Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Beat Gebistorf

Interconnection of Different Network Architectures

ana
autonomic network architecture

Semester Thesis, SA-2008-22
September 2008 until January 2009

Professor: Prof. Dr. Bernhard Plattner
Advisor: Ariane Keller & Dr. Rainer Baumann

Abstract

The future of the Internet is unknown, even more as the currently existing mechanisms are a wonky foundation for the requirements of modern users: Huge efforts have to be taken to manage all participating nodes and to enable the attachment of upcoming network patterns like sensor nodes or delay tolerant networks. The ANA (Autonomic Network Architectures) project tries to introduce a solution to the known problems by handling the protocol stack in a more flexible way. The ultimate goal of the ANA project is to design and develop a novel network architecture that enables flexible, dynamic, and fully autonomic formation of network nodes as well as whole networks. In this semester thesis a new functional block of the ANA protocol stack was implemented which is able to connect the Internet Protocol world with a network equipped with Field Based Service Discovery. The Internet Protocol builds the central protocol of the current Internet whereas Field Based Service Discovery is a novel protocol adequate for service handling in mobile ad hoc networks. Therefore the implemented functional block builds the gateway from an Internet Protocol network to a Field Based Service Discovery network.

For the more general design of the implemented gateway, the Internet Protocol is considered combined with Routing Information Protocol to enable routing within the Internet Protocol network.

Specifications for gateways in general were defined in this thesis as groundwork for the design of the gateway between the Internet Protocol/Routing Information Protocol and Field Based Service Discovery. The general specifications allow prospective developers of gateways to have a simplified initial position for their development. Finally the implementation of this thesis got validated. The validation showed that the implemented gateway works correctly and is ready to extend the ANA protocol stack.

Acknowledgments

First of all I would like to express my sincere gratitude to Prof. Dr. Bernhard Plattner for giving me the opportunity to write this semester thesis in his research group.

I would also like to thank my advisors Ariane Keller and Dr. Rainer Baumann for their constant support during this semester thesis. They always helped me to solve the difficult problems of this thesis. Without their assistance, this work would never have been possible.

Furthermore, I would like to thank my fiancée, my family and my flat share colleagues for supporting and motivating me during this thesis.

Contents

List of Figures

1 Introduction

A growing demand for connections between networking devices exist. These devices base on different network architectures. Interconnecting different network architectures is thus the answer to the existing demand. In this semester thesis devices allowing interconnections of different network architectures are called gateways. The development of gateways involves three steps. The functionality of a gateway has to be specified (1), designed (2) and implemented (3) to be deployable. The specifications are required to allow evaluations of different designs whereas the implementation according to a design is needed to allow practical test.
All three steps are covered in this thesis. The specifications for gateways in general as first step enabled the second step of designing a particular gateway between Internet Protocol [13]/Routing Information Protocol Version 2 [12] (IP/RIPv2) and Field Based Service Discovery (FBSD) [23] networks. IP stands for the Internet Protocol as it is used today in the Internet. Combined with IP the routing protocol RIPv2 can be used as routing mechanism in the Internet. RIPv2 replaced the faulty RIPv1 protocol [7]. The acronym RIP in the rest of this thesis refers to RIPv2. FBSD as novel protocol was designed for mobile ad hoc networks. Implementing the designed IP/RIP & FBSD gateway formed the third step of this thesis. The ANA framework [5] offered an optimal platform for the implementation. The ANA project provides a flexible and autonomic network architecture based on a clean slate approach. Various protocols like IP,RIP and FBSD were already implemented in the ANA framework. In addition to that useful communication APIs (Application Programming Interface) alleviating the implementation of new designs

made the ANA framework predestined for the IP/RIP & FBSD gateway implementation.

1.1 Motivation

Many different network architectures exist like for example TCP (Transmission Control Protocol [14])/IP [11] and GSM [3]. People use an increasing diversity of networking devices based on these network architectures. All the same people want information stored on different devices to be accessable on each device. Therefore the demand of interconnecting these devices increases. This demand in turn is technically a demand for devices enabling interconnections of different network architectures corresponding to gateways as mentioned before. A popular and widely used architecture is the Internet. Therefore it is meaningful to start designing a gateway connecting the Internet with other protocols. Internet bases on IP. A possible routing protocol in addition is RIP. On the other hand mobile and ad hoc devices become more and more relevant, e.g. sensor nodes. A novel protocol for mobile ad hoc devices to handle services is FBSD. Hence it is reasonable with the prospect of future requirements on networking to design a gateway connecting IP/RIP networks with FBSD networks. Furthermore it is reasonable to implement the gateway in a framework prototyping the future Internet to use up-to-date methods. Such a framework is provided by the ANA project. Therefore the gateway implementation of this semester thesis is embedded in the ANA framework.

1.2 Goals

Different kinds of tasks have to be completed within this semester thesis as described in the following.

- Specifications for gateways in general have to be defined to ease the work of developers who want to design gateways. The specifications should form a guideline for developers.
- An example gateway needs to be designed for IP/RIP and FBSD networks.
- To complete the gateway development procedure the previously designed gateway has to be implemented in the ANA framework. This implementation has to pass a validation of several test scenarios.
- On the basis of this semester thesis one has to be able to follow the procedure of an entire gateway development circuit.

1.3 Structure

This Semester Thesis is structured as follows:

- Chapter 2 introduces the ANA framework and the characteristics of the protocols IP, RIP and FBSD. In addition the existing implementations of these protocols in the ANA framework are explained.
- Chapter 3 handles the topic gateway design. Beside introducing the problem space of gateways, specifications for gateways in general are described. Furthermore a particular gateway of IP/RIP and FBSD networks is explained with specifications and design information.
- Chapter 4 allows to understand the implementation of the gateway design explained in chapter 3.
- Chapter 5 illustrates how the validation of the implementation was done by means of test scenarios.
- Chapter 6 lists possible future work and concludes the entire semester thesis.

2
Related Work

In this chapter the network architectures covered in this thesis with their most important features are discussed. The first section gives an overview over the ANA framework embedding the implementation of this thesis. An introduction of IP [13] and RIP [12] is given in the second section. In the third section FBSD [23] which bases on Field Based Routing (FBR) will be elucidated.

2.1 ANA

The ultimate goal of the ANA project [5] is to design and develop a novel network architecture that enables flexible, dynamic, and fully autonomic formation of network nodes as well as whole networks. It will therewith be possible to adapt and reorganize the network dynamically according to the needs of the users. The project is based on a clean slate approach. Clean slate in this context means a new architecture from scratch not only improving or extending an existing network architecture like the Internet.
In figure 2.1 two ANA nodes with their elements are shown. The elements are explained in the following section.

2.1.1 Terminology

Within ANA particular terms are used. In the following the terms MINMEX, KVR, Brick, Compartment, IDP and Information channel are explained [1].

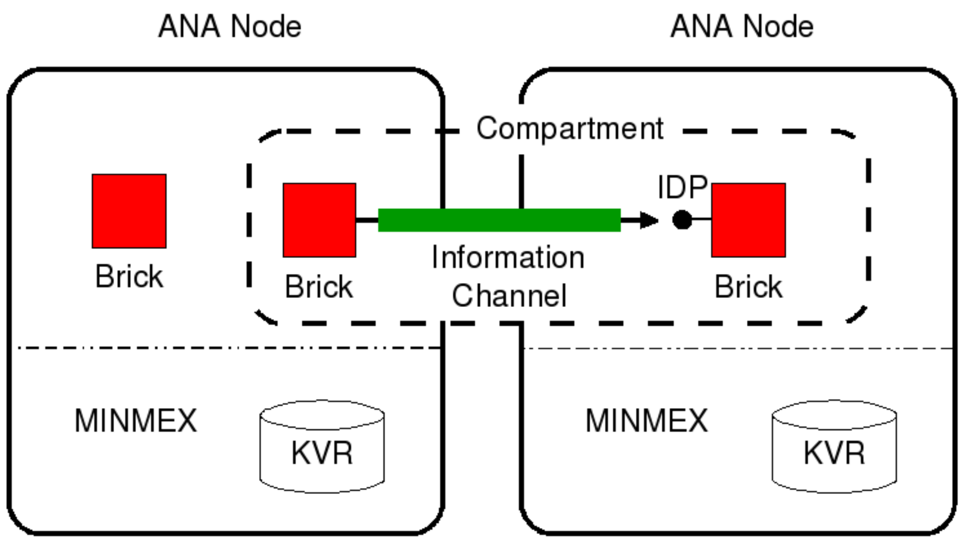


Figure 2.1: Typical ANA nodes with their elements

MINMEX Minimal INfrastructure for Maximal EXtensibility (MINMEX) defines the functionality which an ANA node minimal has to implement. It is the commonality of all ANA nodes. The MINMEX allows Bricks to interact.

KVR The key value repository (KVR) located in the MINMEX allows Bricks to discover each other locally on a node. It can be seen as a small database where entries have unique names, the IDP values, and set of keywords allowing to retrieve these values.

Brick Bricks are the most atomic elements in an ANA node providing functionality. Bricks interact over the MINMEX.

Compartment Bricks within a compartment are able to communicate with each other over the underlying network infrastructure. A compartment envelops several nodes which all have to implement the particular protocol stack of the compartment. Nodes are able to be part of various compartments, as long as they implement all necessary functionality of the compartment protocol stacks.

IDP Information Dispatch Points (IDP) are access points to Bricks. The binding of IDPs and Bricks are managed by the MINMEX.

Information channel Bricks within a compartment are able to communicate with each other over information channels similar to communication channels.

2.1.2 Publish & Resolve

ANA provides APIs for Bricks to communicate with the MINMEX. The two primitives publish and resolve of this API are used in this semester thesis.

Publish A Brick can publish its service with the publish primitive in a compartment to become reachable within this compartment.

Resolve A Brick can learn how to reach a published service and the corresponding Brick within a compartment by using the resolve primitive.

2.1.3 XRP Messaging

A common message format within the ANA framework is the eXtensible Routing Protocol (XRP) [20]. The ANA framework provides an entire engine which uses XRP to encode and decode messages. An XRP message is encoded as illustrated in figure 2.2.

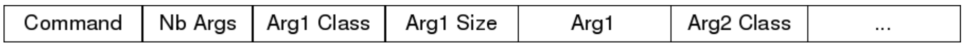


Figure 2.2: XRP message format

Description of the terms:

- **Command:** Determines the purpose of the XRP message.
- **Nb Args:** The number of XRP arguments attached to this command.
- **Arg1 Class:** The XRP class of the first argument attached to this command. Meant by class is a known description (meta-data) about the nature of the argument.
- **Arg1 Size:** The size in bytes of the first argument value.
- **Arg1:** The value of the first argument associated to the XRP command.

Afterwards further arguments can be attached to the XRP message by repeating the three field pattern (class, size, argument).

2.2 IP/RIP

In this section the IPv4 Internet Protocol gets explained. Additionally RIP as a distance vector routing protocol is explained later on in this section. At the end of the section the implementations of IP and RIP within the ANA framework are considered. Henceforward in this documentation the abbreviation IP will be used for the IPv4 protocol, in contrast to the IPv6 protocol.

2.2.1 IP/RIP in General

This section gives an introduction of IP and RIP as they are currently used in the Internet. IP implements the most important protocol on the internet layer in the TCP/IP model [11] whereas RIP has to be split into the routing daemon and the routing mechanism. A daemon is an application running in the background, invisible for the user. The routing daemon can be categorized as application layer protocol (see figure 2.3). The routing mechanism in turn can be categorized as internet layer protocol. IP and the routing mechanism of RIP are internet layer protocols because they both enable nodes to communicate over one connected logical internetwork [16]. The routing daemon of RIP can be seen as application layer protocol because it is a User Datagram Protocol (UDP)-based protocol. UDP [15] - a transport layer protocol - in turn assumes IP as underlying network.

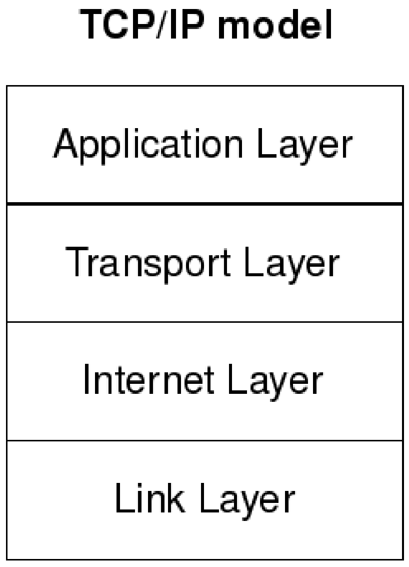


Figure 2.3: Layers of the TCP/IP model

IP

The Internet architecture is also called TCP/IP architecture because TCP and IP are the two most important protocols of the Internet [16]. IP is located on the second layer of the TCP/IP model, the internet layer, whereas TCP is a transport layer (third layer) protocol.
IP allows to connect different and heterogeneous networks regardless of the lower layers (e.g. Ethernet, Wi-Fi, Token Ring) [9]. Therefore IP ensures that IP datagrams are forwarded to the correct next node whereby the transmission of IP datagrams is unreliable[1]. Datagrams are similar to packets or messages. In addition IP ensures that nodes reassemble incoming IP datagrams and if needed fragment outgoing datagrams [11]. To achieve these properties IP encapsulates messages with the IP header which is explained in appendix A.1. To detect errors within the IP header of a received message a node computes the header checksum and compares it to the value stored in the checksum field of the IP header. This header includes the source and destination IP addresses of the message. IP addresses are assigned to nodes implementing IP and are globally unique, except the address ranges reserved for local networks which are not reachable in the global Internet. Local addresses can be reassigned in each local network. To reach a specific destination node routing protocols are required. Based on IP various routing protocols exist. One of them is RIP which will be explained in the following section.
TCP and UDP [15] are the most important protocols based on IP located in the transport layer. Transport layer protocols handle the communication from the source node to the destination node and are therefore also called end-to-end protocols. The transmissions in the Internet mainly are unicast (one-to-one) or broadcast (one-to-all) communications. TCP in contrast to UDP enables reliable message transfers. Reliable in the meaning of guaranteeing messages to reach the destination or at least informing the sending node when the message was not transmitted successfully.

RIP

RIP [12] is a dynamic routing protocol using the distance vector routing algorithm. The routing daemon of RIP maintains a database for all destination nodes and stores them in combination with the shortest path of the respective destination. As important additional information the next node for this shortest path is stored in the database. A node implementing RIP periodically distributes its own routing information to all neighbor nodes. Therefore it takes some time till the routing information is spread and the routing algorithm has found the shortest paths. IP combined with RIP is

[1]The sending node has no indicator whether the message reached the destination.

able to send messages over the shortest known path to the desired destination by sending the message to the next node stored in the database.

2.2.2 IP/RIP in ANA

IP as the most important Internet protocol is implemented in ANA combined with RIP as routing mechanism. Although the IP implementation in ANA does not support all functionality of IP used in the Internet [13]. Supported are the encapsulation of data in the IP header, the checksum computation, the addressing and the forwarding [9]. This enables an ANA node equipped with IP and RIP Bricks to encapsulate messages in an IP header with a correct checksum and send it to the desired destination node. Furthermore a node is able to act as forwarding device if it receives messages addressed to another node.
Within the ANA framework the implemented protocol stack of IP is built as follows: Applications can be run on top of IP. IP in turn runs on the underlying Ethernet protocol (see figure 2.4). But conceptionally IP works independent of the kind of underlying protocol as long as this protocol ensures connectivity to other nodes.

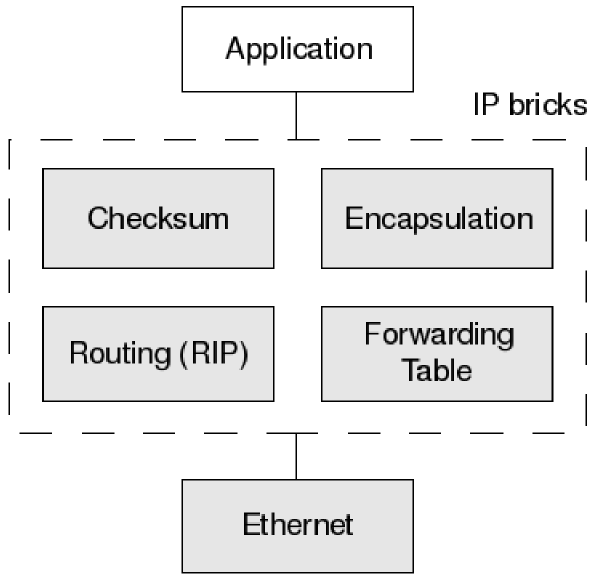


Figure 2.4: Overview of the IP protocol stack in ANA

2.3 Field Based Service Discovery (FBSD)

Field based service discovery is an approach to enable service discovery in mobile ad hoc networks without a central intelligence. Existing service discovery solutions used in the Internet are not well suited for mobile ad hoc networks [23]. FBSD is explained in the first part of this section. FBSD bases on the FBR routing strategy. The second part of this section gives an overview of the implementation of FBR in the ANA framework.

2.3.1 FBSD in General

First the two terms FBSD and FBR are clarified. FBR is a routing strategy, whereas FBSD is the application which makes use of FBR to publish and discover services.
FBR borrows its principle from a principle in physics, the field theory. Beside electrostatic fields other fields can be taken as background, e.g. the temperature field [6]. In analogy of positive point charges within electrostatic fields services generate a potential. These services are published by servers[2]. Clients who want to subscribe to this service can be compared to negative test charges. They are attracted by the services like negative charges are attracted by positive point charges [23]. Mapped to the network architecture FBSD it means that servers publish their services to all neighbor nodes. The service then gets spread from node to node. With each hop the potential decreases. The servers can call their services arbitrarily. These services in FBSD can be seen as equivalents to addresses. As multiple servers could publish the same service a potential field arises like the one in figure 2.5. Clients now can route their messages towards the steepest gradient[3] of the potential field created by the desired service. The message thereby reaches an adequate server.

2.3.2 FBSD in ANA

The FBSD implementation in ANA is distributed in five Bricks. They are the following: a field assembly, routing, forwarding table, forwarding dissemination and information dissemination Brick [9]. These Bricks enable a node which uses them to act as server or client of services distributed over the FBR compartment. The fundamental implementation provides six com-

[2]A node can be a server of services and a client of other services at the same time. It has to be noticed that it makes no sense that a node is a server and client of the same service because the node would serve itself with information.

[3]This routing towards the steepest gradient is called Field Based Routing. This routing strategy results in anycast routing like in [8] described for IP networks.

