ACKNOWLEDGEMENT

Writing a thesis is done by a single person: the one who graduates. However, doing research and gaining knowledge can not be done without the help of other people. Writing a protocol, aimed to be an international standard, is, despite that it probably will be published as an “experimental” instead of a “standard” protocol, is not an effort I would attempt on my own. It is sometimes hard to discriminate between the work I did on my own and the work done by others. I have written the chapter about PAP, CHAP and EAP (called “Supporting various client Authentication Protocols”), but I lost count on the number of times others authors rewrote it. Likewise, I hope some of my input improved the other chapters.

Co-authors

My gratitude goes to all co-authors. In particular I would like to thank John Vollbrecht for his endless questions and patient answers. He did make me feel comfortable, in particular when I presented some slides at the IETF conference for a room full of people. My thanks also go to Walter Weiss, who pushed the drafts and got everybody writing. He astounded me twice by compiling the draft at the last minutes before the deadline, even though that seemed impossible to me. I still wonder how he managed it – even if he didn’t sleep! A special thanks goes to Ravi Sahita who kindly took the time in Salt Lake City to sit down and explain some details about the accessor issues which must have been obvious, but I completely misunderstood. Gratitude to Kwok Ho Chan, who showed in Minneapolis that he is a good technician and lobbyist, and is nice to talk to. A last thanks goes to Dave Durham, who showed, when I feared disaster when I had to give a talk to the RAP group, there was nothing to fear about. In the mean time he elevated my knowledge about what I was doing.

I am in debt to Cees de Laat, who supervised me, and kept an eye on what I was doing, even though he moved from Utrecht University to University of Amsterdam during work. In particular I am very grateful for the way he allowed me to work on my own pace, but still took the decision to get me on track when I got stuck in the original 802.1x work. Also, he was very generous by sponsoring my attendance at two IETF-conferences, held in the USA andreserving money from his research budget to cover my expenses there.

Family and friends

A hug and thanks go to my parents, Hennie and Wil Dijkstra, who showed me that they have faith in me, despite the fact it took me a long time to get to the point where I am now.

I would like to thank everybody who pre-reviewed my thesis and gave valuable feedback, particularly my dad, for catching logical errors and to Floris de Vooys, for keeping overview. Love and kisses to Caroline Mattheij, for all the typos she caught. Additional thanks to John Vollbrecht, Leon Gommans, David van de Vliet and Reinoud Koornstra for their comments. Last, thanks to all others: Richard, Frank, Nils, Thijs, Jeroen, Arjan, Guus, René and Thomas.

My computer

Despite that I’m a Macintosh “Freak”, hardly any thanks goes to the applications on my Mac, by crashing as much as 271 times since I started to write my thesis (101 days). Only thanks to MacsBugs I didn’t loose count. (Least stable apps: Internet Explorer – 82 crashes; MS Word – 34 crashes Opera 5 – 21 crashes; most stable apps: Nifty Telnet, BBEdit and Drag Thing, which I used just as often but “only” crashed respectively 2, 2 and 3 times).

http://folk.uio.no/paalee/
ABSTRACT

In order to gain access to a network, users connect to an access device. Usually, network providers have multiple access devices, and one central device. Such a central device can be used for two distinct functions. First, for provisioning: the central device configures each access device with a certain policy. Second, this central device can be used as an authentication server that authenticates each user who connects to an access device. An example of the first function would be a server that configures multiple edge devices from an ISP. An example of the second function would be a dial-in server (network access device) where users authenticate themselves using PAP or CHAP, and where the dial-in server checks the identity at a central authentication server.

Until now, it was not possible to create a device that combines automatic configuration and per-session authentication using a single protocol. The RAP group, an IETF working group, has described a framework and created a data structure, which can combine these two functions when the data instances are transported between an access device and a central server. The data structure is described in an Internet draft called Framework for Binding Access Control to COPS Provisioning (the “Access Bind PIB” for short).

This thesis will describe the situation from the perspective of a generic AAA-architecture (Authentication, Authorisation and Accounting), as defined in RFC 2903. It will describe the limitations of the draft, and on which existing technologies it is based: the DiffServ model, the COPS and COPS-PR protocol and EAP.

About the author

Freek Dijkstra is a student of Utrecht University and has received his master graduation (doctorandus title) based on this thesis on 9 September 2002. He has been actively involved in creating the Access Bind PIB during 2001 and 2002.
TABLE OF CONTENTS

ACKNOWLEDGEMENT .................................................. 2

ABSTRACT ............................................................. 3

TABLE OF CONTENTS .................................................. 4

I. THE ACCESS BIND PIB ............................................. 7
   1. Why the Internet draft was written ......................... 8
      1.1. Merging Access Control and Provisioning .............. 8
      1.2. Research questions ..................................... 10
      1.3. Involvement of the RAP group ......................... 10
   2. What is the Access Bind PIB? ............................... 11
      2.1. Model ...................................................... 11
      2.2. A data structure ....................................... 12
      2.3. Usage of other technologies ........................... 12
   3. What is COPS? .................................................. 13
      3.1. COPS, COPS-RSVP and COPS-PR ......................... 13
      3.2. PIB's, PRI's, PRD's and PRID's ....................... 14
      3.3. COPS and the Access Bind PIB ......................... 15
   4. The DiffServ model .......................................... 16
      4.1. The DiffServ elements .................................. 16
      4.2. DiffServ and the Access Bind PIB ................. 17
   5. PPP Authentication Protocols .............................. 18
      5.1. Authentication Protocols ............................... 18
      5.2. PPP ....................................................... 18
      5.3. Security concept of PPP ............................... 20
      5.4. PAP ...................................................... 21
      5.5. CHAP ..................................................... 22
      5.6. EAP ...................................................... 23
      5.7. EAP and the Access Bind PIB ........................... 25
   6. Message Sequence ............................................. 27
      6.1. Message sequence overview ............................. 27
      6.2. Configuration of the PEP ............................... 28
      6.3. Triggering of an event ................................. 29
      6.4. Authentication of a user ................................ 30
      6.5. Per-User provisioning of the PEP ..................... 30
      6.6. Combining messages .................................... 31
   7. Data structure ................................................ 32
      7.1. The eventHandler as a classifying DiffServ element .. 32
      7.2. Events and event criteria ............................. 35
      7.3. ContextData pointer and ContextData structures ... 38
      7.4. Authentication Data structure ........................ 40
      7.5. Provisioning data structure ........................... 42
   8. Discussion and Conclusions ................................ 44
      8.1. Fitting the Access Bind PIB in the authorisation models... 44
      8.2. Potential Usage as a generic AAA-protocol .......... 45
      8.3. Potential usage as a NAS authentication protocol .... 46
      8.4. Edge Devices and Network Access Servers ............ 48
      8.5. Suggestions for improvement .......................... 48
      8.6. Combining authentication and provisioning .......... 50
II. ACRONYMS AND BUZZWORDS

1. What is a network protocol? 54
   1.1. The OSI model and the Internet model 54
2. What are standards and who makes them? 56
   2.1. What is a standard? 56
   2.2. Standards in computer science 57
   2.3. Which standardisation bodies are out there? 57
3. What is the IETF? 60
   3.1. The organisation of the IETF 61
   3.2. Network Protocols in the IETF 61
4. What is the IRTF? 63
5. AAAarch! 64
6. Generic Authentication Concepts 66
   6.1. Examples 66
   6.2. Credentials and trust-relations 70
   6.3. Glossary 72
7. Authorisation models 75
   Pull model 75
   Push model 76
   7.3. Agent model 76
   7.4. Internals of an AAA-Server 77
8. What is the RAP group? 79
   8.1. Operation and Management protocols 79
9. Quality of Service 80
   9.2. The sense and non-sense of QoS 81

III. BACKGROUND ISSUES 83
1. Embedding of data in COPS-PR messages 84
   1.1. COPS message sequence 84
   1.2. Embedding of instances in COPS messages 84
   1.3. C-nums, C-types, S-nums, S-types and the others 85
   1.4. Example of an embedded COPS message 85
2. Future of EAP 87
   2.1. IANA considerations 87
   2.2. Interaction between EAP and other layers and state machine 87
   2.3. Efficiency and piggy-backing 87
   2.4. Tunnelling PPP and EAP over other protocols 88
   2.5. Improving security 89
3. Student housing example 90
   3.1. The IBB-complex 90
   3.2. SSHUUnet 90
   3.3. The Access Bind PIB 91

INDEX 95

TABLE OF FIGURES 97

REFERENCES 98
I. THE ACCESS BIND PIB

The goal of this thesis is to explain what the Access Bind PIB [draft-ietf-rap-access-bind] is about, and why it was created. In particular, it reviews the Access Bind PIB for suitability as a component in a generic AAA-model. This thesis will assume basic knowledge of Authorisation models [RFC 2904].

The goal of the Access Bind PIB is to merge QoS provisioning and access control. Provisioning is the ability of a Policy Decision Point (PDP) in a central server to configure a policy on a Policy Enforcement Point (PEP). A PEP is an entity that resides in a device, typically an edge device, like a dial-in server, but it can also reside in a core router. The PDP typically resides in a central server like an Authentication, Authorisation and Accounting server (AAA-server). Access control is the ability to control which services may be used, typically based on who wants to gain access (authentication).

The Access Bind PIB can be used in the AAA-model as an Application-Specific “protocol” between a PEP and a PDP. It is application-specific because the provisioning needs to be specified using elements described in the DiffServ model [RFC 2475]. The Access Bind PIB is capable of binding authentication to provisioning.

Readers who are not familiar with the terminology used in this thesis may wish to read the Appendix “Acronyms and Buzzwords”, which introduces the concept of network protocols and authorisation models.

---

1 “Protocol” between quotation marks, because the Access Bind PIB does not actually define a new protocol, but a data structure. This is explained in the thesis.
1. Why the Internet draft was written

The Access Bind PIB was created by the RAP working group to combine authentication (access control) and per-session provisioning. It effectively combines the outsourcing model (PEP outsources authorisation decision to a PDP) and the provisioning model (a PDP configures a PEP).

The research questions answered in this thesis are to describe and co-author this protocol and to check if it can be used as component in a generic AAA-model.

1.1. Merging Access Control and Provisioning

Access Control is the ability to control which service may be accessed. Examples of access control are a firewall, which only allows access to a limited set of services, and applications that require authentication before a user can use them.

Currently, a common situation is that users who want to access a network, like the internet, need to give their username and password in order to use the network. This is called Authenticated Access Control: there is control over who may use the network and who may not, based on authentication. In these situations, someone who connects to the network has two options: either he or she will get the default access, or no connection at all.

On the other hand, ISP’s for large businesses, usually offer multiple services. For example, an ADSL customer may choose between different bandwidths. These parameters, like bandwidth, latency or fault tolerance, together specify a certain Quality of Service (QoS). For these larger customers, access is often based on the source address rather than username and password combination. However, this solution doesn’t scale very well: it is manageable to manually configure the QoS for a small number of customers, based on source addresses. However, if the number of customers grows or the source addresses change for the ISP, managing all these policies will become a tedious and thus expensive job, with the risk that one of the devices is malconfigured.

In order to scale the two solutions above, most ISP have a central server, which is called a PDP (Policy Decision Point) throughout this thesis. The network access device, the service where users connect to, is an example of a service equipment were the policies of the ISP are enforced. These devices will be called PEP’s (Policy Enforcement Points) throughout this thesis.

Outsourcing the authorisation decision

The first reason to use a central PDP server is to outsource the authorisation decision.

Observe that for network access, most access control mechanism use the pull model [RFC 2904], as shown in Figure I-1: the User only contacts the PEP, a device that offers a service the user want to use. Typically, the Username and a challenge are received by the PEP. However, the PEP does not have a long list with all passwords,

---

2 This is especially true for Internet access providers whose clients use modems, or for mobile telephony providers, but also ADSL providers start to request their users to sign in before they can use the network.

3 The only notable exception is Kerberos. Outside the networking world, the push model is more common. For example tickets, certificates and money use the push model.

4 The challenge is a slightly more secure way of sending a password in the clear, if you don’t know how a challenge/response protocol works, see CHAP in section 0.
so it outsources the authorisation decision to a device specifically tailored to do this job, the PDP.

![Diagram](image)

**Figure I-1. The pull model, with a user who wants to access a service from the PEP (Policy Enforcement Point). The decision to allow or deny the access request is outsourced by the PEP to the PDP (Policy Decision Point)**

The protocol between the User and the PEP may be PPP for the usage of service, preceded with CHAP or EAP as the authentication protocol at the start.\(^5\)

Note that none of these protocols have an option to negotiate the type of service. PPP has the capability to negotiate some options, but none of these options is really concerned with the Quality of the Service (QoS). Currently, differences between the Quality of Service, like a difference between offered bandwidth, are based on fixed parameters configured in the PEP or PDP. For example, a provider may have configured the bandwidth based on the user who logs in (more expensive subscriptions get a higher bandwidth). However, this is not negotiable by the client. If a User has a low bandwidth subscription, he\(^6\) can’t dynamically get a higher one by asking the PEP, even if he is willing to pay for it. He needs to contact a person at the provider who manually changes the configuration.

Today, the conversation between the PEP and the PDP is typically done using the RADIUS (Remote Authentication Dial In User Service). RADIUS is currently being extended and completely rewritten to a similar, but non-compatible protocol called DIAMETER. RADIUS is currently very widely used, and DIAMETER is fiercely pushed forward by its authors.\(^7\)

**Provisioning the PEP**

It is possible to configure one or more devices by a central server. Typically, the SNMP protocol is used for this task. SNMP means Simple Network Management Protocol, but was extended over time and the term “Simple” is probably not a good description for it anymore. SNMP, however, has no option to perform an authentication and provision a PEP on a per-user or per-session basis.

Using DIAMETER or RADIUS, the PDP can say “yes, grant this user access” or “no, deny access for this user”. It is possible to configure the PEP based on the user that

---

\(^5\) PPP is typically for dial-up connections, and to some extend in LAN application using PPP layered over another protocol. The protocol between the User and the PEP may also be 802.11 (Wireless LAN), with EAP as the authentication protocol.

\(^6\) Where the text reads ‘he’, you should read ‘he or she’. Unfortunately, Computer Science is still dominated by males, so I’ll keep the text short and just write ‘he’.

\(^7\) Patrick Calhoun, one of the authors, even bought a license plate for his car reading ‘DIAMETER’.
wants to log in, by using specific RADIUS or DIAMETER attributes. So for example, if a small business user who has a low-bandwidth account logs in, the PDP can provision the PEP to limit the bandwidth usage for this particular session. However, RADIUS and DIAMETER can only do per-user provisioning. It is not possible to configure the PEP at boot time using DIAMETER or RADIUS.

Summarising: a provider who wishes to use Access Control (username of the person who just logged in) to configure the PEP in real time with Quality of Service parameters like bandwidth, is forced to use two protocols: RADIUS and SNMP.

That is an unsatisfying situation. The most obvious problem is the co-operation of the two protocols. RADIUS and SNMP where never designed to work together, so they don’t share their states. For example, it is possible that a user disconnected and the RADIUS interface is already aware of this situation, but the SNMP interface is not. This can lead to unexpected results.

It is possible to make one device which supports multiple protocols, and which has an internal mechanism to keep the states synchronised. However, if one of the protocols is replaced with another one, for example if RADIUS is replaced with DIAMETER, this mechanism needs to be re-examined.

So it would be a much better idea to have a protocol or state mechanism that merges the outsourcing behaviour, in particular the access control outsourcing of RADIUS, and the provisioning behaviour like SNMP exhibits. This merger is what a new internet document, called the Framework for Binding Access Control to COPS Provisioning attempts to do.

1.2. Research questions
My research focuses on two assignments:

1. Help to create a protocol that combines provisioning and access control.

2. Check if the created protocol can be used as a component in the generic AAA-model.

1.3. Involvement of the RAP group
Creating a protocol is done by a group of people. In this case, the creation of the protocol was a work item of the RAP group [RAP-charter], a working group in the IETF, in collaboration with a few people from the AAAarch research group [AAAARCH-charter] on personal basis.

So far, I explained the virtues of adding provisioning to a protocol capable of Access Control. This is exactly the opposite of the protocol that was developed in the RAP group, COPS. COPS is already capable of provisioning. However, it was currently not possible to do access control with it as well. Due to a growing demand for scalability, security and accounting, it is desirable that when someone does a request, such a request is authorised. Therefore, the RAP group wanted to add access control to COPS. In order to gain some input, they contacted a few people of the AAAarch research group, who had existing knowledge about authorisation models and authorisation mechanisms.

---

* A relative harmless situation is that the SNMP interface keeps the high-bandwidth connection up, although the connection was closed and an intruder can gain unauthorised access to this connection.
2. What is the Access Bind PIB?

The Access Bind PIB is defined in an Internet Draft. It describes a data structure that can be sent between a PEP (Policy Enforcement Point) and a PDP (Policy Decision Point). It uses COPS for the transport of the data instances. It is to be deployed in a configuration described in the pull model.

The Access Bind PIB assumes to provide access control to DiffServ based services. It supports authentication protocols like EAP, PAP and CHAP.

The Access Bind PIB is a short name for a document called “Framework for Binding Access Control to COPS Provisioning”.

The document has, at the time of this writing, seen two revisions; each made publicly available as an Internet Draft. These drafts where called draft-ietf-rap-access-bind-00 (published in July 2001), respectively draft-ietf-rap-access-bind-01 (published in March 2002).

Note that an Internet-Draft is a work in progress. Internet-Drafts are publicly available, but they are deleted from the repository after six months. After that time, either they should be published as RFC, or a new revision should be issued. If that’s not the case, it is expected that no one is still working on it, and they are abandoned.

2.1. Model

Figure I-2 below shows a pull model. The Access Bind PIB is a protocol that operates between the PEP and the PDP.

![Diagram](image)

Figure I-2. The Access Bind model (a pull model)

Typically, a user sends traffic to the PEP, and if the traffic is coming from an unknown origin, the PEP expects someone new is trying to log in. The PEP will contact the PDP and sent a set of pre-defined properties of this new connection to the PDP. This might already be a username and password if the User already sent those. The PDP will then authenticate the user, or if it does not have enough information, contact the user via the PEP and an authentication session is started. When that is done, the PDP will signal the PEP of the result, and optionally configure the PEP.

For the configuration, the DiffServ model is used. See chapter 4 for more information about the DiffServ model.
For efficiency, the PDP can also configure the PEP at boot time. For example, the PDP can tell the PEP what properties the PEP needs to send when a new user connects; or the PDP can set up components for a default connection, a meter and a tunnel, using DiffServ elements.

2.2. A data structure

A PIB is actually just a data structure, so all the Access Bind PIB does is define a data structure. To transfer the data between the PEP and the PDP, a protocol, called COPS, is used. PIB, or Policy Information Base, is the COPS terminology for a (policy related) data structure definition. Section 3 will go into a bit more detail about COPS.

2.3. Usage of other technologies

Instead of re-inventing the wheel, the Access Bind PIB makes use of a few existing technologies. For the transmission of the data, the synchronisation of the states between the PEP and the PDP, COPS, and in particular COPS-PR is used.

The main purpose of the Access Bind PIB is to merge authentication with provisioning. For the provisioning, the DiffServ model is used. Even though the Access Bind PIB aims to support any authentication protocol between the User and the PEP, in practice, it needs to describe mechanism for each Authentication Protocol used: PAP, CHAP, EAP, HTTP, SRP, etc. However, due to the extensible nature of EAP, the focus is on EAP.

COPS, The DiffServ model and authentication protocols like EAP are described in the next few chapters.
3. What is COPS?

COPS (Common Open Policy Service Protocol) [RFC 2748] is a protocol developed by the RAP group. Whatis.com describes it as “a standard protocol for exchanging network policy information between a policy decision point (PDP) and a policy enforcement point (PEP)”. Part of this policy information is the Quality of Service (QoS) – information how network traffic resources are allocated to services and users. The policy decision point might be a network server controlled directly by the network administrator who enters policy statements about which kinds of traffic (voice, bulk data, video, teleconferencing, and so forth) should get the highest priority. The policy enforcement points might be routers that implement these policies as traffic moves through the network.

COPS send data back and forth between a PEP and a PDP. COPS messages are in a binary format. The appendix “Background and details” explains how data is encoded in COPS-PR messages.

3.1. COPS, COPS-RSVP and COPS-PR

COPS is a client/server model that exchanges policy information between a PDP and a PEP. The PEP is the client, and the PDP is the policy server.

COPS was created to create a reliable mechanism to provision a PEP by a PDP. It’s most notable virtues are the reliability and its state synchronisation mechanisms. Typically, a PEP sends a request to a PDP and the PDP responds with a decision message. In addition, the PEP can send report state messages to the PDP when the decision is successfully installed, or when a change in the state of a session occurs.

Currently, there is one protocol that use COPS to exchange policy information: COPS-RSVP [RFC 2749], and one protocol, COPS-PR [RFC 3084], that extends COPS to allowing the PDP to provision (configure) a PEP. There are also about five drafts (works in progress), which use COPS, but it is not certain that all these drafts will actually be published as a new standard protocol.

The Access Bind PIB uses COPS, and in particular COPS-PR. Because COPS-PR is used for the transport of data, all the Access Bind PIB needs to do is to define the data structure of the data that needs to be sent. Things like reconnecting and synchronisation of states is left to the COPS protocol.

COPS-RSVP

COPS-RSVP is relative straightforward usage of COPS. It allows for a PEP (in this case often a router) who receives RSVP messages [RFC 2205], to ask a PDP how to respond to these messages. RSVP (Reservation Protocol) is used by a client to reserve bandwidth along a route.

COPS-PR

The other usage of COPS, COPS-PR (short for COPS Provisioning), is very unlike the COPS-RSVP protocol. In fact, COPS-PR does not define any specific usage for COPS, but merely expands it, so that COPS can sent particular objects called PRI’s, where a
PRI can be any type of data. PRI’s (Provisioning Instances) can be seen as records in a table, or –using the terms in the specification– as Instances of Classes. These classes, indeed called Provisioning Classes or PRC’s for short, are defined in a specification called a PIB, a Policy Information Base. A PIB describes what kind of data is sent between a PEP and a PDP.

The way how COPS-RSVP waits for an event (an incoming message) to happen, and then asks another device (the PDP) what to do with it, is called the Outsourcing model: The PEP outsources the decision making to the PDP.

COPS-PR focuses on another model, called the Provisioning Model, or the Configuration Model. At some time, probably boot time, the PEP wants to be configured and requests the PDP for a configuration. This can for example be a set of elements as described in the DiffServ model. Here the PEP does not need a specific incoming message. The advantage is that the PDP does not need to be bothered for each incoming message, because the policy knowledge is already configured in the PEP. This could be time-saving for Network operator who has a lot of PEP’s: when he installs a new PEP device in the network, all he has to do is give the IP address of the PDP device, and boot the PEP. If all goes well, the PEP is then automatically configured by the PDP based on the policies implemented by this particular Network operator. We will later see that this difference between “plain” COPS (using the Outsourcing model) and COPS-PR (using the Provisioning model) is actually not there, and that the methods described in COPS-PR could also be used to implement the Outsourcing model, effectively merging the two models.

3.2. PIB’s, PRC’s, PRI’s and PRID’s

There are many ways to describe a data structure. A PIB, a Policy Information Base, is one way to do this. As the name suggests, the data structure described in a PIB is aimed at describing Policy Data. A PIB is just a plain text, which defines a particular data structure. The syntax of a PIB is defined in SPPI [RFC 3159], Structure of Policy Provisioning Information.

