architecture development cycle. It includes information about defining the scope, identifying the stakeholders, creating the Architecture Vision, and obtaining approvals.
- Phase B: Business Architecture describes the development of a Business Architecture to support an agreed Architecture Vision.
- Phase C: Information Systems Architectures describes the development of Information Systems Architectures for an architecture project, including the development of Data and Application Architectures.
- Phase D: Technology Architecture describes the development of the Technology Architecture for an architecture project.
- Phase E: Opportunities&Solutions conducts initial implementation planning and the identification of delivery vehicles for the architecture defined in the previous phases.
- Phase F: Migration Planning addresses the formulation of a set of detailed sequence of transition architectures with a supporting Implementation and Migration Plan.

- Phase G: Implementation Governance provides an architectural oversight of the implementation.
- Phase H: Architecture Change Management establishes procedures for managing change to the new architecture.
- Requirements Management examines the process of managing architecture requirements throughout the ADM.

There are four main iterations within ADM: Architecture Context iterations that allow initial mobilization of architecture activity by establishing the architecture approach, principles, scope, and vision. Architecture Definition iterations allow for the creation of architecture content by cycling through the Business, Information Systems, and Technology Architecture phases. These iterations also allow viability and feasibility tests to be carried out by looking at 'opportunities and migration planning'. Transition Planning iterations support the creation of formal change roadmaps for the defined architecture. Architecture Governance iterations support governance of change towards the defined Target Architecture.

2.6 How EAF reflects in security?

While Enterprise Architecture Frameworks and Enterprise Architectures are abundant - due to their wide use in enterprise governance - reports of their applicability in the domain of enterprise security are not common. In that particular domain there is currently a fast growing interest in information security management and information assurance. To this end, (i) the Sherwood Applied Business Security Architecture (SABSA) from SABSA Limited, (ii) Control Objectives for Information and related Technology (COBIT) from ISACA, and (iii) Information Technology Infrastructure Library (ITIL) from UK Office of Government and Commerce are currently the leading methodologies in information security and assurance, each focusing on similar problems, but emphasizing and addressing specific questions differently. Each strives to give detailed descriptions of a number of important practices and provides comprehensive checklists, tasks and procedures that an organization can tailor to its needs.

The Sherwood Applied Business Security Architecture (SABSA) is a model and a methodology for developing Enterprise Information Security Architectures and for designing security infrastructure solutions that support business needs (Sherwood et al., 2005).

SABSA relies on a model, which consists of a 6x6 SABSA matrix, where rows represent layers of the architecture model (contextual, conceptual, logical, physical, component and operational), while the columns represent the six major concerns in architecture modeling: assets, motivation, process, people, location and performance (Stango et al., 2011). These concerns are representations of Zachman's six communications questions, respectively: what, why, how, who, where, and when (*Zachman Framework*, n.d.). The topmost contextual layer allows to capture the context necessary to understand the requirements and the business attributes that shape the security required. The conceptual layer defines security concepts, principles and management procedures. In the logical layer, security controls are designed and the management procedures are specified. The logical security services are covered at the SABSA physical layer, in terms of physical security mechanisms. The component layer is related with the selection of products and technology. Finally, the operational layer is concerned with classical system operations work (Stango et al., 2011). The process of developing enterprise security architecture consists in populating the SABSA matrix by following SABSA workflows.

By comprising a number of models, frameworks, methods and high-level processes, SABSA allows to develop risk-driven enterprise information security and information assurance architecture. It can be used for the development of architectures at any level of granularity of scope. Being an open standard, it may be used in industry sector and in private and public organizations. To some extent SABSA is unique by being a risk-driven, enterprise information security and information assurance architecture. The layers of the architecture model and partitioning of concerns allows for two-way traceability of the architecture artifacts, thus allowing to check the architecture development product for completeness, to make sure that every business concern has been properly handled, and that the associated security requirements have been tackled with enough attention. SABSA also provides reverse traceability for business justification. It allows any architecture decision to be linked back to the original business requirements.

2.7 Common approaches to security

The IT industry has developed sets of standards which address security management.

2.7.1 ISO Standards

ISO/IEC 27000-series security standards are probably among the most commonly security standard deployed in enterprises worldwide. The series provides best practice recommendations on information security management, risks and controls within the context of an overall Information Security Management System (ISMS). The documentation structure and design are similar in design to management systems for quality assurance (the ISO 9000 series) and environmental protection (the ISO 14000 series) which were developed earlier. The current state of standard evolved from BS 7799 from 1995 published by BSI Group. The initial document was written by the United Kingdom Government's Department of Trade and Industry (DTI) and was composed of several parts. Currently, the set of ISO 27000 series standards include documents numbered 27000-27006, 27011, 27031, 27033-1 and more. Some selected ones are described below.

2.7.1.1 ISO 27002

ISO/IEC 27002 basically outlines controls and control mechanisms, which may be implemented subject to the guidance provided within ISO 27001 described below. The standard "established guidelines and general principles for initiating, implementing, maintaining, and improving information security management within an organization". The actual controls listed in the standard are intended to address the specific requirements identified via a formal risk assessment described with more details in ISO 27005. The standard is also intended to provide a guide for the development of "organizational security standards and effective security management practices and to help build confidence in inter-organizational activities".

2.7.1.2 ISO 27001

The significant reverse in numbering between BSI standards and ISO standards reflects the truth of the way the security awareness was developing. The increasing level of IT peril caused that simple in the beginning check-list were replaced with more complex and systematic approach. The new response to IT menace was the Part 2 of BS 7799 already mentioned which was adopted later in November 2005 by ISO and named ISO/IEC 27001.