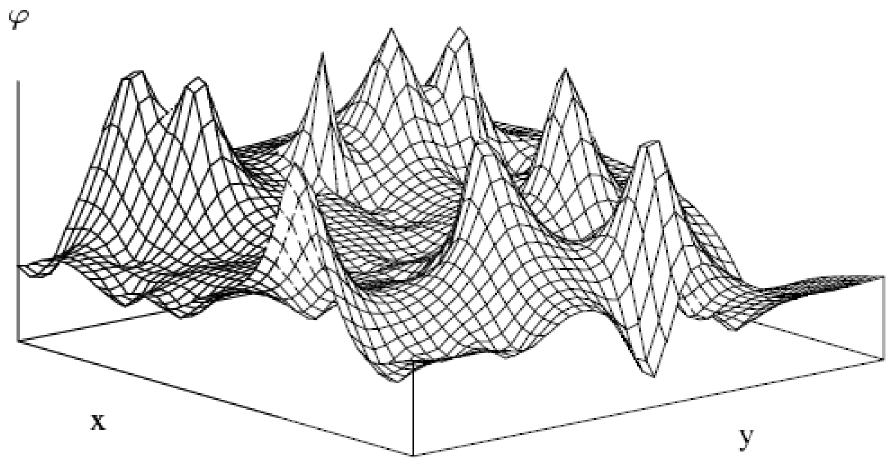


Figure 2.5: Potential field of a FBSD network (src [23])

munication mechanisms for server and three for client applications. Service applications can

- publish their services,
- update published services to keep the field alive,
- unpublish services to retract previous published services,
- send response messages to confirm successful subscriptions,
- receive subscribe messages from clients
- and receive data messages by using the implemented FBSD Bricks.

Clients in turn can

- subscribe to services,
- send data messages
- and get response messages.

An additionally implemented FBR API Brick extends the previously mentioned communication mechanisms. The FBR API Brick provides the standard ANA APIs. Two of the standard ANA API primitives are publish and resolve. The FBR API Brick is based on the previously listed five FBSD

Bricks.
Within this semester thesis the two ANA primitives publish and resolve are used to communicate between nodes in the same compartment, in the FBSD compartment as well as in the IP compartment. Therefore applications - of nodes in the FBSD compartment - implemented within this thesis base on the features of the FBR API Brick (see figure 2.6).
Within the ANA framework the implemented protocol stack of FBSD is built as follows: Applications can be run on top of FBSD. FBSD in turn runs on the underlying Ethernet protocol. But conceptionally FBSD works independent of the kind of underlying protocol as long as this protocol ensures connectivity to other nodes.

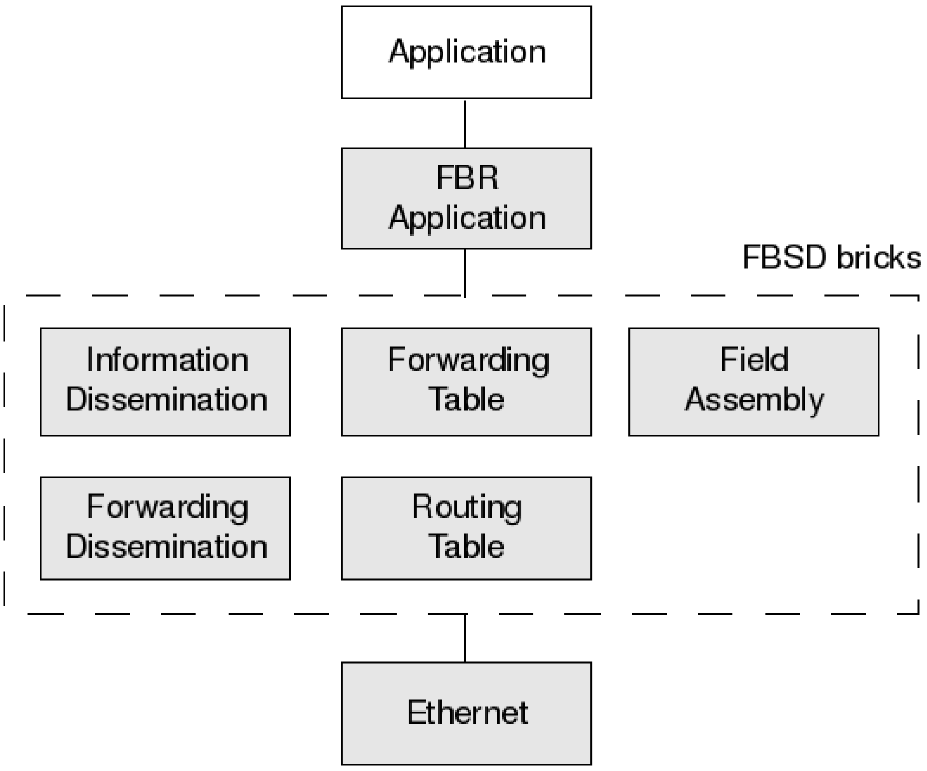


Figure 2.6: Overview of the FBSD protocol stack in ANA

3
Gateway Design

The main part of this semester thesis addresses the design of gateways. Introducing the problem space of gateways forms the topic of the first section in this chapter. The properties of gateways in general are explained in the second section before in the subsequent section the properties of a gateway between an IP/RIP network and a FBSD network are described. At the end of the chapter a concrete scenario gets observed in detail. In this scenario the two networks are connected directly over one gateway. In this last section additionally the protocol procedure for the named scenario is illustrated.

3.1 Problem Space

This semester thesis discusses gateways in general and also in particular for IP/RIP and FBSD networks. Gateways span a problem space which includes 4 dimensions of aspects. Mobility of the nodes as fourth dimension will be skipped because it would go beyond the scope of the thesis. Hence nodes are assumed to be stationary. The 3 remaining dimensions can be illustrated in a problem space (see figure 3.1). The 3 dimensions are defined as follows:

- Number of **gateways** connecting neighboring networks (single or multiple)

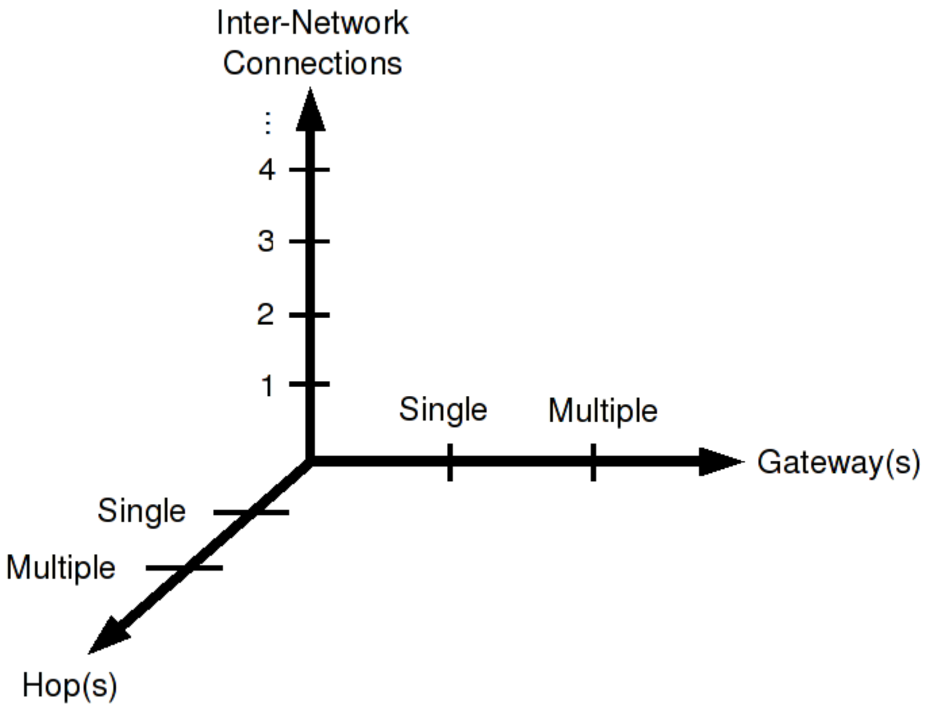


Figure 3.1: Problem space of interconnecting different network architectures

- Number of **inter-network connections** in the meaning of crossed boarders of different architectures between the source and destination nodes[1]
- Number of **hops** the source and destination nodes are away from a gateway respectively (single or multiple). If several gateways are reachable for a node then the gateway furthest away from the node is considered.

In the oncoming paragraphs some important combinations of these dimensions are illustrated. Thereby the following notation is used:
number of gateways / number of inter-network connections (crossings) / hops to the gateway

single gateway/1 crossing/single hop In the most simple case two nodes of different networks are connected via one gateway which is only one hop away from each one of them (see figure 3.2). This scenario is used

[1]In case of routing along the shortest path this number is equal to the number of crossed gateways.

as operation point for the gateway designed and implemented in this thesis namely the gateway between an IP/RIP network and a FBSD network.

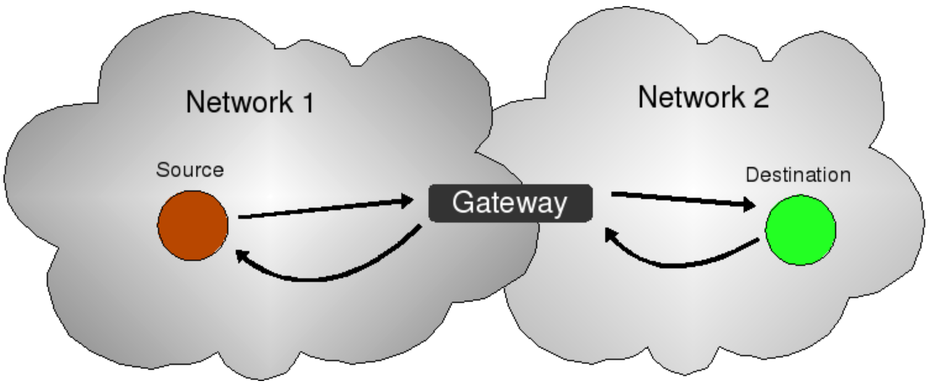


Figure 3.2: single gateway/1 crossing/single hop

single gateway/1 crossing/multiple hops Two different networks are connected via one gateway as depicted in figure 3.3. The source and destination nodes are connected to the gateway over multiple hops. To find the shortest path to the gateway the networks have to implement routing mechanisms. Thus they are able to choose the path to the gateway.

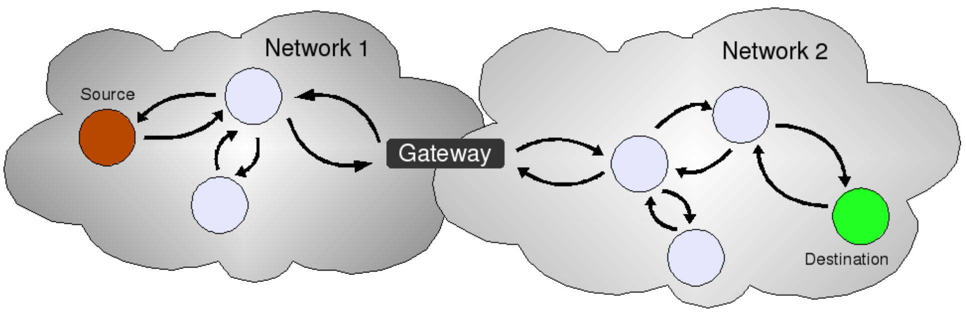


Figure 3.3: single gateway/1 crossing/multiple hops

multiple gateways/1 crossing/multiple hops For the case of nodes connected over multiple hops to multiple gateways the involved nodes have to provide routing mechanisms. Thus they are able to choose the gateway[2] or the path to the gateway respectively. A practical application for such a

[2]Depending on the implementation either the gateways coordinate the selection of the best gateway for a transmission or the nodes select themselves the best gateway.

scenario is the handling of telephone calls over Skype from the Internet to a PSTN[3] telephone or vice versa. In figure 3.4

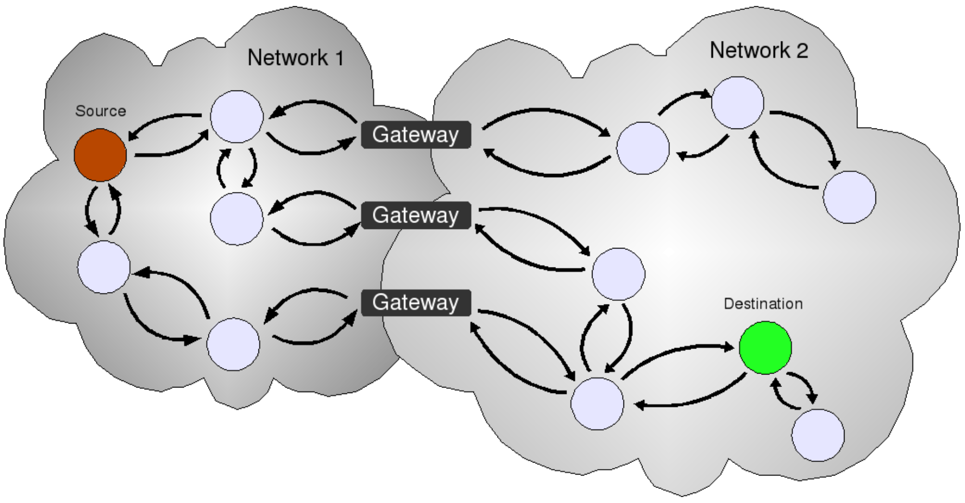


Figure 3.4: multiple gateways/1 crossing/multiple hops

multiple gateways/multiple crossings/multiple hops For the case of multiple gateways, multiple inter-network connections (crossings) and multiple hops (see figure 3.5) one can refer to a similar case with one crossing. The handling of the multiple crossings can thereby be done by tunneling through the intermediate networks. This results mainly in the analogue case of *multiple gateways/1 crossing/multiple hops.*

3.2 Gateways in General

Several networks are usually connected to a gateway. The main task of a gateway is to forward messages from one network to another one according to the transmission destination. The gateway modifies messages in compliance with the network properties. To specify the requirements and assure the functionality of gateways some assumptions have to be constituted. The assumptions made in this thesis as listed in the first part of this section are the following three: unique addresses in the networks connected to the gateway; the sending node knows the destination address; the sending node knows the address of an adequate gateway. Gateways in general have to fulfill basic requirements to work reliable. A gateway therefore has to adhere the specifications listed in the second part of this section. The specifications

[3]Public Switched Telephone Network

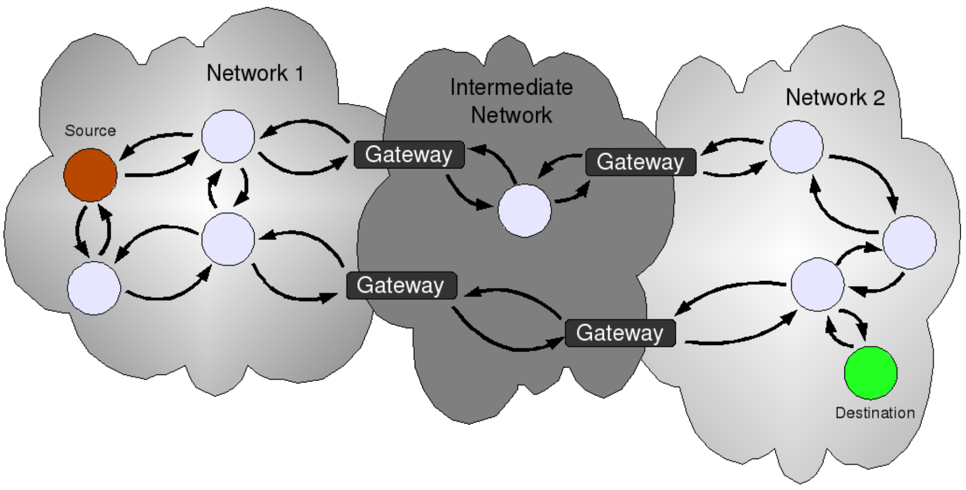


Figure 3.5: multiple gateways/multiple crossings/multiple hops

are to be able to apply protocols used by connected networks, translate data formats and addresses, resolve addresses, handle routing information and ensure coordination of multiple gateways.

3.2.1 Assumptions

Assumptions need to be set to clarify the conditions a gateway requires to function properly. These assumptions are discussed in this section. In the following only networks connected to the gateway are considered by talking about networks.

Unique addresses A network architecture has to ensure for each node in a corresponding network to be reachable over at least one unique address. The uniqueness has to be ensured within the range of all reachable nodes using this architecture. This assumption is important for gateways in order to ensure that forwarded messages are delivered to the correct destination. For many network architecture algorithm exist to ensure unique addresses, e.g. ZeroConfig [21] for IP or the autonomic identifier allocation algorithm developed for ANA in [10].

Knowledge of destination addresses If a node wants to send a message to another node it has to know the address of the destination node. One can assume that the node has knowledge of this address in advance or at least is able to learn the address. To learn the destination address one can use algorithms equivalent to DNS [17] for IP.

Knowledge of gateway address If a node wants to send messages to a node running another network architecture than itself it has to use a gateway. The same assumption as for the destination address holds for the address of this particular gateway. One can assume that the sending node has knowledge of the address of an adequate gateway in advance or at least is able to learn the address. To learn the gateway address one can use again algorithms equivalent to DNS.

3.2.2 Specifications

Under the previously mentioned assumptions the beneath listed specifications can be stated for gateways in general.

Applying protocols Generally spoken the gateway has to be able to communicate to the connected networks. This means to apply the protocols of the connected networks.

Format translation Exchanging data between different network architectures is the main task gateways have to deal with. This data may not only differ in the kind of content it conveys but also in the format of the data. A gateway therefore has to be able to translate data from one format to another one. Obviously a gateway has only to support data formats used in the networks connected to the gateway.

Address translation Beside correct format translations gateways are responsible for forwarding the data to the destination designated by the sending node. As different network architectures use different addressing schemes gateways have to manage the address translation. Address translation becomes important if a source node sends data to a placeholder address[4] of the destination node because the placeholder address matches the addressing scheme of the sending node network.

Address resolution A gateway needs the ability to resolve nodes which means to search for them. Searching for nodes becomes necessary if a gateway has to forward data to a node it does not know yet. Hence it has to search for it and perform an address resolution.

[4]A placeholder or a placeholder address is an address within a network which matches the addressing scheme of this network. This address represents a node in another network which applies another addressing scheme. Because the two networks use different addressing schemes the addresses have to be represented by placeholders and cannot be adopted directly.