The basic structure is just a table with data in it: each row or record contains several fields. For example, you can define a data structure called a person, with fields for the name, date of birth, sex and the address. A more common example in Computer Science: an interface, which consists of an IPv4 field and a port number. The first table can contain zero, one or more persons and the other table can contain any number of interfaces.

PRC and PRI

A PIB uses the terms Classes and Instances, with the abbreviations PRC and PRI, instead of tables and records. A PRC is the Provisioning Class and a PRI is a Provisioning Instance, an instance of a PRC. So a PIB defines one or more PRC’s and the actual data that is sent between a PEP and PDP is one or more PRI’s.

A PIB not only defines the syntax tables, field names, and structure of each field; it also contains some meta-information and semantics about each PRC (table) or attribute (field). For example, it gives an access code for each PRC, explaining if the PEP should understand and check the syntax of contents, or if the contents should just be passed along. It also specified if the PDP may create new instances and sent it to the PEP, or only the PEP may do that. It states if the PEP should keep the data

---

9 SPPI is a variant of SMI [RFC 2578], Structure of Management Information. Because of this close relation between SPPI and SMI, a PIB looks very similar to a MIB, a Management Information Base, which is a data structure that uses the syntax of SMI. Both SMI and SPPI are a subset of ASN.1, a data definition language defined by ISO. MIB’s are the data structure that are used by SNMP. MIB’s and PIB’s are for a great deal interchangeable, as is described in Appendix A of RFC 3159 [RFC 3159].
(stateful information) or if it should process it and throw it away (stateless). Last, a PIB can add a human-readable description to each PRC and each field.

The advantage of creating a PIB instead of describing all PRC’s and fields in a normal text is that a PIB can be read by a parser. In fact, IBM offers such a parser to be downloaded for free on their website, called the Policy Information Base (PIB) Control Interface Generator (CIG), as part of their COPS Client Software Development Kit (SDK). (Yes, more abbreviations here).

**PRID’s**

COPS-PR describes how PRI’s are sent between a PEP and a PDP. Each PRI that is embedded in a COPS-PR message is preceded by a PRID, a Policy Repository IDentifier. The PRID serves two purposes: it tells the recipient what type of Instance to expect (the PRI is an instance of which PRC?), and it also uniquely identifies the particular PRI, so that other classes can refer to it.

3.3. **COPS and the Access Bind PIB**

The Access Bind PIB is actually a PIB: a description of different data classes. For example, one of the classes that is described in the Access Bind PIB is an ContextData class, a class of which each instance contains information about a session (the context of this session). For example, a session is a user who contacted the PEP to gain access, and the context of a session might include the telephone number from where the user is dialing in and the modem number he or she got connected to.

The Access Bind PIB uses COPS as the protocol to manage sessions, keep the state synchronised and to transport data between the PEP and the PDP.

The reason why COPS was chosen as the protocol to transport data structures is greatly due to the very reliable mechanism that COPS has to provide for state synchronisation, roll-back and confirmation of received messages.

The fact that COPS was proposed as a candidate as a generic AAA-protocol (as described in AAA protocol evaluation [RFC 3127]) and the suggestion that COPS is slightly faster protocol then for example SNMP where not convincing arguments.
4. The DiffServ model

The DiffServ model describes a PEP as a collection of elements through which data traffic flows along a data path. The DiffServ model describes these elements. The goal is to describe a QoS (Quality of Service) policy using these elements.

DiffServ is short for Differentiated Services, and the DiffServ model [RFC 2475] specifies elements are needed in a device needed to threat different data streams in a different way. For example, a data stream coming from one user gets priority over other traffic; or the device knows that a certain packet stream is audio, and in case of congestion, packets should be dropped instead of queued. This differentiation of services can be seen as offering a different Quality of Service (QoS) to different customers or to different types of data streams.

4.1. The DiffServ elements

The DiffServ model defines the elements inside a device. Such a device is usually called a PEP, a Policy Enforcement Point, because it enforces certain rules on the traffic. Figure I-3 shows an example of a PEP.

![Figure I-3. Example of DiffServ elements inside a PEP.](source)

In this example, traffic flows from left through right through the device. The flow of a traffic stream is called the **datapath**. The data first encounters a classifier, which is an entity that selects packets based on the content of packet headers according to defined rules. In this example, traffic from subnet A is encapsulated in a tunnel and forwarded to a queue. Premium traffic is sent to a meter, which is an entity that measures the temporal properties of a traffic stream: the rate or bandwidth. If the traffic goes too fast (out-of-profile traffic), the packet is sent to a dropper, which discards the packet. If it is in-profile, the packet is sent to a queue.

Technically, the document doesn’t describe a queue, but a shaper, which is an entity that delays packets within a traffic stream to cause it to conform to some defined traffic profile.

The classifier is one of the most important entities in the PEP, because it is separates one datastream, or one type of datastream from another. The selection may for example be based on the header of a packet. In case of an IP packet, by the source or
destination IP address, or by looking at the part of the IP header that describe the type of the packet: the ToS (Type of Service) bits for IPv4, or the Traffic Class bits in IPv6. Originally IPv6 defined a 4-bits sized Traffic Class (See for example [Tanenbaum (1996)]), with the first bit describing what to do in case of congestion: drop the packet or try to queue it, and the other 3 bits describing the priority (001 should be a low priority NNTP connection, and 110 for high priority telnet connection). The final IPv6 specification [RFC 2460], [Deering (2002)] defines a 8-bit Traffic Class, the same as IPv4, with a complex specification defined in RFC 2474 [RFC 2474]. In fact, so complex, that someone later discovered that for IPv4 it did conflict with other specifications [RFC 2873]. Another IPv6 field, which is lacking in IPv4, is relevant here: the Flow label field, which can be used specifically for this purpose: segregation between different kind of datastreams. An entity called a marker can assign some random number to this flow label, and further down the datapath, or perhaps in an other device, a classifier can use the this value to split this traffic from other traffic.

4.2. DiffServ and the Access Bind PIB

The Access Bind PIB wants to combine authentication (access control) to provisioning (configuration of a PEP by a PDP). It was decided to use the DiffServ model to describe the provisioning of the PEP.

The DiffServ model describes the PEP as a collection of different elements, where the data comes in at one element, and each element can process the data and forward it to another element, or to a port on the device. Examples of elements are a dropper, a queue, meter and a classifier.

In this model, the Access Bind PIB is a complex DiffServ element, a Classifier. The idea of a classifier in the DiffServ model is that it examines data traffic, and classifies is according to the header or it’s contents. For example, it may classify one type of traffic as having a high priority. Based on this classification, it may forward the data to other elements. The high priority traffic is, for example, forwarded to a high-priority queue, while low priority traffic is forwarded to a meter, which may decide to drop the packets if the line is saturated.

You may look at the classifier as a kind of filter: if the traffic data matches a certain filter, it is forwarded to one next DiffServ element, if it matches a second filter, it is forwarded to another next DiffServ element, and if it doesn’t match either filter, it is forwarded to yet another DiffServ element.

The classifier is not a new element in the DiffServ model. Existing classifiers would just classify (filter) data, and forward it. These classifiers are configured by a system maintainer, or are installed by a PDP. That is a static situation, which can only be changed if the maintainer alters the configuration or the PDP installs a new element in the PEP.

What is new, is that the Access Bind PIB classifier has some intelligence, and can communicate directly with both the User (the source of the Data) and the PDP. For example, if unknown traffic is coming in, it is not just forwarded to the next DiffServ element, but the classifier will gather some information about the packet (possibly authentication information), and will forward this to the PDP, which will authenticate this information and sent a decision back to this classifier in the PEP. Based on this decision, the PEP can add this new type of data to one of it’s lookup tables, and either forward it to the DiffServ element matching traffic, or a the element for non-matching traffic (in case the PDP returned a negative decision).
5. PPP Authentication Protocols

For network access, the user talks to a network access server (NAS). There are three commonly used authentication protocols for communication between a User and a NAS: PAP, CHAP and EAP. PAP and CHAP are often used for dial-in access, and EAP is often used in wireless networks.

All these protocols are deployed in the pull model, where the authentication decision is outsourced by the NAS to an authentication server. A major advantage of EAP over PAP and CHAP is that EAP is extensible. If the NAS supports EAP, it is able to support any current or future authentication mechanism. PAP and CHAP only support one authentication mechanism.

This chapter will introduce three protocols, PAP, CHAP and EAP, each of which can be used to authenticate a user to a service. Since these authentication protocols are designed for use with PPP, this chapter will also discuss PPP. Point to Point Protocol (PPP) is used as the datalink layer protocol, mostly in dialup connections. It is a point to point protocol, which means that there are just two devices, called peers, talking to each other. Usually one peer is the User and the other is a Network Access Server (NAS), like a dial-in server. There is no shared medium, like in a wireless application, where many devices can talk to each other over a single wire.

5.1. Authentication Protocols

This chapter will give usage examples of PAP, CHAP and EAP-MD5 (an authentication mechanism for EAP). PAP, CHAP and EAP are all defined by the IETF.

However, these protocols are not the only protocols for authentication of a user to a Network Access Device. Examples of other protocols include Kerberos and TLS, which are both mentioned earlier in this thesis. Kerberos uses tickets, and TLS is a variant of SSL, which is used in web browsers. So there are many authentication protocols, each with its own message format. As we will see later, EAP was designed to structure the large number of different mechanisms by creating an Extensible Authentication Protocol (hence the name EAP).

Since the protocols discussed in this chapter (PAP, CHAP and EAP) are originally all designed to be used over PPP, this chapter will start by introducing PPP and its security concept.

5.2. PPP

The development of PPP is typical for the internet protocols. In the eighties, the first need to transport IP packets over a phone line emerged, which resulted in a protocol called SLIP (Serial Line IP) [RFC 1055]. It is a very simple protocol, and has a few flaws: there is no error detection, only IP packets could be sent over the wire (no IPX, Appletalk, Novell or Windows filesharing protocols), and lastly, SLIP has no authentication support. The authors seemed to be aware of most of these problems, and called their document “A nonstandard for transmission of IP datagrams over serial lines: SLIP”.

The IETF became aware of this problem and chartered a working group to create a new protocol to solve these problems. The protocol they came up with is called PPP (Point to Point Protocol) [RFC 1661]. PPP occupies the Link Control Protocol (LCP) sublayer of the datalink layer, while HDLC occupies the Medium Access Control (MAC) sublayer of the datalink layer. Clearly, the mapping of protocols designed with the Department of Defence (TCP/IP) reference model in mind onto the OSI model is not one-to-one here.

10 Actually, PPP and an underlying protocol, like HDLC, together occupy the datalink layer. PPP occupies the Link Control Protocol (LCP) sublayer of the datalink layer, while HDLC occupies the Medium Access Control (MAC) sublayer of the datalink layer.
with is **PPP** – Point to Point Protocol, and was first defined in RFC 1331 [RFC 1331]. It contains a field for the protocol, so more protocols could be used in the data field; it had a checksum, and a start pattern to allow modems to recognise the start of a packet.

### PPP redescribed

PPP was to a great extend based on a protocol called HDLC. Only the protocol and data field were PPP specific. Shortly after the publication, a new RFC, RFC 1661 [RFC 1661], was published, redescribing PPP. Only, this time, the HDLC and PPP fields were more clearly separated. This would allow PPP to be sent over other protocols. In fact, later documents were published to describe how PPP packets could be sent over an ethernet (PPPoE, [RFC 2516]) and over AAL5 [RFC 2364]. AAL5 is version 5 of a layer in ATM networks, similar to the transport layer in the OSI model, just a lot worse, according to critics.

Before the PPP client and server (the peers, as they are called in PPP terminology) start sending data, they first start negotiating options. For example, they negotiate the maximum packet length, whether or not they want to skip some unused data fields and the length of the checksum field\(^\text{11}\). So PPP is very flexible. The negotiation is done using a special protocol called **LCP** (Link Control Protocol), which is defined in the same document as where PPP is defined. LCP also has an option to negotiate the authentication protocol. If the NAS (the dial-in server) wants the user to identify himself, it will request an authentication protocol. Currently, about ten types of authentication protocols are defined [IANA PPP-numbers], where three of them are defined by the IETF: PAP, CHAP and EAP. An example with a NAS that prefers to use CHAP, but agrees to switch to PAP when the User doesn’t support CHAP is shown in figure I-4. Here, the NAS first asks the user if he can use CHAP. The user says he can’t. Then the NAS asks if PAP is OK then. This time the User complies. Though not shown here, the NAS can ask for other options at the same time. LCP is so efficient that no more than two or three round trips (request plus response) are needed to finish the negotiation.

Usually the message sequence, before a user can actually use the protocol goes in three steps:

- The LCP negotiation;
- The Authentication;
- The configuration of the desired protocol, using a configuration protocol, especially targeted on the desired protocol. These protocols are called NCP’s (Network Control Protocols).

\(^{11}\) Using a “Quality Protocol”, two peers can measure the quality of a connection. If there is a lot a data loss, it can be beneficial to increase the checksum to detect and correct transmission errors.
An example of a configuration of IP is shown in figure I-5. The Network Control Protocol (NCP) for IP traffic is called IPCP (IP Control Protocol) [RFC 1332] and [RFC 1877]]. Despite all the fancy names, all it does is just assign an IP address to the User12.

After this is finished, the transmission of data using the desired protocol can start, for example IP, IPX or Appletalk. Even multiple protocols at the same time may be used. In order to be sure that the NAS is still talking to the same user, the CHAP authentication protocol specifically says that the NAS may request re-authentication anytime it likes.

5.3. Security concept of PPP
PPP uses the philosophy that you first set up a connection, authenticate yourself, and then can start using the established connection for as long as you like. This is a common security design. Most application protocols like FTP, telnet, SSH, POP3 and IMAP (the last two are used for picking up e-mail) also use this approach.

This seems secure enough: PPP was designed with just serial lines in mind. The only way to break into such a line is by physically wiretapping it. That doesn’t seem very plausible. However, these days PPP is often layered on top of other protocols, usually for the benefit that you can identify whom is on the other end of the line. For example, PPP can be embedded on LAN’s using PPPoE (PPP over Ethernet), [RFC 2516] or over existing IP connections using PPTP (PPP over TCP/IP) [RFC 2637]. A typical usage of this layering is used to allow authentication of users on ADSL networks.

So clearly, the PPP protocol is incredibly popular, and it seems reasonably secure at first sight. However, it is possible to attack a PPP session. See the side note Hacking a session.

Example of an attack: Hacking a session
Despite that other users on the network can sniff packets, this “log-in once” authentication mechanism seems reasonably secure at first sight. If someone wants to ‘break in’ into your session, he will send it from a machine with another IP address13 than the IP address of your machine. The server will detect this and just discard the packet. However, it’s not so hard these days to forge (spoof) the originating IP address. So when an attacker sends a message with your IP address as the source address, the receiver will assume it’s a legitimate packet and process it. However, the reply will be sent to you; not to the attacker, so mostly it will be useless anyway. Unfortunately, the attacker may also be able to sniff the packets you send, and get the answer he wants. Despite newer

12 The current options include getting an IP address, getting the IP address of the primary and secondary name server, and optionally turning IP header compression on.
13 Or MAC address, depending on the type of protocol PPP is layered on. For PPTP and L2TP this will be an IP address, for PPPoE this will be a MAC address. I will use “IP-address” throughout this example for clarity.
techniques, like switched instead of shared networks, most attackers also invent newer techniques (and especially newer tools, which they spread over the internet in no-time), to hack into sessions.

Most security experts claim that now that sniffing and spoofing tools are so common, a session that is authenticated only once is not secure. For a secure connection, each and every packet should at least be encrypted (so that no one but the recipient can read the contents) and contain a message integrity checksum (for proof that it was really the sender who sent it\textsuperscript{14}). This technique should guarantee confidentiality and authenticity. However, it may still be prone to replay-attacks (an attacker sniffs a message, and resends the same –fully encrypted– message just an hour later). Also, if such a system uses a lot of cryptography for the transmission, it might be prone to a DoS attack: an attacker can send you just a couple of thousand messages at once, all of which you first need to cryptographically decode before you can see they are bogus. This may cost so much time, that your computer will have no time for anything else, and it may collapse due to the heavy load\textsuperscript{15}. In addition, cryptography requires that both parties share a common secret, the key. How to transmit such a key? Thankfully, this problem is solved with public key cryptography, where the decoding and encoding keys are different. However, now the problem is that if someone sends you a public key, how can you be sure that the person who sent you the key, really is who he says he is? And what about privacy: maybe you just don’t want to say, let alone prove, who you are if all you are going to do is visit a website.

Instead of inventing really complex protocols, most protocol designers stick to a well-known, reasonably secure protocol, like PPP with its authentication protocols. They say that if a more secure option is needed, well, it should be implemented at higher level protocols, like the network layer or the application layer. Some people in the IETF don’t like that they are still saying it, but the fact is: they still do.

5.4. PAP

The most simple authentication protocol is PAP (Password Authentication Protocol). All it does is send the username and the password. Both in the clear: there is no encryption whatsoever.

PAP is still sometimes used as the authentication protocol for dial-in using a modem.

An overview of PAP is shown in Figure I-6. On the left side is the user, in the middle is the NAS and on the right side is the authentication server. The vertical axis represents the time: message above others are sent earlier. At the top, the User connects to the NAS using the PPP protocol and they use LCP on top of that to negotiate the options. In this case they decide (see for example Figure I-4) to use PAP for authentication. Then the User uses PAP to sent the identity (the username) and the password in the clear to the NAS. The NAS may check this itself, but it may also forward it to the authentication server, for example using the RADIUS protocol (as shown in the Figure I-6). The authentication server will respond, and the NAS will configure itself to allow the user to use the service, at least in case of a positive

\textsuperscript{14} A message integrity code is a checksum which is calculated using both the content of a packet and a secret key only known by the sender and receiver. Because an attacker does not have to know this secret key, it can not create the integrity checksum, and can not alter a message or create a bogus message. Of course, provided that the checksum is large enough. Otherwise, the attack can send a message using all possible checksums, knowing that one of them will be the correct one.

\textsuperscript{15} Public key cryptography will cost a lot of computing power. For most desktop PC’s this is not a big issue, but for less powerful devices like handheld or mobile phones or for routers, this is a big problem. A common solution is to use public key cryptography to negotiate a shared secret, and that this secret is used as a temporary session key. This session key is then used to encrypt and decrypt messages during the rest of the session. Usually, this reduces computational overhead. An example authentication protocol which negotiates a session key, even though it uses simple passwords rather than public and private keys is SRP [RFC 2945].
response. Then, he will forward the accept- or reject decision to the User, and the User can start using the network (see also Figure I-5).

Figure I-6. Authentication using PAP.

5.5. CHAP

The most obvious disadvantage of PAP is that the password is sent in the clear. To improve for that, CHAP, Challenge Authentication Protocol, was invented to encrypt the password.

Figure I-7. Authentication using CHAP, Challenge Authentication Protocol.

The user and the NAS don’t have a shared secret, and the password is encrypted using a one-way encryption. That is an encryption mechanism, which can’t be reversed\(^\text{16}\). So the NAS can’t decrypt the password either, but that is no problem: it will just encrypt the password using the same method and compare the result to the information that the user sends. If the result is the same, the password encrypted by

---

\(^{16}\) A common example of a one-way encryption method is MD5 encryption, which is the default encryption mechanism for CHAP. Microsoft used a non-compatible, but still RFC-compliant variant of CHAP, which uses another type of encryption.
the user must have been correct. However, if only the password is encrypted, this
would still allow an intruder to steal access: he could sniff the encrypted password,
and resent that to the NAS\textsuperscript{17}. Instead of just encrypting the password, the NAS will
send a random string, the \textit{challenge}, to the user. The user will encrypt the string
made up of both the challenge and the password, and return this response. Since the
challenge will change with each request from the NAS, the response will change too,
while the NAS can still check the result. Figure I-7 shows this process.

It should be noted that CHAP is still not completely secure. It may still be possible
for an attacker to gain access to the network, by pretending to be someone else.
Sebastian Krahmer [Krahmer (2002)] has published an example of such an attack.
However, only sessions in progress can be hacked, and the attacker won’t be able to
find out the password itself.

The advantage of CHAP over PAP is that an attacker can not get the password itself
by sniffing the data packets over the network. A disadvantage of CHAP over PAP,
which has not been discussed, is that for CHAP, the password must be stored in the
clear on the server (either the NAS or the authentication server). With PAP, the
server can store the passwords using one-way encryption.

CHAP is currently the protocol most often used to authenticate dial-in (modem)
network access.

\subsection*{5.6. EAP}

One of the most versatile authentication protocols which can be used between the
user and the NAS, is \textbf{EAP}. EAP is short for \textit{Extensible Authentication Protocol} [RFC
2284].

A problem encountered by producers of Network Access Devices is that for each
authentication protocol, they need to add a specific module to support that
authentication protocol. However, all the NAS needs to do is pass messages from the
user along to the authentication server, and back, without need to analyse the
content itself. The only message the NAS must understand is the final result-
message: accept or reject.

The solution to this problem was to create an extensible authentication protocol,
which could not only support current authentication mechanisms, but also future
authentication mechanisms, without need to update the NAS.

All EAP did was create four common message types: request, response, accept and
reject. EAP is extensible (hence the name Extensible Authentication Protocol),
because the EAP request- and response messages contain a code, which usually
define the type of authentication used. For example, type 4 denotes “MD5
Challenge”, which is similar to CHAP-authentication. This mechanism, EAP-MD5, is
explained later in this section.

EAP was developed as an authentication protocol to be used over PPP; however,
currently it is mostly used for Wireless networks. In particular, Wireless networks
use a specification, called IEEE 802.1x [IEEE 802.1x] for Access control, and IEEE
802.1x uses EAP messages for authentication.

The big advantage of EAP is that all the NAS needs to know is the message type (it
should recognise accept and reject messages): if it supports EAP for authentication, it
can support any authentication EAP is capable of.

\textsuperscript{17} This is an example of a so-called replay-attack.
The disadvantage of EAP is that all existing authentication protocols (PAP, CHAP, Kerberos, TLS) will need to be rewritten or extended to be used over EAP. For example, a new document was published to explain how TLS [RFC 2246] should be used over EAP (PPP EAP TLS Authentication Protocol [RFC 2716]).

**EAP-MD5**

Unlike PAP or CHAP, EAP is capable of supporting multiple authentication mechanisms. One of these mechanisms is EAP-MD5, which is very similar to CHAP. The EAP specification states that every EAP implementation needs at least to support EAP-MD5. The message sequence of EAP-MD5 is explained in Figure I-8, which shows how EAP-MD5 is used over PPP.

![Figure I-8. Sequence of messages for the EAP-MD5 authentication protocol.](image)

Though CHAP and EAP-MD5 are similar in design, EAP-MD5 has more overhead, which is a trade-off for the increased flexibility of EAP and security of EAP-MD5. As with CHAP, the PPP conversation between the User and the NAS starts with the option negotiations using LCP. Then the NAS can let the authentication server know that a new EAP session is starting, and the authentication server will first ask for the identity of the User. Since every EAP implementation will first ask for the identity, the NAS may skip this connection to the authentication server and ask the user for its identity immediately. The user responds and the NAS forwards this response to the authentication server. Next, the authentication server will ask for the password by sending a challenge. Again, NAS forwards, the User responds, and the NAS will forward the response to the authentication server. Now, the authentication server can make the decision to either grant or deny access to the user. In case of granted access, the authentication server can send an EAP-Success message, embedded in a Radius Access-Accept message. If access is rejected, the authentication server will send an EAP-Failure message, embedded in a Radius Access-Reject message.
Background: EAP extensions

EAP-MD5 is just one of the authentication mechanisms supported by EAP. The original EAP specification defined six request/response types, as shown in Figure I-9.