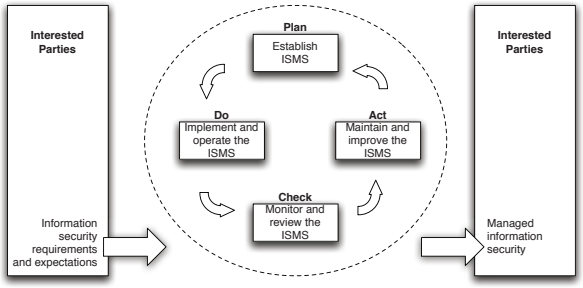

Fig. 7. Deming cycle applied to ISMS processes

The initial BS 7799 Part 2 (aka BS7799-2), entitled "Information Security Management Systems - Specification with guidance for use." was published by BSI in 1999. The main focus of BS 7799-2 was how to implement an Information security management system (ISMS). The document really brought a new quality to BS 7799-1 giving implementation guidelines to the information security management structure and controls identified in BS 7799-1. The most significant change of next revision form 2002 of BS 7799-2 is the Plan-Do-Check-Act (PDCA) (Deming quality assurance model), aligning it with quality standards such as ISO 9000. The phases or activities of the Deming cycle are:

- PLAN - Establishing the ISMS.
- DO - Operating the ISMS.
- CHECK - Monitoring and reviewing the ISMS.
- ACT - Improving the ISMS.

Even if periodical checks and proactive planning were present in some places of the previous versions BSI documents, its formal introduction brought significant change.
Other important aspects which are defined in ISO 27001 is the responsibility of the management for the ISMS, placing risk management as integral part or standard ISMS operation and economic justification of security-related expenses .

2.7.1.3 ISO 27005

The ISO/IEC 27005 standard, published in 2008, provides guidelines for information security risk management. It supports the general concepts specified in ISO/IEC 27001. It is designed to assist the implementation of information security based on a risk management approach. It does not specify, recommend or even name any specific risk analysis method. Its value is in specifying a structured, systematic and rigorous process from analysing risks to creating the risk treatment plan.

2.7.2 FISMA/NIST

The next important set of standards and best practices comes from USA. The Federal Information Security Management Act of 2002 called "FISMA" requires each federal agency to develop, document, and implement an agency-wide program to provide information security for the information and information systems that support the operations and assets of the agency, including those provided or managed by another agency, contractor, or other source.(*Federal Information Security Management Act (FISMA) Implementation Project*, n.d.)

FISMA similarly as described above ISO standards looks to the financial justification of the security means and explicitly emphasizes a "risk-based policy for cost-effective security."
FISMA requires that agencies have an information systems inventory in place. The head of each agency shall develop and maintain an inventory of major information systems, including major national security systems, operated by or under the control of such agency. The guidance on determining system boundaries can be found in NIST SP 800-18, Rev. 1, Guide for Developing Security Plans for Federal Information Systems.

2.7.3 Categorize information and information systems according to risk level

According to FISMA, all information and information systems should be categorized according to the objectives of providing appropriate levels of information security according to a range of risk levels. The definitions of security categories are defined in the mandatory security standard FIPS PUB 199 "Standards for Security Categorization of Federal Information and Information Systems". The more detailed practical guidelines are provided by NIST SP 800-60 "Guide for Mapping Types of Information and Information Systems to Security Categories".

2.7.3.1 Security controls

FISMA requires that all federal information systems must meet the minimum security requirements, defined in the mandatory security standard FIPS 200 "Minimum Security Requirements for Federal Information and Information Systems". NIST Special Publication SP 800-53, "Recommended Security Controls for Federal Information Systems" defines the minimum security requirements which have to be met by organization by selecting the appropriate security controls and assurance requirements. The process of selecting the security controls to achieve adequate security is a multifaceted, risk-based activity involving management and operational personnel within the organization. Agencies have some choice in application the baseline security controls in accordance with the tailoring guidance. This allows agencies to adjust the security controls to more closely fit their mission requirements and operational environments. Practical implementation help guiding through the process can be found in NIST SP 800-53A "Guide for Assessing the Security Controls in Federal Information Systems and Organizations, Building Effective Security Assessment Plans". The controls selected or planned must be documented in the System Security Plan.

2.7.3.2 System security plan

Agencies are obliged to develop policy on the system security planning process. NIST SP 800-18 introduces the concept of a System Security Plan - a collection of living documents that require periodic review, modification, and plans of action and milestones for implementing security controls. Procedures should be in place outlining who reviews the plans, keeps the plan current, and follows up on planned security controls.
Without having the System Security Plan a proper security certification and accreditation process for the system is impossible.

2.7.3.3 Risk assessment

FIPS 200 along with NIST SP 800-53 requires a foundational level of security for all federal information and information systems. The agency's risk assessment validates the security control set and determines if any additional controls are needed to protect agency operations.

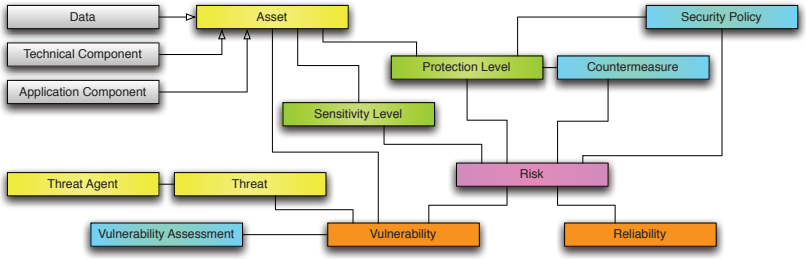

Fig. 8. Risk Analysis Model

2.7.3.4 Certification and accreditation

The certification and accreditation process is defined in NIST SP 800-37 "Guide for the Security Certification and Accreditation of Federal Information Systems". Security accreditation is the official management decision given by a senior agency official to authorize operation of an information system. The set of mandatory conditions which system must fulfill in order to receive it includes: proper system documentation, completed risk assessment and the review of system's controls to be functioning appropriately.

2.7.3.5 Security standards comparison

If we try to compare the approaches to security of the ISO 27k series and NIST Special Publications dealing with security we can see that they are influenced by each other. They are close to each other in their motivation and objectives, but differ in the primary focus, and in the level of details. The target reader is not the same, too, which causes some important changes in their use. The NIST standards are mandatory for all federal agencies by law so they are limiting choice of an agency where ISO series mentions only possibilities as addressed to a business not bound by legal regulations. Other difference is caused by the budgeting. The level of details and structural organization of the NIST series results in noticeably higher level of details, and quality. Also these documents reflect, to a larger than the ISO series, the new approaches and methods.