Property	IP/RIP	FBSD
Addressing scheme	Unique IP addresses	Arbitrary names
Communication method	Unicast or broadcast	Anycast
Routing scheme	RIP	Field based routing
Role	Server & Client similar	Server oriented

Table 3.1: Differences between IP/RIP and FBSD

Routing information Various routing mechanisms exist according to different network architectures. Multiple metrics are defined within these routing mechanisms. To deal with protocols which use different metrics a gateway must implement functions to convert the used metrics.

Gateway coordination Between two networks multiple gateways might be located. A node resident in one of the two networks has to know which one of the multiple gateways it should use to maximize its transmission performance[5]. The coordination of the multiple gateways can either be done by the gateways or by the node itself. Anyway a gateway needs to be aware of the coordination problem. If only one gateway is in between two networks it obviously can ignore the coordination.

3.3 Gateway: IP/RIP & FBSD

More detailed specifications than for gateways in general are required to define a concrete gateway protocol. Within this semester thesis a gateway between IP/RIP(see chapter 2.2) and FBSD (see chapter 2.3) is considered. In this section the differences between IP/RIP and FBSD are listed and subsequently the specifications of its gateway protocol are described.

3.3.1 Difference: IP/RIP & FBSD

IP/RIP and FBSD were designed by differing motivations. IP/RIP is designed for the use in connected systems of packetswitched computer communication networks [13]. Whereas FBSD is designed to handle the service discovery in ad hoc networks. According to their backgrounds the two protocols feature unequal properties as listed in table 3.1.

[5]Transmission performance can have various meanings. E.g. best bandwidth or shortest response time. The gateway implementation defines how the performance will be measured and weighted.

3.3.2 Properties: IP/RIP & FBSD

A gateway for IP/RIP and FBSD has to obey the specifications for gateways in general. Additionally it must be able to handle the differences between IP/RIP and FBSD (see table 3.1). Gateways rely on fundamental assumptions being fulfilled (see section 3.2.1). Due to these exigencies the compliance with the above mentioned assumptions are verified in a first part of this section. In a second part the specifications resulting from the above named requirements are described.

Assumptions Verification

In the following three paragraphs the assumptions listed in section 3.2.1 are verified for the network architectures IP/RIP and FBSD.

Unique addresses The Internet as it is today uses globally unique IP addresses except for the locally used address spaces. Protocols like DHCP [19] or ZeroConf [21] provide the allocation of unique IP addresses in local networks. Nodes using FBSD are able to choose arbitrarily their service names which are equivalents to addresses. This evokes a problem if it is assumed that at least one unique address exists for each node. However one can assume that nodes are able to obtain a unique identifier as address, e.g. by using the autonomic identifier allocation algorithm developed in [10].

Knowledge of destination addresses There are several possibilities to ensure that nodes know the destination addresses of the nodes they want to reach. Three possible solutions are listed beneath.

- Hardcoded destination addresses if destination nodes use fixed addresses
- Usage of DNS [17] or DNS like mechanisms
- Ask for the destination addresses via a userinterface assuming that users know them

Knowledge of gateway address The knowledge of a gateway address is similar to the knowledge of an arbitrary node address. Therefore the same approaches as for the knowledge of destination addresses can be used.

Specification Refinement

In the following paragraphs the specifications for gateways in general are refined for the case of a gateway between IP/RIP and FBSD networks.

Applying protocols A gateway for IP/RIP and FBSD networks has to apply the protocols used in the IP/RIP network and FBSD network to be able to communicate with them. This means to embed the protocol stacks of the two network architectures (see figure 3.6).

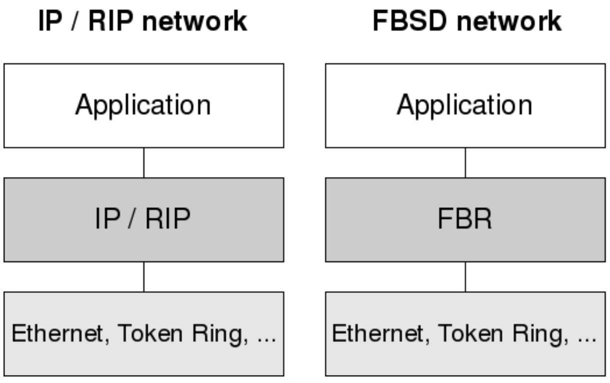


Figure 3.6: Protocol stacks of an IP/RIP network and a FBSD network

Format translation Within an IP/RIP network data messages are enveloped in IP headers (see appendix A.1). The important parts of the IP header are the source address, the destination address and the data message itself. For FBSD no similar standard was defined till now. Due to the lack of a standard protocol for FBSD the format translation depends on the implementation of FBSD. Even though no standard protocol exists one can assume that the most important information types are delivered. As mentioned for the IP header these most important types of information are the source address, the destination address and the data message.
The address translation gets explained in the adjacent paragraph. In case of the data handling IP uses fragmentation and padding methods whereas for FBSD no standard exists. For security reasons IP may even encrypt its messages. FBSD most probably is not able to decrypt and defragment the IP data. And vice versa IP might not be able to decode the FBSD data format. Therefore the gateway has to extract the plain data from one format and encode it in the other format to be able to handle both network formats.

Address translation In the simplest case of address translation IP nodes know the address of the FBSD nodes they want to reach or vice versa. The sender then creates a new header appending it to the already existing headers. This new header contains the destination address. Hence the gateway only has to extract the destination address from the message header.
Otherwise if networks only can manage their own addressing schemes the translation has to be done with placeholders. For the address translation of a gateway it means the following. On the one hand FBSD services have to be representable as IP addresses in the IP/RIP network. The gateway protocol has to define whether this is done by assigning IP addresses to FBSD services or by setting up a server which ensures the conversion like a NAPT [18]. On the other hand IP addresses have to be representable in the FBSD network. Again the protocol has to define whether this is done by assigning service names to IP addresses or by setting up a server which ensures the conversion.

Address resolution For the case of IP/RIP networks the address resolution can be achieved by using RIP. If an IP address exists and is reachable then the RIP mechanism will find it. As IP addresses are arranged hierarchically (spatial), RIP routes messages to the region where the node with the address of interest should be located. The closer the message travels to the destination the higher is the congruency of the address with the addresses in this region until reaching the destination address. By considering FBSD networks one can observe that the address resolution in FBSD has to be done by broadcasting a request message if the address is not published in the network as a service. If the address is published in the network as service one can simply resolve the address by routing towards the steepest gradient of this service.

Routing information RIP offers the possibility to use complex metrics if desired. The most used metric for RIP is the simple hop-count[6]. As metric in FBSD networks the potential of a service counts. In consequence a gateway between FBSD and IP/RIP has to be able to convert potential values in hop-count values and vice versa.

Gateway coordination To coordinate multiple gateways between IP/RIP and FBSD networks one can build an ad hoc control mechanism among the gateways. Thus the gateway with the best transmission performance gets the permission to operate as gateway for this transmission. If transmissions have to be reliable like in TCP[7] then the gateways have to worry more about

[6]The hop-count counts the number of network devices between the source node and the destination node. Technically each point-to-point link counts as one hop.

[7]TCP is a reliable transport layer protocol based on IP [14]

the coordination than about the unreliable connections like UDP[8]. For reliable transmissions the control mechanism additional to the transmission performance has to take care of the states of established connections. A gateway therefore keeps the permission to act as gateway for a transmission of an established reliable connection although another gateway might gain a better transmission performance. The gateway coordination depends on the type of connected network architectures. The particularities for IP/RIP and FBSD are the following:

- FBSD nodes route in an anycast manner to the next gateway. If a connection has to run over a particular gateway then this gateway has to be uniquely identifiable within the FBSD network
- As mentioned above IP supports reliable TCP connections which have to remain on an established channel and therefore must flow over the initially dedicated gateway

3.4 Protocol: IP/RIP & FBSD

In this section a concrete protocol between IP/RIP and FBSD will be presented. In the first part of this section the chosen operation point of the problem space in which one FBSD node sends messages to one IP node over one gateway will be commented. In the second part the protocol procedure of this operation point gets illustrated.

3.4.1 Problem Space

To build a gateway protocol it is recommended to start with the simplest scenario, having the remaining scenarios and requirements in mind. The problem space of gateways described in section 3.1 shows the problems concerning a gateway and wraps up the possible scenarios. For the gateway protocol between IP/RIP and FBSD designed in this semester thesis one operation point will be considered. Namely the case where an IP/RIP network is connected to a FBSD network over one gateway, with no intermediate networks. All nodes are thereby connected directly to the gateway (see figure 3.7 and 3.8).
This simplest scenario performing a connection from a FBSD node to an IP node, initialized by the FBSD node, will be discussed. It is not necessary to discuss the case in which an IP node starts the connection. Because this case has no practical application yet. This is founded on the fact that FBSD is mainly designed for mobile ad hoc networks. Whereas IP/RIP is designed

[8]UDP is an unreliable transport layer protocol based on IP [15]

for stationary located nodes. Nodes in a mobile ad hoc network are mainly sensors or at least devices running generally client applications. Clients in turn have to subscribe to a server and hence initiate the connection. That is why one can assume that FBSD nodes primarily start connections.
The few server applications in FBSD networks are most probably located on base stations which are fixed located and consequently are predestined to own an IP address and be part of the IP network. Or even to implement a gateway.
In the adjacent section the procedure of the gateway protocol for IP/RIP and FBSD is explained in detail.

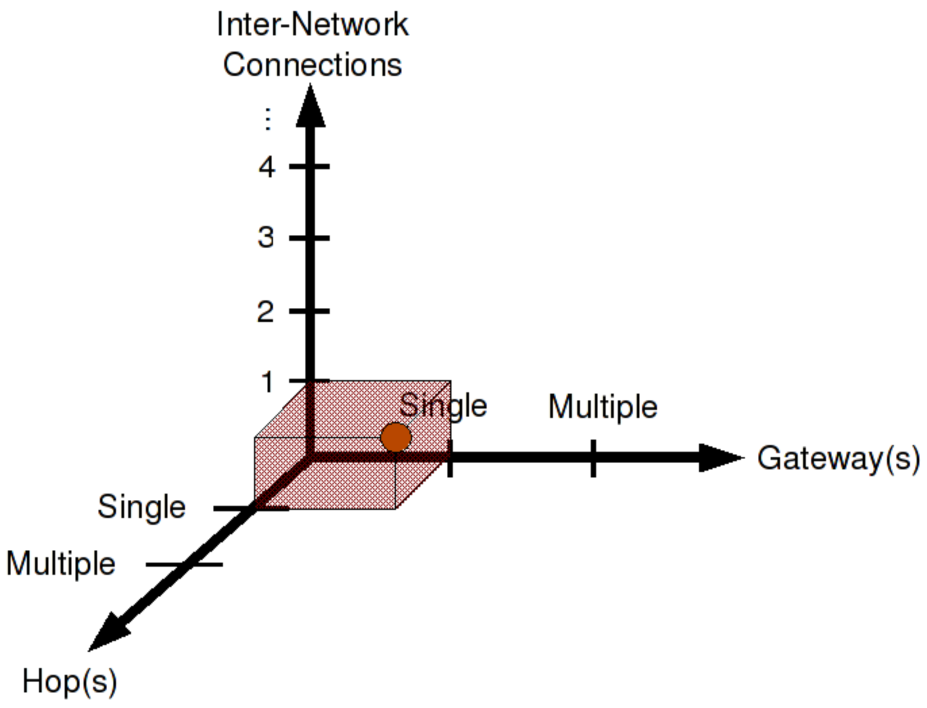


Figure 3.7: Problem space of gateways with the red marked operation point of this semester thesis

3.4.2 Protocol

In this section the procedure of connecting a FBSD node to an IP node will be explained step by step.

Initial situation In the initial situation three nodes are available (see figure 3.9). One node in an IP/RIP network, one node in an FBSD network

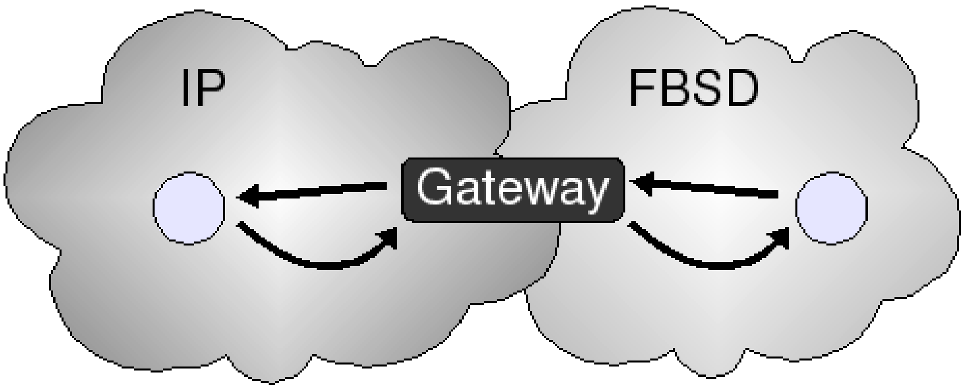


Figure 3.8: Scenario for the operation point of this semester thesis

and one node in between these networks and connected to both which will act as gateway. The FBSD node knows the address of the gateway and of the IP node it wants to be connected to. The gateway maintains a repository to store connection information.

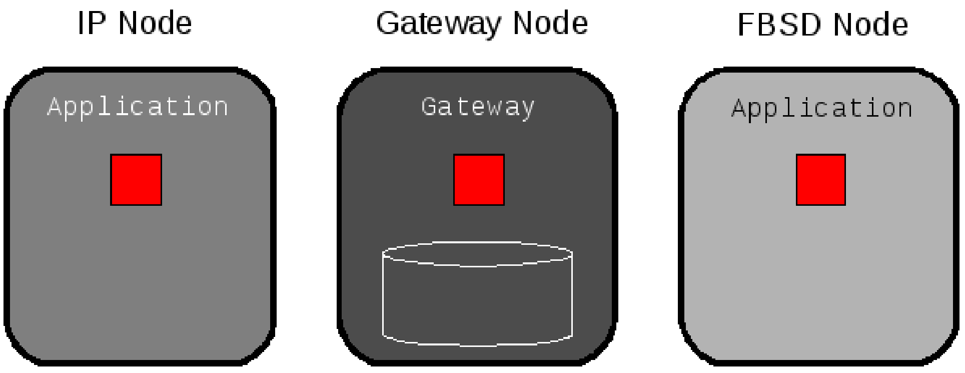


Figure 3.9: Initial situation of the connection of a FBSD node to an IP node over a gateway

Resolution As explained before (see section 2.1.2) a resolution has connection information as result. This resolution is important to prevent confusions between different addressing schemes. With the node identifier (see next paragraph) the addressing of message destinations happens uniformly, e.g. independent of IP or FBSD addresses. The resolution mechanism additionally prevents overhead caused by message losses. A node can test the existence of a destination node by resolving it before sending messages.
Concretely the FBSD node resolves the IP node by sending a resolve mes-

sage to the gateway ① (see figure 3.10). Afterwards one has to distinguish two cases. In the first case the gateway knows the IP node already, in the meaning of having a valid routing information for this node. In this case the resolution phase ends here. In the second case the gateway does not know the IP node yet. That is why it has to resolve the IP node itself. The used resolution functionality depends on the implementation of the IP/RIP network. In ordinary implementations of IP one assumes that the node exists. The gateway hence sends a request message. If the IP node does not exist the gateway will receive an ICMP error message. If the gateway receives an answer it stores the available information in the repository.

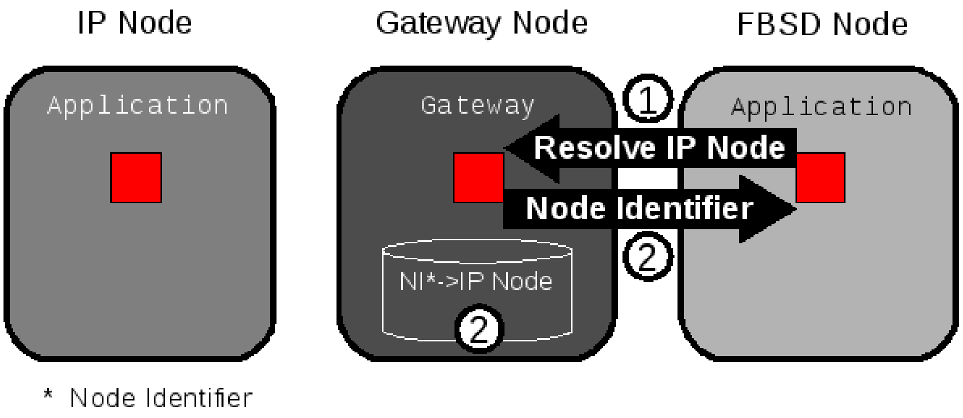


Figure 3.10: Resolution of an IP node and creation of a unique identifier

Unique node identifier To answer to the resolve message of the FBSD node the gateway creates a unique node identifier. This node identifier is unique within the gateway and is coupled with the resolved IP address stored in the repository. A message containing the node identifier now will be sent to the FBSD node ②.

Message creation The FBSD node is from now on able to send messages to the resolved IP node by employing the node identifier. To identify and distinguish different correspondences the FBSD node is forced to choose a unique message identifier. In this context unique means unique within the FBSD node. If this condition is fulfilled then the identifier is unique in the whole network because the combination of one unique source node and a unique message identifier therein is unique.

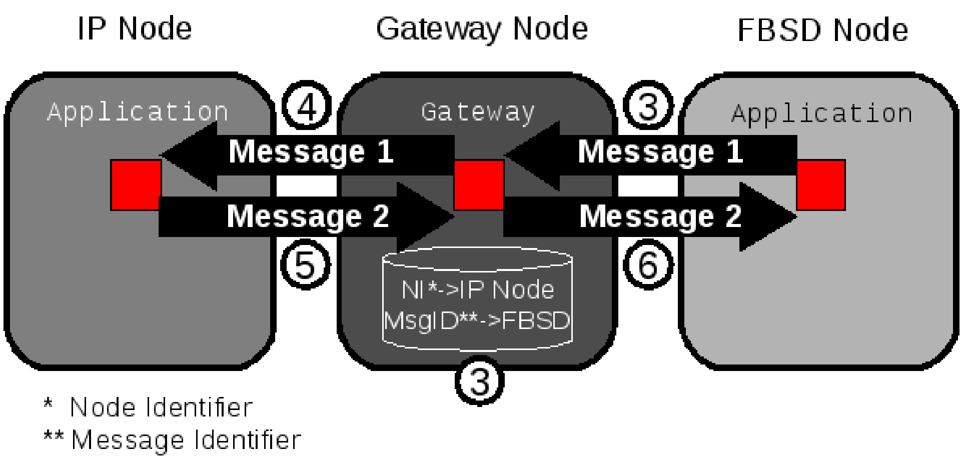


Figure 3.11: Delivery of messages with a message identifier

Message delivery The FBSD node now sends the data message including the message identifier, the node identifier from the resolve procedure, the source address and the data to the gateway (see figure 3.11). To enable the transmission of a response message from the IP node the gateway stores the message identifier combined with the source address ③. The gateway translates the node identifier into the corresponding IP address and sends the message further to the IP node ④. The IP node can respond to the received message or send additional messages by using the message identifier ⑤ ⑥.