1. Identity
2. Notification
3. Nak (Response only)
4. MD5-Challenge
5. One-Time Password (OTP) (RFC 1938)
6. Generic Token Card

**Figure I-9. EAP Message types defined in RFC 2284.**

Currently, IANA assigned another 26 message types [IANA PPP-numbers], including the afore mentioned EAP-TLS. Most of these message types, however, denote a non-documented vendor-specific protocol. OTP and Generic TokenCard are hardly used [EAP-BOF].

If a User encounters a request for an unknown type, it will return a Nak (Not Acknowledged) message. By sequentially requesting different information, the authentication server and User negotiate the authentication mechanism. To ensure that each EAP has at least one mechanism in common with every other implementation, EAP requires that each implementation is able to use EAP-MD5.

Future of EAP

Because EAP is being deployed increasingly, some minor issues have arisen. For example, the interaction with other layers (layering EAP on top of other protocols) is not documented, the state machine and the security requirements are not always clearly documented. At the time of writing, the IETF has just set up a new working group to discuss and resolve these issues [EAP-charter]. Some of these issues are explained in more detail in the appendix “Background and details”.

5.7. EAP and the Access Bind PIB

In an ideal situation, the PEP does not need to know anything about the authentication protocol used by the client, regardless PAP, CHAP, HTTP authentication, EAP, SRP or any other authentication protocol. The PEP should just forward the messages from the user on to the PDP, and back, if necessary. All the PEP needs to know is the final decision: if the User is authorised or not. This is also true in the situation of the Access Bind PIB.

Regrettably, each authentication protocol has a different message and result format, so in practice, the PEP needs to know about the details of each authentication protocol it implements. EAP is just one of these protocols, but it is capable of supporting most authentication mechanisms already designed by the other authentication protocols. What’s more important, EAP is also capable of supporting future authentication protocols, without need to change or modify the way the PEP treats the EAP messages. (Of course, the PDP and the User must support the new EAP authentication mechanism, only the PEP can keep functioning the same way).

This, along with the widely used deployment of EAP, in particular in wireless networks, has made the Access Bind PIB authors decide to primarily focus on the EAP message format for authentication messages. Having said that, in order to facility networks which does not yet use EAP, the data structure can be extended to supporting other authentication protocols. In particular, it has been shown that the Access Bind PIB can work with PAP and CHAP too. A data structure to show it can
work with HTTP authentication is in the pipeline, but was not finished at the moment of this writing.
6. Message Sequence

Data transmission between a PEP and a PDP occurs in a certain sequence. The first transmission is at boot time of the PEP, which causes the PDP to configure the PEP. There can be communication between a User and the PEP and between the PEP and the PDP. The PEP sends an access request (event) tot the PDP when an unknown user sends data to the PEP. The transmission of the event may be followed by an optional authentication sequence, and an optional per-user provisioning. Every access request is concluded by an authentication decision that is sent from the PDP to the PEP. If the decision was positive, the user can use the PEP to connect to a network.

The message sequence between the PEP and the PDP in the current version of the Access Bind PIB is optimised for speed, by reducing the number of messages that are sent back and forth.

This chapter will give an overview of the messages that are sent between the PDP to the PEP and between the PEP and the User.

### Background: Round trips

For most network applications, the speed limit is not the bandwidth, but the round trip time. The bandwidth is a measure for how much data can be sent per second over a certain wire, and the round trip time is a measure for the time that a single packets takes to travel from one host to another host and back (one leg).

Sometimes, the time a single tip takes (from one host to another) is called the latency. This latency is caused by the speeds that signals travel along the wire (about two third of the speed of light), and by the time it takes devices along the way, such as routers, to examine the data packet.

A typical round trip time ranges from 0.5 milliseconds for computers in the same building to 50 milliseconds for a packet to travel from Europe to the United States and back. Compare this with the 8 ms time a transmission of a typical internet package of 1500 bytes takes on a T1 connection. In addition, the time it takes a computer to calculate a response to your messages may roughly take a few milliseconds (that’s a few millions of processor cycles these days). You can improve the transmission time by adding more bandwidth, and you can improve the processing speed by adding CPU’s or replacing them with faster ones.

However, there is no way you can change the latency. You can not change the speed of light.

So a significant speed-up can be achieved by reducing the number of round trips (legs) that are needed in the communication between two devices, ranging from 0.5 milliseconds for an ISP with the PEP and the PDP in the same building to as much as 50 milliseconds for a roaming situation, where the two devices may be on different continents.

6.1. Message sequence overview

The PDP and the PEP exchange messages. A stepwise overview of actions and events is shown in Figure I-10.

The list can be divided into two parts: 1 and 2 happen at boot time of the PEP, and are meant to configure the PEP. The other events and actions occur during normal operation of the PEP, and mainly deal with the way a user connects to the PEP, is authorised and uses the service the PEP offers.
1. PEP sends configuration request
2. PDP configures Event Handlers on PEP
3. PEP notices ‘new’ traffic that triggers an event (lookup table)
4. PEP sends info to PDP
5. PDP talks EAP via PEP to client
6. PDP sends decision to PEP (optionally with new DiffServ elements)
7. PEP adjusts lookup table (only for positive decision)
8. PEP notifies user of result
9. PEP notices ‘known’ traffic, and passes it to next DiffServ element

Figure I-10. Sequence of actions and events that may occur in this order at a PEP.

6.2. Configuration of the PEP

The COPS protocol was designed that the PDP can automatically configure the PEP, with as little user interaction as possible. The PDP is the central device that not only handles authentications, but also is responsible for configuring the PEP with Policies. A typical policy is rule-based, and may specify rules like ‘Users dialling in, must always be authenticated. Users from a known IP range must authenticate as well, and some of these users should always get a premium connection. Other users should get premium connections in weekends and a best effort connection on weekdays. Users connecting from other IP address are always denied access.’ The function of a PDP (the Policy Decision Point) with COPS-PR support is to translate these rules into data structures, like the ones defined in the Access Bind PIB. These data structures are then configured (installed) in the PEP, where they are enforced. Hence the name Policy Enforcement Point (or PEP).

In this philosophy, system administrators do not need to configure each PEP manually, but rather design one organisation-wide policy, implement this in a central PDP server, and let this PDP server provision each PEP. All that needs to be manually configured in the PEP is the hostname or IP address of the PDP server.

![Diagram showing the sequence of actions and events at boot time of the PEP.](image)

**Figure I-11 Message sequence at boot time of the PEP. The PDP configures the PEP by installing policies.**

When the PEP boots up, the messages sequence shown in Figure I-11 occurs. This figure (which shows the messages related to the first two actions or events in Figure I-10), shows three types of message. First of all, the PEP will connect to the PDP with a request “Please, configure me”. However, because the PDP at that point does not know what type of device the PEP is and what its capabilities are, the PEP will request a list of capabilities of the PEP. For example, a router that does not work with dial-in connects shouldn’t have support for the ctxtDialupInterface class. The
capability exchange is done using by sending instances of a special table, called frwkCapabilitySetTable. This class and its behaviour is not Access Bind PIB specific, so it is out-of scope to describe this process in detail. Currently, it is described in the Framework Policy Information Base [draft-ietf-rap-frameworkpib].

After the capability exchange is finished, the PDP will configure the behaviour of the PEP by installing instance of classes onto the PEP. These classes contain generic Roles, as described in [draft-ietf-rap-frameworkpib] and DiffServ elements. If the PEP wants to support access control based provisioning (i.e. support different kind of QoS or security, based on the person who has just logged in, rather than based on static IP address), it must included instance as described in the Access Bind PIB.

In particular, the PDP installs the following classes:
- Event Handlers and event Handler Elements.
- Context Data “pointers”
- Scopes, defining filters and lookup
- Other DiffServ elements

In any case, the PEP must notify the PDP of a successful or unsuccessful configuration by sending the appropriate COPS response message. If the configuration failed for some reason, the PEP can sent an error message along with it, and the PDP may decide to retry the configuration based on the type or error it received. Note that the COPS protocol explicitly specifies that if just one data structure in a particular COPS message failed to get installed, the installation of all other data structures in the message should be cancelled. In database terminology, the PEP should do a full roll-back if it was not possible to commit all data structures.

6.3. Triggering of an event

When a user want to use the services the PEP offers (like network access or access to a fileserver behind the PEP), it will sent a message to the PEP. This first message triggers a whole set of messages that are sent between the PEP, PDP and User. Figure I-12 shows all these messages.

---

**Figure I-12.** The messages that may be sent between a PEP, PDP and User when an unauthenticated user connects to a PEP, and EAP is used for authentication. For PAP and CHAP, the scenario is slightly different.
The message that is sent by the User to the PEP can be anything. The Access Bind PIB is designed that it can work with data traffic on different layers. The PEP can be configured that only specific RSVP (Reservation Protocol) messages trigger an event (The Access Bind PIB has few classes for RSVP support), but the PEP can also be configured that any traffic coming on an unknown triggers an event.

Upon receipt, the PEP checks which eventHandlerElement was responsible for the triggering of the event\(^1\), and will gather the User information, as described by the context data pointers. The PEP will then create a COPS request message, and includes the event, and instances of all specified contextData classes. Implementations will send both data structures in the same COPS message, so the yellow and green arrows in Figure I-12 are combined.

### 6.4. Authentication of a User

For EAP, the authentication is communication between the PDP and User as the end-points. For PAP and CHAP, the message exchange needed for the authentication takes place between the User and the PEP, and the PEP forwards this than to the PDP.

Figure I-12 shows the sequence for EAP. Here, the PDP decides that the user must be authenticated, and sent an EAP Identity request to the PEP. The PEP checks the COPS client handle, and based on that, forwards the EAP request to the correct user. Then, the user replies, and the PDP forward the EAP packet to the PDP. This request and reply of various credentials can be repeated as many times the PDP like.

For PAP and CHAP, the PEP initiates the message sequence to the User, and contrary to the EAP situation, does this before the event is sent to the PDP. The PEP will gather all the credentials needed and fill either an authPapExt or authChapExt instance for PAP respectively CHAP. This instances is then sent along with the event and the contextData instances to the PDP.

### 6.5. Per-User provisioning of the PEP

When the PDP has authenticated the user, it either authorises or denies him or her access to the service\(^2\). Before this decision is sent back to the PEP, the PDP has the option of installing additional provisioning rules in the PEP.

For example, if the message sent by the User is a special message that contains a request to use a certain Quality of Service (QoS), and this message type is recognised by the PEP (and forwarded to the PDP, using a contextData instance), the PDP may decide to grant this request and provision the PEP for the required QoS. This would eliminate the need to install very complex filters in the PEP to recognise every request.

However, for normal operation, this provisioning is not necessary, and the PDP can just send the decision is back to the PEP. If the decision was positive, the PEP will add the (virtual or physical) interface where the User connected on to the lookup table. In any case, the PEP will let the user know of the result.

#### Usage of service

As soon as the User receives the acknowledgement that he or she may use the desired service, it can connect to the PEP. The PEP now recognises the user data

---

\(^1\) The Access Bind PIB does currently not specify which eventHandler should be used if two or more eventHandlerElements matches. Since it seems unlikely that the PEP should sent two events, a new version of the draft will probably include a priority attribute in the eventHandlerElement class.

\(^2\) Note that the PDP only sends one result: the PEP has no means of knowing whether an access request failed because the authentication failed or the authorisation failed.
traffic, because the User interface was added to the lookup table, and will forward
the Users messages along the desired datapath to the next DiffServ element.

If the User disconnects, or the PEP recognise in some other way that the interface on
which the user has connected is terminating the connection, it will remove the User’s
interface from the lookup table, and notify the PDP of this event by sending a COPS
Delete message with the appropriate COPS Client Handle.

6.6. Combining messages
In order to decrease the number of round trips, the Access Bind PIB specifies that a
few data structures can be combined in the same COPS message. Particularly for
usage with mobile telephony, the speed up by removing one or two round trips is
significant. In mobile telephony, round trips can become very long due to roaming
agreements (the PDP may be on another continent), and because a delay of a fraction
of a second can become noticeable for speech.

This applies to three cases. First, when the PDP configures the PEP, the configuration
request message and the capability exchange may be embedded in the same COPS
message.

The other situations are shown in Figure I-13. Both the Access request from the PEP
to the PDP, the context Data instance, which describe the User and interface
information, and the PAP or CHAP authentication data (if applicable) are sent in the
same COPS message.

Figure I-13. Combining messages.

In addition, the authorisation decision may be combined with the per-user
provisioning. If should be noted that if this provisioning fails, however, that the PEP
must not forward a success message to the user, but just sent an error to the PDP and
act like the message was never sent. Now, the PDP can try again, for example using
another type of provisioning or an access denied accompanied with an error
messages to the User telling it was unable to fulfil the User’s access request.
7. Data structure

The Access Bind PIB defines a data structure. The basic class is an event handler which acts like a Classifier (a DiffServ element). The data is event-based: the PDP can install event triggers onto a PEP, using scopes (filters), and the PEP sends an event to the PDP each time the event handler is triggered. An event can be accompanied with data describing the context of an event.

The PDP authenticates a session, which is defined by a COPS Client Handle, not directly a user account (a username may be sent, but is not stored by the PEP; the PEP does keep track of the client handle).

The Access Bind PIB defines a data structure. This structure will be explained in this chapter. Alternatively, you can read the Access Bind PIB itself.

Section 4.2 explained that the Access Bind PIB data structure is conceptually a complex classifier, which we will call eventHandler21, which is a DiffServ element inside a PEP used to classify data by filtering certain traffic from other traffic. The complexity comes from the fact that this element is not just a passive filter, but is capable of communicating with the User and the PDP.

This chapter will start with this concept of the eventHandler, shows the data structure immediately associated with it, and link other data structures, like filters, events, context -, and authentication data structures to this central concept.

Each section is accompanied with one or more diagrams of data structures. The legend used to correlate data structures to each others is shown in Figure I-14.

Legend

- Inherentence, the referring class is an extension of the referred class.
- Reference by Instanceld to another instance of specified type (a many to one relation)
- Reference by a tag to a group of instances of specified type (a many to many relation)
- Reference by Prid to another, unspecified type of instance (a many to one relation)
- Reference by Prc to a classname

Figure I-14. Legend for the data structure figures.

7.1. The eventHandler as a classifying DiffServ element

The main concept of the Access Bind is an eventHandler, a complex DiffServ Classifier element. A classifier filters or ‘classifies’ traffic and, based on certain criteria, forwards this data traffic to other DiffServ elements. In the data structure, every eventHandler (classifier) is represented by exactly one eventHandler record, and zero22 or more eventHdlrElement (event handler element) records. Figure I-15 shows an overview of some of the data structures, including the eventHandler and eventHdlrElement. As you can see, the eventHandler contains an attribute called eventHandlerElements. This is a so-called TagReferenceId, which is an arbitrary number. Each eventHdlrElement record associated with this eventHandler contains the same number (it’s in an attribute of type TagId called eventHdlrElementGrpId).

---

21 The DiffServ PIB [draft-ietf-diffserv-pib] already defines a classifier. To distinguish that classifier from the one defined in the Access Bind PIB, this thesis will call the classifier in the Access Bind PIB an eventHandler.

22 An eventHandler with zero eventHandlerElements has no filtering capability, and is thus a trivial and rather useless classifier. So in practice, this will be one or more.
This way, the eventHandler has zero or more are associated eventHdlrElements with it.

**TagIds and TagReferenceIds**

We will see more uses of this concept of TagIds and TagReferenceIds. People who are familiar with database design will recognise this as a many-to-many relation. Another approach would have been to put a common ReferenceId in the eventHdlrElement data structure (note that the reference and the id are in the opposite classes here). That way, the eventHandler could have recognised the associated eventHdlrElement records as well. That would have been a many-to-one association from the eventHdlrElement to the eventHandler. The advantage of using a Tag and TagReference is that you can reuse the sets of eventHdlrElements. For example, there may be two eventHandler records with the same TagReferenceId. See RFC 3159 [RFC 3159] for the definition of TagIds and TagReferenceIds.

---

**Figure I-15. Event classes and eventHandler classes.**

Each eventHdlrElement is a filter. If data traffic matches a filter, the data will be forwarded to the next DiffServ element associated with this particular filter. The eventHdlrElementMatchNext attribute contains a pointer to this next DiffServ element. If the incoming traffic does not match any filter, it will be forwarded to the DiffServ element specified by the eventHandlerNonMatchNext attribute in the eventHandler class.

**Filters and Scopes**

Each eventHdlrElement is a filter, or actually, a composite filter made from multiple filter objects. For example, you may have two basic filters, one filtering source IP addresses in the range 10.0.0.0 to 10.0.255.255, and another filter, filtering on ethernet (MAC) addresses. Each basic filter is called scope, so each eventHdlrElement contains a set of scopes. In fact, each eventHdlrElement contains two very similar sets of scopes, but for now we will only discuss one type of them, the eventHdlrEventScopes. A set is defined by all eventHdlrEventScopes with the same
eventHdlrEventScopeGroup attribute (a TagId, which is referred to by the eventHdlrElement using a TagReferenceId attribute called eventHdlrEventScope). In the example, the set of scopes associated with our eventHdlrElement contains two eventHdlrEventScopes: one to filter the range of IP addresses and another to filter MAC addresses. As you can see in Figure I-15, each Event Scope records contains a few attributes: An InstanceId (a unique number, which is not used in our data structure, but which is useful to database designers), the group TagId (identifying the set of scopes), a PRID to Filter, A Precedence field and finally, a Boolean called ChangeFlag.

Figure I-16. A few Filter classes. The rsvpFilter is defined in the Access Bind PIB [draft-ietf-rap-access-bind], while the frwkBaseFilter, and the two extended classes are defined in the Framework PIB [draft-ietf-rap-frameworkpib]. The frwkIpFilter can be used to filter data traffic based on source or destination IP addresses, while the frwk802Filter can be used to filter data traffic based on hardware addresses, in particular MAC addresses defined in IEEE 802 working groups.

A PRID is the COPS equivalent of a ReferenceId, and contains a pointer to another class instance. Unlike a ReferenceId, a PRID can point to an arbitrary instance type. So any type of filter can be used, and in fact, this allows for extensibility: in the future, the PRID in an eventHdlrEventScopeFilter can point to a filter data structures.

---

23 An extended class does not only contain the attributes listed in their definition, but also inherit the attributes of the extending class. In this case, each frwkIpFilter instance also contains the attributes frwkBaseFilterPrid and frwkBaseFilterNegation.

24 Most class instance have two identifiers: an InstanceId which is mentioned in our data structures, and a PRID an identifier which is automatically created by the COPS service layer. The PRID is and transmitted using the COPS protocol just before the actual Instance itself, the PRI is transmitted. A PRID not only uniquely defines a PRI, it also contains information about the type of Instance that is sent (it contains the class name), so that the other devices knows what type of data will follow. It is very confusing, and in my opinion incorrect that the Framework PIB named an attribute frwkBaseFilterPrid, while it is an InstanceId, not a PRID. The correct name should have been frwkBaseFilterId. Hopefully, this will change in a new draft.
that are not yet defined. Figure I-16 shows a few example filter classes that can be referred to.

We haven’t discussed how two filters are combined: using a logical AND or a logical OR. In our example, will the eventHdrElement match data traffic that matches both the IP address and the MAC address, or will it match data traffic that either matches the IP address or the MAC address? This is what the Precedence attribute in the eventHdrEventScope decides: if two scopes have the same precedence value, the match is an AND, if the two scopes have different precedence value, the match is an OR.

Finally, the ChangeFlag boolean is discussed in the next section, where events and event triggering mechanisms are discussed.

7.2. Events and event criteria

So far, we only talked about static filters, without interaction with the PDP. All data structures mentioned so far, eventHandler, eventHdrElement, eventHdrEventScope and filters, can be installed at boot time by the PDP into the PEP. However, the goal of the Access Bind PIB is to allow the PEP to differentiate the offered services, based on the authorisation of users, not based on predefined filters, like IP address or MAC addresses.

Two drafts

At the moment of writing of this thesis, there have been two revisions of the Access Bind PIB, draft 00 and draft 01. The 00 draft dealt with the concept of a user who is granted or denied a certain service level by introducing the concept of a session. The session was a central object which pointed to an accessor, an object slightly similar to the eventHandler in the 01 draft, which among other attributes, contained the next DiffServ element.

The 01 revision moved to another concept, that of events. In this draft, the PEP has no knowledge of individual users, but is just told by the PDP when to create a new event and to signal the PDP about this event. Usually, an event is generated at the moment a new user connects. In this sense, the creation of a new session and the generation of a new event are similar.

The data structure presented in this thesis is the data structure used in the 01 draft, which uses events, because that is the latest revision of the Access Bind PIB at the moment of writing. However, a few times this thesis will discuss the implications of this choice. Readers should bear in mind that the current data structure is still a data structure in a draft, which is a work in progress. It is very likely that in a future draft, the data structure will again change. This thesis can (and will) only speculate on future changes.

Event concept

An event is triggered by the PEP as soon as a new user wants to gain access to the service provided by the PEP, like access to a network. The PDP configures the PEP by telling when a trigger should occur, and what information the PEP should sent to the PDP. The PDP will examine this information, and based on that, either immediately grant or deny access to a new user, or first start an authentication negotiation with the User, possibly by exchanging EAP messages. This is the basic concept; what follows is a more detailed description of this process.

Event Triggering

Filters play an important role in the generation of events. The same filters that are used to filter data traffic to decide what the next DiffServ element is, are used to filter the traffic packet that will trigger an event.
However, in most cases, we don’t want that every data packet coming from a certain client triggers an event. Usually, we only want that unauthenticated data traffic coming from a certain client triggers an event. Therefore, each eventHdlrElement contains an attribute called eventHdlrElementEventCriteria. Currently three values are defined: one_time, every_time and on_change. If this attribute is set to every_time, every time a packet that matches the event scope, a new event will be generated by the PEP. If this attribute is set to one_time, only the very first data packet matching the filter will generate an event. This is also probably not what you want: if the filter filters traffic coming from source IP address in the range from 10.0.0.0 to 10.0.255.255, and traffic comes in from 10.0.18.231, then an event is sent. However, if later traffic arrives from 10.0.73.108, you probably want to generate a new event. With the Event Criteria set to one_time, this won’t happen. You probably want to set Event Criteria to on_change, which means that an even will be generated for each unique set of filter values. For example, if the PEP first get traffic from 10.0.18.231, then from 10.0.73.108, and then again from 10.0.18.231, the first two packets will generate an event, but the third won’t, since that address is already known. This means that the PEP must keep a list of values that already known. This list is not defined in the PIB data structure, since it is never transferred between the PEP and the PDP.

This approach using on_change brings a problem: what for example, if I have two filters: one filter filtering traffic on source IP address in the range from 10.0.0.0 to 10.0.255.255, and another filter matching traffic going to destination port numbers ranging from 21 to 80. I want to generate a new event for each new source IP address, but not for every new destination port number. The solution to this problem is to introduce a boolean attribute in the eventHdlrEventScope table called eventHdlrEventScopeChangeFlag. This Change Flag must be set to true for filters which should generate a new event for each unique value, and it must be set to false for filters which should not generate a new event. In this example, the event scope instance for the IP address filter will have this attribute set to true, while the event scope instance for the destination port number will have this attribute set to false.