3. Risk Assessment

EA can be useful to describe the structure of the Enterprise and to map it to architectural and technical components able to fulfill the Enterprise goals. The security of the whole Information architecture, however, needs to be analyzed in a little more specific detail.
When analyzing the security of the CII, the model in Figure 8 should be used. In this model the prime component is the definition of Assets, as is the union of the Data, Technical (hardware and software) components and the Application Components. The difference between the latter two is the use of Data: whereas the Applications modify and use Data assets, the Technical components does not.
In this model every asset have some desired properties (i.e., its Protection and Sensitivity levels) and some inner properties (i.e., its Reliability and Vulnerability). The Sensitivity Level is defined by the importance of the asset for the operations of the CI, while the Protection level is defined and is the outcome of the Risk Analysis process. The sum of Countermeasures and Security Policies contribute to quantify this property. Thus, it is not an intrinsic but rather an

extrinsic property. The most interesting part, probably is the Vulnerability, as is the asset's resistance to faults due to attacks or misbehaving entities.
In the diagram are shown also the Security Policies and the Countermeasures. Both are actively contributing to define the Risk by lowering it to acceptable levels. The main differences between the two are that while the Security Policies are aimed at preventing a dangerous event, the Countermeasures have the target to react to an event and mitigate the consequences.
Furthermore, assets can be divided into *primary* and *secondary*, depending on their relative importance for the CI operations. According to the EA results, a set of scales based on the threat estimation and the vulnerability of each asset have to be set, and opportune policies (Security and Countermeasures) should be applied in order to lower the risks to acceptable levels (also to be defined in the EA).
A fundamental part of this process is thus the evaluation of the Threats and the Vulnerabilities.

3.1 Threat estimation

The Threat estimation is a very difficult part, as it involves assumptions on the Threats like the exploitability of a Vulnerability, the capabilities of an attacker and so on. It is extremely important to have a realistic Threat Model (Rescorla & Korver, 2003), otherwise the threat estimation might lead to over or underestimation. An extremely important point, however, is to never consider exploitation probability as dependent on the knowledge of the vulnerability itself. Always assume that the attacker have the maximum knowledge possible.

3.2 Vulnerability Assessment

Vulnerability Assessment (Thompson & Chase, 2005) is an integral part of the Risk Analysis process. Each Application or Technical Assets should pass some tests in order to measure their functionalities. The most common ones are the conformance tests, while vulnerability analysis tests are less used.
The conformance testing aims at verifying that the system is behaving correctly to a set of known inputs. It is a common practice to require a conformance against some protocol or some reference implementation in order to ensure interoperability. Vulnerability testing, on the contrary, aims at discover unexpected behaviours when the system have unexpected inputs. On a simpler scale one can consider a vulnerability test as a search for backdoors, possible bugs and so on. Since this kind of tests involve unknown variables, they are usually much more complex and costly than the conformance tests. Nevertheless it is imperative to adopt some sort of vulnerability assessment system, as those are exactly the kind of vulnerabilities an attacker could use to violate a system. At the moment the vulnerability testing is by far the most difficult and challenging part of an asset verification. Ni2S3 EU project (http://ni2s3.kt.agh.edu.pl) developed a methodology and a full framework for Vulnerability Assessment. The interested reader can check Ni2S3 outcomes and deliverables.

4. Vulnerabilities in IPv6 networks

Although EAF can (and should) be used to describe the structure of the Information systems and the network, when it comes to deploying a lot of problems might occur. Whilst most of them are "known", we think that IPv6 might be one of the major risks, mainly due to underestimation. In this section the main vulnerabilities in IPv6 networks are pointed out.

4.1 Differences between IPv4 and IPv6

We will assume that the reader is familiar enough with IPv6, so we will not describe IPv6 details. The goal here is to summarize the main points that are related to security.

The main and, probably, most known difference between IPv4 and IPv6 is the addressing space size. Although this is one of the key points, it is not at all the most important one. Among the new IPv6 features, there are some more interesting and under evaluated features that might be a concern for security.

In the pure spirit of the Internet of Things concept, IPv6 designers decided to push the auto-configuration features to the limit, allowing a near-seamless plug and play network model. This has been reached by defining and enforcing the concept of auto-configuration for all the relevant network layer features, from IP addresses acquisition to network knowledge (e.g., routers, gateway, DNS discovery).

4.1.1 IP addresses

The structure of the IPv6 address is described in (Hinden & Deering, 2006). An IPv6 address is logically divided into two main parts: a network part and a node address part. Each of them is (or can be) auto-configured.

Multicast and Anycast addresses are particularly important, as they play a major role in the security of the IP protocol. As one could expect, a multicast address is a one-to-many address. The relevant point is about the scope of this address. In IPv6 multicast can be restricted to having a local or global scope (or more complex scopes, not to be described here).

About anycast addresses, it is interesting to consider the definition (Hinden & Deering, 2006): *An IPv6 anycast address is an address that is assigned to more than one interface (typically belonging to different nodes), with the property that a packet sent to an anycast address is routed to the "nearest" interface having that address, according to the routing protocols' measure of distance.*

Multicast in IPv4 is a quite rarely used system. On the other hand in IPv6 it is used to contact routers, DNS, address configuration and so on. Anycast is a completely new addressing scheme in IPv6. Their importance is related to their use in network operations, as all the plug and play IPv6 features rely on them. Due to this, it is quite evident how a malicious user could leverage them to attack the network infrastructure and cause potential damages.

4.1.2 IP address acquisition

IPv4 defines various methods to get a valid network address. However, if two network elements try to use the same IP address, there is no automatic way to fix the conflict. In IPv6 the changes are radical about this point. First and foremost, the main thing to keep in mind is that a single interface has always more than one address, and they are all valid.

Any networked entity has at least one link-local IP address and one multicast address per interface. The first is algorithmically built and allows communications across the link (either a point-to-point or a switched network), the second is closely related to the first and is used to solve the possible conflicts between nodes that, by chance, might end up with the same address.