Clean repository To keep the repository of the gateway clean the gateway necessarily has to delete old entries. Entries like message identifiers and node identifiers are called old if they are not valid any more. Cleaning up the repository is done by setting a timer for each entry in the repository. If a timer reaches the value zero the entry will be deleted. The timer can be reset by updating the entry. The interval length of the timer depends on the implementation and might differ for the message identifiers and the node identifiers.

4 Implementation

In this chapter the implementation of a gateway between a IP/RIP network and a FBSD network within the ANA framework is described. The implementation follows the procedure of the gateway protocol illustrated in section 3.4.2. The implementation focuses on the gateway Brick which connects the two networks. Within this thesis the gateway Brick and a sample application Brick for the IP/RIP and FBSD network respectively got implemented. The sample Bricks are templates to test the gateway functionality. The achieved task is to enable a FBSD node to ask an IP node for a file and to receive the file afterwards.
Because of the lack of a standard for FBSD the name FBR sometimes is used for the protocol although FBR only describes the routing mechanism. Within the ANA implementation the protocol is called FBR whereas in this chapter both abbreviations FBSD and FBR are applied.

4.1 Protocol in ANA

To establish a connection from a FBSD node to an IP node the IP, FBSD and gateway nodes have to follow a given procedure. In this subsection the detailed steps of this procedure in the ANA framework are explained. Abbreviations are used in the procedure illustrations and are therefore explained in the following.

- GW: Gateway

- IP: Internet Protocol
- FBR: Field Based Routing
- FBSD: Field Based Service Discovery
- App: Application
- IP NID: IP Node Identifier
- Msg ID: Message Identifier

Initial connection

1. First of all the Bricks within the gateway node have to publish themselves to the MINMEX (see Figure 4.1).

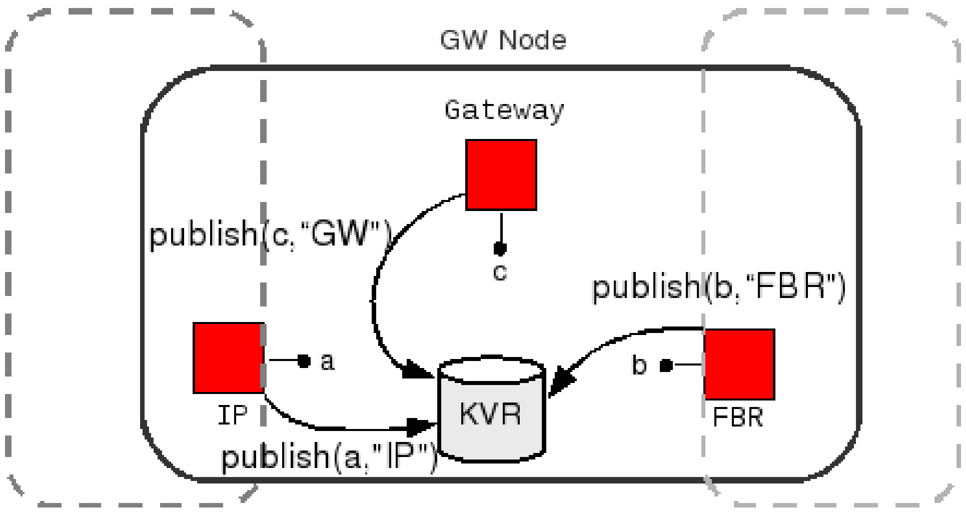


Figure 4.1: Publish the gateway, IP and FBR Bricks to the MINMEX in the GW Node

2. In a second step the gateway Brick has to resolve the IDPs of the FBR and the IP Bricks (see Figure 4.2).
3. To establish a connection to the FBR and IP compartment, the gateway Brick publishes itself to the respective Bricks (see Figure 4.3).
4. In the same sense the FBSD application and the FBR Brick within the FBSD node have to publish their existence to their MINMEX (see Figure 4.4). Analog the IP application and the IP Brick inside the IP node have to publish themselves to their MINMEX (see Figure 4.5).

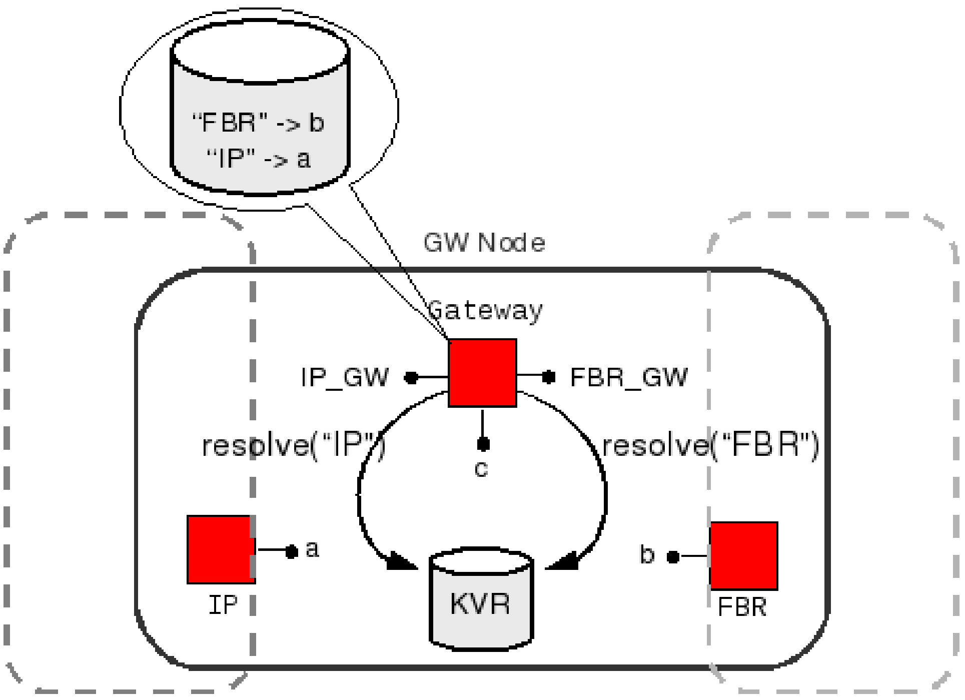


Figure 4.2: Resolve the IP and FBR Bricks in the GW Node

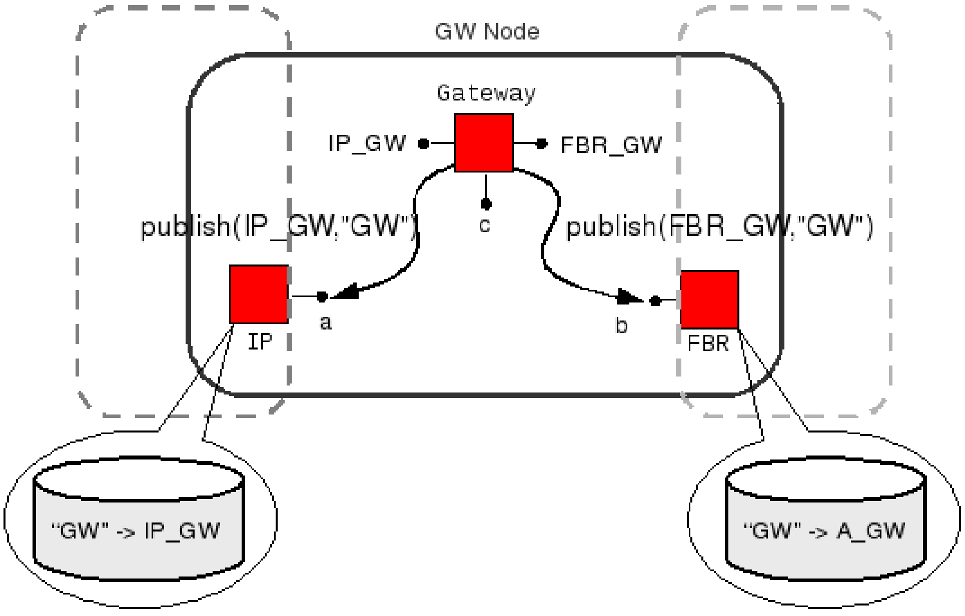


Figure 4.3: Publish the gateway service in the IP and FBR compartments

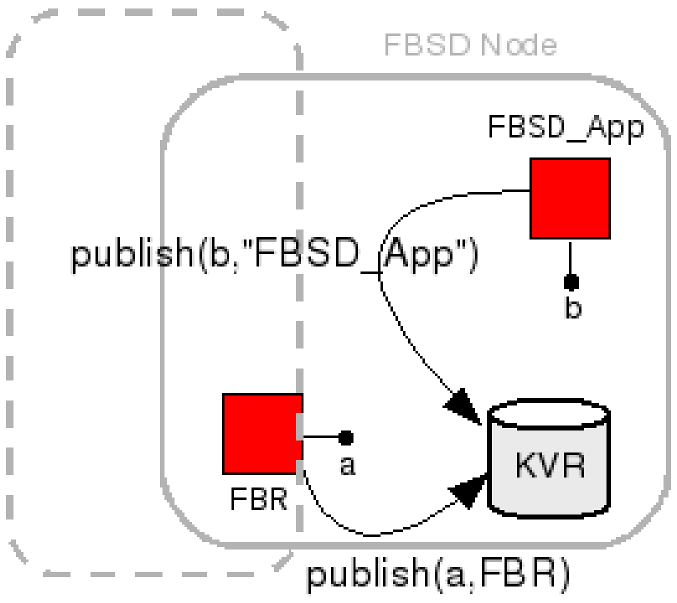


Figure 4.4: Publish the FBSD application and the FBR Brick to the MINMEX

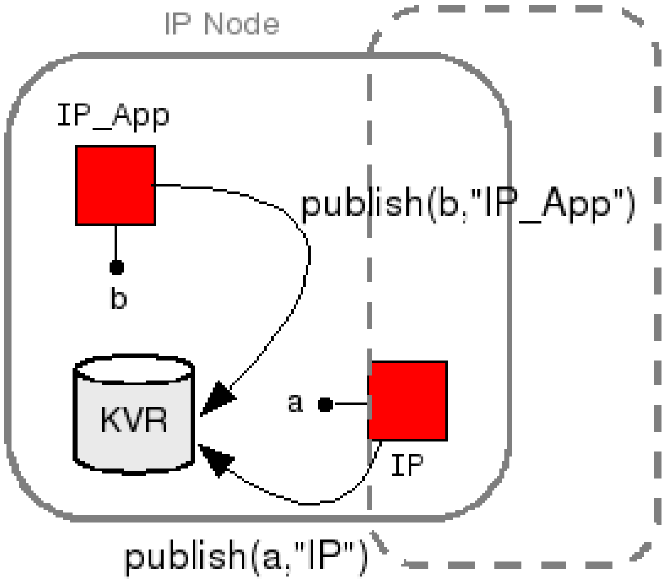


Figure 4.5: Publish the IP application and IP Brick to the MINMEX

5. To be able to connect to the gateway, the applications resolve the network compartment (Brick) they want to use (see Figure 4.6 and 4.7)

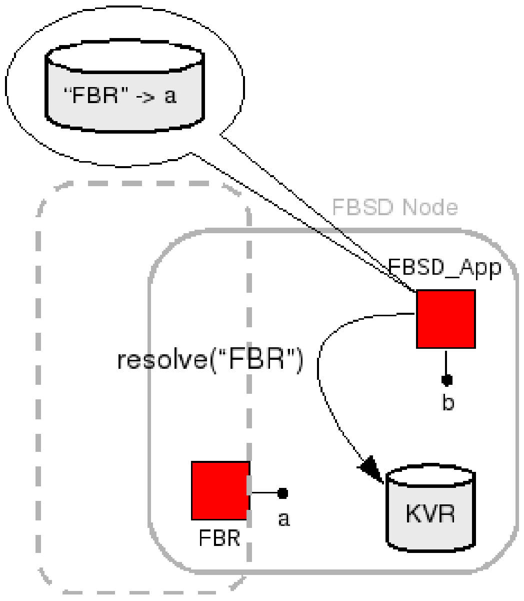


Figure 4.6: Resolve the FBR Brick in the FBSD node

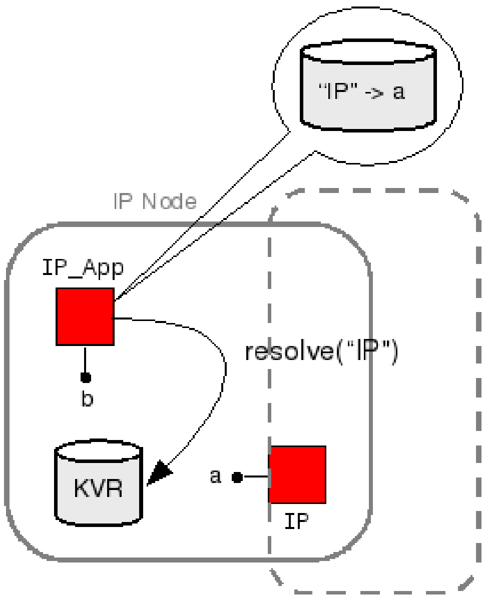


Figure 4.7: Resolve the IP Brick in the IP node

6. In order to be able to get responses the applications publish themselves locally to the compartment Bricks (see Figure 4.8 and 4.9).

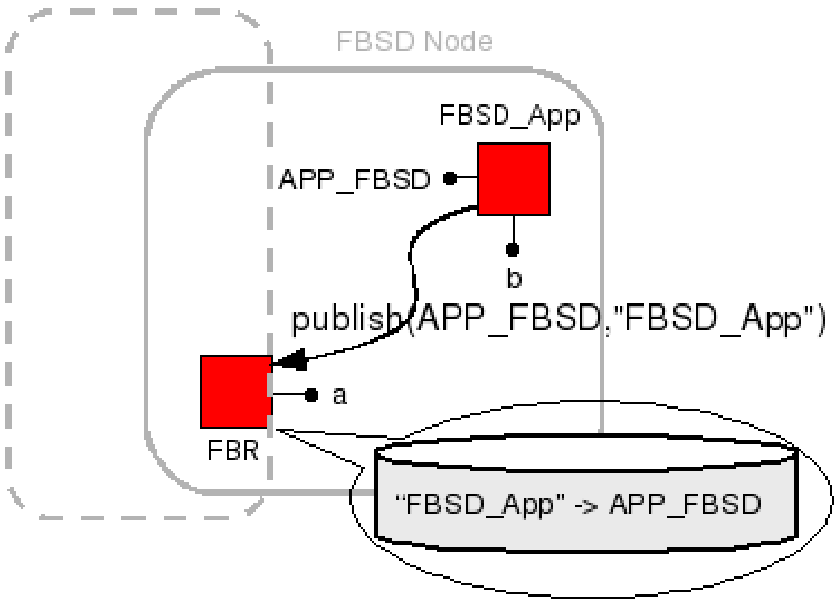


Figure 4.8: Locally publish the FBSD application to the FBR Brick

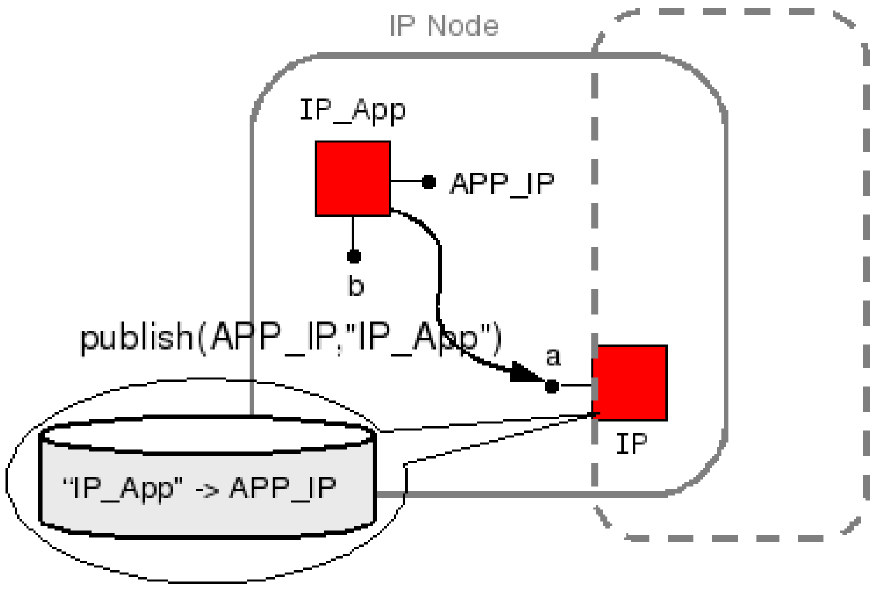


Figure 4.9: Locally publish the IP application to the IP Brick

7. The applications now can resolve the gateway service in the FBR compartment and IP compartment respectively by sending a resolve message (see Figure 4.10 and 4.11)