As soon as the PEP detects that a new event should be generated, it creates a new instance of the event class, and sends this instance to the PDP. Note that events are created by the PEP, while other instances where created by the PDP.

A problem with lookup tables

Technically, by setting the eventHdlrElementEventCriteria attribute to on_change, the PEP will need to create a lookup table, and it will have to include the parameters of all filters that have the eventHdlrEventScopeChangeFlag attribute set to true.

Unfortunately, the 01 draft is not clear to specify exactly which fields of the filter should be included. For example, if you have a filter that specifies to filter on source IP addresses, you will probably use the class frwkIpFilter, as defined by the Framework PIB [draft-ietf-rap-frameworkpib], shown in Figure I-16. However, this class also defines destination IP addresses, and source- and destination ports. It is not obvious which parameters the PEP should include in the lookup table.

I expect that a fourth value will be defined in a future version: never.

Therefore, on_change is not a good name. On_unique would probably be more correct.

Actually, the PEP needs to create a lookup table for each precedence value. Section 4.6 of the 01 draft gives a good overview of this, although there is an omission: the * next to the instance number means that the eventHdlrEventScopeChangeFlag is set to true, and there are a few minor errors the boolean expression used: 1 should be "((A=5 or A=6) and (B=15) and (C=10 or C=11))"; 2 should be "((W=20) and (X=2 or X=3 or X=4)) or ((W=14) and (Y=400-500)) or ((Q=4-9) and (R=92))". 
It is tempting to specify that the PEP should ignore all parameters that are wildcarded (The Framework PIB says that “Wildcards may be specified for those fields that are not relevant,” although it is not obvious how this should be done). However, this is not entirely correct. For example, we may want to say allow any source IP address (don’t filter on that, but do trigger on it), and allow any destination port (don’t filter on that, and neither trigger on it).

So we want to make a difference between:
- Filter on it, and trigger on it
- Don’t filter on it, and trigger on it
- Don’t filter on it, and don’t trigger on it

The fourth option, filter on it, but don’t trigger on it is already dealt with, using an attribute called eventHdlrEventScopeChangeFlag.

The last two options can be rephrased into the main question:
Given a filter, can I distinguish between “ignore a field” and “don’t ignore a field, but match every possible value”?

This is not obvious. I belief that we cannot rely on the fact that every filter is able to differentiate between these two options. So, a more workable solution might be to use another mechanism to specify on which header fields the scope should trigger if they change. A more radical approach would be to drop support for the on_change options altogether. In that case, the PDP needs to install a separate filter for each known combination of header fields. This is essentially the same as leaving the event model in favour of the session model.

Unrelated to the above problem, there is a difficulty that there is not a one-on-one relation between the attributes that a filter class has and the header fields in a data packet. For example, to filter on destination port, the frwkIpFilter defines two attributes: DstL4PortMin and DstL4PortMax, and to filter on source IP address it defines three attributes: AddrType, srcAddr and srcPrefixLength. This is not a real problem, since the PEP should have knowledge about a specific filter to use it anyway, so a PEP already needs to have a list to which data packet header fields the filter applies to.

When, what and how
We have seen that eventHandler and the eventHandlerElements describe when a new event should occur. They do that by using scopes and filter objects. In subsequent sections, can also describe what should be sent to the PDP and how this information should be retrieved. Section 7.3 shows that ContextData tables are used to describe what is sent, and section 7.4 explains how it is retrieved can optionally be described using the eventHandlerProtocol.

COPS Client Handle: a Session concept
The protocol that is used to transport all data structures between the PEP and the PDP is COPS. COPS does support other data structures then those specified in a PIB. For example, just before an instance is sent, a PRID is sent. This PRID is a unique identifier for this particular instance. It helps the receiver to know what data is sent, because the PRID contains the name²⁸ of the class.

But all these transmissions of PRI (Instances) are defined in an extension of COPS, COPS-PR. However, COPS itself does also define message types. One of these message types is the COPS client handle (See sections 2.2.1 and 2.4 of [RFC 2748]). The client handle contains a value, chosen by the PEP, and refers to a single state. For

²⁸ Actually, it contains the OID, the Object Identification Number.
example, if the PEP makes two requests to the PDP, and both refer to a single request made by a user, both requests to the PDP will contain the same Client Handle. If this state is terminated, for example because the User disconnects, the PEP can just issue a delete message for the corresponding Client Handle. Then all corresponding states (and data structures) related to this particular Client Handle will be deleted by the PDP, and the PDP may even remove or alter some other data structures as well.

Effectively, the Client Handle acts as a session identifier. All requests from the PEP corresponding to a certain user connecting have the same Client Handle. As soon as a Client Handle is deleted, the session is effectively deleted. It acts as a message to delete all states and data structures corresponding to this session.

It is even possible to have two events correspond to the same Client Handle. To facilitate for this, the eventHdlrElement refers to two sets of Scopes: one TagReferenceId (eventHdlrElementEventScope) to define when a new event must be sent to the PDP, and another TagReferenceID (eventHdlrElementHandleScope) to define when a new Client Handle should be defined. By default, these two scopes are the same, and every time a new event is sent to the PDP, the message contains a new Client Handle.

### Start of a session

It is interesting to notice how the term session is used here, and in the draft 00 version. Within the IETF, the term session has two, slightly different meanings: Either a session includes all communications from one device to another device, or in the other meaning, the devices first start with some initial communications (like authentication messages) that are used to set up a session. In this latter case, the word session only signifies all communication that is related to the actual usage of a service, but excludes all ‘overhead’ messages.

The difference becomes clear when you consider a user connecting to a service, but access to this user is denied for some reason. In the first meaning, the user did create a session, even though the session was never authorised to use the service. In the latter meaning, there was just no session: all that happened was that a user unsuccessfully tried to create a session.

The Access Bind PIB, both the 00 draft, which has a actual data structure called a session, and the 01 draft, which uses COPS Client Handles for the session concept, start a session at the time a user connects. So they use the first meaning of the word session: A session starts as soon as the user connects, but not all session are always authenticated.

### 7.3. ContextData pointer and ContextData structures.

In the previous section, I showed in detail how the data structure defines when an event should be sent from the PEP to the PDP. However, it is often very useful if the PEP does not only sent an event to the PDP (which includes a pointer to the eventHdlrElement that caused the trigger), but also includes some information about the packet itself that caused the event.

This is possible. The Access Bind PIB defines a series of classes, just for this purpose. These classes are called ContextData classes, since they provide the context why an event was triggered. All ContextData classes that are defined by the Access Bind PIB are shown in Figure I-17. You may wonder why there is no pointer to the InstanceId of the event, in order to associate a contextData instance with a particular event, but this association is already made using the COPS Client Handle: The PEP sends the event and the contextData instance in the same message, or in multiple messages with the same COPS Client Handle (as described in the previous section).
Optimisation

Figure I-17 shows a lot of contextData that can be sent. Sending it all would be inefficient, so there is a mechanism for the PDP to tell the PEP which contextData needs to be sent along with each event.

This can be specified by instances of the contextData class. Despite the same name, the contextData class, which is shown in Figure I-15, itself is not one of the contextData classes as shown in Figure I-17. An instance of the contextData class itself is a pointer to the name of one of the contextData classes. The instances contextData classes contain the actual context data. To avoid further confusion, I will refer to the contextData class from now on as the contextDataPointer class.

![Figure I-17. contextData classes](image)

Figure I-17. contextData classes

Each eventHdrElement contains a pointer to a set of contextDataPointers, using the eventHdrElement attribute (a TagReferenceId). When the PEP sends an event, it will examine the set of contextDataPointers associated with the eventHdrElement that triggered the event, and create the appropriate contextData classes (as mentioned in the contextDataIfElement) for each contextDataPointer. The name IfElement comes from the word Interface; each contextData class describes one element of the Interface (either a physical or virtual port in the PEP device). An element (described by a single contextData class) might for example be a layer in the internet model. Examples of such element are the network layer (layer 3 in the OSI model, thus described by the name ctxtL3Hdr), or information about the phone connection for dial-in (described by ctxtDialupInterface).

Encapsulation

A problem with layers is that they can be stacked on top of each other, like PPP over ethernet (PPPoE) does. In fact, it is even possible that one interface is layered on top of another. I believe this is very confusing. A better name for the contextData class would have been contextDataPointer class.
of the same interface (IP over IP). In this case, if the PDP asks the PEP to sent information about the IP layer (ctxtL3Hdr), it is not clear which of the two interfaces it wants to have. Therefore, the contextDataPointer contains an attribute called contextDataEncapsulation, that describes which of the interface should be taken in case of encapsulated layers.

### 7.4. Authentication Data structure

Last, the Access Bind PIB has an extensive data structure to support authentication structures. Just like contextData classes, these classes are associated with a certain event by the COPS client handle. Figure I-18 shows the authentication classes.

![Figure I-18. Authentication classes](image)

Each and every class is an extended class of the base class, the authExt (Extensible Authentication) class.

For an introduction to PAP, CHAP and EAP, see chapter 5.

### Support for EAP

Two simple classes are used to support EAP. EAP is by design, a protocol where the authentication is done between the end-point, that is, the User and the PDP. So the PEP is not involved in the content of the messages, with only one exception: the PEP should know the final authentication result: allow or deny.

Because the PEP does hardly have to parse the EAP messages (it should only check the messages type, which signifies whether it is a request, a response, failure or success), the EAP just forwards the whole EAP messages from the User to the PDP, or the other way round (from PDP forwarding to the User).

Despite that the EAP messages are identical, it was decided to use two different classes to embed the EAP messages: authEapReqExt to embed EAP messages that are forwarded from the PEP tot the PDP, and authEapRespExt to embed EAP messages that are sent from the PDP to the PEP.
Authentication versus authorisation result
For EAP (and also PAP and CHAP), the authentication result (EAP message type 3 – and type 4 – ) is also used for the authorisation result. It is not possible that the PDP can signify that the authentication succeeded, but the authorisation failed. This is the case if someone want access, and provided the correct credentials, but he or she is not allowed access, for example because access is restricted to working hours, or because the required services is temporarily unavailable. I believe this is an omission in the data structure.

Support for PAP and CHAP
PAP and CHAP are two PPP authentication protocols between a User and a Network Access Server (NAS), which is in this case the same device as the PEP.

PAP is a simple protocol where the username and password are sent in the clear. CHAP is slightly more complex: the username is sent in the clear, but the password is encrypted using a NAS-generated challenge code.

Unlike EAP, the design of the PAP and CHAP protocol requires that the PEP has knowledge about the contents of PAP and CHAP messages. Therefore the Access Bind PIB authors decided not to embed the full PAP or CHAP message, but create classes for each PAP or EAP message that is forwarded to the PDP, with each part of the message in a separate attribute in the class.

Since the response for both PAP and CHAP is the same (success or failure), they share the same class: authExtResult, with just a boolean indicating a successful authentication or a failure. Currently there is no support to discriminate between a successful authentication, but a denial to use the services. For example, because someone only has access rights from 9:00 to 17:00 on weekdays, but tried to use the service in a weekend (an authorisation failure, as opposed to an authentication failure).

Both PAP and CHAP sent the username, and optionally realm, like the DNS domain, to the PDP. So both the PAP and the CHAP request object share the same request class which contains the username and realm: the userAuthExt class.30

For PAP, the extended class, called authPapExt, contains the unencrypted password, as well as of course, the attributes that it inherits from userAuthExt and authExt.

For CHAP, the extended class, called authChapExt, not only contains the encrypted password (in the authChapExtResp attribute), the attributes inherited from userAuthExt and authExt, but also the challenge code, which is generated by the PEP and the identifier.31

The identifier is used by the PAP protocol to match challenge request and response from the user. However, it is not really necessary to forward this information to the PDP. There is no harm done, so future version of the draft may still include it.

The challenge must be sent for two reasons. First of all, the PDP needs the challenge to encrypt the internally stored password in order to verify it with the result sent by

30 I expect that this class will be renamed. For example to authExtRequest.
31 It should be noted that the current CHAP class does not yet include a code to signify the algorithm used to encrypt the password and challenge code. By default, this algorithm is MD5, but an other algorithm can be negotiated as part of the Authentication –Protocol Configuration Options during the PPP LCP negotiation. For example, Microsoft uses an other algorithm for CHAP, which is known as MS-CHAP. In order to support MS-CHAP (which is a valid variant of CHAP) as well, the authExtChap class needs to include a one-byte attribute to signify this algorithm (default value is 5, CHAP with MD5).
the User via the PEP. Secondly, the security of CHAP relies on the fact that the challenge code is changed with each request that is sent to the User. A potentially malicious PEP could be detected by the PDP by checking if the challenge code does indeed change with every request.

<table>
<thead>
<tr>
<th>Why the can PDP specify HOW to retrieve credential info</th>
</tr>
</thead>
</table>
| There is a big difference between the support for EAP, and all other authentication protocols. For EAP, the PEP does not sent any authentication data along with the first event, but the PDP does initiate the authentication sequence. For PAP and CHAP, however, it is the PEP that initiates the authentication sequence. To allow the PDP to configure the PEP and let it know when to use a particular authentication mechanism, the eventHdrAuthProtocol class was created and a extension of the eventHandler class, the datapathEventHandler class was created. Now for each eventHandler, the PDP can specify which authentication protocol the PEP should initiate before it should trigger an event.  

Support for HTTP authentication
The Access Bind PIB strives to support HTTP authentication as well. However, the current draft does not yet contain classes for this support.

There is an issue with HTTP authentication though. Just like the PEP creates a CHAP challenge, it it also possible that it creates an HTTP response of type 401 (Unauthorized) [RFC 2616], to which the user may responds with valid authentication credentials.

However, it is suggested in section 10.4.2 of RFC 2616 that such an HTTP response not only includes a WWW-Authenticate header, but also a HTML page (actually, any entity according to the RFC) explaining that authentication is necessary. In particular, a User Agent (a webbrowser) should display such a page in case the authentication failed. Such a page may indicate where the user can retrieve his or her password. It may be a bad idea if the PEP is capable of generating such a complex response with webpage and all. Alternatively, traffic of unauthorised session may use a different datapath by forwarding it to a different DiffServ element. This DiffServ element may forward the data to an external HTTP server.

Currently, the draft does not support the ability to use a different DiffServ element for Authorised and unauthorised data traffic, since each eventHandlerElement has just one pointer to a DiffServ element, and each eventHandlerElement may (using the lookup table mentioned earlier) filter both unauthorised and authorised data traffic. Therefore, the current draft does not support an external HTTP server that returns a page for authentication. The previous 00 draft did support this.

7.5. Provisioning data structure
The Access Bind PIB does not define per-user provisioning data structures. The only provisioning-related data structures that the data structures that are provisioned as boot time of the PEP, like the event Handler.

---

32 In my opinion, not the eventHandler class, but the eventHdrElement class must be extended. That would allow different (PAP, CHAP or no) authentication for different eventHandlerElements. Useful if –for example– traffic coming from a server does not need to be authenticated. Also, the current data structure allows to specify PAP or CHAP, but does not allow to specify options. For example this means that MS-CHAP is not supported, which is a perfectly valid CHAP authentication but with another encryption instead of MD5.
33 Note that it blatantly violates the idea of layers in the first place to return a HTTP page if the traffic the user send might have been a completely different protocol, like one of the email protocols.
Currently, the provisioning is done by installing DiffServ elements in the PEP. The data structures denoting DiffServ elements can be found in for example [draft-ietf-diffserv-pib].

**Per-session provisioning**

In the Access Bind PIB, the result of an access request is a boolean: yes or no. If the result is affirmative, the source address is added to the lookup table specified by the eventHandlerElement. However, in the current data structure it is not possible to connect a specific next DiffServ element to a certain session (COPS Client Handle)\(^\text{34}\).

If the PDP wants to provision a different QoS at the PEP for each session, it will have to install a new eventHandlerElement, Scope and Filter onto the PEP, with the nextMatch attribute in the eventHandlerElement pointing to the desired DiffServ element. It is my opinion that it is desirable that the PDP can specify a different next DiffServ element for each COPS Client Handle, without the need to install a new eventHandlerElement. After all, one of the purposes of this draft is to allow per-user provisioning, and this seems like a relatively complex way to do this. In particular if each eventHandlerElement can change the look-up tables, and with each provisioning, the PEP will need to check if it needs to update it’s look-up tables.

---

\(^{34}\text{This was possible in the 00 revision of the Internet Draft.}\)
8. Discussion and Conclusions

The Access Bind PIB is an Application-Specific data structure. It is DiffServ-specific. Together with COPS for transport, it is an Application-Specific protocol as described by the AAA-model: a protocol between an AAA-server (PDP) and an service equipment (PEP). It can not be reused for a protocol between two PDP's.

There are still a few issues in the current version (02) of the Access Bind PIB, which can be resolved in a new revision.

This chapter will compare the Access Bind PIB to the generic AAA authorisation models. It will look at the possible application of the Access Bind PIB as the protocol between a service equipment (PEP) and AAA-server (PDP). In particular it will check if the data structure is able to combine authentication data with QoS provisioning and how generic this combination is.

8.1. Fitting the Access Bind PIB in the authorisation models

The AAAarch research group has defined three basic authorisation models, the pull model, the push model and the agent model [RFC 2904]. The appendix “Acronyms and Buzzwords” describes these models in detail.

The Access Bind PIB describes a PEP, a PDP and a User, where the User wants to get access from the PEP, and the PEP outsource the authentication and authorisation of the user to the PDP. This maps neatly onto the pull model (See Figure I-2), as described by the AAAarch [RFC 2904]. The PEP maps onto the service equipment and the PDP maps onto the AAA-server. It is not co-incidental that the terms PDP and PEP, which are used in the Authentication Framework Documents [RFC 2903 to RFC 2906], are used in the COPS and the Access Bind PIB terminology as well.

The Access Bind PIB as an Application Specific data structure

The Access Bind PIB classes embedded in COPS messages is the protocol between the PEP and the PDP. This is protocol number 5 (See Figure I-19), as defined by the internet draft Structure of a Generic AAA server [draft-irtf-aaaarch-generic-struct].

This model suggests that this protocol is application specific, because the PEP communicates with an Application Specific Module in the PDP, not with a generic Module.

This is true for the Access Bind PIB: The focus on the DiffServ model is inherent to the data structure and philosophy: the Access Bind PIB functions as one complex DiffServ element in a PEP, a complex kind of classifier. The whole concept only makes sense as part of a datapath. If you want to provision for other applications than DiffServ elements (for example, for ordering a pizza instead of ordering network access), it is likely that the combining must be done using other methods then using the nextMatch elements in the eventHandler and eventHdlrElement. This is not trivial, and I belief such a new revision can never be created while maintaining compatibility with the current version. Therefore, I claim that the Access Bind PIB is only useful when used in conjunction with the DiffServ model.

Therefore, I conclude that the Access Bind PIB is Application Specific data structure, because it relies on the DiffServ application.
The three authorisation models
So far, we have seen example usage’s of the Access Bind PIB in the pull model [RFC 2904], where the PEP outsources decisions about configuration and authorisation to the PDP. However, since COPS-PR allows the PDP to configure the PEP, it is also possible to use this same protocol for the agent model. The third model, the push model, does not specify communication between the PEP and the PDP, so it is not applicable to use the Access Bind PIB in that model.

So despite that the Access Bind PIB is application (DiffServ)-specific, it does not seem limited to one authorisation model.

8.2. Potential Usage as a generic AAA-protocol
Although the Access Bind PIB was not designed to be used as a generic AAA-protocol, it may be interesting to check if (components of) it can be used as a generic AAA-protocol, a protocol between two AAA-servers.

Communication between two AAA-servers
If the Access Bind PIB is to be used as basis for the protocol that handles the communication between these two AAA-servers, two immediate problems arise:
- First, the Access Bind PIB uses the concept of events, which are triggered by an eventHandler. This means that the AAA-server needs to install eventHandlers at the other AAA-server. That seems badly scalable.
- Secondly, the eventHandlers and the result are based on the concept of a DiffServ datapath with DiffServ elements in it. An AAA-server has no notion of a datapath: there is no next DiffServ element to forward new traffic to and no packets to trigger the filter.

Perhaps some of these issues can be evaded, but it would take quite some effort to make a sensible concept for two AAA-servers using the Access Bind PIB as a

---

35 I am aware that the creation of a generic AAA-protocol is currently not a work item in any working group of the IETF, nor is the IRTF chartered to create standard track protocols. This check is only relevant this study, and is not an attempt to create such a protocol.
protocol. Therefore, I am forced to conclude that the Access Bind PIB is a bad choice as the basis of a generic AAA-protocol, because of the huge conceptual misfit.

**Reusable elements**
The Access Bind PIB does not make a good protocol to be used as basis for a generic AAA-protocol between two AAA-servers. However, it may be possible to re-use some of its concepts, like the contextData classes, and the ability to carry EAP packets. However, the main classes, like events, eventHandlers, eventHdlrElements and Scopes, do not seem re-usable.

This does not mean that the concept of events is always a bad concept to use in a protocol between two AAA-servers. It is just the event class as defined in the Access Bind PIB that is a bad concept, because this event class is based on event triggers, and event triggers is a bad concept for use between two AAA-server. That would scale very bad, because each AAA-server needs to install event triggers for every possible event in every other trusted AAA-server.

Also note that between two AAA-servers the Accounting part of AAA is increasingly important, along with security concepts like non-repuditation. The Access Bind PIB does currently not deal at all with accounting or non-repuditation.

**A meta-data approach**
It is interesting to notice what section 5.3.1.2 of [RFC 3169] says about resource management and “The ability for the NAS to allocate and deallocate shared resources from a AAA server servicing multiple NASes.”: “This feature is primarily intended for NAS-local network resources. In a proxy or multi-domain environment, resource information should only be retained by the server doing the allocation, and perhaps it’s backups. Authorization resources in remote domains should use the dynamic authorization features to change and revoke authorization status.”

Translated to the DiffServ model, this means that it is indeed a good concept that the AAA-server provisions QoS (and thus network resources) at the (local) NAS (the service equipment or PEP). But the protocol between two AAA-servers should be more generic. Perhaps a combination of provisioning meta-data (which does not depend on the DiffServ model) along with policy data and authentication data structures (perhaps using EAP) is a good idea.

I believe that a generic AAA-protocol can very well be created with COPS as the transport protocol. In particular because COPS is a reliable and fast protocol, which is, among other things, able to carry XML data and has an excellent state synchronisation mechanism.

**8.3. Potential usage as a NAS authentication protocol**
Because the Access Bind PIB is a data structure that is transported between a PEP (like a NAS) and a PDP, it is tempting to think of it as an authentication protocol, similar to RADIUS and DIAMETER.

I am aware that, in the IETF context, this is a political statement. The Access Bind PIB was not intended to even compete with DIAMETER or RADIUS. However, for the purpose of this academic study I will consider its potential usage nonetheless.

The rationale for this check is based on a discussion in the IETF, sometime around 2000, when different protocols where considered by the AAA Working Group for use.

---

Anyone who has read an RFC to the last line, knows that each RFC states that "[...] THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, [...] INCLUDING [...] ANY IMPLIED WARRANTIES OF [...] FITNESS FOR A PARTICULAR PURPOSE.". With this discussion in mind, this legal mumbo jumbo seems especially applicable to the Access Bind PIB.
as the AAA protocol for NAS, Roaming and Mobile IP applications [RFC 2989]. One of the considered protocols was COPS (other were RADIUS+, DIAMETER and SNMP) [RFC 3127]. It seems attractive to reason that since COPS was considered as the basis for a NAS AAA-protocol, and the Access Bind PIB is aimed at the first A (authentication), the result might be reusable as part of a potential AAA-protocol.