To clarify this, it is worth explaining how an address is built. There are a number of ways to build a valid address, but the most used are: 1) Auto-configuration, 2) DHCP, 3) Manual configuration. The first of them is probably the most used. It provides a number of ways to build a valid address. The first and probably the easiest way is to use the MAC address as part of the IPv6 address. This, however, have the drawback of identifying almost uniquely the

device, so its communications can be tracked by a malicious user in a quite easy way. Another approach is to randomly choose the address (MS Windows does that), however in this case the drawback is the total opposite: you can not identify the entity by its address anymore, and every time there is a network restart, the address will be different.
Another interesting way is to use the so-called Cryptographically Generated Address (CGA) (Aura, 2005; Bagnulo & Arkko, 2006), where part of the address is a hash of a public key. Although the use of CGA is very interesting and can be used in a number of ways, their processing does require an extra load in the routers. Hence, they can be successfully exploited to trigger fancy network attacks.
In any case, auto-configuration does require a protocol to solve possible conflicts among the addresses, and this is handled by the Neighbor Discovery protocols (Arkko et al., 2005; Narten et al., 2007). The discovery is based heavily on multicast, and a malicious user can use this as well in order to trigger a denial of service attack (see section 4.3.1).
Thus, in the auto-configuration case, each node is able to build a valid link-local address in the form of FE:80::[auto-configured 64 bits address]. In order to gain a global-scope address, i.e., an address valid for the global Internet, the entity shall contact a router through a well-known multicast group and receive a valid prefix. Again, this is a nifty but potentially dangerous feature.
DHCP approach is not different from the IPv4 one, but it can also be used by autoconfigured nodes to acquire the DNS address. DHCP as well can be a vulnerability, as it relies on a well-known multicast address.

4.1.3 IPv6 address scope

The last key point to keep in mind when dealing with IPv6 networks is the absence of Network Address Translators (NATs). Although NAT was originally proposed as a technique to delay the inevitable IPv4 address exhaustion, it quickly became a way to separate network segments and "hide" parts of it from the public internet. NATs, however, can be bypassed in a number of ways, some actually raising quite dangerous exploit possibilities (Huston, 2004). In IPv6 there is no need of NATs (the address space is large enough to use global addresses), even tough for some security or multihoming configurations a NAT could be still a viable solution (see (Thaler et al., 2010)).
The real point, however, is that all the IPv6 hosts can have a *global scope* address, thus exposing them to the traffic from the public internet. As a rule of thumb, the network planners/administrators should keep in mind that everywhere there was a NAT in IPv4, in IPv6 there should be a firewall. Moreover due to the global address use, it becomes imperative to implement Intrusion Protection (or Detection) Systems, so to quickly react to unexpected user's behaviours. It should be expected, as a matter of fact, an increased number of peer-to-peer systems and connections, not easily blocked by firewalls, and not at all unwanted per-se.

4.2 Vulnerabilities of the IPv6 infrastructure

In order to reduce the security to a manageable problem it's useful to divide it among its basic components The basic level of separation shall be between the vulnerabilities arising from the design and the ones related to the implementation.

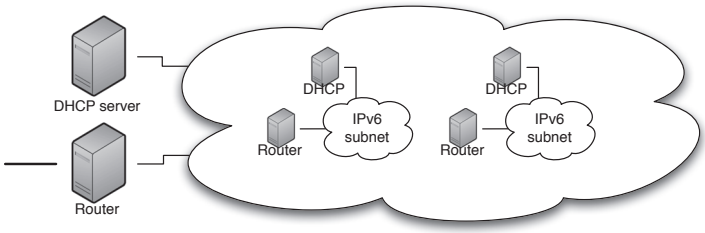

Fig. 9. Subnetwork hierarchy

4.2.1 Design flaws

The first and probably the most important aspect is about the network design. There is not a single way to build an IPv6 network and each design might lead to potential security issues. From a general point of view, there is no real difference between an IPv6 and an IPv4 network at LAN level. The main difference arises when we look at the larger "network", depicted in Figure 9. As we might notice, there is a hierarchy of routers and DHCPs servers. Also this architecture might not seem different from a classical IPv4 network, however in IPv4 most of the subnet routers would have been NATs and the local DHCPs administratively disjoint from the main DHCP server.

Due to the lack (or disappearance) of NAT, the role of network segmentation goes to routing and firewalling. On the other hand, the lack of NATs makes it central the role of firewalls. These have to be placed practically in every single router, and their configuration has to be consistent.

On the other hand, only a handful of firewall/routing configuration managers are able to properly handle IPv6 routing tables, and this can be quite a problem.

A second point is about the address configuration policies. Some network parts could need fixed addresses, while some other might need a more flexible auto-configuration mode. Due to the differences in address configuration, the firewall rules might be quite complex to setup.

Again about network obfuscation, some policies for the DNS have to be adopted. It is a well-defined policy in IPv4 to have reverse-address available by default, so that the DNS knows about all the possible network addresses. This is clearly impossible for IPv6 networks due to the address space. This poses a design problem. Should the DNS store all the numbers or just the relevant ones? It is out of the scope to give an answer, but it is reasonable to assume that the DNS should store the direct and reverse mapping only for the well-known addresses, as is the manually configured and fixed ones. The other ones should be left unmapped. It is of course possible to update the DNS in real-time coupling it with the DHCP (if any), but this does not seems to give any real benefit beside keeping this "fake" address existence.

Another design point is about address assignment delegation. In large networks it seems reasonable to divide the network in sub-networks, much like in IPv4. The existence of global scope address is based on the Router Advertisement process (i.e., a router will periodically or on-demand provide RA messages with the valid network prefix). A frequent design flaw is thus related to the decision about administratively separated domains, as is about the subnetwork topology. It is rather obvious that each network part should not have more

than one router answering with RAs.[1] Network design should carefully choose the size and topology of the subnets in order to avoid excessive signaling overhead for each subnetwork. A wrong design might expose the network to Denial of Service attacks or unexpected failures. About the DHCPs, it is rather obvious that their parameters should be consistent. IPv6 do admit the use of DHCP delegates, or slaves. In this case, as in the previous one, the signaling overhead should be minimized and checked.
Last but not least, there is the issue related to where and how to insert security probes in the network. While the previous design decisions where mainly related with the overhead analysis, as is "intrinsic" faults due to overheads, the probes should look for anomalies in the network. Thus, it is necessary to look for the main possible architecture attacks.