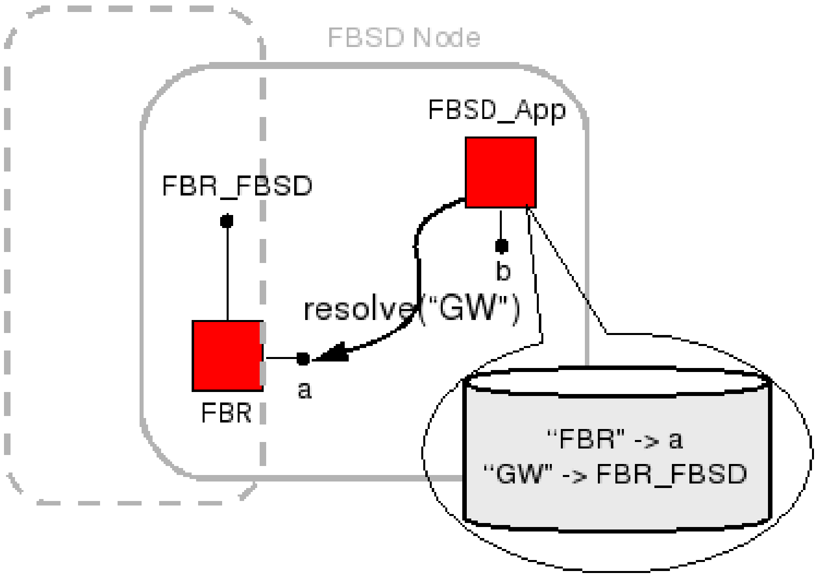


Figure 4.10: Resolve the gateway service from the FBSD node

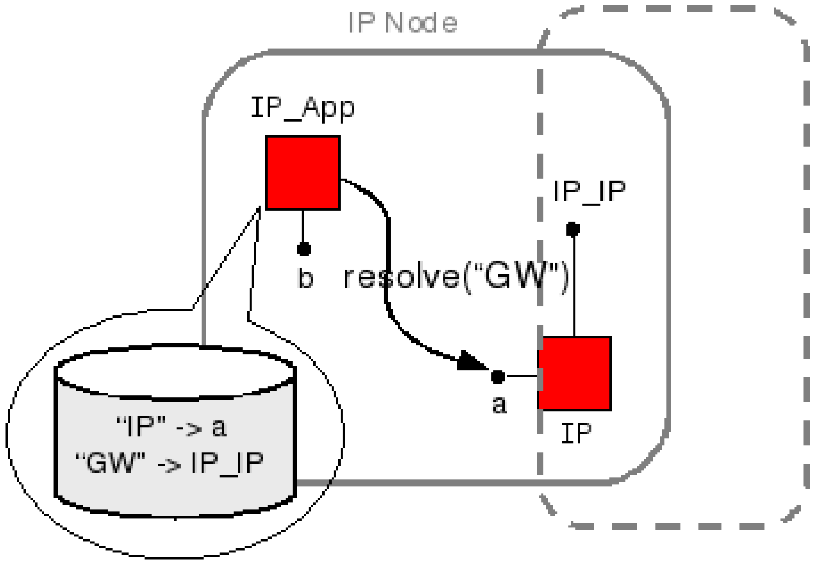


Figure 4.11: Resolve the gateway service from the IP node

8. Now the information channels from the applications to the gateway service are established (see Figure 4.12 and 4.13).

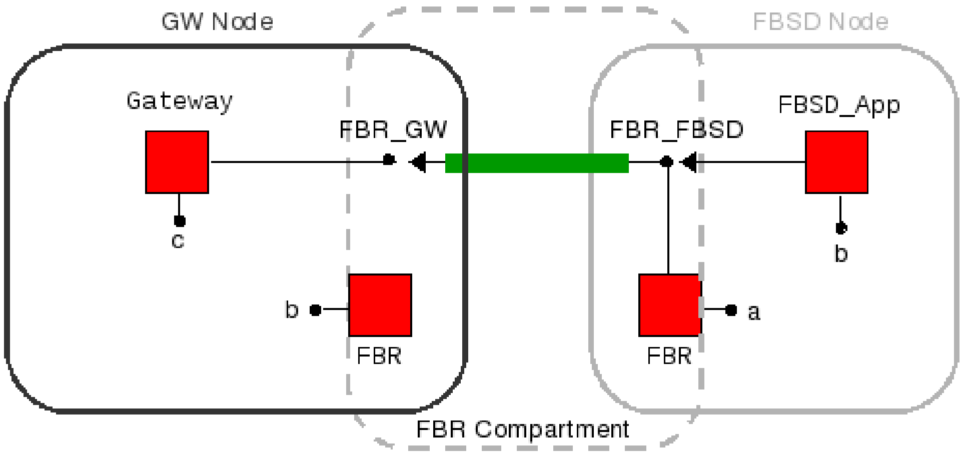


Figure 4.12: Information channel established between the FBSD application and the gateway

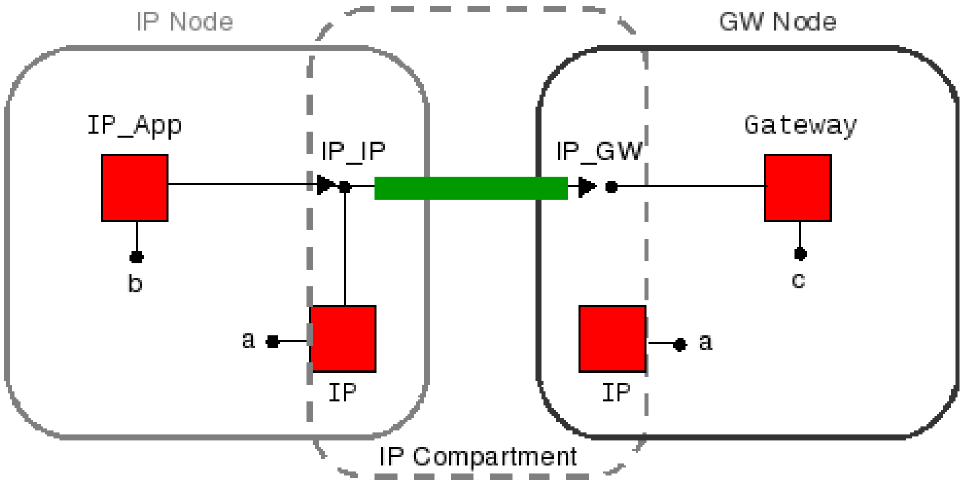


Figure 4.13: Information channel established between the IP application and the gateway

Resolution of the IP node The FBSD node now is able to resolve an IP node via the gateway by sending a resolve message over the information channel. To ease the description in the rest of the section the term node refers to the application running on the node, e.g. if a node sends a message it means that in fact the application on that node sends the message.

9. The FBSD node sends a resolve message to the gateway to resolve the IP node. The gateway on its part stores the response IDP[1] of the message to forward messages later on to the FBSD node. (see Figure 4.14).

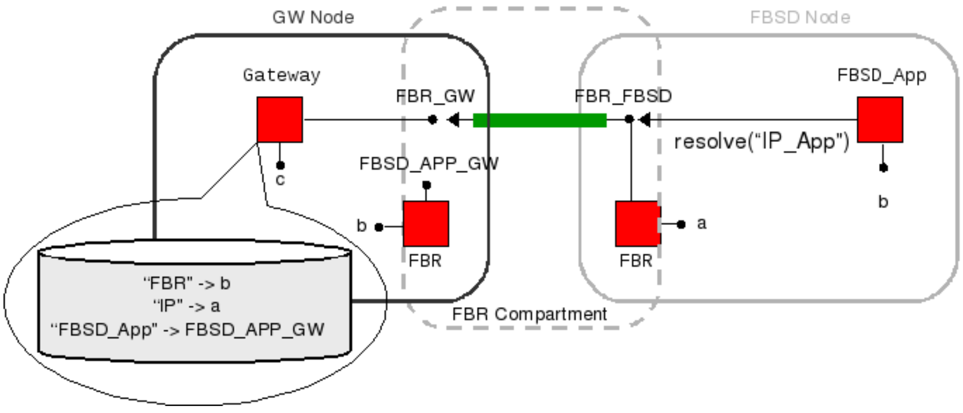


Figure 4.14: The FBSD node resolves the IP node via the gateway

10. (This step only becomes necessary if the gateway does not know the IP node yet) The gateway resolves the IP node according to the destination information in the resolve message. As result the gateway gets an IDP to forward messages to the IP node. If the gateway knew the IP node beforehand it just takes the already stored IDP[2] (see Figure 4.15).

11. Now the gateway creates a unique node identifier for the IP node. This identifier is unique within the gateway and is coupled with the IP IDP stored in the repository. The identifier is sent afterwards to the FBSD node to enable it to transmit messages to the IP node via the gateway (see Figure 4.16). If the resolution of the IP node was not successful the gateway sends a message to the FBSD node with the information that it did not find the IP node.

[1]In the ANA framework for each sent message a response IDP is created to enable sending acknowledgments or other data. The acknowledgment does in turn create no new "response" IDP.

[2]The gateway can know the IDP of an IP node because the IP node sent a message to the gateway and delivered a response IDP or because the gateway resolved the same IP node before.

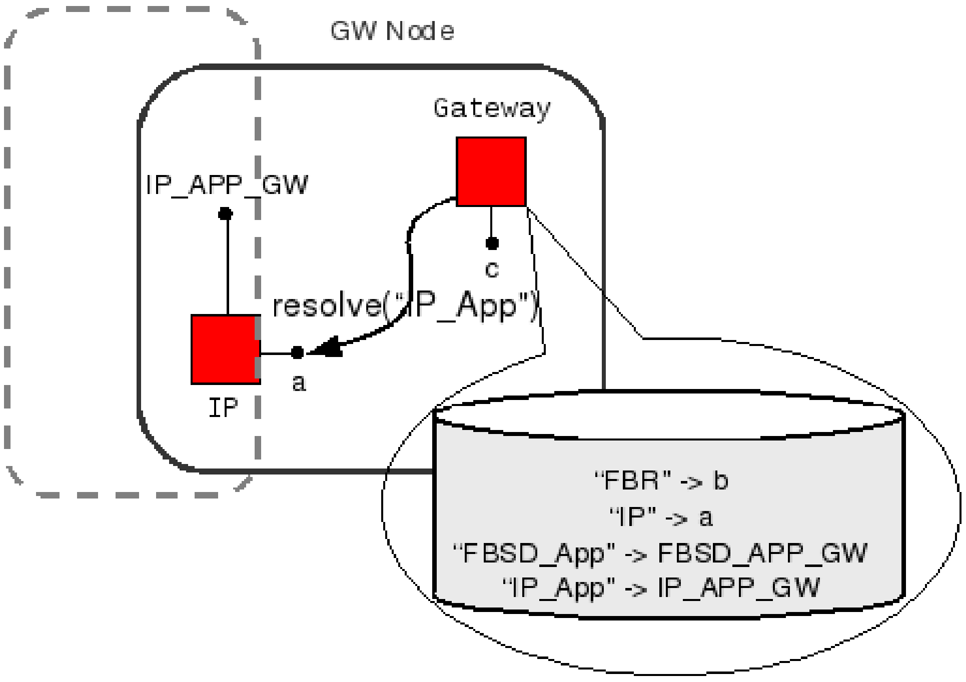


Figure 4.15: The gateway resolves the IP node

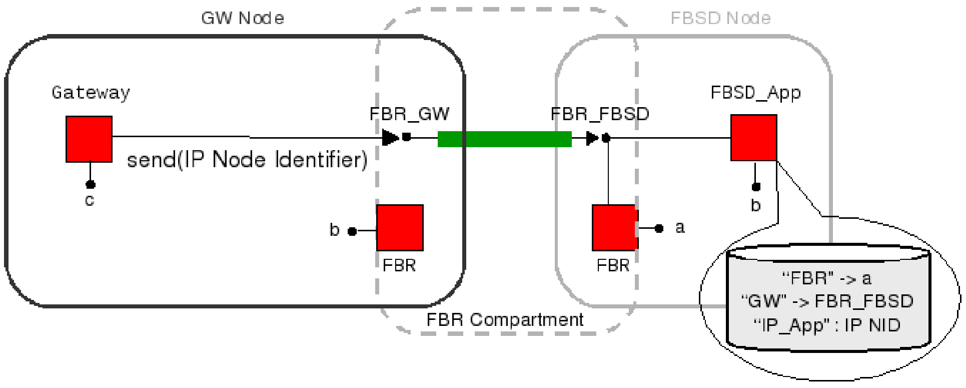


Figure 4.16: The gateway sends the unique node identifier to the FBSD node

Connection from FBSD to IP From now on the FBSD node is able to send messages to the IP node it resolved applying the learned node identifier. To distinguish different correspondences with the same IP node one has to introduce unique message identifiers. This message identifier has to be unique within the FBSD node. It then is unique network wide in combination with the unique source address of the FBSD node[3].

12. The FBSD node creates a unique message identifier. Afterwards it can send a message to the IP node via gateway applying the node identifier (see Figure 4.17). The message format of all messages are described in section 4.2.

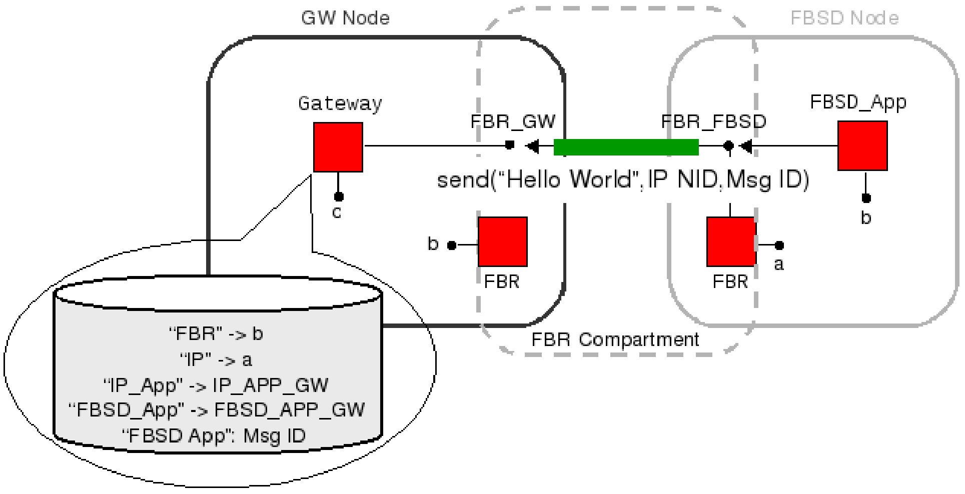


Figure 4.17: Message from FBSD to gateway

13. The gateway reads the node identifier and forwards the message to the corresponding IP node (see Figure 4.18).

14. The IP node receives the message and can send messages back to the FBSD node by using the message identifier included in the received message (see Figure 4.19).

15. The gateway in turn reads the message identifier of messages sent by the IP node and forwards it to the corresponding FBSD node (see Figure 4.20).

[3]The autonomous identifier allocation got implemented simultaneously with this semester thesis. Therefore static names were given to the FBSD nodes within the implementation of this semester thesis. Furthermore it is assumed that the number of possible identifiers is huge enough to ensure with a high probability that the identifiers are unique. In this implementation the number of possible identifiers is $2'147'483'647 = 2^{31}$.

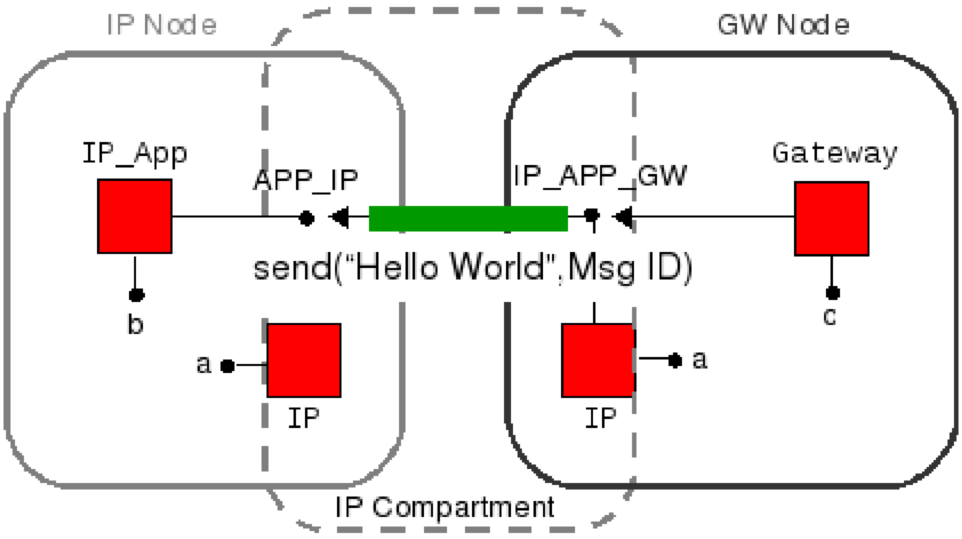


Figure 4.18: Message from gateway to IP

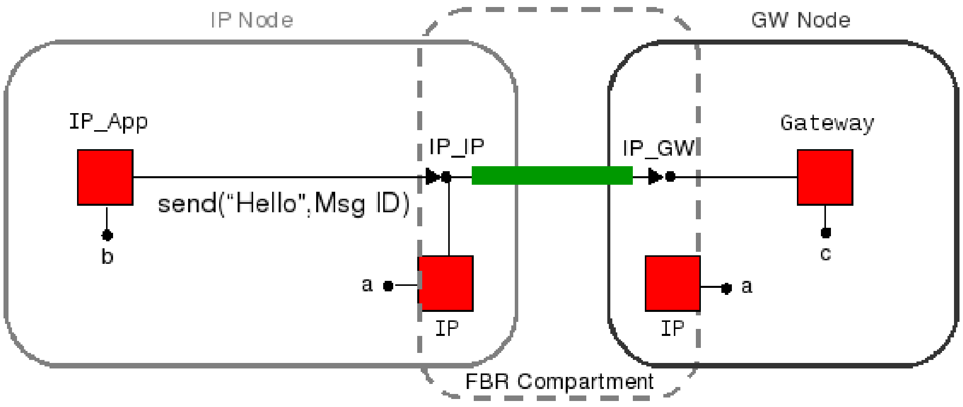


Figure 4.19: Message from IP to gateway

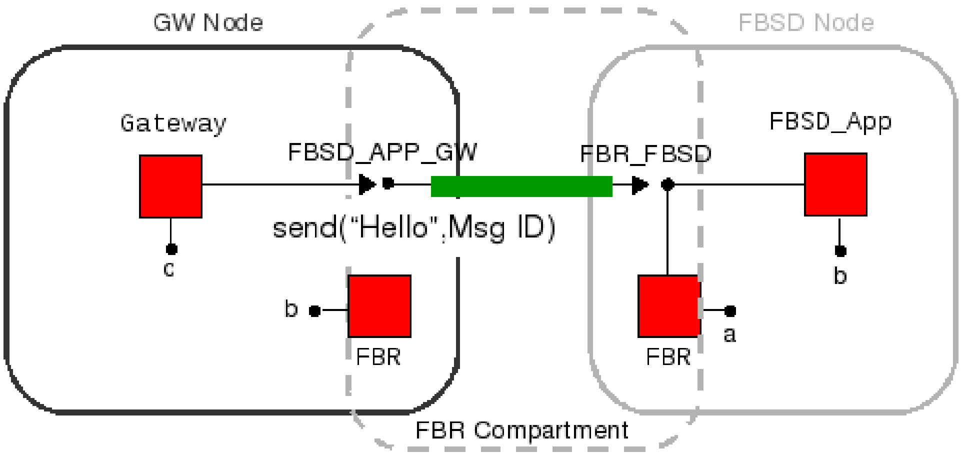


Figure 4.20: Message from gateway to FBSD

4.2 Message Format

The communication between nodes and the gateway has to be uniform to work properly. The already in ANA implemented XRP engine provides a suitable platform for this communication (see section 2.1.3). In this section the communication interfaces of the gateway encoded in XRP messages are illustrated (see figure 4.21). These interfaces are active after the initial connection phase described in the precedent section (see paragraph *Initial connection* in section 4.1). Therefore the resolution phase is the initial situation for this section (see paragraph *Resolution of the IP node* in section 4.1). As shown in figure 4.21 the gateway as intermediate node handles the resolve messages coming from the FBSD node. The other messages containing data or information are only modified as far as needed to reach the destination. Thereby the gateway processes parts of messages containing the ID or the label. Only the data itself is forwarded unmodified.