Despite the fact that the Access Bind PIB was designed with the idea of binding access control and provisioning in mind, the discussion in the IETF made it interesting to see where this classification of the Access Bind PIB as an authentication protocol between a NAS and an AAA-server came from, and if this classification is justified. This does not have to be the case; after all I’ve shown in the previous section that the Access Bind PIB is clearly a bad AAA-protocol to be used between two AAA-servers.

**Access Bind PIB Protocol Evaluation against RFC 3169**

The best way to judge the Access Bind PIB is by comparing it to the criteria set forward in “Criteria for Evaluating Network Access Server Protocols” [RFC 3169]. This answers the question if the Access Bind PIB is a good “AAA-protocol between the NAS and an AAA-server”, as RFC 3169 terms is.

Because the check was very rough, I have not included the table here: for a thorough conclusion, each requirement should be checked and discussed. Therefore, I will just give the preliminary conclusion:

The general protocol characteristics requirements (section 5.1 in RFC 3169) are almost all fulfilled, or could be fulfilled by extending COPS. The main problem is that the protocol can only be used between a PEP and a PDP, not between two PDP’s, which seems a roaming requirement (section 5.1.1.4). The Authentication and User Security Requirements (5.2) are also all supported, even though 5.2.1.3 (Multi-Phase authentication) does not seem to be met: there is no reason given for a failed authorisation (the Access Bind PIB does not differentiate between authentication and authorisation). Authorisation, Policy and Resource management requirements (5.3) are only partly met, mostly because the Access Bind PIB has no support for timed events: nor is it possible to reserve future access nor can the PDP specify a time-out. Accounting and auditing requirements (5.4) are not met.

Although quite a few requirements are not fulfilled, I belief that most or even all requirements can be fulfilled by extending COPS, COPS-PR, the Access Bind PIB or the DiffServ PIB. Concluding: the Access Bind PIB is not a NAS AAA-protocol, but it possible to use it as a basis for such a protocol.

Whether or not this is a desired direction, is another discussion.

---

37 The AAA Working group focuses on three scenario’s [RFC 2989]: Network Access Server Requirements, Roaming Operations and Mobile IP. The AAAarch research group defines six usage examples in RFC 2905: PPP Dialin with Roaming, Mobile-IP, Bandwidth Broker, Internet Printing, Electronic Commerce and Computer Based Education and Distance Learning. PPP Dialin with Roaming is comparable to the two examples in RFC 2989, Network Access Server Requirements and Roaming Operations. If you count them as two, you’ll get seven usage scenario’s; John Vollbrechts once mentioned eight different scenario’s.

38 After RFC 3127 was published, the AAA Working Group decided to use DIAMETER as a future protocol between the NAS and an AAA server, and it was decided not to create any generic AAA-protocol in the IETF.

39 The AAA working group in the IETF uses the term AAA-protocol for the authentication protocol between the NAS (PÉP) and the AAA-server (PDP). This is a different location then the AAA-protocol mentioned sometimes in the AAAarch research group. The AAA-protocol in the AAAarch is the protocol between two AAA-servers (PDP’s). Therefore, I use the term “NAS AAA-protocol” or just “NAS-protocol”, even though it is not limited to Network Access, as described in footnote 37.
8.4. Edge Devices and Network Access Servers

Although the Access Bind PIB mostly discusses edge devices and network access servers, these two are not equivalent. The Access Bind PIB does not make implicit assumptions that the PEP needs to be an edge device. It does however, use the DiffServ model, which clearly defines a datapath, and the PEP, therefore, needs to be a network device (either a Network Access Server or a core router). This section will discuss the difference between edge devices and network devices and network access servers.

A NAS is a Network Access Server. It should be noted that the Access Bind PIB is designed not only to deal with network access, but with other types of access as well, like access to a server. As long as DiffServ can specify the next service (egress port), the Access Bind PIB can handle it. Usually, this would indeed be network access, but there is no explicit requirement for this.

![Diagram](https://via.placeholder.com/150)

Figure I-20. A typical edge device. The network access server is located at the edge of the network, and acts as a portal for the user to access the network.

The Access Bind PIB talks about Edge Devices in the introduction. An Edge Device is a device on the edge of the network, usually connecting the user to the Internet. See Figure I-20, which shows the location of an edge-device like a dial-in server.

![Diagram](https://via.placeholder.com/150)

Figure I-21. Example of an Access Server which is not located between the User and the network. In order to access the server, the User needs to connect to the access server, which authorises the user before letting him or her access to the application server. The connection between the Access Server and the Application Server can technically be just a very small network, so the access server can be a Network Access Server (NAS).

However, an access device does not necessarily have to be a portal between a user and the rest of the network. It can also be located near the service. See for example Figure I-21, which shows an access server just in front of some application server. Such an application server could for example be a server that produces a streaming video of a university college. It is desirable that this server can only be accessed by someone who is identified as a student. Another example is to use it for measurements of physical for high school students over the internet. In an experiment set up by Utrecht University [Van Yperen (2000)], students could control a computer connected to an experiment used to measure the speed of light, as well as an other computer to view the experiment with a web camera. Both computers are behind an Access device, to prevent multiple students from controlling the experiment at the same time.

8.5. Suggestions for improvement

By examining the data structure of the Access Bind PIB, a few suggestions for improvement where discussed, mostly in footnotes.
Rephrasing, the following limitations were observed:

- FIt may not be able to use filters to specify lookup tables. I believe it is better to explicitly specify the fields of the lookup table in the eventHandler (see section 7.2).
- There is no interim behaviour for traffic coming for a yet unauthorised datapath. I believe this should be added. That implies that, unlike in the current revision, each eventHandlerEment has the same lookup table. I don’t believe this is a problem, because two eventHandlers with different lookup tables can be placed in serial using the DiffServ model.
- I believe it is desirable that the PDP can specify a different next DiffServ element for each COPS Client Handle. It currently can only be done by installing a new eventHandlerElement, Scope and Filter (see section 7.5).
- It may be beneficial, in particular for HTTP authentication, to have a different next DiffServ element for authenticated a non-authenticated data. This may also solve the previous issue.
- I believe that the specification of PAP of CHAP as a PEP-initiated authentication protocol should be done by extending the eventHandlerElements, instead of the eventHandler itself.
- There is no option to configure authentication protocol options, like the algorithm option for CHAP (MD5 by default, but different for MS-CHAP).
- It is not possible to provision option, other then using DiffServ element. For example, an LCP option like the IP address can not be specified for each session.

Non-generic design of the Access Bind PIB

Beside these issues listed above, a few issues that are more general have been spotted; ironically, most were found by comparing the Access Bind PIB to RFC 3169. Given the usage goal of the Access Bind PIB, I do not believe that these issues need to be fixed, but it is good to realise which design decisions limit a generic usage of the Access Bind PIB.

I claim that the Access Bind PIB is not truly generic. There are four reasons for this. Two of these reasons are fundamental; two of them can be fixed.

- First, the Access Bind PIB is based on the DiffServ model. This is a problem discussed in section 8.1, which can not be fixed without completely altering the design of the Access Bind PIB.
- My second observation is that there is no ‘global’ namescope: all requests that come in from a User at the PEP apply to this very PEP. The design of the PEP assume that the access request from the User comes in through the same datapath as that will be used for the actual usage of the service later. It is possible for the PDP to install a usage filter for an ingress datapath, other then the one associated with the event that was just triggered. It is unclear what kind of result should be returned in this case: a success result would add the datapath from the event to a lookup table, while a failure result would leave the user unaware of the fact that the other datapath was successfully installed. I believe that this can be fixed as long as the draft is not published as RFC. I think it is possible, but harder, to fix it while maintaining backward compatibility as soon as it is published as a standard.
- Related to this second observation, but harder to fix in the current design, is that it is now assumed that each request is supposed to be for a connection that is required immediately. It is not possible to request access for one hour later. It is possible that the PDP will reognise the request, and provision the correct

---

Again, this does not signify a problem as such: the Access Bind PIB was not designed to be generic.
options an hour later. However, in the meantime, the original requester is added to the lookup table. This might be undesirable, in particular, if the User makes two such reservations: one for today and another for tomorrow. It seems undesirable that the PEP should use the same Client-handle for both sessions. In the current design it is possible to specify this (using the eventHdlrElementEventCriteria attribute).

- Last, there is no options for the User to request a particular type of service or QoS (Quality of Service). Only requests for a service that has been pre-installed with a particular trigger (eventHandler) can lead to a trigger and can thus be fulfilled. This demand is perhaps more a requirement for the protocol between the User and the PEP rather then the protocol between the PEP and the PDP, since for this request, the User must have some means of specifying the desired QoS. Using a protocol specific filter, it is possible to recognise a protocol that requests a certain QoS, like the RSVP protocol [RFC 2205]. The Access Bind PIB has in fact support for RSVP, showing that it is possible to support these kind of requests. However, in the current implementation it is ambivalent in my opinion: each type of RSVP message is sent in an other instance (PRI type), so the PEP needs to be aware of the meaning of RSVP messages, and thus needs to be upgraded if the RSVP protocol is extended with new types of messages. On the other hand, the actual content of the RSVP messages is not parsed into a more generic request, but are just forwarded in contextData instances to the PDP, so the PDP needs to implement RSVP as well. Idealistically, the protocol should define some generic classes that describe the desired QoS, like the desired bandwidth, the desired time of usage (now or later) or the desired latency.

There are in fact a few other minor issues, some of which are listed at the start of this section. They include support for the PDP to request a status about current sessions, like the amount of data that has been sent by a specific user so far. Related, I miss a few options in the DiffServ PIB to specify limitation for a particular session, like a time-limit. In addition, I missed an option in any of the protocols, for the PEP to specify its limitations. It is possible to specify the layers and protocols it recognises, but not for example, its bandwidth of the number of concurrent sessions it can handle. I belief that these are no major problems and can be fixed anytime by extending the protocol.

8.6. Combining authentication and provisioning

Despite it’s limitation, which have been stressed in the previous section, the Access Bind PIB also has a few very powerful characteristics. For one thing, the combining of access control (authentication) and provisioning (configuration of QoS) seems to work very well. In most cases, this eliminates the need to create complex devices that use multiple existing protocols to communicate with each other. The result is more stable and less error-prone implementations because there will be no risk of two state machines (one for each protocol) that don’t correspond.

The Access Bind PIB is unique in the fact that it can both handle start-up configuration of a PEP as well as per-session configuration based on authentication results. This is a very scalable solution: it eliminates the need to configure each PEP individually (the PDP automatically configures each PEP), and is also able to adjust the configuration based on the user who has logged in, rather then based on static information, like IP addresses.

Because of the provisioning capabilities of COPS and COPS-PR, the same protocol (COPS) can be used in a pull model and in an agent model. In the pull model, the Access Bind PIB can be used along with DiffServ provisioning. In the agent model, just the DiffServ provisioning is used.
Client handle as a session

Instead of keeping a session data structure, the Access Bind PIB uses the COPS Client Handle as a session, as was discussed in section 7.2. By using these COPS Client Handles, the PDP can authorise a user and let the PEP know of the result, even though the username is never stored on the PEP nor on the PDP. It is possible to offer differentiated services, based on the authentication result. This can be done in two ways: by pushing a QoS policy onto the PEP using DiffServ element, and only allow authorised users, using a lookup table. Alternatively, this can be done by using this same COPS Client Handle, and configure a QoS policy onto the PEP using DiffServ element on a per-user basis. This configuration can be managed by using the COPS Client Handle; for example, the PDP can send a (named) remove decision for a certain session by specifying the associated COPS Client Handle, and the PEP must send a delete message for a client handle as soon as the associated state is removed.

Conclusion

Concluding, the Access Bind PIB is not an all-purpose (generic) protocol to be used between the PEP and the PDP, mainly because it can only be used in conjunction with the DiffServ model. It can, however, provision a QoS policy onto a PEP, either in advance or per-session, based on an authentication result. It successfully combines access control and provisioning.
II. ACRONYMS AND BUZZWORDS

Computer science is an area often overcrowded with terminology and abbreviations. Whether this is due to the exact nature of the topic (detailed definitions are compulsory, since a single wrong bit can make your computer crash), or due to the people practising the topic (nerds), may be a study on its own. I take this for granted, but will try to give an overview of all the terminology used in this thesis. Feel free to skip it if you are familiar with it.

The goal of this appendix is not to give an exhaustive overview of network protocols in general. It will give you a basic understanding of the RAP working group in the IETF and the AAAarch research group, by explaining what they are and what they do. It should also give you an idea on terms like authentication and authorisation.

Other, more general, but fairly readable sources for introductions to networks and network protocols include the excellent book “Computer Networks” by Andrew S. Tanenbaum [Tanenbaum (1996)], a must-have for anyone who wants to understand how networks work. (The third edition was published in 1996, and translated into many languages, including Dutch; a fourth edition will be published in the summer of 2002). Furthermore, Cisco maintains a site called “Introduction to Internet” [Cisco], part of their Internetworking Technology Handbook, which mainly goes through the OSI model. Furthermore, whatis?com [Whatis?com] is a website with lots of definitions and descriptions of computer related words. Though the quality between definitions differs, they usually provide a good overview in one or two paragraphs for any unknown term you encounter.

This appendix expects that you have a basic understanding of a computer network, like the Internet, and are familiar with commonly used terms like IP-address and router.
1. **What is a network protocol?**

Network protocols define how computers communicate with each other. When computers communicate with each other, they use multiple types of protocols at the same time, each type of protocol focused on a particular task, each protocol layered on top of each other.

When computers communicate with each other, it is imperative that they exactly understand what the other means. For humans, confusion of the tongue can be inconvenient, but is usually manageable. For computers, it can be disastrous. If I order my e-banking program to transfer 50 Australian dollars to fund a purchase, I may hope that the computer of my bank does not withdraw 50 US dollars. But confusion can take place at different levels: not only should the amount and the unit be defined correctly, the whole concept of withdrawal should be understood by both devices as well.

Instead of going into detail about syntax and semantics, I will state that each time two devices communicate with each other, the communication between these devices should be well-defined. These definitions are described in network protocols.

1.1. **The OSI model and the Internet model**

Implementing a single network protocol that handles all aspects of communication, ranging from voltage, addressing, routing, connection set up to error-handling and actual applications, would be an enormous task. Therefore, most network architectures define a protocol stack, where lower layers (like ethernet, wireless or dialup) define physical properties, and higher levels handle properties like routing and sessions.

The protocols that make the internet use stacking as well, and they were originally divided in four layers, the host-to-host layer, the IP layer, the TCP layer and the application layer. However, this model has a few flaws, so nowadays, most Internet protocol designers refer to the OSI model, as shown in figure II-1. The IP layer mapped to the network layer, and the TCP layer mapped to the transport layer.

The idea is that one layer on the one computer will only talk to the same layer on the other computer, without knowledge of any of the other layers. So the Network layer, which in the internet world handles the addressing (using IP addresses) and routing, does not know about the protocols used in the application layer: that can be HTTP, streaming video or e-mail. It is also not aware of the layers below: that can be a dialup connection, wireless LAN or ethernet. Similarly, an datalink layer protocol like ethernet can carry the Internet Protocol (IP) on top of it, but also IPX (part of Novell Netware) or AppleTalk.

In practice, any layer will use the services of the layer beneath it\(^1\), which will take care of either passing it on to the layer beneath that one, or for the actual transmission. The concept of layered network design is explained in an excellent way in

\[^1\] Except of course layer one.

---

Figure II-1. The OSI reference model for networks
Tanenbaum’s “Computer Networks” [Tanenbaum (1996)], mostly in sections 1.3 and 1.4.

This thesis will refer to the layers in the OSI model, even though that reference models has its own flaws. For example, the datalink layer can be considered as two separate sublayers\(^{42}\) and the session- and presentation layer are hardly used. Tanenbaum [Tanenbaum (1996)] even uses a mixed reference model, which omits these layers.

When I talk about network protocols, I refer to the syntax and semantics on a single layer. For example, how TCP messages on the application layer will talk to another device on the same layer who is also talking TCP. I do not refer to the services TCP is offering to the layer above, unless that is specifically mentioned. In the internet world, these two concepts are easily mixed. That’s because the drive for creating a new protocol is the new services it can offer, but you end up with the definition of the actual protocol that defines the messages that are sent between the devices.

\(^{42}\) The datalink layer is sometimes split in the LLC (Logical Link Control) layer and the MAC (Medium Access Control) layer. The MAC layer functionality is only used in broadcast networks, not in point-to-point networks.
2. What are standards and who makes them?

Standards are documented or commonly used specifications. There are many organisations who produce standards.

When people or organisations agree on something they make or do, they create something called a standard. For example, they agree on how an object should look like or behave, or on how and when certain communication should occur, or even what the behaviour of someone should be. This thesis will go into detail about standardisation of network protocols, but this is surely not the only place where standards are developed. Imagine a world without standards: not only would the plug of your VCR not fit into the wall outlet, you wouldn’t even have TV (PAL, NTCS and SECAM are standards for television recordings). Not that you would care, since you won’t be able to go to work by train, for it would be quite a rough ride if the standard gauge wouldn’t be defined as 1435.5 mm (actually 4ft, 8 1/2 inch, since that’s the gauge of the first railway, in Britain). And don’t think of going by bike, since the bolts and nuts wouldn’t fit, just as the tire wouldn’t fit on the rim. So clearly, standards are a Good Thing.

2.1. What is a standard?

Most people will think of a standard as something that is agreed upon by everyone. Though that is close to the concept, it is not entirely true. For example, not everyone does need to agree: there may be competing standards, just like the above-mentioned PAL and NTCS for TV-systems. As Andrew Tanenbaum once commented, “The nice thing about standards is that there are so many of them to choose from”. According to Merriam-Webster [Merriam-Webster], a Standard is “something established by authority, custom, or general consent as a model or example.” I will make a few observations about this definition.

De-jure and de-facto standards

It makes sense to expect that a standard is properly documented. However, there are two types of standards: de-facto (commonly used) standards and de-jure (officially issued) standards. A de-jure standard must be issued by an authority. This authority may be an ad-hoc co-operation between multiple organisations, as long as it is recognised by these organisations as the authority when it comes to a particular standard. Authorities whose sole purpose is to create or ratify standards are called standardisation bodies. That doesn’t mean that such a standardisation body must create the document itself: that may be done by anyone, as long as the author is willing to hand over the rights to the authority, and the authority is willing to publish it as a standard.

A de-facto standard, on the other hand, might not even be documented: it is just the way people and organisations do things. The more people use it, the more standard it will be. This makes things rather vague: what is common enough to call something a standard?

Therefore, this thesis will differ from the definition given by Merriam-Webster, and will only consider a de-jure standard as a ‘standard’: a standard is a documented specification that is acknowledged by multiple organisations as a standard. Not all organisations do need to implement it. For example, a published specification by a single person can be used by almost everyone. However, that does not make it a standard with regards to this definition; it will just be a commonly used specification. All standards are specifications, but not all specifications are standards. In addition, a standard that has become out-of-date, and is hardly used is still a
standard, albeit an unused, obsolete or deprecated standard. The key here is that multiple organisations agreed on something as the way to do things, at the time the specification was created.

Public or Private information
A common mistake is to think that all standards are public and free information. To a student doing this research, like me, this seemed logical: I was used to the academic world, where knowledge was almost by definition public. Moreover, the body I have been working with, the IETF, publishes all standards for free on the internet. However, there are also standardisation bodies who fund themselves by charging a fee for their documents, kind of the way an on-line magazine would work. Specifications, like most standards, are usually copyrighted, usually by the organisation body, and sometimes by the organisations where the authors belong to or the authors themselves. Finally, standards may very well contain or even describe patented techniques. That means that you may have to pay for retrieving the document and for using it in your products.

2.2. Standards in computer science
Standards are used everywhere, from physical properties, like the former mentioned bolts and nuts, to administrative properties, like the ISBN numbers for books. In computer science, they are also used from the physical properties of the connectors at the back of your computer to the format of the file you just created.

For network protocols, all layers as mentioned in the section about The OSI model and the Internet model are described in different specifications, most of which are a standard.

2.3. Which standardisation bodies are out there?
Since there are many standards out there, you might assume there must also be many standardisation bodies to make all these standards. You are right! There are. Most of these bodies are independent organisations, even though some of them are embedded in larger organisations, like a national government. For example, the telecom standardisation body ITU is embedded in the United Nations. Most of the organisations discussed here operate on an international scale, although some of them also have national equivalents, particularly in the telecom world.

When it comes to standard bodies involved in computer networking, most of these organisations focus on a specific type of standards, like the standards used in one or two layers of the OSI model. I will cover a few examples in this section.

ISO
ISO, the International Organisation for Standardisation\textsuperscript{43} [ISO], publishes and has published a lot of technical engineering standards. It was founded in 1947 and has 93 countries as member, each represented by the national standardisation body, like the Dutch NNI, the German DIN and ANSI (USA). They are most likely the best known organisation, since they create standards which greatly improve every persons daily life, like the maximum thickness a credit card may have, to prevent your wallet from getting to bulgy (0.76 mm). Of course, they also create less important standards like the earlier mentioned gauge of a railroad track, the pitch of nuts and bolts or the 2-

\textsuperscript{43} Clever readers will point out that the abbreviation for International Organisation for Standardisation is IOS, not ISO. According the ISO website [ISO], “ISO” is not an abbreviation, it’s a word, derived from the Greek iso, meaning “equal”, which is the root of the prefix “iso-” that occurs in a host of terms, such as “isometric” (of equal measure or dimensions) and “isonomy” (equality of laws, or of people before the law). However, I expect that also political motives have played a role, similar to the fact that the standard time is called UTC, just because it was neither English (CUT; Coordinated Universal Time) nor French (TUC; Temps Universel Coordonné)
IEEE
IEEE, the Institute of Electrical and Electronics Engineers [IEEE], is originally an American professional association and is mostly active in electric and electronic engineering. It organises meetings and publishes magazines. One of the six interest groups is the standards association, which is divided in working groups. One of the most notable of the almost one hundred working group areas is called IEEE 802 [IEEE 802] and defines standards for LAN’s (Local Area Networks). Being creative from the start, the name comes from the fact that the working group area was formed in February 1980 (80-2).

ITU-T
ITU, the International Telecommunication Union [ITU], is, unlike volunteer organisations such as ISO and IEEE, formally established by a treaty between nations, and is formally part of the United Nations. The ITU is divided in three parts: ITU-R, which distributes radio frequencies, ITU-D, which support developing countries and ITU-T, which creates telecom standards.

3GPP
3GPP stands for Third Generation Partnership Project [3GPP], and this cryptic name refers to mobile communication: the first generation cellular telephones used analog mobile phone networks (With networks named TACS, AMPS and NMT). Among the second generation mobile phones are – in Europe – at least the GSM phones (in the USA, TDMA and CDMA are used as well, and are often referred to as PCS – Personal Communication Services). The third generation mobile phones are said to offer not only voice and simple messages services, but also broadband applications like video. The suite of protocols 3GPP is developing to make this all possible is called (in Europe) UMTS, Universal Mobile Telecommunications Service (The USA calls it either 3G or Wideband CDMA).

To add to the general confusion, some USA companies formed a competing organisation called 3GPP2. Possibly, they want to elevate the apparent lead that European companies like Nokia and Ericsson are said to have in mobile telephony, but maybe they just want to create opportunities to invent even more challenging abbreviations.