4.2.2 Architectural attacks

The main attacks possible aimed at disrupting the IPv6 architecture use "creatively" the plug and play IPv6 features. Unfortunately any decentralized or consensus system is somewhat subject to attacks, and IPv6 is not different.

- Router's advertisements can be forged, and a malicious user (or a malfunctioning unit) could inject in the network false or wrong RAs. The possible attacks range from Denial of Service to Man-in-the-Middle.
- DHCP architecture suffers the very same issue. A malicious user could impersonate the local DHCP and inject false data in the network. According to the address construction methods this could lead to different attack kinds.
- The last attack is related to the ND protocol (Narten et al., 2007). IPv6 nodes must, periodically, check for duplicates in the network. A malicious node can easily exploit this mechanism to implement a Denial of Service simply replying to all Duplicate Address Detection messages.

Those three kinds of attacks make it clear the importance of continuously monitoring the network in order to promptly discover attackers or malfunctioning nodes. It is also rather important to point out that also simply misconfigurations or malfunctioning nodes could exhibit the above behaviours. The network administrator shall not assume that the absence of attackers make the network immune from those issues. More insights on the specific attacks will be in section 4.3.

4.2.3 Implementation flaws

The kind of attacks toward IPv6 implementations are mostly arising from bugs or vulnerabilities in the IP or in the application stacks, and should, in theory, not be too difficult to find and eliminate. On the other hand there is a bad habit among the implementors, as is to give for granted that the underlying software is working as intended. If this might be true for IPv4 stacks, it is not anymore so for IPv6. As a matter of fact, IPv4 stacks have been in use for decades, so there are well-known and well-tested implementations that seldom are changed. On the contrary IPv6 has been around for decades as well, but with much less testing and use. Moreover, the additional functionalities in IPv6 like autoconfiguration, multicast, anycast, IPSec, etc. make the protocol quite more complex. Hence, it can be expected that some implementations might contain bugs or non-optimized code (the latter can lead to DoS).

[1] A special case is where there are double or triple exit points, or fallback routers, but this case is rather specific.

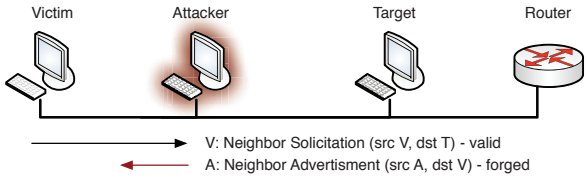

Fig. 10. Neighbor Discovery attacks

At L4 and above layers, as is TCP, UDP, Application Level Protocols, etc., things are not looking brighter. Theoretically IP stacks should be agnostic with respect to the IP version, with application level protocols "thinking" in terms of URIs and not bothering with actual IP numbers. In practice there are a number of cases where this is not possible or simply the programmers did violate this principle. A good rule of thumb is to never consider a properly working in IPv4 system (i.e., passing vulnerability and conformance testing under IPv4) as "safe" for IPv6. Tests should be re-evaluated and considered as independent.
In order to minimize the impact of implementation flaws, two main techniques should be considered: conformance testing and vulnerability testing.

4.3 IPv6 specific attacks

This section will summarize some new attacks specific to IPv6. The aim is not to be exhaustive, but to show what kind of threats can be expected reasonably in an IPv6 infrastructure. Moreover the attacks listed here are well known, so one can expect that an attacker will try them.

4.3.1 Address resolution attacks

In IPv6 the ARP protocl is substituted by Neighbor Discovery (ND) protocol (Narten et al., 2007). To resolve the IP - MAC address mapping, two new packets exists: ICMP6 Neighbor Solicitation / Neighbor Advertisement. The use is quite straightforward: if A needs B's MAC address it will send an NS using multicast "solicited-node" address (or unicast, depending on the link capabilities). B will reply with an unicast NA. Since anyone can reply to the NS message, an attacker can forge NA replies and claim to be either the victim or even all the machines in the network. The countermeasure is quite obvious, but it requires to monitor the network continuously. Moreover false positive could arise frequently, as it is quite difficult to find a generic rule to spot this kind of attack.
Another "flavor" of this attack is related to the IP assignment procedure. Any host, at boot time, will initiate a double Duplicate Address Discovery (DAD) procedure, in order to make sure no duplicate address is in the local network. This is particularly important in case of auto-assigned addresses, but it is used also in case of DHCP assigned addresses. The attacker can send an ICMP NS and pretend to be any host in the network. In this case the result is a Denial of Service to the victim, as it will not be able to acquire any IP address. The basic ND attacks scheme is depicted in Figure 10.
It should be noted that, since NA can be also be unsolicited, in case an interface changes its IP address, this kind of attack can be also target working machines, triggering unnecessary DAD procedures. If used on a whole network it can be quite dangerous.

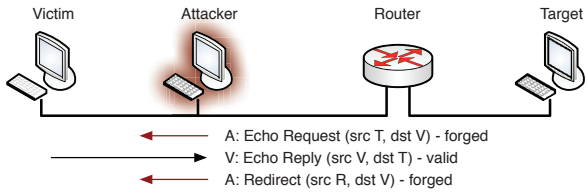

Fig. 11. ICMP Redirect attack

4.3.2 Router attacks

The attacks involving routers are strictly related to the previous set of attacks. In this case the attacker can use ICMP Router Advertisement in order to claim to be a router, or to inject in the network false prefixes. The result is different, as in the first case the attacker will become a router, thus allowing an easy man-in-the-middle. In the second case the network will be blocked. A different approach is to implant a bad route through creative use of ICMP redirect. This kind of attack is a bit trickier than the previous ones, but it is still quite simple. Figure 11 shows this attack.

A similar class of attacks involves the use of DHCP. As DHCP are in IPv6 replying to requests on a specific multicast address, also in this case an attacker can forge DHCP replies in order to inject in the network fake informations. In particular the DNS association is sensible, as one could point to an almost-valid DNS, meaning a DNS replying correctly to all the requests but some specific ones, thus redirecting only some specific addresses to a fake server.