4.2.1 Interfaces to the FBSD Node

The FBSD application has to be able to resolve an IP node and its application, send and receive messages, ask for files and receive files. In this section these abilities are shown represented in encoded XRP messages.

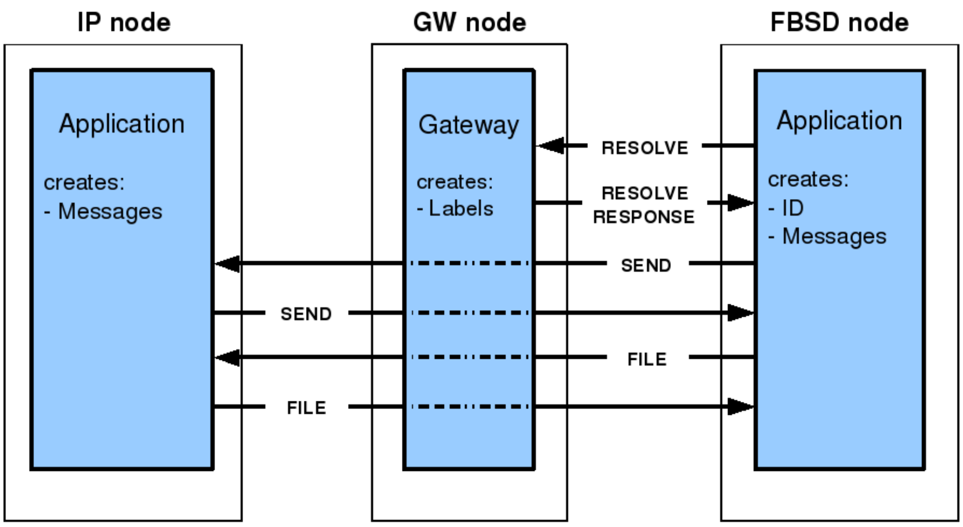
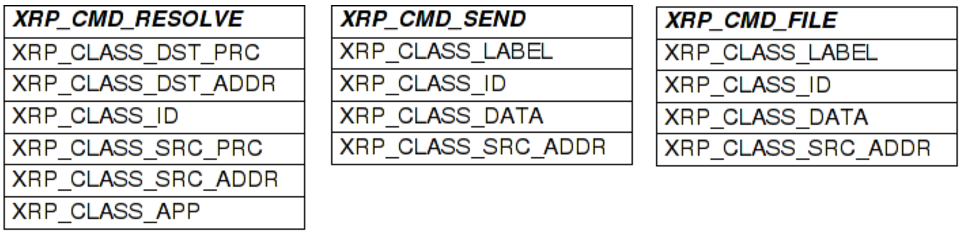


Figure 4.21: XRP messages used in this thesis

From the FBSD Node

The FBSD application starts the communication with a resolve message. As soon as it gets the node identifier for the application running on the IP node the FBSD application can ask the IP application for a file or send messages to the IP application. In figure 4.22 the structure for this XRP messages coming from the FBSD node are shown.



XRP_CMD_RESOLVE
XRP_CLASS_DST_PRC
XRP_CLASS_DST_ADDR
XRP_CLASS_ID
XRP_CLASS_SRC_PRC
XRP_CLASS_SRC_ADDR
XRP_CLASS_APP

XRP_CMD_SEND
XRP_CLASS_LABEL
XRP_CLASS_ID
XRP_CLASS_DATA
XRP_CLASS_SRC_ADDR

XRP_CMD_FILE
XRP_CLASS_LABEL
XRP_CLASS_ID
XRP_CLASS_DATA
XRP_CLASS_SRC_ADDR

Figure 4.22: XRP messages from FBSD to gateway

RESOLVE The RESOLVE message is used to resolve an IP node (and its application) via the gateway and is encoded as follows:

- **XRP_CMD_RESOLVE:** Identifies a RESOLVE message.
- **XRP_CLASS_DST_PRC:** Determines the protocol used by the destination IP node which has to be resolved.

- **XRP_CLASS_DST_ADDR:** Determines the address of the destination IP node which has to be resolved.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active resolve processes, created by the initiator of the resolve process.
- **XRP_CLASS_SRC_PRC:** Determines the protocol used by the source node, i.e. the FBSD node which is sending this message.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node, i.e. the FBSD node which is sending this message.
- **XRP_CLASS_APP:** Determines the application description which the gateway might need to resolve the application running on the destination IP node.

SEND The SEND message is used to send plain text messages to the IP application via the gateway and is encoded as follows:

- **XRP_CMD_SEND:** Identifies a SEND message.
- **XRP_CLASS_LABEL:** Determines the node identifier of a destination, belonging to an already resolved IP node. The label was created by the gateway (the label is a number generated at random).
- **XRP_CLASS_DATA:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange (the FBSD or the IP node).
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier. This information in addition to the ID provides the uniqueness of the ID.

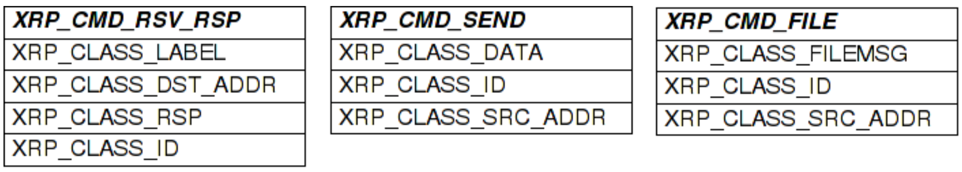
FILE The FILE message is used to send file requests or file data to the IP application via the gateway and is encoded as follows:

- **XRP_CMD_FILE:** Identifies a FILE message.
- **XRP_CLASS_LABEL:** Determines the node identifier of a destination, belonging to an already resolved IP node. The label was created by the gateway.
- **XRP_CLASS_DATA:** Determines the data to be delivered. In case of a file request the file name.

- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

To the FBSD Node

The FBSD application receives resolve response, text and file messages. In figure 4.23 the structure for this XRP messages sent or forwarded by the gateway to the FBSD node are shown.



XRP_CMD_RSV_RSP
XRP_CLASS_LABEL
XRP_CLASS_DST_ADDR
XRP_CLASS_RSP
XRP_CLASS_ID

XRP_CMD_SEND
XRP_CLASS_DATA
XRP_CLASS_ID
XRP_CLASS_SRC_ADDR

XRP_CMD_FILE
XRP_CLASS_FILEMSG
XRP_CLASS_ID
XRP_CLASS_SRC_ADDR

Figure 4.23: XRP messages from gateway to FBSD

RESOLVE RESPONSE The RESOLVE RESPONSE message is used to send a node identifier generated at random called label to the FBSD application from the gateway and is encoded as follows:

- **XRP_CMD_RSV_RSP:** Identifies a **RES**OLV**E** **RESP**ONSE message.
- **XRP_CLASS_LABEL:** Determines the node identifier of the IP node which got resolved in this request. The label is created by the gateway.
- **XRP_CLASS_DST_ADDR:** Determines the address of the destination IP node which got resolved.
- **XRP_CLASS_RSP:** Determines whether the resolve process was successful or not.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active resolve processes, created by the initiator of the resolve process.

SEND The SEND message is used to forward plain text messages to the FBSD application and is encoded as follows:

- **XRP_CMD_SEND:** Identifies a SEND message.

- **XRP_CLASS_DATA:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

FILE The FILE message is used to forward file data messages[4] to the FBSD application and is encoded as follows:

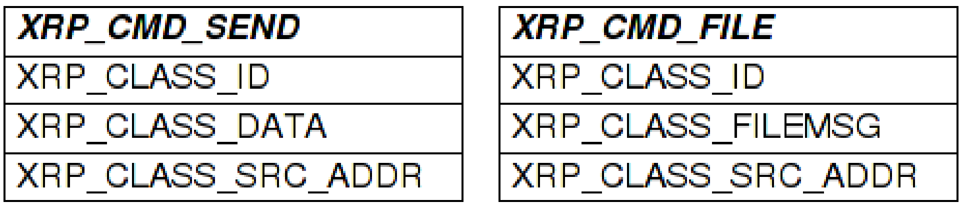
- **XRP_CMD_FILE:** Identifies a FILE message.
- **XRP_CLASS_FILEMSG:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

4.2.2 Interfaces to the IP Node

If the IP application receives a file request from the FBSD application it can send a message containing the requested file. In addition the IP application is able to send and receive text messages. In this section these message handlings are shown represented in encoded XRP messages.

From the IP Node

The IP application sends plain text or file messages. In figure 4.24 the structure for this XRP messages coming from the IP node are shown.



XRP_CMD_SEND
XRP_CLASS_ID
XRP_CLASS_DATA
XRP_CLASS_SRC_ADDR

XRP_CMD_FILE
XRP_CLASS_ID
XRP_CLASS_FILEMSG
XRP_CLASS_SRC_ADDR

Figure 4.24: XRP messages from IP to gateway

[4]The gateway does not care whether the delivered message contains file data or a file request.

SEND The SEND message is used to send plain text messages to the FBSD application via the gateway and is encoded as follows:

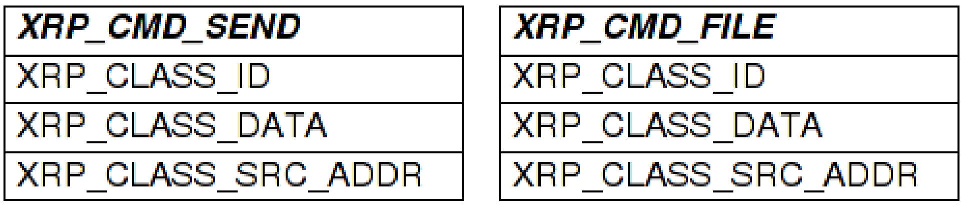
- **XRP_CMD_SEND:** Identifies a SEND message.
- **XRP_CLASS_DATA:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

FILE The FILE message is used to send file data messages to the FBSD application via the gateway and is encoded as follows:

- **XRP_CMD_FILE:** Identifies a FILE message.
- **XRP_CLASS_FILEMSG:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

To the IP Node

The IP application receives plain text or file messages. In figure 4.25 the structure for this XRP messages sent or forwarded by the gateway to the IP node are shown.



XRP_CMD_SEND
XRP_CLASS_ID
XRP_CLASS_DATA
XRP_CLASS_SRC_ADDR

XRP_CMD_FILE
XRP_CLASS_ID
XRP_CLASS_DATA
XRP_CLASS_SRC_ADDR

Figure 4.25: XRP messages from gateway to IP

SEND The SEND message is used to forward plain text messages to the IP application and is encoded as follows:

- **XRP_CMD_SEND:** Identifies a SEND message.
- **XRP_CLASS_DATA:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

FILE The FILE message is used to forward file request or file data messages to the IP application and is encoded as follows:

- **XRP_CMD_FILE:** Identifies a FILE message.
- **XRP_CLASS_DATA:** Determines the data to be delivered.
- **XRP_CLASS_ID:** Identifier to distinguish multiple active transmission processes, created by the initiator of the message exchange.
- **XRP_CLASS_SRC_ADDR:** Determines the address of the source node which created the identifier.

Development Advices

To extend the existing IP and FBSD applications or even create new ones one has to obey some advices concerning the XRP messages. XRP messages have to follow the structure explained in section 2.1.3. One is free to define arbitrary XRP commands or classes, e.g. XRP_CMD_MY_COMMAND or XRP_CLASS_MY_CLASS. It is strongly recommended to follow the expression guidelines. The guidelines are the following:

- Use only capital letters.
- Only attach letters to the descriptive root (e.g. XRP_CMD_ and XRP_CLASS_) and do not generate entire new expressions.

Newly introduced expressions have to be implemented in all affected applications, i.e. in a subset of the FBSD, IP and gateway applications. It is advisable to use the already implemented XRP commands and classes or the ANA framework common expressions as far as possible. A list of common expressions can be found in the ANA core documentation [1].

5 Validation

The validation of the implemented gateway between a FBSD and an IP network (see chapter 4) are illustrated in this chapter. The first section lists the requirements on the implementation. In the second section the validation with tested scenarios takes place. The last section discusses the performance of one test scenario.

5.1 Requirements

Most important for an implementation is that it functions correctly. This can be shown with a validation. To validate an implementation first of all the requirements which have to be fulfilled have to be determined. These requirements are described in this section as hard and soft criteria. The hard criteria are derived from the specifications of gateways in section 3.2. Whereas the soft criteria are derived from the protocol design in section 3.4.2.

Criteria The specifications mentioned before result in the following hard criteria which a gateway has to fulfill.

1. A gateway has to apply the needed network protocols used by the connected networks to be able to communicate with them.

2. A gateway has to extract the message data from the sender network and inject it into the destination network respectively correct formatted.

3. The previous criteria implies that a gateway has to be able to forward a message correctly.
4. A gateway has to ensure that source nodes are able to address the destination nodes by using their own features.
5. A gateway has to implement the metric translation needed for the routing mechanisms in the connected networks to work properly.
6. If multiple gateways exist between the source and destination networks then each of these gateways has to ensure that the proper coordination between them is guaranteed.

Additionally to the hard criteria some soft criteria should be fulfilled as listed next.

a. A gateway can handle multiple transmissions simultaneously, which means active and current processes.

b. A gateway can handle multiple connections simultaneously, which means connections which are currently not in use. Whereas the properties of the connections (e.g. message identifier) have to be maintained nonetheless.

c. The previous criteria imply the handling of multiple source or destination nodes simultaneously.

5.2 Tested Scenarios

In this section the hard and soft criteria are applied to the implementation of this semester thesis. They are validated by arranging test scenarios. The first part of this section introduces the four test scenarios whereas the second part discusses the validation according to these scenarios.

Scenarios The four tested scenarios as depicted in figure 5.1 permit to validate the implementation according to the hard and soft criteria. The first and simplest scenario represents a FBSD node connected to an IP node over one gateway. The second scenario tests the case where a FBSD node transmits messages over one gateway to two IP nodes simultaneously. The third scenario tests the case where two source FBSD nodes communicate simultaneously to one IP node over one gateway. The fourth and last scenario represents a case which is not included in the considered operation point. It shows that even multihop connections to the gateway are possible. In this scenario a FBSD node is connected via another one to the gateway. The

source FBSD node communicates over the gateway with two IP nodes.
In all four scenarios each FBSD node sends a request for a file to all available IP nodes. The IP nodes answer with the file data.

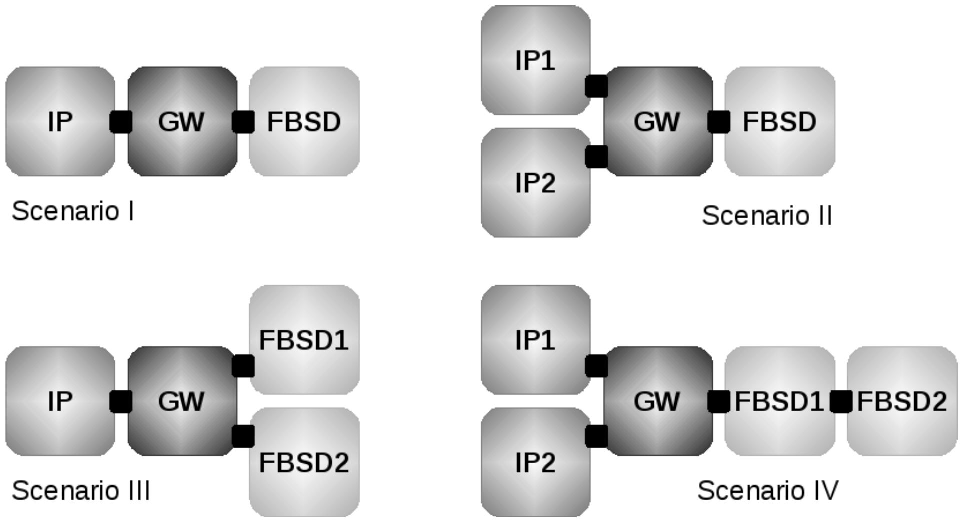


Figure 5.1: Four validated test scenarios

Results All four test scenarios were tested on one single system (an IBM ThinkPad T43p, Pentium M 1,86GHz with 1024MB RAM to be precise). That implies that all nodes were located on the same system too, connected with virtual ethernet links (refer to [1] chapter 5). In all four scenarios the gateway was successful with respect to the criteria. The FBSD nodes received in most of the cases the file correctly. The emerging errors can be assigned to the ANA framework. Because by considering the log files of the gateway and the IP application one can conclude that all emerged errors were neither caused by the gateway nor by the IP application. The emerged errors were the following:

- Errors by starting Bricks, most of them because of concurrency and race conditions. In some cases the Bricks were not able to find other Bricks they needed to run properly because the other Bricks did not finish their own loading phase. In other cases Bricks tried to access the same ressources like unix sockets at the same time. These problems can be avoided by choosing the timing of loading Bricks more carefully.
- Errors because of disordered messages within the communication channel between the gateway and the FBSD node. In large networks reordering of messages is normal. But because this validation took place

on one system where only one path without packetloss exists, it was not expected to observe reordering. However this problem can be managed by implementing and applying adequate transport protocols like TCP [14] or SAFT [22].

The four scenarios show that the hard and soft criteria are fulfilled and the implementation works fine. To start with the hard criteria:

1. The implemented gateway applies the needed network protocols used by the connected networks, namely the protocols of the IP and FBSD protocol stacks. Otherwise the messages would not have reached their respective destinations.

2. The gateway forwarded the messages correctly, extracting them from the source network format and injecting them into the destination network format. This is shown because the FBSD node received files sent by the IP node.

3. As assumed (see section 3.2.1) a FBSD node knows the address of the destination IP nodes or a placeholder address of it. In the implementation the IP address is used without using placeholders. Thereby the gateway had not to translate the addresses. The IP nodes use the message identifier of the received messages to send messages back to the source FBSD node. That is why an IP node in this implementation has not even to know the FBSD address.