IETF
IETF, the Internet Engineering Task Force [IETF], is unlike the previous organisations. It’s very informal. While attendees of ITU meetings are business or governmental employees wearing suits, the dress code for an IETF meeting is described at their site as “Since attendees must wear their name tags, they must also wear shirts or blouses. Pants or skirts are also highly recommended.” It’s origin goes back to 1969, when the ARPANET was created, a military and university network. Back then, people would come to the IETF in military uniforms or jeans. These days, there are no military officials at IETF meetings, and the corporate employees are in the majority. However, as far as the dress code is concerned, the de-facto standard is still jeans. Another difference is that unlike the ISO and IEEE, the IETF standards are freely available for download on the internet. The IETF is funded by an external organisation, the ISOC, the Internet Society. Organisations like ISO and IEEE generate funding by making members pay for the documents they download. More information about the IETF can be found in the next chapter. The IRTF, the Internet Research Task Force, is the research cousin of the IETF and doesn’t publish any standard documents. It therefore can’t be regarded as a standardisation body. It is described in the section “What is the IRTF?”
**Global Grid Forum**

GGF is the Global Grid Forum [GGF], an organisation that defines standards to create a computing grid: a way for scientists to spread a large single computation over multiple computers. Problems involved here are: ways to split data, how to tell a computer when to perform which calculations on what data, and how to tell where to transport it to, and how the actual transportation goes, and of course how all this is done securely by a trusted person. Using a super computer may not be free of charge, and the owner likes the invoice to go to the right person.

**W3C**

W3C, the World Wide Web Consortium [W3C], was only recently created to define application layer protocols for use on the “World Wide Web”*. In particular, they defined HTML, the standard that defines the structure and contents of most websites. Though the World Wide Web, or just the Web, is not clearly defined, you can regard it as all that you can view with your webbrowser. The Internet and the Web are not the same: you can use e-mail over the internet, but it's not part of the web. W3C is similar to the IETF because it originates from the scientific community and still publishes all its standards to download for free on its website. In the application layer, you will see specifications which are sometimes not obviously network protocols, like mark-up languages and images formats, such as PNG and SVG, which are defined by W3C as well.

There are still many more standardisation bodies, on either side of the protocol stack. Like, on the one side, Web3D, which does similar work as W3C, only for 3D graphics, or like ECMA[ECMA], a standardisation organisation which published ECMAScript, an attempt to make Netscape’s JavaScript and Explorer’s JScript more compatible. On the other end, there are bodies like IEC [IEC], that create standards in electrotechnology.

Of course there is sometimes overlap between the areas two organisation are active in. For example, the IETF and GGF will sometimes share interest in the same topics. Standardisation bodies want to avoid creating two competing, but incompatible protocols, so they usually have a few people who try to keep in touch with other standardisation bodies to prevent this from happening. Usually, collaboration between two bodies will have as the result that one body creates a specific protocol, and the other body will refer to that protocol. Another option is that a protocol or set of protocols is renamed. For example, ISO has a protocol suite, called ISO 8802, which is essentially just (a reference to) IEEE 802.

Summarising these bodies, ISO is very generic, the ITU and 3GPP are telecom based. From the authorities who focuses on computer networks, the IEEE 802 focuses on the physical and datalink layer; the IETF and IRTF focuses on network and transport layer; while the GGF, W3C, Web3D and many other, mostly smaller organisation develop protocols for the session and application layer.
3. What is the IETF?

The IETF is a standardisation organisation who publishes documents about the operation of the internet. These documents are called RFC’s. Some, but not all, RFC’s describe protocol standards.

The IETF is a standardisation organisation, and is fully called the Internet Engineering Task Force. It is the organisation that defines all the protocols needed to make the internet function smoothly. The IETF is generally only concerned with the protocols on the network- and transport layer, and does not create protocols on lower layers, like ethernet or wireless LAN protocols. Just as the name suggests, it also does not define protocols for other protocol stacks. For example, telecom networks may use non-Internet protocols on the network and transport layer, but the IETF isn’t concerned with those issues.

There is an ongoing debate whether or not the IETF should be concerned with protocols on a higher layer: the application layer. So far, they created quite a few application layer protocols, like HTTP, HTTPS, protocols for e-mail (SMTP, POP3 and IMAP) and protocols for file-transfer (FTP). Still, there are application layer protocols on its way, like Instant Messaging and Presence protocols and a Scheduling protocol, but there is a tendency to let other standardisation bodies make these kind of protocols.

The IETF is formally described in a document called “Defining the IETF” [RFC 3233] (See section 3.2 for information about what an RFC is), but there are other documents which will explain the details in a much more readable fashion, in particular “The Internet Standards Process -- Revision 3” [RFC 2026] and “The Tao of IETF - A Novice’s Guide to the Internet Engineering Task Force” [RFC 3160]. I will not repeat all that information here.

It is interesting to note how the IETF would describe itself. To quote The Tao of IETF [RFC 3160]: “The Internet Engineering Task Force is a loosely self-organized group of people who contribute to the engineering and evolution of Internet technologies. It is the principal body engaged in the development of new Internet standard specifications. The IETF is unusual in that it exists as a collection of happenings, but is not a corporation and has no board of directors, no members, and no dues.”

And in case you are wondering what this Internet really is, “The Internet Standards Process” [RFC 2026] gives you the answer: “The Internet, a loosely-organized international collaboration of autonomous, interconnected networks, supports host-to-host communication through voluntary adherence to open protocols and procedures defined by Internet Standards. There are also many isolated interconnected networks, which are not connected to the global Internet but use the Internet Standards.”

Note the stress on the word “loose” in these documents. The Internet community, unlike organisations like IEEE and ITU, does seem to regards standards as necessary evil. Standardising is making trade-offs, getting people to accept your ideas and implementing it. That’s very much like politics. Not something most of the IETF attendees like, but something that must be done. This attitude can be explained by looking at the history of the Internet. See for example the “All About The Internet: History of the Internet” webpage at the ISOC website [ISOC], Tanenbaum’s...

3.1. The organisation of the IETF
The introduction on the website of the IETF tells us that “The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. It is open to any interested individual.”

“The actual technical work of the IETF is done in its working groups, which are organized by topic into several areas (e.g., routing, transport, security, etc.). Much of the work is handled via mailing lists. The IETF holds meetings three times per year.”

The details of the structural organisation can be found in “IETF Working Group Guidelines and Procedures” [RFC 2418], and “The Organizations Involved in the IETF Standards Process” [RFC 2028].

Working groups are formed by a number of volunteers and have no strict membership. The closest thing to a membership is being on the mailing list of the working group. They are short-lived, and are started with a very clear set of milestones in mind, which are described in a document called their Workgroup Charter. As soon as all milestones are delivered, the working group will be resolved. Despite of this guideline, there are a few working groups who still exist for a longer period of time, by just adding more and more work items to their charter. Usually, this doesn’t help to speed up acceptance of a new idea or a new protocol: as long as the workgroup still exist, their work is seen as a “Work in Progress”.

For management reasons, each working group (there are over 50) has one or two chairs and falls into an area. The current areas are: Applications, Internet, Operations and Management, Routing, Security and Transport. Partly, the areas are divided on basis of the OSI layer model, as described in section 1.1. Each Area has two directors (the ADs or Area Directors. All directors together form the IESG, the Internet Engineering Steering Group, who reviews drafts just before they are either published as RFC or handed back if they need more work.

3.2. Network Protocols in the IETF
The main task of the IETF is to publish documents, which are called RFC’s. The name RFC comes from “Request for Comments”, but that absolutely doesn’t cover the documents, since they don’t really inquire for feedback. They are now just referred to as RFC. An RFC can describe a protocol, but it can also be an informational document, which for example, describes a glossary or a model.

Before an RFC is published, it will go through some revisions. The last revision is available on a website and is called an Internet-Draft, or I-D for short. These drafts are temporary: either they should be published as RFC or they will expire after six months and deleted from the site.

When an RFC is published it belongs to a certain category: Proposed standards, Draft standards, Internet standards (also called Full standards), Experimental, Informational, Historic or Best Current Practice. For an exhaustive list of the different categories, see The Internet Standards Process [RFC 2026].

44 Omitting the General Area and Sub-IP Area. The first only contains the NomCom, the Nominating and Recall Committee. The Sub-IP Area is explicitly a temporary Area.
Standards are reviewed before they are published. Once a RFC is published, it is never replaced. If a new version needs to be published, it is published using a new RFC number. Currently there are over 3500 RFC’s. Unlike formal organisations like the IEEE and ITU, there is no voting on standards in the IETF. There is an advantage: it is not possible for a company to send lots of employees to a voting just to push a certain protocol. Instead, drafts are carefully reviewed by various people: working group chairs and IESG members. Only if the IESG approves it, a document is sent to the RFC Editor, who assigns an RFC number and publishes the document. The disadvantage of this lack of voting by large companies is that it is very well possible that a document would make it to a Standard, but is actually never used or implemented. To prevent this, a standard comes in three flavours, starting at Proposed Standard (no known technical omissions), moving to Draft Standard (at least two independent and interoperable implementations available) to finally a real Internet Standard (significant implementations and successful operational experience). Not all standards reach this stage.

All of this is in the spirit of the IETF, which has once been phrased as “rough consensus and running code”. Clearly, the aim of the IETF is not to debate too long about issues, but just to cut the Gordian knot, take a decision and implement it. With the growth of the IETF and the financial investment of the companies pushing certain technology, this is often the way it is supposed to be. In a typical IETF meeting, when there is a polling for which of two options to choose in a protocol, only 10 or 20 out of 100 people will raise their hands: all others don’t care, as long as the decision is made.
4. **What is the IRTF?**

The IRTF is a volunteer-based research organisation, working on Internet related research.

In July 1989, the IETF split in the IETF and the IRTF, the Internet Research Task Force. Where the IETF sets short-term design goals for creating Internet Standards, the IRTF sets the long-term goals to study research problems in the Internet, like architectures. To do this, the IRTF has chartered small Research Groups to work on a particular topic, without setting strict milestones, as the working groups in the IETF do.

Detailed information about the organisation of the IRTF research groups can be found in “IRTF Research Group Guidelines and Procedures” [RFC 2014]. Currently the IRTF has about ten research groups, with sometimes intriguing names like AAAarch (Authentication Authorisation Accounting Architecture), Group Security, Internet Digital Rights Management and Interplanetary Internet.

### Example: Interplanetary Internet

For example, Interplanetary Internet [IPNRG-charter] studies the problems we will face when we want to build a network using Internet technology in space. This is science, not fiction: satellites will communicate with the earth, and why not use well-known technology, like internet protocols. One of the tougher problems is how to deal with time-outs. On the internet, each (TCP) packet that is sent generates an acknowledgement, which is sent back to the original sender. If no acknowledgement is received after a short period of time, the sender will assume the packet was somehow lost and will try to resend it. When dealing with planetary internet, these round trip times will become unmanageable long, so other techniques must be used. Not to mention problems like severe asymmetry in the transmission capacity of bi-directional channels, severe variation in interference due to solar storms and predictable loss of connectivity due to celestial motions of the sender and recipient (behind the moon).

In chapter 5, the AAAarch research group will be discussed in detail.
5. **AAAarch!**

AAAarch is an IRTF research group, researching generic Authentication Authorisation and Accounting (AAA) architectures.

AAA is short for Authentication, Authorisation and Accounting: respectively telling who you are (identification); what you are allowed to do; and finally keeping track of what you did. Many internet protocols support some sort of authentication and authorisation: usually, after a connection is established, the first thing you do is send your username and password, and the service checks if you are known. If you are not, the connection is simply cut off. If you are known, you are granted to use the service, like Internet access (dialup or otherwise), access to files of a server (FTP or other file sharing protocols), if you are allowed to view a certain web page (HTTP Authentication), or even if you are allowed to use a high bandwidth connection.

AAAarch [AAAarch-charter] is an acronym for the Authentication, Authorisation and Accounting ARCHitecture research group of the IRTF. That’s quite a mouthful.\(^45\)

The AAAarch research group was started in 1999 as a spin-off from the AAA Working group in the IETF. It appeared that some authentication methods where unclear and in order to reduce the workload on the engineering group, the AAAarch research group was formed in the IRTF. The aim of the AAAarch was to research generic authentication concepts, which can later by used by the AAA working group. For example, AAArch research group has identified three distinct authorisation models, as is described in chapter 7. Cees de Laat and John Vollbrecht became and still are chairs of the research group.

Most of the protocols used today only focuses on a very specific usage, and even for this specific usage, the possibilities are very limited. There is hardly any support for roaming or distributed services.\(^46\)

**Roaming** is the ability for a user to access a service from a location other than his or her home location. For example, if you have a mobile telephone and travel abroad, it is sometimes possible to use the mobile phone network from a company in the country you are travelling to. If you access this foreign network (in this case a GSM network, but it could be the internet as well), it checks with your home organisation if you’re a valid user, and if that’s the case, they will allow you to use the network. The foreign provider should have an agreement with your home organisation in order for this to work. Such an agreement is called a roaming agreement.

**Distributed services** is the combination of services from several providers for one person. Let’s use for example a user who temporarily wants a 1 Gbit/s connection

---

45 Evil tongues more than once suggested that this name was specifically chosen because no-one seemed to know how to pronounce it.

46 For example, the AAA working group in the IETF limits itself to Mobile IP and network access.
from Berlin to Amsterdam. However, that requires operation of multiple organisations: one organisation for the connection from Berlin to Frankfurt and another for the connection from Frankfurt to Amsterdam. Instead of asking both organisations for a 1 Gbit/s connection, it would be much easier just to ask one of them a request: “Give me a 1 Gbit/s connection from Berlin to Frankfurt, but only if you can also arrange a 1 Gbit/s connection from Frankfurt to Amsterdam for me”.

Unfortunately, in practice things are not that simple. Sending data along a connection with a non-guaranteed bandwidth is, thanks to the internet, very easy these days. However, when you require a pre-configured connection with a guaranteed bandwidth, for example because a physics experiment, like two interconnected radio-telescopes part of an very large array (VLA) requires so, the situation is very different. For example, for a pre-configured connection from Amsterdam to Jülich\textsuperscript{47}, you will need as much as seven different organisations (which each may use different network protocols) to work together [De Laat (2001)]. These organisations, or “kingdoms” include parties like the University Campus network maintainers and the department network maintainers. Getting all these organisations to work together may be a bureaucratic nightmare. Anyone who ever attempted to make something like this work would probably wholeheartedly agree that there is a definite need for a generic AAA-protocol, which would allow people to make such a pre-configure connection possible in seconds instead of months of frustration.

\textsuperscript{47}Jülich is a city in Germany, and is in fact closer to Amsterdam then Frankfurt.
6. Generic Authentication Concepts

Authentication is identifying, telling who you are. Authorisation is granting or denying someone permission to a service, telling what you may or may not do. Accounting is keeping track of who actually uses which services.

For most authentications, three organisational entities can be recognised: the User (you), the provider of the service (for example a store) and the users “Home organisation” (for example, a bank). The service provider trusts the Home organisation, and the Home organisation trusts you. These pre-established trust relations must exist for an authorisation.

A simplified situation is that the Home organisation and the service provider are the same.

Even in daily life, you will encounter many examples of authentication. To make you aware of the generic concepts and terms used in authentication, this chapter will use examples to introduce concepts, and will then formally declare a list of terms.

Alternatively, you can read RFC 2903 through 2906 [RFC 2903; RFC2904; RFC 2905; and RFC 2906], which describe the general model of AAA-servers. In particular “AAA Authorization Framework” [RFC 2904] defines the basic authorisation models (also described in chapter 7 of this document) and “AAA Authorization Application Examples” [RFC 2905] defines six network-related application examples in more detail.

6.1. Examples

To understand these examples, you just have to know these (slightly simplified) definitions of authentication and authorisation:

Authentication, or Identification, is telling who you are. A user who wants some sort of service first tells who he is –he identifies or authenticates himself.

Authorisation is granting permission to a service. As soon as the identity of the requester has been established, the provider of a service, will then grant (or deny) the requester permission to use the service.

Buying a pizza (organisational entities and trust relations)

Probably everyone who is reading this document has ever visited a shop, bought something and paid for electronically. For example, you paid for a pair of shoes using a pinpas (pinpas is a Dutch bank card, when paying, the device checks by phone line if your account has enough credit), or you browsed to www.pizza.it and ordered a cheese pizza

Obviously, there is some authentication going on: pizza.it likes to know if you are really going to pay before they sent the pizza on its way. However, that’s strange: neither pizza.it nor the shop owner knows who you are nor asks for that information. Technically: there is no pre-established trust-relation between you and the shop. So how could he authorise you? Well, because all that is checked is not your identity, but just the fact that you have an account with a known bank and the bank is asked if your balance is sufficient for a pizza or your shopping needs. Because the shop trusts the bank, and the bank trusts that you will pay your debts, it is possible to commit the transaction. Technically: there is a pre-established trust-relation between the shop and the bank and between you and the bank (see figure II-4).

48 Preferably large with extra cheese if you are like me.
Three comments here: the only information you divulge to the shop is the name of your bank, the account number and the pincode. However, a protocol can be made where the bank first sends a random number to the shop (who forwards it to you), and that you encrypt your account number and pincode. So all the shop can decipher is the bank name and the final response from the bank if there is enough money on your account or not.

Secondly: the three organisational entities here are: the client (you); the service provider (the shop) and your home organisation (the bank).

Thirdly, if you are looking at the trust relations (see figure II-4), you will see that the bank can be seen as some kind of intermediate. This model can be extended to include another intermediate. The advantage of an intermediate is that not every shop needs to have a contact with every bank. For example, in the Netherlands, an organisation called Interpay acts as an intermediate between the shops and the banks.

If a trust relation consists of a contract, describing the kind of service that one of the parties offers the other, this agreement is called a Service Level Agreement (SLA).

**Tickets (authorisation model)**

Tickets are another example of authorisation. If you plan to visit a concert, you will go to a ticket vendor and pay for the tickets. Later you use these tickets to get authorised entry to the concert.

In this example, the person who checks your ticket at the entrance does not contact the issuer of the ticket, but only uses the ticket itself to verify its validity. That means that the ticket needs to be “signed” by the issuer: only owners of tickets that where created by the issuer are allowed to pass. For example, the ticket could contain a signature of the issuer or it could be printed on special paper, only available to the issuer. A signature might be a digital signature, using public key encryption. Protecting tickets against duplication is by no means trivial, and I will not go into details here.

In the previous example (shopping using a credit-card), authorised transmissions between people unknown to each other are possible, with the aid of a common trusted authority, like a bank. In that case, the the one who performed the authorisation had to make a check with the authority in real time. This is not the case if tickets are used. These two situations are two different **authorisation models**. We will later see that there are three distinct models.

There is only one Internet Protocol which uses tickets: Kerberos [RFC 1510], developed at MIT. In contrast, in the real world, tickets are very common, including your drivers license and banknotes. The trusted authority for money is the central bank: anyone who uses money to pay, trusts that the central bank guarantees the value of the money, and will look for characteristics of the banknotes, like a watermark, to check that the banknote was really issued by the central bank.
Dial-in

For most home-users of internet, there are two situations when they need to authenticate themselves. The first is when they dial in to their ISP, the Internet Service Provider, where the modem software requires a password. The second is when they check their e-mail, which require them to enter a password in the e-mail program. This example will focus on this first example, a dial-in connection, even though this authorisation model can be applied to many similar examples.

When a user wants access to the internet, he or she will use a modem to dial a special phone number of the ISP. On the other side of the phone line is a special device, called the Network Access Server (or NAS for short), which answers the phone and lets traffic through from the user onto the network where the device is attached to (usually the Internet). However, to prevent anyone from calling in, the device will first ask who is on the other side of the phone line, since it will only allow customers who have a service agreement with the ISP. The procedure is illustrated in figure II-5. First, the user connects to the NAS (the Network Access Server), sending his username and password (Connection ① in figure II-5). The NAS can check the username and password in a table and respond to the user. However, most ISP’s have more then one dial-in connection, and have multiple Network Access Servers. Maintaining an up-to-date list of all usernames and passwords in each device would be very inefficient. Therefore, ISP’s have a central authentication server that stores all this information. When a user dials in to a NAS, the NAS sends this information to the authentication server, asking if it is correct (②). The Authentication Server responds (③) with a positive or negative result. If it is positive, the NAS grants the user access to the network. Regardless of the result, the NAS notifies the user of the result (④), so that the user’s computer can either present an error message, or the user knows he can start using the network connection to send and receive data (⑤).

49 In practice, the user does not send the password in clear text. Usually the Network Access Server will sent a challenge. The exact method depends on the protocols and authentication mechanism. For modem connections PPP is used, with CHAP or EAP as the authentication mechanism. See chapter 5 of part I for more information.
Sometimes a user wants to gain access to a network which does not belong to his or her home organisation. For example, an employee from the University of Michigan visits the University of Chicago and wants to log in there. Fortunately, the two universities signed an agreement to permit that each employee can log in at either location. This is called roaming, the ability to access a service from another location then your home location, but using your home username and password. Of course it is possible that the authentication servers at both locations know the employees of both universities. However, that would again create management problems: how to keep all data synchronised? The solution is that the user does not only send his or her username, but the name of his home organisation as well. For example if John Doe wants to log in at Chicago University, he will use john@umich.edu as his username. The NAS just checks the username and password with the local Authentication Server (Connection ① in figure II-6). The Authentication Server will recognise that it is a non-local username, and see in its configuration that the domain the user belongs to is an organisation where it has a valid Service Agreement with. It will then forward the authentication request on to the Authentication Server of the User’s Home Organisation (umich.edu) (Connection ②). This Authentication Server responds (③), and the response is forwarded to the Network Access Server (④), which does the same thing as in the non-roaming situation.

**Bandwidth broker**

The last example deals with bandwidth brokers and agents. In chapter 5, which introduced the AAAarch research group, distributed services where mentioned. Let’s review the example that you want to get two services together, a 1 Gbit/s line from Berlin to Frankfurt from Deutsche Telecom, and a 1 Gbit/s line from Frankfurt to Amsterdam from the Dutch KPN. In such a case, you might first want to set up the first connection, and then the second. However, it would be more efficient to only ask one organisation: either one of these two or a third organisation. Such an organisation would be a bandwidth broker. A bandwidth broker does not have to own lines itself, but has agreements with multiple owners to make bandwidth reservations in real time. A possible model is shown in Figure II-7. The user makes a request to (the authentication server of) a bandwidth broker. The bandwidth broker evaluates the request and delegates the request to (the Authentication servers of) the organisations it has an agreement with. Those servers will evaluate the request against their policy and, if approved, will configure their routers to make the
bandwidth reservation. As soon as the Authentication Server received all confirmations, it will send the final approval to the user.

If one or more organisations disapprove the request from the bandwidth broker, it is the responsibility of the bandwidth broker to cancel the other reservations it made. This can be a problem if a malfunctioning bandwidth broker constantly requests a bandwidth reservation and cancels the request a second later. In the mean time, no-one is able to use the bandwidth, and reservation requests from other bandwidth brokers are denied as well. In the end, no-one can use any bandwidth, effectively making this a Denial of Service attack (DOS-attack). A possible solution for the bandwidth broker is to first ask if there is enough bandwidth, but postpone making the actual reservation until all organisations have confirmed that enough bandwidth is available. Of course, a possible problem is that in the mean time someone else might have taken the bandwidth that was available just a second earlier. A possible solution here is to allow timed reservations in the future. The exact solution will depend on the policy implementation of the bandwidth broker.