This kind of attacks are quite dangerous and are well documented (see (Nikander et al., 2004)), but they have something in common: they all come from local network. There are some passive countermeasures, like SEND (SEcure ND, (Arkko et al., 2005)), but SEND use can not be considered as a definitive solution, as its applicability is not universal.

The best countermeasure to those attacks is for sure to secure the local network, not allowing an attacker to join the network. This can and should be done by disabling the physical unused ports and monitoring the network for unexpected behavior of the local hosts. An implementor should never consider the local network as secure, and always check for compromised hosts. An attacker gaining control of an host in the local network can easily block it.

4.3.3 Traffic amplification attacks

This class of attacks can be triggered either from local network or from remote network. They all involve triggering automatic replies to legitimate messages, like Echo Requests, to the specified victim, flooding its interfaces. Another target could be a specific link, in an attempt to consume all the available bandwidth. A quite handy way to do that involves using Routing headers in order to force a communication between two controlled hosts (even external to the attacked network) passing through one of the attacked routers. The two hosts can generate enough traffic to fill the available link bandwidth.

4.3.4 Fragmentation, mobile and tunnel attacks

One misconception in IPv6 is about fragmentation not existing anymore. It is true that fragmentation has been changed, but it is still possible to have fragmented packets.

In IPv6 the *routers* are not allowed anymore to fragment, but the source can still use it. Hence, fragmentation and reassembly is limited to the source and destination hosts and it is used

when the L4 layers does not (or can not) honor the discovered MTU size. An attacker can use this in order to mask a malicious routing header, hence it is imperative to defragment the packets in a firewall in order to inspect the real packet content and block dangerous communications.
About Mobile IPv6 (see (Johnson et al., 2004)) attacks, the protocol itself is to be considered secure, as it involves a massive use of IPSec in order to protect the communications. On the other hand all implementations have an option to disable IPSec. The implementor should carefully check to reject non secure communications, as any unprotected Mobile IPv6 link can be easily redirected to a different destination.
Last but not least, tunnel attacks do involve injecting packets in a IPv6 over IPv4 link. Also in this case the tunnel should be protected with proper cyphering, otherwise an attacker can inject packets easily just guessing the two tunnel endpoints.

4.3.5 Dual stack attacks

Attacks involving dual stack hosts (i.e., hosts with both IPv4 and IPv6) are quite common, but not so different from the other ones presented before. They have been separated mainly because they do involve network design and management in order to be coped with.
The point is that IPv6 and IPv4 infrastructure could be different due to NAT use in IPv4. Hence, the firewalls could be placed in different points or have different setups. This of course should be avoided as much as possible, but the problem of keeping firewall rules consistent between the two domains still remains. The network administrator should always keep in mind that the attacker could use a double stack attack (i.e., part of the attack on IPv4 and part on IPv6) in order to gain access to a victim host. Hence, any security solution should consider as a priority to keep consistent rules in both domains.
On a different perspective, it should be kept in mind that most O.S. have dual stacks already implemented and IPv6 ready to run. The Administrators should disable the unwanted stack or, at least, make sure that the wanted one have precedence. On this point, it is worth pointing out that IPv6 is normally the preferred stack by default. This last point could not be the case for aeronautical networks, where IPv6 use is mandatory, but it could be still a problem for part of the network in the airport. As a general rule IPv4 traffic should not be allowed outside limited areas (e.g., public internet areas). Hosts should not be dual stack enabled unless necessary and appropriate synchronized security policies should be used.

4.3.6 Network discovery attacks

The last consideration concerning peculiarities about IPv6 is the network discovery procedure. In IPv4 an attacker had a number of ways to discover the network structure, the most known being network scanning through a brute-force search for assigned IP numbers. This was feasible in IPv4 since the number of hosts in a subnet is typically quite limited, so it was possible to use pings or port scanning on all the IP numbers of a LAN.
In IPv6 this is simply not feasible, as the number of hosts to scan for is typically extremely large. A “normal” IPv6 subnet is a /64, meaning the attacker would had to scan about 2^{64} hosts (more than 18 millions of millions). This means that a brute-force discovery would take around 500 millions of years. Using clever techniques one could lower the time to some months, but it is still clearly not a feasible attack. This point, however, should not give a sense of false security, as the attacker will probably try to recover the needed informations from public-available sources: the DNS.

The network administrators and security planners should always keep in mind that the attackers will target:

- Public hosts, discovered through google, DNS, etc. and Anycast address hosts.
- All standard services hosts (DHCP, routers, time servers, etc.) through local multicast addresses.
- All hosts though local multicast (can be still time consuming but is feasible).

The most interesting point is about the DNS. It is expected that DNS servers will be one of the major source of informations, hence the public DNS should never contain any detail about the "internal" hosts. Even tough internal hosts could have names and globally valid IP addresses, their presence should not be made public unless really necessary. Moreover whatever does not need a name resolution but only an IP address should not, from a security perspective, have a canonical name. This will increase network obfuscation and will make it more difficult for the attacker to find the network structure.

5. AAA architectures

AAA stands for Authentication, Authorization and Accounting and is an important area in commercial telecommunication networks. There are, however, some misconceptions about AAA that should be considered, as its application is not consistent in all networks, nor it is equally considered important. Especially in the Internet community AAA systems are seldom considered as an important part of the network.
First we should point out the differences between each 'A'. *Accounting* is generally confused with billing, thus omitted in the networks where there is no need for it. Nevertheless, Accounting is mainly responsible for resource usage tracking, which can be used for a number of other purposes like network planning and resource optimization. The second set of mistakes is around the confusion between Authentication and Authorization. Keeping things simple, *Authentication* is the process of validating an entity identity. *Authorization*, on the other hand, is about the verification of the entity rights to use a specified resource. As an example, consider to have to use a network printer. The printer could ask an Authentication (e.g., ID and password), then it could check through Authorization if you can use all its features or just a subset (e.g., print in color or just black and white) and finally it could log the resource usage (number of pages printed) through Accounting. The very same model can (and should) be applied on network use, differentiating access privileges (e.g., the rights to use all Internet ports, firewall setups and so on).
IEEE 802.1x (see Figure 12) is one of the most widely used AAA frameworks, at least in the Internet. We can identify three actors:

- the *Supplicant*: the entity needing an Authentication.
- the *Authenticator*: the entity the Supplicant is authenticating with.
- the *Authentication Server*: the entity actually performing the Authentication.