4. The remaining two hard criteria are labeled meaningless for this implementation as explained in section 5.1

For the concrete operation point of the implementation within this semester thesis the following criteria become meaningless.

5. The metric translation becomes meaningless because the discussed operation point describes only one hop connections to the gateway. Therefore no routing mechanisms are needed within the networks.

6. The gateway coordination becomes meaningless because the discussed operation point implies only one gateway between two networks.

Coming to the soft criteria:

a./b./c. Scenario 2,3 and 4 demonstrate that the implemented gateway is able to handle multiple transmissions and connections simultaneously. Independent whether the multiple connections and transmissions originate from multiple FBSD nodes or multiple IP nodes.

5.3 Measurements

To get an idea of the abilities of the implementation the measurements of 1000 runs are discussed in this section. The runs were made within scenario one. Out of the 1000 runs 891 were observable. The remaining 109 caused an error in the initial phase of the measurement which means in the starting phase of the Bricks. The errors emerged because a Brick "A" depended on the functionality of a second Brick "B". "A" started before "B" ended its initial phase. Hence "B" did not provide the functionality "A" would have needed at the moment. Therefore Brick "A" caused an error. By choosing a better timing of starting the different Bricks these errors would have been erasable.

Setup Again all three nodes - the FBSD, IP and gateway node - run on a single system. In each run the FBSD node requested a file with 2721 bytes of size from the IP node. The IP node in turn sent the file in small fragments because too large fragment sizes turned out to be susceptible to errors. If too large fragments are chosen parts of fragments get lost. In this setup the chosen fragments have a size of 500 Bytes. Each run was started separately after the previous one had finished. In each run the Bricks were started anew to guarantee similar circumstances for each run.

Results As illustrated in figure 5.2 most of the runs had an execution time lower than 70ms. The thereby best achieved throughput was about 100 mega bytes per second ($\frac{2721\ bytes}{25ms} = 108.84MBps$). Measurements of runs with execution times higher than 70 ms had a significant longer execution time as can be seen in figure 5.3. These outliers can be explained by the workload of the processor which at the same time as the execution might had to compute other tasks.
Summarizing the section it can be claimed that the gateway functions according to the requirements.

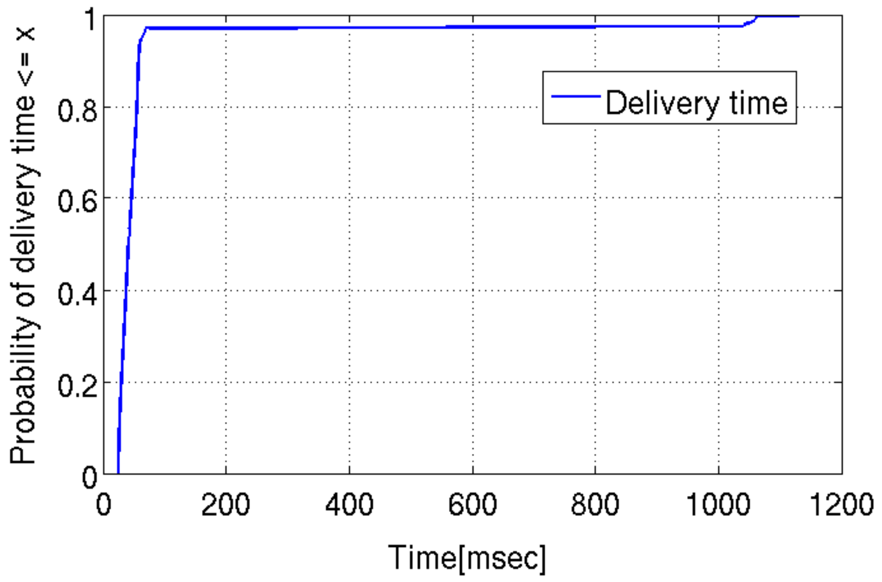


Figure 5.2: CDF of 891 measurements within scenario 1

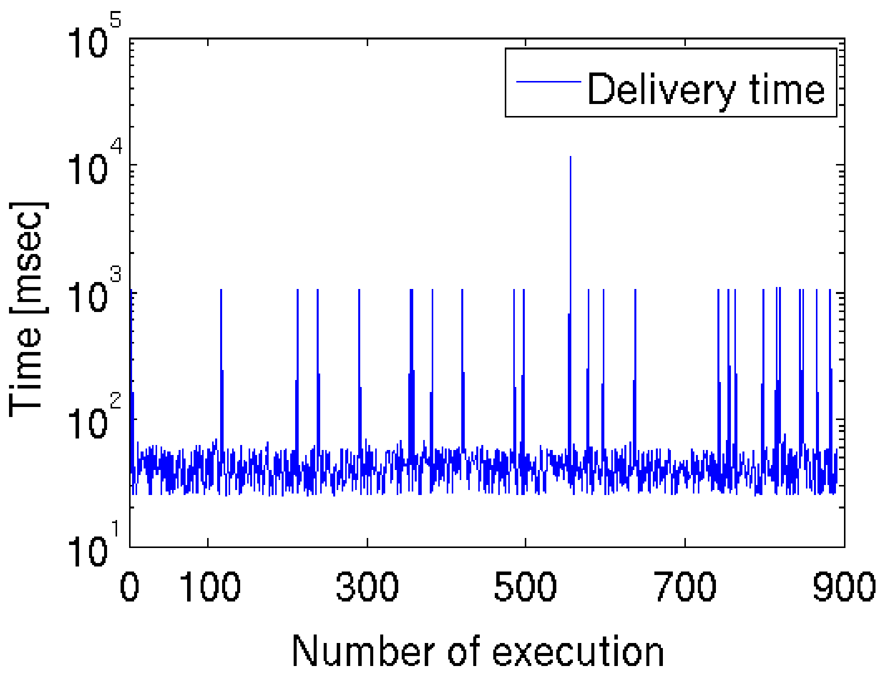


Figure 5.3: Execution times of the chronological sorted measurements

6 Future Work and Conclusion

Completing this semester thesis the first section of this chapter gives an overview of meaningful future work. To conclude the thesis on the topic of *Interconnection of Different Network Architectures* the second section summarizes the main discussed subjects and lists the achieved contributions.

6.1 Future Work

The gateway properties discussed in this semester thesis focus mainly on one operation point of the gateway problem space. Many remaining operation points are left open. Similar to that the gateway implementation of this thesis covers only a part of a gateway between IP/RIP and FBSD networks. The resulting open topics are described in the following sections as possible future work. In the first section potential extensions to the ANA implementation are listed. The second section gives a prospect to open work regarding the operation points for gateways.

6.1.1 Extensions

The implementation within the ANA framework of the gateway between IP/RIP and FBSD networks can be extended in various ways. The most meaningful suggestions are listed in the following.

- An extension for the gateway of IP/RIP and FBSD networks is to allow multiple gateways between the networks. Hence the implementation of a coordination mechanism between the gateways becomes necessary.

- Another extension is to enable IP nodes to initiate communications to FBSD nodes. This implies to enable the IP nodes to resolve FBSD nodes, e.g. in a DNS-like manner.
- Although it is far more than an extension, one very helpful implementation for the ANA framework would be to implement a gateway which enables various additional network architectures to be connected. If possible even a gateway which is able to connect arbitrary network architectures, e.g. by providing generic interfaces.

6.1.2 Work Prospects

The design of the gateway between IP/RIP and FBSD networks bases on one operation point. All remaining operation points allocate new topics for future work. The most significant topics are listed in the following.

- A future work can be to find ways to handle more than one inter-network connection. That means that messages have to cross other networks between the source network and the destination network.
- Mobility is an ongoing research area. Therefore it could be interesting to examine how multiple gateways can handle mobility of nodes which are connected to them, e.g. by using coordination mechanisms.
- An interesting future work could be to define specifications for gateways in various operation points. Whether for the case of nodes connected to the gateway over multiple hops[1], for multiple gateways or for more than one inter-network connection[2].
- It would be very helpful for future gateway designs to work out a generic gateway protocol.

6.2 Conclusion

The growing diversity of networking devices leads to an increasing demand on interconnections of different network architectures. To design devices

[1]This case is challenging for network architectures with routing schemes like source routing (refer to [16] pages 177-180)

[2]In contrast to the first topic of finding ways to handle multiple inter-network connections this means to define specifications for gateways, independent of the way it will be handled.

for these interconnections - within this semester thesis called gateways - one needs general specifications. These specifications facilitate the design of gateways for particular network architectures.
This semester thesis focused on the implementation of a gateway between an IP network combined with RIP and a FBSD network. IP/RIP is used in the current Internet whereas FBSD forms a cutting-edge and novel protocol. A suitable framework for the implemented gateway between IP/RIP and FBSD is the framework of the ANA project. The ANA project prototypes a flexible and autonomic network architecture based on a clean slate approach. Various protocols like IP, RIP and FBSD were already implemented in the ANA framework. In addition to that useful communication APIs alleviating the implementation of new designs made the ANA framework predestined for test purposes.
The contributions made in this semester thesis meet the need of people to interconnect network architectures.

- First of all by giving specifications for gateways in general. These specifications are in the future reusable for the design of gateways of arbitrary network architectures.
- By using these specifications the protocol for the gateway between an IP/RIP network and a FBSD network was designed.
- Thereby the differences between IP/RIP and FBSD were considered and described.
- Finally this design got implemented within the ANA framework.
- The implementation consists of the gateway and a generic application for IP and FBSD respectively.
- The correct functionality of the implementation has been validated by several test scenarios.
- FBSD nodes in the ANA framework are thanks to this implementation able to communicate with IP nodes over a gateway.

Final Remark

In my opinion ANA becomes a very powerful framework by implementing gateways like this because of providing even more flexible ways of communicating.

A
Protocols

A.1 IP Header

Following the various fields of the IP header are explained [13].

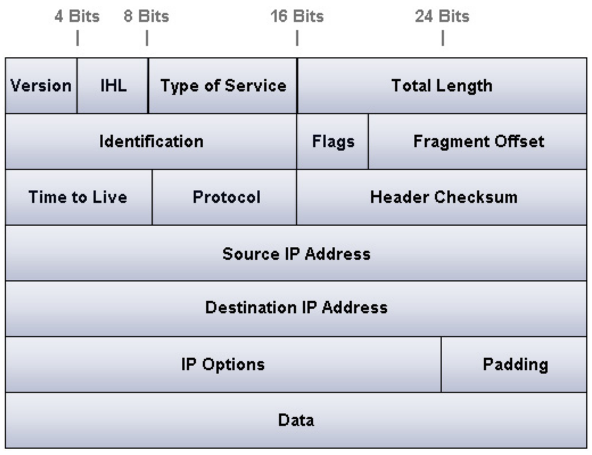


Figure A.1: IP header [src [4]]

- **Version:** A 4-bit long binary number to indicate the version of the used IP protocol. Currently IP version four (IPv4) is used, although IP version six (IPv6) will soon gain more importance than IPv4.

- **IHL (Internet Header Length):** This 4-bit field indicates the length of the IP header measured in 32-bit words. The minimum header length is five 32-bit words.
- **Type of Service:** This 8-bit field indicates special routing information, e.g. priority of a datagram.
- **Total Length:** This 16-bit field includes the length of the IP datagram, including the header and the data.
- **Identification:** To identify pieces of a packet that has to be fragmented, this 16-bit field contains the identification number of the whole data message. This mechanism is necessary to ensure that data can be rebuilt on the receiving node properly.
- **Flags:** This 3-bit field is used to select or deselect options for the fragmentation, e.g. whether fragmentation is enabled or not.
- **Fragment Offset:** If fragmentation is enabled the receiving computer has to reassemble the data pieces in the correct order. Therefore this 13-bit field is used to assign a number to each fragment.
- **TTL (Time to Live):** This 8-bit field indicates the number of hops a packet should go through before it gets discarded. Each router on the way of the packet decrements this field by one. A router which observes a TTL of zero discards the packet.
- **Protocol:** This 8-bit field determines which protocol should be used next at the receiver to process the message.
- **Header Checksum:** The checksum is a calculated value used to verify whether a header still is valid. This field has a length of 16 bits.
- **Source IP Address:** This field holds the 32-bit long IP address of the sending host identifying the sender and obtaining a return address in case of an error.
- **Destination IP Address:** This field holds the 32-bit long IP address of the receiving host. This address is used to route the packet towards the correct destination.
- **IP Options:** The option field with variable length gives the opportunity for optional settings.
- **Padding:** Because the header has to end after a full 32-bit word, this field is included to occupy left over bits.
- **Data:** At the end of the IP Packet the data takes place.

B
Task Settings

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Semester Thesis

Interconnection of Different Network Architectures

Beat Gebistorf

Advisor: Ariane Keller, ariane.keller@tik.ee.ethz.ch
Co-Advisors: Dr. Martin May, may@tik.ee.ethz.ch
Dr. Rainer Baumann, baumann@tik.ee.ethz.ch
Professor: Prof. Dr. Bernhard Plattner, plattner@tik.ee.ethz.ch

September 2008 - January 2008

B.1 Introduction

This semester thesis is in the context of the ANA project. The goal of the ANA project is to explore novel ways of organizing and using networks beyond legacy Internet technology. The ultimate goal is to design and develop a novel network architecture that can demonstrate the feasibility and properties of autonomic networking.

Future networks will interconnect different network architectures each of which is designed for a different network environment. In the ANA context we refer to all nodes using the same set of network protocols as a network compartment. The communication between two nodes belonging to different network compartments is not automatically possible, since they may use incompatible protocols. To allow this communication an extra translation step is necessary. This "gateway" interconnects the two compartments by translating information received from one compartment to information meaningful in the other compartment.

In this semester thesis we will develop such a gateway protocol. We will focus on IP/RIP as representative of today's Internet architecture and Field Based Routing (FBR) as a representative of network architectures designed for mobile ad-hoc networks. Depending on the students interest the developed translation mechanism will be implemented in the ANA framework or a more general translation mechanism will be designed that allows the interconnection of several different network architectures.

B.2 Assignment

This assignment aims to outline the work to be conducted during this thesis. The assignment may need to be adapted over the course of the project.

B.2.1 Objectives

The objective of this semester thesis is to design a gateway protocol between two different network compartments. In a first step the two specific network compartments "IP/RIP" and "FBR" are considered. If time allows we will either implement the developed gateway protocol or enhance it to be usable for different network compartments.

B.2.2 Tasks

This section gives a brief overview of the tasks the student is expected to perform towards achieving the objective outlined above. The binding project plan will be derived over the course of the first three weeks depending on the knowledge and skills the student brings into the project.

Familiarization

- Study the available literature on ANA [1, 2].
- Study the available literature on FBR [3, 4].
- Study the available literature on FBR and IP/RIP on ANA [5].
- In collaboration with the advisor, derive a project plan for your semester thesis. Allow time to study related work, design your gateway protocol between FBR and IP, generalize your gateway protocol or implement and evaluate your protocol. At the end of your semester thesis you will need some time to write your documentation and prepare the presentation.

Protocol Design

- List all the areas where IP/RIP needs to communicate with FBR
- List all the possible events when the two compartments need to exchange information.
- For each area listed above develop a translation mechanism, take into account that it may need to be different with respect to the data flow direction.
- For each event listed above show how your protocol achieves the communication between the different network compartments.
- Optional: do the same analysis for different network compartments.

Optional: Software Design

- The software should be as generic as possible.
- Divide your protocol into network compartment specific and unspecific bricks.
- Think about possible test scenarios for the functional verification and the performance analysis.

Optional: Implementation and Validation

- Implement the core functionality of your protocol.
- Implement additional functionality of your protocol.
- In several iterations optimize your implementation.
- Provide a simple validation script, that determines whether your Bricks work correctly.
- Validate the correct operation of your implementation.
- Check the resilience of the implementation, including its configuration interface, to uneducated users.
- Document your code with doxygene [8] according to the ANA guidelines.
- Adhere to the Linux coding style guide [6].
- Optional: Try whether your implementation works also in the Linux kernel space.
- Optional: Do a performance evaluation of your implementation. What is the impact of different parameters?

B.3 Deliverables

- Provide a "project plan" which identifies the mile stones.
- Mid semester: Intermediate presentation. Give a presentation of 10 minutes to the professor and the advisors. In this presentation, the student presents major aspects of the ongoing work including results, obstacles, and remaining work.
- End of semester: Final presentation of 15 minutes in the CSG group meeting, or, alternatively, via teleconference. The presentation should carefully introduce the setting and fundamental assumptions of the project. The main part should focus on the major results and conclusions from the work.
- End of semester: Final report describing the semester thesis.
- Any software that is produced in the context of this thesis and its documentation needs to be delivered before conclusion of the thesis. This includes all source code and documentation. The source files for

the final report and all data, scripts and tools developed to generate the figures of the report must be included. Preferred format for delivery is a CD-R.

B.4 Organization

- Student and advisor hold a weekly meeting to discuss progress of work and next steps. The student should not hesitate to contact the advisor at any time. The common goal of the advisor and the student is to maximize the outcome of the project.
- The student is encouraged to write all reports in English; German is accepted as well. The final report must contain a summary, the assignment and the time schedule. Its structure should include the following sections: Introduction, Background/Related Work, Design/Methodology, Validation/Evaluation, Conclusion, and Future work. Related work must be referenced appropriately.
- The source code will be published under the ISC license.