The bandwidth broker is an example of an agent: an organisation performing reservations on your behalf. An agent may just as well make reservations for storage capacity or computer power on a supercomputer. The model used to describe the authorisation and trust relations is therefore called the agent model (see chapter 7 for more information). Unfortunately, this bandwidth broker example is still a hypothetical one. Currently, making a reservation for bandwidth will require weeks or months instead of a couple of seconds.

6.2. Credentials and trust-relations

To understand authentication, let’s elaborate on the concept of identity attributes and credentials.

Binding attributes to a session

In the dial-in example in the previous section, an authentication server performed an authorisation to allow or deny access to the network. However, it was not clear who or what was authorised: the user or the session. As soon as anyone contacts the Network Access Server (NAS), the NAS creates a session. As soon as the user sends his or her username and password, the NAS binds these information pieces to the
session. The username and password are now attributes of the session. When the NAS wants the authentication server to verify the username and password against the registered identity, it only sends a part of all the session attributes. For example, it may not reveal information about the telephone number to the authentication server.

**Authorities and trust relations**

On the other hand, it is also possible to create new attributes. In the example of tickets, you may contact an authentication server in order to obtain a certificate that allows access to a certain service. In this case, you will contact the authentication server, presenting your username and possibly the name of your organisation (identity attributes), and most likely a password as well. The authentication server will verify these attributes (authenticate you), check if you are allowed to use this particular service at this time (authorise) and return a certificate to you, possibly logging the transaction in order to later bill you for it (accounting).

The certificate you obtained in this example is a **credential**, a **qualified identity attribute**, because it is signed by the authentication server, and this authentication server is willing to bind its **reputation** to it. You just created a new **trust-relation**: as soon as you present the certificate to someone else, you effectively say: “look, this organisation trusts me. If you trust that organisation, you should also trust me.” Any organisation that issues credentials is called an **authority**. You may or may not trust an authority.

The password you sent earlier is also a credential, since it was issued by the organisation you belong to. By presenting it, you may obtain certain rights. So you got the certificate because the authentication server did trust the authority that issued your password (maybe because he did it himself). This sequence of trusts is called a **chain of trust-relations**. It is not necessary to completely grant any access to anyone you trust. You can for example grant certain rights to anyone you trust, and some other (fewer) rights to anyone who displays a certificate signed by anyone you trust.

**Examples of trust relations: Web certificates and PGP**

One of the main applications of chains of trust-relations is the security mechanism built in web browsers. This mechanism is used when you visit a secure website (a HTTPS website; most browsers display a lock icon when visiting a secure site). Explorer and Netscape have a few certificates built in of known trustworthy organisations. Website builders can buy a certificate from these organisations and send them to a user when this user is requesting their secure website. The browsers checks if this certificate was signed by the secret key of a built-in authority certificate. If this is true, this proves that the creator of the website is really who he claims it is, and the requested website is shown. Otherwise, an error message is presented to the user.

Some people claim that for browsers, these trust relations are pretty useless, and wonder why just these organisations are to be trusted. In fact, one of the opponents of this mechanism was able to get himself a certificate in the name of “Microsoft, Inc.”, even though he was by no means a Microsoft employee, by just sending the right amount of money to VeriSign, one of these “trusted” organisations.

PGP (Pretty Good Privacy) is another security product, mainly used to encrypt e-mail, which also uses trust-relations, but without a central authority. With PGP, you can sign public keys of your friends, and your friends can sign your public key. Now if you decide to trust a friend of you, you may also decide to trust the people your friended trusts. In the end, you can trust that the sender of a PGP-signed
e-mail is who he says who he is, if the chain of trust between you and him is very short. So for PGP, you can indeed make yourself more trustworthy by having your public key signed by people who are trusted by many other people. These people may decide to sign your key so if they’re convinced you are who you say you are, probably because you showed them your passport or driver’s license. The IETF usually has a late-night session where “famous” IETF members can sign others people’s public keys. Needless to say that the clear disadvantage of PGP is that it requires quite a good knowledge of security and trust relations to use it effectively. It’s not enough to just create your own public key. You have to go out and make it public, and get it signed as well. Not to mention that you should check possible chains of trust before you accept an e-mail.

The debate about security approaches has been going on for over a decade, and the number of complaints about central authorities who seem only eager to gain more money or more control is countless. It is not expected that this debate will end soon. See Tanenbaum, chapter 7.1 for a more extensive introduction to security mechanisms and cryptography, or any one of the many, many books and websites published on this topic.

6.3. Glossary

In the previous sections, a few examples where discussed, and a few conceptual ideas regarding authentication where introduced. In this section, these terms will be described in a formal way. Most of these terms are based on Private Communications with Leon Gommans [Gommans (2001)].

**Identity**: Any differentiating information that allows further selection of members belonging to a particular group of entities. This may be a person, or just “any employee of the physics department of Utrecht University”.

**Identity attributes**: attributes that describe differentiating features of an entity. Identity attributes may or may not be apparent, may or may not be secret, and may or many not be revealed when requested. For example: a username, a home organisation, a ticket, a password or a certificate.

**Credentials**: Qualified identity attributes, thus identity attributes that are acquired from and signed by an authority. By using security methods, the attributes are bound to the authoritative organisation’s reputation. Credentials may have one or more (groupable) elements that can be exposed, hidden or abstracted depending on the context in which they are used. For example: a certificate, ticket or password.

**Registered Identity**: Any set of identity attributes that is defined by some application for registration and future usage. For example: a list of usernames and passwords in an authentication server.

**Identification** is the act of presenting identity attributes by one entity to another trusted entity.

**Authentication** is the process of validating the authenticity of credentials, usually by comparing credentials to a registered identity.

**Anonymous authentication**: Authentication based on credentials that have been abstracted in such a way that it is impossible for any third party to link the credentials to the original registered identity. The credentials can only be linked by the registration authority or by an explicit desire of the entity.

**Authorisation model**: push, pull or agent model as described by the AAIAarch research group (see chapter 7) plus implicit trust-relations.
**Authorisation**: the process of deciding to grant or deny a User’s request for services or resources. For example: the outcome of rule based policy engine; “yes, you may use the network now”.

**Accounting**: Generation of relevant information regarding the authorisation decision and the associated authorised session (if any) that represents the ongoing consumption of those services or resources.

**Service Level Agreement**: A contract between two organisations specifying, in measurable terms, what service the network service provider will offer to the other organisation. For example, a roaming agreement is a form of a service level agreement.

**Policy**: A formal set of rules that defines how the network resources are to be allocated among clients. Clients can be individual users, groups of users or applications. Resources can be allocated based on time of day, client authorisation priorities, availability of resources, and other factors. Policies and policy rules are created by network managers and stored in a policy repository. During network operation, the policies are retrieved and used by network management software to make decisions. [Whatis?com]

**Policy Repository**: A database containing the available services and resources about which authorisation decisions can be made, and the policy rules to make them. [RFC 2903]]

**Policy engine**: A piece of code capable to evaluate logical formula’s and to parse access requests. A policy engine will evaluate an access request, and based on factors like current policy, available resources and currently reserved resources, will come to a logical conclusion. Note that we silently assume that the policy engine is a rule based policy engine.

**Policy Decision Point (PDP)**: The point where policy decisions are made. For example: an entity in an AAA server. The AAAarch research group defines in chapter 4 of AAA Authorization Framework [RFC 2904] where a PDP may reside within each authorisation model. That can either be at the user or at the AAA server. For the rest of this document, we will use the term PDP exclusively for an AAA server, similar to the use of this term in the RAP working group (an working group of the IETF). So unless specifically stated otherwise, AAA server and PDP are the same in this document.

**Policy Enforcement Point (PEP)**: The point where the policy decisions are actually enforced. For example: an entity in a Network Access Device, or an entity in a Service Equipment. In this document, the term PEP refers to the access device which enforces a policy presented by a PDP, unless specifically stated otherwise.

**Authentication Server**: A part of an AAA server, involving the authentication of users. Most of this document will use the more generic term AAA server or PDP, even though our main concern is with the authentication part of the AAA server, and not with the account part and policy based engine used for authorisation.

AAA Authorization Application Examples [RFC 2905] defines a few more terms, like **Distributed Service, Trust Relation** (dynamic and static), **roaming** and **security association**.

COPS-related terms, like PRID, PRC and PRI are defined in the main thesis (part I).
Finally, security related definitions like non-repuditation are defined in any good textbook or website related to security.
7. Authorisation models

There are three basic authorisation models, each model defining the interaction between the User, the Service Provider and the Home organisation (the authenticator and its AAA server).

These models are the pull model (the service pulls information about the user from the AAA server), the push model (the user pushes some kind of ticket onto the service) and the agent model (the AAA server acts as an agent who handles the service request on behalf of the user).

The AAAarch research group identified three distinct authorisation models. This chapter will briefly review each three of them. Also it will quickly sketch which interfaces and protocols can be identified when looking at the inner working of the PDP. Note that this document uses a different terminology than RFC 2903 to 2906.

Alternatively, you can read RFC 2903 through 2906, which describes the general model of AAA-servers. In particular AAA Authorization Framework [RFC 2904] defines the basic authorisation models and Generic AAA Architecture [RFC 2903] partly defines the inner-working of a generic AAA-server. However, an internet-draft describing the inner working in more detail is still a work in progress.

7.1. Pull model

![Figure II-8. The AAA pull model.](image)

The most commonly used authorisation model in protocols is the pull model (Figure II-8), which is typically used for mobile IP and dial-in, and for situation where a user wants to get access to network resources. A user requests access from the PEP (the service equipment), which forwards it to the PDP (the service provider’s AAA server). The PDP evaluates the request and returns a response to the PEP, which sets up the services and tells the User it is ready. Note that the response may either be ‘approve’ or ‘deny’ (the diagram shows approve). A ‘deny’ may be accompanied by further information, like ‘not authenticated’ (probably a typo in the password), ‘User already consumed this month’s quota’ or ‘Not allowed between 9:00 and 17:00’.

Section 3.1.2 of AAA Authorization Framework [RFC 2904] describes the pull model.
7.2. Push model

Figure II-9. The AAA push model.

Figure II-9 describes the push model. Here, the user first contacts the PDP to get a ticket or certificate, which states that the User is allowed to have access to a particular service. The PDP verifies the User’s identity and issues a ticket (certificate), possibly including time restrictions. When the user requests the PEP to use a service, he includes the ticket in the request. The PEP examines the ticket and sends an OK to the User. In the Push model, the communication between the PEP and the PDP is relayed through the User. Section 3.1.3 of AAA Authorization Framework [RFC 2904] describes the Push model. The push model might for example be used for a network printer. It is also used as the authorisation model in Kerberos [RFC 1510], and is used often in real life (for example concert tickets or licences).

7.3. Agent model

Finally, in the agent model (Figure II-10), the PDP acts as an agent between the User and the PEP. In this model, the Users wants to use a service offered by the PEP, but sends a request to the PEP via the PDP (the Server provider’s AAA server), which will verify the request using the policy for the services and this user. If approved, it will first send a configure request to the PEP, which might respond if all that is requested is set up (not shown). The PDP will now reply to the user that the requested services is set up. A typical application includes a bandwidth broker. Section 3.1.1. of AAA Authorization Framework [RFC 2904] describes the Agent model.

Figure II-10. The AAA Agent model.
7.4. Internals of an AAA-Server

So far, we regarded the PDP as an autonomous device, which took the decisions, sometimes on behalf of the PEP, sometimes directly on the user’s request, however, always according to a policy of the Service Provider. In this section, we will call the PDP an AAA-server, to stress that decision making is just a part of the job of the AAA-server. Another part is accounting (logging). This section is largely based on Generic AAA Architecture [RFC 2903], private discussion with Cees de Laat [De Laat (2002)] and on a Work in Progress, Structure of a Generic AAA Server [draft-irtf-AAAarch-generic-struct], which now seems stalled and doesn’t seem to be published as informational RFC.

One of the design goals of the AAAarch research group is to sketch how a generic AAA-server would look like. A user should be able to request a set of services from an AAA-server, which might be in different domains or concerns completely different type of services. For example, a user may want a bandwidth provided by one organisation and access to a storage device provided by another organisation, whereas the request is managed by a third organisation. So generic AAA-servers should be able to communicate to each other using a protocol which is very generic in structure, and is able to provide multiple types of service requests, including still to be developed requests types, that are not known yet. Such a protocol is a generic AAA-protocol, and does not exist yet. Figure II-11 shows the internal structure of an AAA-server, with protocol 1 the generic AAA-protocol.

The heart of each AAA-server would be a rule-based policy engine, which must be able to parse requests from a user or from other AAA-servers. When a request arrives, the engine would inspect the contents, determine what authorisation is requested, retrieve policy rules from the policy repository, perform various local functions, query the event log or forward (parts of) the request to other AAA-servers for evaluation. The interaction with other AAA-servers, and possibly also the user (in the agent authorisation model) is done using protocol 1 in Figure II-11, the generic AAA-protocol. The querying of the policy repository and the event log uses interface or protocol number 3. If part of the request is for some local service, the generic AAA-server does not need to have knowledge about the precise content of the message, so it will let this part of the request be evaluated by an Application Specific Module (an ASM). All the AAA-server needs to know is which module it should pass this part to. So there must be some global structure to recognise the different types of request: request for bandwidth, for storage, for computing power, etc. The communication between the generic rules based policy engine and the application specific module is done using the interface of protocol 2 in the figure. Most likely, this will not be a protocol, but just an interface or API; a set of documented functions in the software.
Figure II-11. Internal structure of an AAA-server, with five types of different interfaces or protocols identified: the AAA-protocol (1), an interface to an application specific module (2), an interface or protocol to read and write policies and events (3), optionally support for legacy ‘AAA-protocols’ (4), and finally an interface between the application specific module and the Service (5).

There will be an Application Specific Module (ASM) for each type of usage request. So network requests would use a different ASM than a request for storage capacity. The ASM would be able to communicate with the outer world by using a protocol (number 5 in the figure) customised for the functionality of the specific ASM. This protocol may either request more information from another device and probably will be able to handle configuration requests from the ASM for the Service.

Finally, the ASM may or may not be able to communicate with other ASM’s using a legacy AAA-protocol (4 in the figure), i.e. any AAA-kind of protocol which is not generic in structure, but can only be used for one specific function, provided by the ASM. In an ideal situation, there would be no need for such a legacy AAA-protocol. However, if a generic AAA-server will be built, it is expected that it will be able to communicate using existing protocols in the beginning. Protocols like RADIUS, DIAMETER and COPS are examples of these kinds of legacy AAA-protocols.

It is my belief that there will be no need for these legacy AAA-protocols, and that these protocols will only be used for communication between the service and the ASM, not between different ASM’s. It is however possible that one of these protocols, either DIAMETER or RADIUS will later be extended to serve as the basis for a generic AAA-protocol.

Right now, the biggest issue is to make such a very generic protocol. Most people in the IETF seem to focus on problems they encounter in their own working groups, and creating a generic AAA-protocol, which could even be used to order a pizza with an addition parameter (“only allow my request if you are able to deliver it in 30 minutes”) still seems too futuristic. We will see this in our final conclusions.
8. What is the RAP group?

The RAP group is an IETF working group who defines protocols to allocate network resources, like bandwidth. The RAP group has created the COPS protocol.

One of the many working groups within the IETF is the RAP group. Just like all other groups, it was formed with the intention of being a short-living working group with a charter with a timeline and a few deliverable goals.

RAP means Resource Allocation Protocol, even though there is no protocol with this name. The resource referred to, are network properties such as bandwidth: the goal of the RAP group is to create a protocol which for example can assign a certain bandwidth, like 10 Gbit/s to a certain user or organisation.

8.1. Operation and Management protocols

The RAP group resides in the Operations and Management Area, or Ops for short. The internet consist of several interconnected devices, like routers (devices which operate at the network layer) and switches (devices which operate at the datalink layer). For example, a router will take care that a packet originating from one computer is routed along the correct route to the destination computer, by looking at the destination address. So these devices must be configured to know some of it’s neighbouring routers. Similar, other devices like switches, must be configured in some way. Next to configuring a device, most operators want to monitor them as well, for example to monitor for problems, count data traffic to bill their customers or just to create an impressive data traffic usage graph.

So routers and switches usually not only speak one or more protocols on their layer, they also speak some management protocol on a higher layer. These kind of protocols, used to configure or monitor devices are sometimes referred to as operation and management protocols. There are a more of these protocols, like non-standard protocols made by Cisco, but the protocol with the widest deployment is SNMP.

It should be noted that SNMP primary use at the moment is to manual configuration. Sometimes a device (in particular dial-in devices operated by ISP’s) is configured to allow users who authenticate themselves and are known by some authentication server to sent and receive traffic, and to deny all other traffic. Currently, the protocol most widely used for the communication which such a dial-in device (called a NAS or NAD: Network Access Server or Network Access Device) and the authentication server is done using the RADIUS protocol ("Remote Authentication Dial In User Service). There is a working group creating a successor of RADIUS, called DIAMETER. A large part of this thesis is devoted to the combination of configuration and authentication of a user, even though other protocols then SNMP and RADIUS are concerned.

The RAP concerns itself with a management and operating protocol, other then SNMP: the COPS protocol. The charter of the RAP working group [RAP-charter] can be found at the IETF website.

---

50 The RAP group was previously named RSVP Admission Policy Working Group, but this name was quickly abandoned. Presumably to able to include more and more work items in their charter, some completely unrelated to RSVP.

51 Although the terms routers and switches are usually defined this way, some switches also have some network layer functionality and are called IP-switches, or Layer-3 switches. In ATM networks, the term router is not used, and switch is used instead.
9. Quality of Service

Quality of Service (QoS) is the idea that network characteristics can be measured, improved and, to some extent, guaranteed in advance.\(^{52}\)

Quality of Service, or QoS for short, is the term used to describe parameters of a service, usually the service the transport layer is offering to the application layer. For example, it can describe the transmission rate (for example 2 Gbit/s), the transmission time (30 ms), the deviation in the transmission time, the error rate (0.1% of the packets is corrupted), and ability to rebuild a lost connection.

QoS can be seen as the property of a service, and it can also be used to negotiate about the options of the service. For example, if you are using the network for telephony, you probably want a fast transmission time to avoid high latency (a noticeable delay between the time that you say something and the time that the person on the other side responds). In addition, you want a very low deviation in the transmission time to avoid choppy sound or images if you also sent a video stream. On the other hand, you probably don’t care if one or two packets are dropped if there is congestion: the human ear is capable of interpolating in case there is a very short silence. It is not needed to retransmit lost or corrupted packets. Another example: if you are sending a large file to someone, you are not so interested in transmission times: if the file arrives a few seconds later, that will fine. However, you are very concerned about error rates: you want absolute certainty that the file arrives just as you have sent it, and want the protocol to correct for errors that occurs along the way. In addition, it must be able to cope with aborted connections. For example, if 500 MB of your 600 MB file is copied, it is more then convenient if you don’t have to start all over if the connection is aborted. There are multiple techniques to recover from this.

Quality of Service is usually only relevant for the end user, or to be exact, for the application layer. Therefore, QoS usually refers to the quality of the service the transport layer offers. However, QoS can also apply to services other layers offer. For example, the Quality of Service the physical layer offers will most likely be fixed, depending on the characteristics of the medium: you can’t increase the bandwidth of a 10 Gbit/s line. However, the datalink layer on top of it can split the bandwidth for a shared 10 Gbit/s line in five chunks of 2 Gbit/s each, for five distinct clients; or, it could split it in four chunks of 1 Gbit/s and one chuck of 6 Gbit/s.

Likewise, the Quality of Service the network layer offers is usually pretty fixed: it is either connection-oriented or connection-less, and the error reduction is not part of the network layer in the OSI model. However, one of the most important functions of the transport layer is to improve the QoS of the network layer, for example by re-sending lost or corrupted packets (each packets sends and acknowledge that is has been received. If the sender does not get the acknowledgement, the packet is resent. TCP works approximately in this way). Another way to reshape traffic is by giving priority over one type of traffic to another type of traffic.

Latency and bandwidth

Two main properties of a network are latency and bandwidth: the latency (usually measured as the round trip time) is the time it takes a packet to reach it’s destination, while the bandwidth signifies the number of packets that can be sent in a certain time. These two properties are not the same. Compare it to transport vehicles: a van is fast, but can carry a limited amount of goods (low latency – which is good, and

---

\(^{52}\) Based on the definition by Whatis?com [whatis?com]
low bandwidth – which is bad). On the other hand, a ship is slow but can carry many goods (high latency and high bandwidth). You can easily add bandwidth by adding more devices and more connections. But you can’t change the latency. This is explained in [Cheshire (1996)] and [Tanenbaum (1996)]. It is possible to manage the bandwidth, but it is not possible to manage the latency.

### 9.1. The sense and non-sense of QoS

Priority routing means that you try to reshape the QoS of your traffic by giving certain type of traffic priority over other type of traffic. For example, a switch or router would give precedence to traffic from or to the customer with a more expensive account over traffic from or to other customers.

However, routers try to be fast, and tend to forward incoming traffic as fast as possible. Today, traffic typically resides for less than 1 millisecond (0.001 second) in a router or switch, while (due to the limited speed of light) the time for a packet from Amsterdam to reach New York will take at least 20 milliseconds. So the gain is only a few percent, likely even less, and the trade-off you have is a lot of added complexity, and above all: added calculation to check to whom certain traffic belongs. These calculations will make a device slower, and in the end, all traffic will be delayed, including the traffic of the customer with a priority account. In particular if there is little traffic on the line (no congestion), it would be fastest just to forward all traffic as fast as possible, without doing any of these checks. Only if there is congestion, and a device is forced to drop or queue packets, then QoS may make sense.

TCP has flow control based on congestion: if no confirmation of the arrival of a packet is received within a certain time, the TCP layer will assume that packets are dropped due to congestion, and will send packets at a lower rate. So, by dropping the correct packets, it is possible to give a certain connection a lower bandwidth, and thus the other connections a higher bandwidth. It may be arguable that it might be cheaper to just add more bandwidth, instead of adding complex QoS checking, which may increase the latency.

Another argument is that is generally a bad idea for any device on a backbone to drop packets: the reason is that a packet reaching a device on the backbone has already travelled through a lot of other devices, and dropping the packet later, would be a huge resource loss: all the previous devices spend time checking and forwarding the packet, and it would have been more efficient if the packet was dropped at the first device the packet encountered (called the edge-device). A better approach might be to first negotiate the route and check if there is enough bandwidth available along the way. One of the protocols which takes this approach is RSVP [RFC 2205].

Concluding, trying to improve QoS is almost impossible: you are stuck with the physical properties of the devices and the wires. It is possible to manage QoS resources, although only the bandwidth can really be managed. Trying to improve QoS by priority routing only makes sense if there is congestion and packets needs to be dropped, either in the current device or along the way. QoS makes most sense if it is done by an edge device, and this edge device only uses it to drop packets which do not fit into a negotiated profile (negotiated bandwidth). If it is deployed further along the route, either there is no gain (if no packets are dropped) or resource-inefficient (if packets are dropped). It may be sensible to re-route packets based on QoS parameters (low priority traffic over an other wire then the high priority traffic).