It is worth noticing that the Authenticator and the Authentication Server are different functions, and must be separate hosts for security purposes. As a matter of fact the Authenticator needs the Supplicant to be authenticated, but does not require to know anything about the Supplicant for real. All it needs to know is if the Supplicant has been authenticated and if it has the rights to access a particular resource.

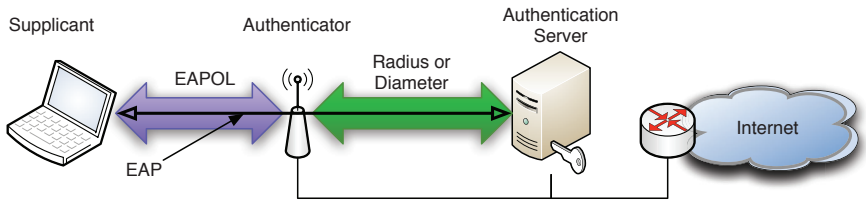

Fig. 12. 802.1x schematic architecture

The process is rather simple. The Supplicant needs to send a set of credentials to the Authenticator. The credentials are contained in an EAP (Aboba et al., 2008) envelope. EAP messages are exchanged between the Supplicant and Authenticator through the EAPOL protocol. The Authenticator will forward these to the Authentication Server using either RADIUS (Congdon et al., 2003; Rigney et al., 2000) or DIAMETER (Calhoun et al., 2003) protocols. The Authentication Server will send back to the Authenticator the keys needed to start a proper secure channel. EAPOL is not limited to the IEEE 802.11, but it can be used in IEEE 802.3 (Ethernet) and other kinds of media. The Authenticator can be any entity requiring an authentication / authorization from the user in order to use its resources.

The architecture, per-se, is not so complex. On the other hand, from a security perspective, a number of aspect must be take into account. It is fairly clear that the Authenticator does not need to know any information about the Supplicant identity, and should not either. On the other hand the Authenticator will have access to the EAP messages exchanged between the Supplicant and the Authentication Server. Hence, it is imperative that these messages are secure, meaning a compromised Authenticator must not be able to discover the Supplicant authentication informations. This is particularly important, as EAP itself defines a number of methods (about 40) to identify the Supplicant and/or the Authentication Server. Some of these methods are less reliable than the others, allowing man-in-the-middle or password discovery though rainbow tables (e.g., MSCHAP). Unfortunately there is not a clear and simple solution about which EAP method to use, as the various devices can have different supported methods. The only suggestion in this case is to make sure to disable the unwanted methods (i.e., the ones considered too weak) in the Authentication Server, and to prefer methods based on challenges and/or hardware, like EAP-AKA.

The second point to be taken into account is about the Authentication Server itself. It seems worth noticing that this network element is for real the most important one to be secured. Since it is the core of all the authentication and authorization process, no other services should run on the host in order to minimize the risk of a successful attack on the server. Moreover its architecture should be redundant and failsafe.

AAA protocol requirements have been defined in (Aboba et al., 2000). The basic requirements are: (1) Scalability, (2) Fail-over, (3) Mutual authentication between the client and the server, (4) Transmission level security, (5) Data object Confidentiality, (6) Data object Integrity, (7) Certificate transport, (8) Reliable AAA transport mechanism, (9) Run Over IPv4 and IPv6, (10) Auditability, (11) Ability to carry service-specific attributes. About this, it is necessary to outline the two architecture of RADIUS and DIAMETER, as they are quite different, although similar.

5.1 RADIUS protocol and architecture

The RADIUS protocol was created in 1997 with Dial-In users as primary target. The protocol run over UDP and defines a set of messages to authenticate the user: *Access-Request*, *Access-Accept*, *Access-Reject* and *Access-Challenge*. The Access-Challenge is used to request additional information to the Client in order to complete the authentication. The authentication information are included in a list of Attribute-Value pairs. The protocol itself defines some of them, but there is the capability to define vendor specific ones. On the other hand the Attribute is coded in a single byte, so only 255 attributes are possible.

RADIUS defines also an Accounting mechanism (Rigney, 2000), moreover it is possible to define roaming policies by using *Realms*. A Realm is defined by prepending or appending a Realm information to the user identity, e.g., `userid@company.com` or `company.com\userid`. Upon receiving a request, a RADIUS Proxy will compare the realm information with a table and will forward the request to an appropriate RADIUS server. It is worth noticing that the realm is a simple test string and does not need to pair with any actual real internet domain.

From a security point of view, RADIUS shows some weakness. First and foremost, the cyphering method used in messages is quite weak (MD5), so all the connections must be tunneled via stronger methods, like IPSec. The second point is about authentication. There is no mutual authentication between the client and the server. Hence, the client must "trust" the server. While this might as well be solved through secure tunnels, it is still a weakness. The third point is concerning UDP. Since the transport layer is unreliable, complex timeouts are necessary, and there is no guarantee that a request is being processed. Last but not least, RADIUS does not supply any "easy" method to provide failover or redundancy. Hence, it have scalability issues.

Despite the above mentioned points, RADIUS is one of the most used protocols for AAA in Internet, with worldwide AAA federations (e.g., eduroam (*Eduroam - Education Roaming*, 2011)) serving users all around the world. Moreover, it is a well-supported system, with a good user-base and many implementations, thus providing some reliability, at least for what concerns the limitations and implementation / deployment knowledge.

5.2 DIAMETER protocol and architecture

DIAMETER has been developed in 1998 to overcome the limitations of RADIUS. The main difference is the use of TCP and SCTP instead of UDP. DIAMETER has been chosen by 3rd Generation Partnership Project (3GPP) as the AAA protocol for IP Multimedia Subsystem (IMS) (3GPP, 2011), and should be expected to be the reference protocol in future AAA IP systems.

DIAMETER supports application-layer acknowledgements, and defines failover algorithms and the associated state machine. It makes IPSec mandatory, thus enforcing transport-level security, and defines explicit roles for Agents (Proxies, Redirects and Relays). Moreover it defines a correct framework to support Capability negotiation and Auditability, among a number of other features. Last but not least, the attributes space has been increased in order to support a greater number of vendor-specific data.