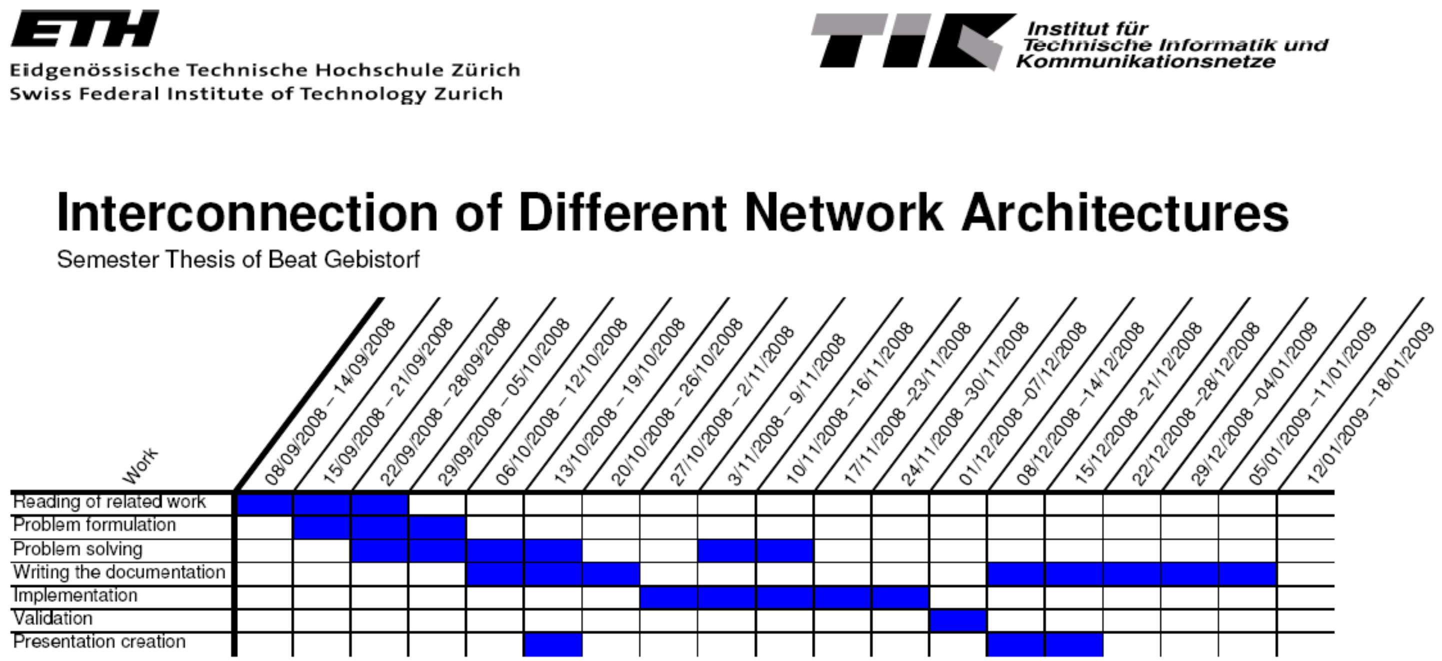
B.5 References

[1] ANA Core Documentation: All you need to know to use and develop ANA software. Available in the ANA svn repository.
[2] ANA Blueprint: First Version Updated. Available from the ANA wiki
[3] Service Discovery in Mobile Ad Hoc Networks: A Field Theoretic Approach, Vincent Lenders, Martin May and Bernhard Plattner, Elsevier Journal on Pervasive and Mobile Computing (PMC), Volume 1, Issue 3, September 2005.
[4] Density-based vs. Proximity-based Anycast Routing for Mobile Networks Vincent Lenders, Martin May and Bernhard Plattner Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Barcelona, Spain, April 2006.
[5] Stephan Dudler: New Protocols and Applications for the Future Internet, Master Thesis [MA-2007-39], ETH Zurich
[6] Available on your Linux box: file:///usr/src/linux/Documentation/CodingStyle
[7] https://www.ana-project.org/wiki
[8] http://www.stack.nl/ dimitri/doxygen/

C Timetable

C.1 Schedule

In figure C.1 the timing of the different kinds of work for this semester thesis is illustrated.



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Technische Informatik und Kommunikationsnetze

Interconnection of Different Network Architectures

Semester Thesis of Beat Gebistorf

Work	08/09/2008 – 14/09/2008	15/09/2008 – 21/09/2008	22/09/2008 – 28/09/2008	29/09/2008 – 05/10/2008	06/10/2008 – 12/10/2008	13/10/2008 – 19/10/2008	20/10/2008 – 26/10/2008	27/10/2008 – 2/11/2008	3/11/2008 – 9/11/2008	10/11/2008 –16/11/2008	17/11/2008 –23/11/2008	24/11/2008 –30/11/2008	01/12/2008 –07/12/2008	08/12/2008 –14/12/2008	15/12/2008 –21/12/2008	22/12/2008 –28/12/2008	29/12/2008 –04/01/2009	05/01/2009 –11/01/2009	12/01/2009 –18/01/2009
Reading of related work	■	■	■																
Problem formulation		■	■	■															
Problem solving			■	■	■	■			■	■									
Writing the documentation					■	■	■							■	■	■	■	■	
Implementation								■	■	■	■	■							
Validation													■						
Presentation creation						■								■	■				

Figure C.1: Timetable of the Semester Thesis

D How To

D.1 Compilation

To use the IP-FBSD gateway within the ANA framework first of all the source code has to be downloaded and compiled:

1. Get the source code with subversion [2]:

   ```
   svn checkout https://subversion.cs.unibas.ch/repos/ana/
   ```

   You need a username and a password to get access to this repository. These are provided by Christophe Jelger from the University of Basel.

2. Switch to the `devel` directory and get the configuration file:

   ```
   cd ana/ana-core/devel/

   cp C/bricks/fbr_ip_gateway/config.txt .
   ```

3. Compile the program for user space usage[1]:

   ```
   make
   ```

[1]`make` compiles the program for user space and/or kernel space depending on the configurations defined in the configuration file `config.txt`.

D.2 Gateway Node

To operate an IP-FBSD gateway a node has to load the IP, FBSD and gateway Bricks. Root privileges are needed to start the MINMEX and the Bricks.

1. Start the MINMEX:

   ```
   ./bin/minmex
   ```

   You can start the MINMEX as background process with “&” or open a new terminal for the following commands.

2. Start the virtual link Brick, vlink [1]:

   ```
   ./bin/mxconfig load Brick ./so/vlink.so
   ```

3. Configure the vlink:

   ```
   ./bin/vlconfig create 1
   ./bin/vlconfig add_if vlink1 eth0
   ./bin/vlconfig up vlink1
   ```

   For more detailed information to configure the vlink, refer to [1].

4. Start the Ethernet Brick:

   ```
   ./bin/mxconfig load Brick ./so/eth-vl.so
   ```

5. Start the FBSD (FBR) Bricks:

   ```
   ./bin/mxconfig load Brick ./so/fbr_diss.so
   ./bin/mxconfig load Brick ./so/fbr_ftab.so
   ./bin/mxconfig load Brick ./so/fbr_forw.so
   ./bin/mxconfig load Brick ./so/fbr_rtab.so
   ./bin/mxconfig load Brick ./so/fbr_potf.so
   ./bin/mxconfig load Brick ./so/fbr_api.so
   ```

6. Start the IP Bricks with IP address 10.0.1.2 (changeable) for the gateway node:

   ```
   ./bin/mxconfig load Brick ./so/ip_enc.so
   ./bin/mxconfig load Brick ./so/ip_sum.so
   ./bin/mxconfig load Brick ./so/ip_fwd.so
   ./bin/mxconfig load Brick ./so/ip_cfg.so i=10.0.1.2 \
   m=255.255.255.0 e=eth01
   ```

7. Start the gateway Brick:

   ```
   ./bin/mxconfig load Brick ./so/gw_serv.so
   ```

   This Brick implements the gateway application.

D.3 IP Gateway User

To operate a node in an IP network one has to load the IP Bricks. To enable the node to use the gateway (being a gateway-user) it needs to implement some basic functions. A sample gateway-user Brick for IP is provided. Root privileges are needed to start the MINMEX and the Bricks.

1. Start the MINMEX:

   ```
   ./bin/minmex
   ```

   You can start the MINMEX as background process with "&" or open a new terminal for the following commands.

2. Start the virtual link Brick, vlink [1]:

   ```
   ./bin/mxconfig load Brick ./so/vlink.so
   ```

3. Configure the vlink:

   ```
   ./bin/vlconfig create 1
   ./bin/vlconfig add_if vlink1 eth0
   ./bin/vlconfig up vlink1
   ```

   For more detailed information to configure the vlink, refer to [1].

4. Start the Ethernet Brick:

   ```
   ./bin/mxconfig load Brick ./so/eth-vl.so
   ```

5. Start the IP Bricks with IP address 10.0.1.3 (changeable) for the IP node:

   ```
   ./bin/mxconfig load Brick ./so/ip_enc.so
   ./bin/mxconfig load Brick ./so/ip_sum.so
   ./bin/mxconfig load Brick ./so/ip_fwd.so
   ./bin/mxconfig load Brick ./so/ip_cfg.so i=10.0.1.3 \
   m=255.255.255.0 e=eth01
   ```

   For more detailed information to configure the IP Bricks, refer to [9].

6. Sample IP gateway-user Brick:

   ```
   ./bin/mxconfig load Brick ./so/gw_clie_ip.so
   ```

   This Brick implements a sample IP gateway-user application and sends a file to the FBSD node via the gateway as soon as the FBSD application asks for the file.

D.4 FBSD Gateway User

To operate a node in an FBSD network one has to load the FBSD Bricks. To enable the node to use the gateway (being a gateway-user) it needs to implement some basic functions. A sample gateway-user Brick for FBSD is provided. Root privileges are needed to start the MINMEX and the Bricks.

1. Start the MINMEX:

   ```
   ./bin/minmex
   ```

   You can start the MINMEX as background process with "`&`" or open a new terminal for the following commands.

2. Start the virtual link Brick, vlink [1]:

   ```
   ./bin/mxconfig load Brick ./so/vlink.so
   ```

3. Configure the vlink:

   ```
   ./bin/vlconfig create 1
   ./bin/vlconfig add_if vlink1 eth0
   ./bin/vlconfig up vlink1
   ```

   For more detailed information to configure the vlink, refer to [1].

4. Start the Ethernet Brick:

   ```
   ./bin/mxconfig load Brick ./so/eth-vl.so
   ```

5. Start the FBSD (FBR) Bricks:

   ```
   ./bin/mxconfig load Brick ./so/fbr_diss.so
   ./bin/mxconfig load Brick ./so/fbr_ftab.so
   ./bin/mxconfig load Brick ./so/fbr_forw.so
   ./bin/mxconfig load Brick ./so/fbr_rtab.so
   ./bin/mxconfig load Brick ./so/fbr_potf.so
   ./bin/mxconfig load Brick ./so/fbr_api.so
   ```

   For more detailed information to configure the FBR Bricks, refer to [9].

6. Sample FBSD gateway-user Brick:

   ```
   ./bin/mxconfig load Brick ./so/gw_clie_fbr.so
   ```

   This Brick implements a sample FBSD gateway-user application. It tries to resolve two IP nodes with the addresses 10.0.1.3 and 10.0.1.4. It asks afterwards the successfully resolved IP nodes for a file. This files (including a log file) will be stored in the directory `/root/IP_FBSD_Gateway/`. To access this directory one has to type `cd /root/IP_FBSD_Gateway` into a terminal.

D.5 Start Nodes with Shell Scripts

To ease the start of an entire functionality for a node scripts are applicable. Root privileges are needed to start the scripts correctly.

1. Start a gateway node:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/gateway_start.sh \
   VLINK_ID INTERFACE IP_ADDRESS NETMASK
   ```
   Description:
   - `VLINK_ID`: Virtual link ID of the INTERFACE.
   - `INTERFACE`: Desired interface, e.g. eth0.
   - `IP_ADDRESS`: IP address of the node.
   - `NETMASK`: Netmask for the IP_ADDRESS.

2. Start an IP gateway-user node:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/ip_start.sh \
   VLINK_ID INTERFACE IP_ADDRESS NETMASK
   ```
   Description: (see above).

3. Start FBSD gateway-user node:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/fbsd_start.sh \
   VLINK_ID INTERFACE FBSD_ADDRESS
   ```
   Description:
   - `VLINK_ID`: Virtual link ID of the INTERFACE.
   - `INTERFACE`: Desired interface, e.g. eth0.
   - `FBSD_ADDRESS`: FBSD address of the node.

D.6 Start Scenarios with Shell Scripts

ANA allows to start multiple virtual ANA nodes on one physical node. This feature of ANA is applied in the following explained scenario scripts. To ease the start of entire scenarios 3 scripts are applicable. Root privileges are needed to start the scripts correctly.

1. Start an IP gateway-user node and a FBSD gateway-user node with a gateway node in between:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/ip_gw_fbsd.sh \
   NUMBER
   ```

   Description:
   - `NUMBER`: Number of runs in this scenario.

2. Start 2 IP gateway-user nodes and a FBSD gateway-user node with a gateway node in between:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/2ip_gw_fbsd.sh \
   NUMBER
   ```

   Description: (see above)

3. Start 2 IP gateway-user nodes and 2 FBSD gateway-user nodes with a gateway node in between:

   ```
   sh C/bricks/fbr_ip_gateway/scripts/2ip_gw_2fbsd.sh \
   NUMBER
   ```

   Description: (see above)

D.6.1 Sample Output

Terminal After starting a scenario script the last lines of the output in the terminal should look similar to the following (reordering possible):

```
########################
# ipconfig successful! #
########################
########################
# ipconfig successful! #
########################
logfile /var/log/ANA/15610_gw_serv
-> Brick's thread launched!

[sending command to udp port 10102]
```

```
logfile /var/log/ANA/15616_gw_clie_ip
-> Brick's thread launched!

[sending command to udp port 10101]

logfile /var/log/ANA/15613_gw_clie_fbr
-> Brick's thread launched!
```

To access the ANA logfiles one can enter for example `less /var/log/ANA/15613_gw_clie_fbr` into the terminal to look at the logfile of the FBSD gateway-user. The needed path after `less` can be found in the terminal output of the started Brick after the term `logfile`.

Transmitted files Files which the FBSD application receives from the IP application are stored in the directory `/root/IP_FBSD_Gateway/`. But the scenario scripts delete automatically all files which arrived correctly (only for the default file `agent.txt`). The `agent.txt` files found in the directory `/root/IP_FBSD_Gateway/` therefore are faulty.
To compare the faulty file with the original file one can access the original file in the directory (relative to the svn checkout directory): `ana/ana-core/devel/C/bricks/fbr_ip_gateway/`.

Logfiles 2 logfiles are created by starting a scenario script if they do not exist yet in the directory `/root/IP_FBSD_Gateway/`. The first one (log.csv) gives information about the file transmission and will be overwritten with each script start. A sample `log.csv` file looks as follows:

```
####### gatewayDemo log file #######
Date:  Fri Jan 9 17:33:00 2009

Global seetings:
Application segment size:  500


Received files:

Filename, Time seconds, Time useconds, Nb.  of Data Messages, Data
bytes, Total bytes, Sending interval
agent.txt, 0, 36481, 0, 0, 22, 3
```

The second logfile (`***log.csv`, where * are either `1` or `2`) can be used for measurements. For each of the three scenarios an own logfile exists. The delivery time of the file within the scenario is stored in the according logfile of the started scenario. Each new script start attaches the new values to the end of the respective logfile.

Bibliography

[1] *ANA Core Documentation.* http://cn.cs.unibas.ch/software/anacore-doc.pdf [Online; accessed 23-January-2009].

[2] *Subversion.* http://subversion.tigris.org/ [Online; accessed 5-January-2009].

[3] *Global System for Mobile Communications (GSM)*, 2008. http://www.gsmworld.com/technology/gsm/index.htm [Online; accessed 3-January-2009].

[4] *How the Internet Layer Works*, January 2008. http://learn-networking.com/tcp-ip/how-the-internet-layer-works [Online; accessed 10-October-2008].

[5] *The ANA Project*, October 2008. http://www.ana-project.org/ [Online; accessed 10-October-2008].

[6] Rainer Baumann. *Building Scalable and Robust Wireless Mesh Networks (Diss. ETH No. 17306)*, 2007. Dissertation, ETH Zurich, Switzerland. http://rainer.baumann.info/public/eth_diss_17306.pdf [Online; accessed 12-January-2009].

[7] C.Hedrick. *Routing Information Protocol.* Network Working Group, June 1988. http://tools.ietf.org/html/rfc1058 [Online; accessed 10-October-2008].

[8] C.Partridge, T.Mendez, W.Milliken. *Host Anycasting Service.* Network Working Group, November 1993. http://tools.ietf.org/html/rfc1546 [Online; accessed 2-January-2009].

[9] Stephan Dudler. *New Protocols and Applications for the Future Internet (MA-2007-39)*, March 2008. Master Thesis, ETH Zurich, Switzerland. ftp://ftp.tik.ee.ethz.ch/pub/students/2007-HS/MA-2007-39.pdf [Online; accessed 12-January-2009].

[10] Mathias Fischer. *Autonomic Identifier Allocation for ANA (AIA) (SA-2008-23)*, January 2009. Semester Thesis, ETH Zurich, Switzerland.

[11] Internet Engineering Task Force. *Requirements for Internet Hosts – Communication Layers.* Network Working Group, October 1989. http://tools.ietf.org/html/rfc1122 [Online; accessed 2-January-2009].

[12] G.Malkin, Bay Networks. *RIP Version 2.* Network Working Group, November 1998. http://tools.ietf.org/html/rfc2453 [Online; accessed 20-January-2009].

[13] Information Sciences Institute, University of Southern California. *Darpa Internet Program Protocol Specification*, September 1981. http://tools.ietf.org/html/rfc791 [Online; accessed 10-October-2008].

[14] Information Sciences Institute. *Transmission Control Protocol.* Network Working Group, September 1981. http://tools.ietf.org/html/rfc793 [Online; accessed 22-December-2008].

[15] J.Postel. *User Datagram Protocol.* Network Working Group, August 1980. http://tools.ietf.org/html/rfc768 [Online; accessed 22-December-2008].

[16] Larry L.Peterson, Bruce S.Davie. *Computernetze: Eine systemorientierte Einführung.* dpunkt.lehrbuch, 2004.

[17] P.Mockapetris. *Domain Names - Implementation and Specification.* Network Working Group, November 1987. http://tools.ietf.org/html/rfc1035 [Online; accessed 22-December-2008].

[18] P.Srisuresh, K.Egevang. *Traditional IP Network Address Translator (Traditional NAT).* Network Working Group, January 2001. http://tools.ietf.org/html/rfc3022 [Online; accessed 30-December-2008].

[19] R.Droms. *Dynamic Host Configuration Protocol.* Network Working Group, March 1997. http://tools.ietf.org/html/rfc2131 [Online; accessed 19-December-2008].

[20] Richard Gold, Per Gunningberg, Christian Tschudin. *A virtualized link layer with support for indirection.* In *FDNA '04: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, pages 28–34, New York, NY, USA, 2004. ACM.

[21] S.Cheshire, B.Aboba, E.Guttman. *Dynamic Configuration of IPv4 Link-Local Addresses.* Network Working Group, May 2005. http://tools.ietf.org/html/rfc3927 [Online; accessed 19-December-2008].

[22] Simon Heimlicher, Rainer Baumann, Martin May, Bernhard Plattner. *The Transport Layer Revisited.* 2nd IEEE International Conference on Communication System Software and Middleware, IEEE COMSWARE 2007, Bangalore, India, January 2007.

[23] Vincent Lenders, Martin May, Bernhard Plattner. *Service discovery in mobile ad hoc networks: A field theoretic approach.* Technical report, ETH Zurich, Switzerland, 2005.