Despite the obvious superiority of DIAMETER, its use in Internet is still a niche, with IMS implementations being the only real test case. This is probably due to the lack of support by clients and the lack of free and supported servers, with only a handful of exceptions. Nevertheless, from a general point of view, it is fairly clear that the RADIUS weaknesses

should suggest to work toward its substitution, so any implementor should consider RADIUS only as a temporary solution.

6. Conclusions

Aeronautic communication can be treated as a kind of critical infrastructure and as an enterprise. In order to enable the protection of this infrastructure it is mandatory to identify the enterprise mission, the organization structure, critical business processes and relationships inside and outside. The more complete picture of the enterprise the better, because the context and interdependencies need to be properly reflected in enterprise description. The description, will have an impact on the infrastructure security planning and deployment. There exist known means to prepare enterprise descriptions - these are enterprise architectures. The security planning and deployment should be also based on a deep knowledge of the threats, the threats model and, most important, of the possible vulnerabilities an attacker can exploit. As integral part of the risk analysis process, vulnerability assessment tools should always be used, especially for IPv6 architectures. Last but not least, the implementors should always take particular care about the new IPv6 features, as the most common risk is to underestimate the changes in IPv4 to IPv6 transition.

7. Acknowledgments

The research leading to these results has been partially funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement n° 233679. The SANDRA project is a Large Scale Integrating Project for the FP7 Topic AAT.2008.4.4.2 (Integrated approach to network centric aircraft communications for global aircraft operations). The project has 31 partners and started on 1st October 2009.
The work presented in this chapter conveys partial results from the FP7 NI2S3 Collaborative Project (FP7-ICT-SEC-2007-1, contract 225488) carried out within the Security and Critical Infrastructure Protection area managed by DG Enterprise & Industry during years 2009-2011.

8. References

3GPP (2011). TS 23.228 - IP Multimedia Subsystem (IMS); Stage 2.

Aboba, B., Calhoun, P., Glass, S., Hiller, T., McCann, P., Shiino, H., Zorn, G., Dommety, G., C.Perkin, B.Pati, D.Mitto, S.Mannin, M.Beadle, P.Wals, X.Che, S.Sivalingham, A.Hamee, M.Munso, S.Jacob, B.Li, B.Hirschman, R.Hsu, Y.Xu, E.Campell, S.Baba & E.Jaques (2000). Criteria for Evaluating AAA Protocols for Network Access, RFC 2989.

Aboba, B., Simon, D. & Eronen, P. (2008). Extensible Authentication Protocol (EAP) Key Management Framework, RFC 5247.

Arkko, J., Kempf, J., Zill, B. & Nikander, P. (2005). SEcure Neighbor Discovery (SEND), RFC 3971.

Aura, T. (2005). Cryptographically Generated Addresses (CGA), RFC 3972.

Bagnulo, M. & Arkko, J. (2006). Cryptographically Generated Addresses (CGA) Extension Field Format, RFC 4581.

Calhoun, P., Loughney, J., Guttman, E., Zorn, G. & Arkko, J. (2003). Diameter Base Protocol, RFC 3588.

Congdon, P., Aboba, B., Smith, A., Zorn, G. & Roese, J. (2003). IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines, RFC 3580.

Eduroam - Education Roaming (2011).
URL: *http://www.eduroam.org/*

Federal Enterprise Architecture (FEA) (n.d.).
URL: *http://www.whitehouse.gov/omb/e-gov/fea/*

Federal Information Security Management Act (FISMA) Implementation Project (n.d.).
URL: *http://csrc.nist.gov/groups/SMA/fisma/index.html*

Hinden, R. & Deering, S. (2006). IP Version 6 Addressing Architecture, RFC 4291.

Huston, G. (2004). Anatomy: A look inside network address translators, *The Internet Protocol Journal* 7(3).

Johnson, D., Perkins, C. & Arkko, J. (2004). Mobility Support in IPv6, RFC 3775.

Narten, T., Nordmark, E., Simpson, W. & Soliman, H. (2007). Neighbor Discovery for IP version 6 (IPv6), RFC 4861.

Nikander, P., Kempf, J. & Nordmark, E. (2004). IPv6 Neighbor Discovery (ND) Trust Models and Threats, RFC 3756.

Oda, S., Fu, H. & Zhu, Y. (2009). Enterprise information security architecture a review of frameworks, methodology, and case studies, *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*, pp. 333 –337.

Rescorla, E. & Korver, B. (2003). Guidelines for Writing RFC Text on Security Considerations, RFC 3552.

Rigney, C. (2000). RADIUS Accounting, RFC 2866.

Rigney, C., Willens, S., Rubens, A. & Simpson, W. (2000). Remote Authentication Dial In User Service (RADIUS), RFC 2865.

Sherwood, J., Clark, A. & Lynas, D. (2005). *Enterprise security architecture: A Business Driven Approach*, CMP Books.

Stango, A., Pacyna, P., Rapacz, N. & Prasad, N. (2011). Proposed Risk prioritization based on SABSA (Sherwood Applied Business Security Architecture) for Critical Information Infrastructures, *3rd International ICST Conference on Security and Privacy in Mobile Information and Communication Systems*.

Taleb, N. N. (2010). *The Black Swan: The Impact of the Highly Improbable*, 2nd edn, Random House.

Thaler, D., Zhang, L. & Lebovitz, G. (2010). IAB Thoughts on IPv6 Network Address Translation, RFC 5902.

The DoDAF Architecture Framework Version 2.02 (n.d.).
URL: *http://cio-nii.defense.gov/sites/dodaf20*

The Open Group Architecture Framework 9 (TOGAF9) (n.d.).
URL: *http://www.opengroup.org/togaf/*

Thompson, H. H. & Chase, S. G. (2005). *The Software Vulnerability Guide*, Charles River Media.

Wolthusen, S. (2004). Modeling critical infrastructure requirements, *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*, pp. 101 – 108.

Zachman Framework (n.d.).
URL: *http://www.eacoe.org/*