---

53 Note that we do not need to drop packets based on the user, but may also drop packets based on the type of service we requested, for example drop a telephony packet, but queue a packet used for a file transmission.
This conclusion (priority routing is only useful if packets are dropped) is counterintuitive, since the main idea was to get as much packets from a certain customer through a device as possible, not to drop (other) packets.
III. BACKGROUND ISSUES

This appendix aims to give some detailed information about topics that do not fit into the main text. It contains three chapters: the first explains how data is embedded in a COPS-PR messages, the second chapter will go into detail about future improvements of EAP and the third will sketch a usage scenario for the Access Bind PIB.
1. Embedding of data in COPS-PR messages

Data, in the form of one or more PRI’s (Provisioning Instances), each prepended by a PRID (PRI Identifier), can be embedded in COPS, using the COPS protocol and the COPS-PR extension to COPS. Each COPS message contains a header and one or more COPS objects. Two particular types of objects are a Named Clients Specific Information Data and Named Decision Data. COPS-PR defines the contents of these objects, by explaining that one of these objects can contain multiple COPS-PR objects, which may contain the actual PRID or PRI.

The data structures that send between the PEP and the PDP are send using COPS messages. COPS is a client-server protocol, described in RFC 2748 [RFC 2748], with the PEP the client and the PDP the server.

1.1. COPS message sequence

In the COPS model, the client requests something from the server with a COPS message with operation code “Request”. A request may be a request to authenticate a user or a request to be configured. The server will always respond with a COPS message with operation code “Decision”. In the COPS-PR model, this decision may contain data instances that provision the PEP. Finally, the PEP must respond to the decision with a COPS “Report State” message. This report state tells the PDP if the PEP was successful in carrying out the PDP’s decision. In the message sequence listed earlier in this chapter, these Report States are left out for readability.

If the PDP wishes to authenticate a user using EAP, before it sends a final response to an access request, it could respond with a COPS Decision message which includes an EAP request. Though it may seem strange that a request can be embedded in a decision, this is perfectly logical when you look at the message sequence on both the EAP and the COPS ‘layer’. When the User responds to the (forwarded) EAP message, the PEP should embed this EAP response in a COPS request message with the same client handle as the previous request.

After the PDP has gathered enough information about the User, it sends a COPS “Decision” message to the PEP. The PEP takes actions: it makes provisions the PDP optionally requested, notifies the PDP of the result (using a COPS “Report State” message) and informs the User of the authorisation result.

1.2. Embedding of instances in COPS messages

A typical embedding of PRI (the instances) in a COPS message is shown in Figure III-1.

The embedding is rather complex. The basic message structure is a COPS message. Each COPS message contains one header followed by one or more objects. The header contains the operation code, like “Request”, “Decision” or “Report State”. Each message contains one or more object, depending on the type of COPS message. Examples of these objects are the Client Handle, Error, (Named) Clients Specific Information (ClientSI) Data and (Named) Decision Data. These two named data objects are further defined in COPS-PR [RFC 2309].

This is described in the COPS protocol [RFC 2748]. COPS-PR [RFC 3084] describes a method to embed data structures in two of these objects, the Named ClientSI object and the Named Decision object. To be precise, COPS-PR describes objects that can be

---

54 Or a COPS Report State Message. The RFC is not 100% clear to me.
embedded in the COPS objects. So one or more COPS-PR objects are embedded in specific COPS objects, and all COPS objects are embedded in a COPS message.

![Figure III-1. Typical embedding of PRI (instances) in a COPS-PR message.](caption)

There are two primary COPS-PR objects, the Encoded PRID and the encoded PRI. The PRI (Provisioning Instance) is an instance of a class, thus the data itself. The PRI and PRID can be encoded using different methods. By default the encoding mechanism is BER [BER], but the authors of COPS-PR anticipated that in the future, PRI can also be encoded using XML [XML].

Since the PRI (the data itself) is encoded without any metadata, like headers, class name or attribute names, the receiver of this data has no means to know what kind of data is sent. Therefore, each PRI is preceded by a PRID. The PRID is an identifier that uniquely identifies each instance and also identifies the type of object that is sent, so the recipient knows what kind of instance is following.

### 1.3. C-nums, C-types, S-nums, S-types and the others

To define the type of message and the content that is sent, the C-num and C-type are defined. For example, the original COPS protocol defines 29 kind of messages, using 17 different C-nums, and one or more C-types for each C-num. The COPS authors went into great length to allow new message types, and did not just define these three parameters (client-type, C-num and C-type). In fact, IANA, the organisations who keeps track of parameters in IETF protocols, keeps track of as much as 21 kinds of parameters just for the COPS protocol!

### 1.4. Example of an embedded COPS message

As an example, Figure III-2 shows all attributes of a COPS request message. This particular request message embeds two instances (PRI’s): an event and a ctxtL3Hdr instance.

---

55 IANA keeps track of a ridiculous amount of COPS parameters, namely Client Types Values; C-Num and C-Type Values; R-Types, Reason-Codes, Report-Types, Decision Object Command-Codes, Decision Flags, and Error-Codes; M-Types, Reason Sub-Codes, and Error Sub-codes; S-Num, S-Types, and Error-Codes (2 kinds); A-types, SubTypes (2 kinds); P-types; and finally ErrorValues. The only parameter they don’t seem to track is the so-called Op-code (Operation code), which occurs in the header of each COPS message.
A COPS-PR Request messages is defined by RFC 3084 as:

\[
\text{<Request>} ::= \text{<Common Header>} \\
\text{<Client Handle>} \\
\text{<Context>} \\
\text{*(<Named ClientSI>)} \\
\text{[<Integrity>]}
\]

In here, the Named ClientSI is defined as:

\[
\text{<Named ClientSI: Request>} ::= \text{*(<PRID> <EPD>)}
\]

---

### COPS-PR Objects

<table>
<thead>
<tr>
<th>COPS common header</th>
<th>Version</th>
<th>Flags</th>
<th>Op Code = REQ</th>
<th>Client-type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Client handle obj.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Context object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Named ClientSI object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrity object (optional)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure III-2. Encoding of an event and a ctxtL3Hdr instance in a COPS-PR request message**

---

Note that all C-num, C-type (defined by COPS) and all S-nums and S-typed (defined by COPS-PR) can have a different meaning, depending on the COPS client-type (an attribute in the COPS common header). A client-type must be defined for each area of usage of COPS. Currently, only one client-type is defined, for COPS-RSVP. Since this model uses COPS-PR, and COPS-RSVP doesn’t, a new client-type must still be defined for this model. It is likely that all usages of COPS-PR in conjunction with the DiffServ model will use the same, yet-to-be-assigned client-type.
2. Future of EAP

EAP, the Extensible Authentication Protocol has become widely used, and due to this increased deployment, a few smaller problems are identified. These issues deal with future extensibility, more precise response codes, efficiency and improved security.

EAP has a few problems, and at the moment of writing, the IETF will form a new working group to solve a few of these problems. Thankfully, some other working groups, like pppext (Extensions to PPP) have already contributed to a revision of EAP [draft-ietf-pppext-rfc2284bis].

I will sketch four problems here. For a concise description, watch the EAP group in the IETF, and possibly read their charter [EAP-charter] or the minutes of the introduction meeting [EAP-BOF].

2.1. IANA considerations

A major problem is a pure practical one: the EAP request/response type is one byte, so only 256 different authentication mechanisms can be defined. Currently, 32 types have been defined, most of these not needing a new type. Since there was no policy, IANA would just give anyone who asked for it a new type numbers. The distribution will be more limited in the future (you will need to publish a specification document for the new type).

2.2. Interaction between EAP and other layers and state machine

Another strange issue is that EAP messages are often embedded in other protocols. Take for example RADIUS in the EAP-MD5 example (In the main thesis). There, the EAP-Success message is encapsulated in a RADIUS-Accept message and a EAP-Failure message is encapsulated in a RADIUS-Reject message. Now what if I’m a NAS and receive an EAP-Success message embedded in a RADIUS-Reject message. What should I do: deny or grant access to the user? And how should I tell this to the User? Maybe this is even a valid message! Maybe the user did type the correct password, but is just not allowed to log in between 18:00 and 21:00 in the evening. Problems would arise if the NAS would block traffic because of the RADIUS-Reject, but the User would think he can sent information because of the EAP-Success message. To solve this problem, possible new documents will contain a section on interoperation with other protocols, describing issues per-protocol, like if creation of EAP-Success or EAP-False messages by the NAS is allowed.

Also, the revised document will contain a state machine, which unambiguously show in what state the NAS and client are (busy authenticating, connection allowed, etc.) and when exactly the NAS may or may not change to another state. The main purpose is to describe how to deal with lost or duplicate messages, but hopefully, it will also eliminate the potential problem that the NAS and Client are each in a different state, as described above.

2.3. Efficiency and piggy-backing

When comparing EAP-MD5 to CHAP, you will see that EAP-MD5 is slightly less efficient because EAP uses a separate leg (round trip) for requesting the identity and the challenge response. CHAP does this in one leg. Piggy-backing it sending

---

A BOF is a meeting at an IETF conference aimed to form a new working group. The name BOF is an acronym for Birds Of a Feather is chosen because people can see what “plumage” a new group has before joining it.
information along, in a packet that already had to be sent anyway. So CHAP is piggy-backing the identity information in the challenge, and EAP-MD5 is not. Currently, EAP is not capable of doing any piggy-backing; an EAP message can only contain one type of information. It is suggested that EAP should be capable of piggy-backing as well. The main reason is to allow faster negotiation, because the User can not only say “no, I don’t know this authentication mechanism”, but it can specify multiple mechanisms it does speak. Also, it will be very useful to allow including of a message or more specific error code in the EAP-Failure message, so that the user can be told why he was denied access (“wrong password”, “no login allowed between 18:00 and 21:00” or “you can’t login in from this ISP’s network” for roaming situations).

Even though piggybacking will become possible in EAP, it may still not be possible to combine the identity and challenge request together, because EAP is a lockstep protocol, meaning that the NAS must not send a new request until a new response is received to the outstanding request. It may be possible, but requires a thorough look how this would affect the state machine.

2.4. Tunnelling PPP and EAP over other protocols

One of the most obvious needs for protocols these days is a need for more security. To prevent malicious attackers from using services or accessing restricted information, users only want to communicate with another party if they can be sure it is a legitimate user on the other side of the line. An easy way is to let the other side first authenticate himself or herself before the actual communication starts.

Instead of re-inventing the wheel, or extending protocols, a common solution is to tunnel an existing over another, more secure protocol. This secure protocol is often PPP, because PPP is able to authenticate users. Again it must be stressed that PPP is not very secure: it can be hacked, it is just more secure than the alternative: doing no authentication at all.

An example of a tunnel is shown in Figure III-3, where PPP is tunneled over an existing UDP/IP layer. This is how L2TP works. Even though PPP offers much more options, usually the key reason for using PPP for tunnelling is the added security. Adding security by adding a PPP layer on top of other protocols is extremely popular. The most popular examples of its usage include:
- PPPoE (PPP over Ethernet LAN’s), [RFC 2516],
- PPTP (PPP over TCP/IP) [RFC 2637], which is often used for ADSL connections to your home and which was a protocol developed by a consortium led by Microsoft and 3Com,
- L2TP (PPP over UDP/IP, and other transport protocols) [RFC 2661], which is more generic than PPTP, but was published just a month later and is thus less popular;
- PPP over AAL5 (an type of ATM network) [RFC 2364]

The most obvious disadvantage of this layering can clearly be seen in Figure III-3: there is a lot of overhead because of the additional layers. Another approach is just to use EAP directly, without the use of PPP. The most common example of this usage is IEEE 802.1x [IEEE 802.1x], which defines how an access point can allow or disallow someone access to the network. IEEE 802.1x is used in the IEEE 802.11 [IEEE 802.11]
protocols, which define Wireless LAN’s. So Wireless LAN’s use EAP for authentication.

There is work in progress to allow direct usage of EAP as the authentication mechanism for other protocols, including EAP over IP [draft-engelstad-pana-eap-over-ip], EAP over UDP [draft-engelstad-pana-eap-over-udp], EAP over ICMP [draft-tsirtsis -eap-over-icmp] and EAP for HTTP authentication [draft-torvinen-http-eap].

2.5. Improving security

A disadvantage of PPP and EAP is that it is not completely secure, although it does make other protocols more secure by adding authentication. Someone watching the packets between the User and the NAS can not get hold of the password, but it is possible to just view all traffic. In the case of CHAP and EAP-MD5, Sebastian Krahmer [Krahmer (2002)] shows that is even possible for a hacker to impersonate a legitimate user. This problem is bad, since an attacker may be able to inject packets into the network which causes the User and the NAS to use a weaker authentication mechanism, so that they will end up using the weakest possible authentication mechanism. Even though there are ways to circumvent part of this problem, the real solution lies in a mutual authentication of both the User and the NAS, so that as soon as the connection is set up, they know no-one else can retransmit, alter or spoof their packets.

A solution proposed by some people from RSA security is PEAP, Protected EAP, which uses EAP over a TLS-encrypted connection. There EAP is not used to authenticate the user, but to negotiate a TLS connection. As soon as the negotiation is complete, then another EAP connection (within the EAP-TLS connection!) can be set up to authenticate the user.
3. Student housing example

This chapter will give an example usage of the Access Bind PIB. The example is fictitious, although it is very much based on an actual situation: the student campus where I live, the IBB-complex in Utrecht. [IBB-SSHU]

3.1. The IBB-complex

The IBB-complex is a housing premise owned by the SSH, the Stichting Sociale Huisvesting in Utrecht [SSHU], a building society in Utrecht owning 7500 homes, mostly (5500) student rooms. The IBB complex is named after an Utrecht writer, Ina Boudier-Bakker. The street bears her name.

Despite the fact that the complex is mainly built for student housing, about 14% of the tenants is not a student [UU FRW (2001)], usually because they already graduated.

My background

I have lived on the IBB-complex since 1996, and am actively involved in the tenant organisation (the Woonbestuur IBB). In addition, I’ve been involved in two successive networks for the tenants: the former IBBnet, a University sponsored network with a backbone using existing television cables, and it’s successor, the Intercom 2000 network, connecting 4 complexes of the SSH, with an all-fiber backbone and a 100 Mbit/s connection to each dormitory. For the first network, I’ve been involved in the helpdesk and wiring individual rooms. For the tenant organisation, I’ve been one of the negotiators with the housing organisation about the cost and offered services (helpdesk and servers). In the new network, I’m involved in server maintenance, server finance and tenant representation.

3.2. SSHUnet

In 2000, the SSH decided that each room should get a broadband internet connection. It was decided that in the first stage 4400 rooms on 4 student complexes should get a 100 Mbit/s network connection. The backbone consists of 1 Gbit/s and 2 Gbit/s connections (the criterion was that 20% of the tenants should simultaneous be able to view a videostream of 2 Mbit/s). This network became later known as the SSHUnet.

Utrecht University and Hogeschool van Utrecht (an institute for higher professional education) offered the connection between the local SSH network to the Internet. The connection from Utrecht University (SOLISnet) to the Internet is by using the Dutch research network (SURFnet), as is shown in See figure III-4. SURFnet is subsidized by the Dutch government.

Per user provisioning

Because SURFnet is subsidised by the Dutch government, only students and personnel of higher education institutions, like universities, are allowed to use this network. Since up to 14% of the inhabitants is not a student or university staff, they are formally not allowed to use the SURFnet backbone.

58 The books of Ina Boudier-Bakker (1875-1966) [MNL Boudier-Bakker] are still often read, even though one of her books also got the shortest review ever: “Do not open”.

59 In 2001 Utrecht University students showed that about 14% of the inhabitants of the four connected premises is neither part-time, nor-full time student, and is either employed or unemployed. However, they did not distinguish between employees of institutes for higher education, like Utrecht University, who are allowed to use the SURFnet backbone, and other employees.
Therefore, these people should use a different backbone to connect to the Internet. Authentication is necessary to discriminate between students and university personnel on one hand and the non-students on the other hand.

Figure III-4 shows a potential overview of the SSHUnet and the networks it is connected to. The bottom half is the SSHUnet, with the four interconnected building premises: IBB, Warande, Tuindorp and Cambridgelaan. Cambridgelaan has a 2 Gbit/s connection to the Data Center of the network operator. This network operator (a third party contracted by SSH), has direct connections to SOLISnet (Utrecht University network), HvUnet (Hogeschool van Utrecht network, not shown) and the Dutch research network (SURFnet). Each of these connections has a high bandwidth (1 Gbit/s). For non-students, a lower bandwidth (2 Mbit/s) connection is drawn.

Before a user is allowed on the network, he or she should authenticate his- or herself. If the user is not a university student, he or she will be provisioned to use the lower bandwidth (2Mbit/s) line.

3.3. The Access Bind PIB

There are three features which COPS and the Access Bind PIB have, which are required in this situation:
- First, the Access Bind PIB is capable of configuring a PEP (the Access Device) at boot time. Since all 4400 rooms have a distinct physical wire going to many devices located at each building, this is a big advantage, because it can be done remote and there is certainty that each device is configured the same.
- Second, the Access Bind PIB is capable of forwarding the authentication data.
- Third, the Access Bind PIB is capable of doing per-user provisioning, so that each user will use the egress port and thus the connection associated with his or her status (2 Mbit/s connection or 1 Gbit/s connection).

Currently no other single protocol is capable of combining these three functions.

In order to allow for a standard authentication solution, the recommended protocol would be to use PAP or CHAP over PPPoE [RFC 2516]. PPPoE, PPP over Ethernet is
a way to allow users to authenticate themselves before using their ethernet connection. PPPoE is supported by all major operating systems, like Windows, MacOS X and Linux.

![Figure III-5. Possible model for authenticating all students and university personnel.](image)

Figure III-5 shows a possible architecture how students and university personnel are authenticated and how the access servers are provisioned. To allow all Dutch students, and not just students of Utrecht University to use the high bandwidth connection, all university authentication servers should be trusted. This is a roaming situation. Thankfully, SURFnet has a central RADIUS server which can query other university RADIUS servers. So the configuration server of the SSHUnet just needs to authenticate users with this central server. Alternatively, for better performance, it can also be configured to recognize username in the Utrecht University domain (uu.nl) and connect directly to the Utrecht University RADIUS server. If it is undesirable that users who do not authenticate themselves use the slow 2 Mbit/s connection by default and thus saturate that line, the configuration server can also be configured to authenticate non-students as well. In that case, and additional non-student authentication server is necessary as well.

**Per user provisioning**

Currently, as soon as a user is authenticated, the SSHUnet configuration server will have to add a filter in the access server that made the request. This filter should filter all data coming from the authenticated user and forward it to the appropriate connection (either the 1 Gbit/s SURFnet connection or the direct 2 Mbit/s connection).

Because it is not possible to tell if a student or non-student is about to log in, only one eventHandlerElement filter can be set up, to trigger an access request for every possible user. Because the eventHandlerElement in the Access Bind PIB can only point to one next DiffServ element, and it is not possible to specify a distinct next DiffServ element per COPS Client Handle (per authentication request), the next
DiffServ element must specifically be specified by the PDP by installing a new eventHandlerElement, scope and filter for each user who successfully logged in.

Alternatively, it is possible to use a lookup table with all wall-outlet numbers (specifying the port on the access device where the users connects to), and assign a different IP range to users with a high bandwidth connection and to users with a low bandwidth connection. The next DiffServ element would then be a filter that forwards the traffic to the correct egress port (line) based on the IP address. The disadvantage of this approach is that users who log in can get the other bandwidth by changing their IP address.

**Filters**

The Access Bind PIB is very flexible and the filtering of certain traffic can be done using either IP address, MAC address or even wall outlet number. The most secure option would of course be to base a session on two all outlet. Otherwise, someone could get unauthorised access by using the IP address of someone who already legitimate logged in. For added security, the Access Server could be configured with a filter which both filters on port number and assigned IP address (a logical AND).

**Servers**

Currently, the eventHandler can only denote one set of allowed authentication protocols (i.e. PAP or CHAP). It is not possible to use an eventHandlerElement to specify that users connecting to a certain range of wall outlet numbers do not have to authenticate themselves using PAP or CHAP.

This behaviour would be desirable for servers that reside in the SSHUnet but do not need authentication.

It is possible to circumvent this by putting another Classifier in front of the eventHandler in the datapath.

**PPPoE**

One of the advantages of PPPoE is that it is not necessary that users are connected to a special VLAN and are transferred to a different VLAN as soon as they are logged in. For PPP, and thus for PPPoE as well, a user can be authenticated before he or she gets an IP address. So it is possible that the IP address depends on the authentication result and whether the user is a student or non-student.

This is in fact required behaviour because the IP address of students and non-students should be in a different subnet, because routers on the internet should route differently to the SSHUnet for students and non-students: traffic for students should be routed over the fast SURFnet, while other traffic should be router to the slower 2 Mbit/s line.

Currently, the Access Bind PIB does not have an option to provision the PPP options for a certain session, and can thus not specify the IP address that LCP (the PPP configuration protocol) should assign to the User. However, it is possible that these options (IP address, but also name server address) are specified in another object, and it is possible to include this object in the Decision from the PDP to the PEP. By

---

60 Maybe this can be avoided by also adding the IP addresses to the lookup table, but then, the user does not have an IP address at the moment he logs in. This is out of scope of this document. Maybe, when the authentication module for the SSHU network is really going to be implemented, I will write a bit more about it on the website accompanied with this thesis. See the abstract page for the URL.

61 This can be achieved by using the same value in eventHdlrHandleScopePrecedence attribute for both filters.

62 I am currently unaware of a PRC specifying LCP options. I will update the on-line version if I encounter one. Alternatively, you can search the Internet-Draft directory at the IETF website.
using the correct COPS Client Handle, the PEP can associate this decision with the correct user.
| **LCP** | 21 |
| **Leg.** | See Round Trip |
| **Lookup Table** | 58 |
| **Message Sequence** | 29 |
| **COPS** | 90 |
| **NAD** | See NAS |
| **NAS** | 21, 50, 71, 74, 83 |
| **Network Access Server** | See NAS |
| **Network protocol** | 56, 64 |
| **OSI model** | 56, 59 |
| **Outsourcing** | 8 |
| **PAP** | 21, 24 |
| **Patent** | 59 |
| **PDP** | 8, 14, 76, 80 |
| **PEAP** | 95 |
| **PEP** | 8, 14, 17, 77 |
| **PGP** | 75 |
| **PIB** | 13, 14, 15 |
| **Pinpas** | 69 |
| **Policy** | 76 |
| **Policy Decision Point** | See PDP |
| **Policy Enforcement Point** | See PEP |
| **Policy Engine** | 76 |
| **Policy Repository** | 76 |
| **PPP** | 94 |
| **PPPoE** | 21, 22, 94, 99 |
| **PRC** | 16 |
| **PRI** | 16, 91 |
| **PRID** | 16 |
| **Proprietary** | 59 |
| **Protocol** | See Network protocol |
| **Protocol stack** | 56 |
| **Provisioning** | 9, 14 |
| **Boot-up Configuring** | 30 |
| **Per-user Configuring** | 32, 45, 97 |
| **Using DiffServ** | 18 |
| **Public information** | See Proprietary |
| **QoS** | See Quality of Service |
| **Quality of Service** | 9, 14, 17, 85, 86 |
| **RADIUS** | 24, 83, 93 |
| **RAP working group** | 10, 83 |
| **Registered Identity** | 75 |
| **RFC** | 64, 65 |
| **Roaming** | 67, 72, 77 |
| **Round Trip** | 29 |
| **Combining Messages** | 33 |
| **Round Trip Time** | 29, 85 |
| **Router** | 50, 83, 86 |
| **RSVP** | 86 |
| **Scope** | 35 |
| **Service Level Agreement** | 70, 76 |
| **Session Concept** | 39 |
| **SLA** | See Service Level Agreement |
| **SLIP** | 20 |
| **SNMP** | 83 |
| **Specification** | 58 |
| **SSHUnet** | 96 |
| **Standard** | 58, 59, 65 |
| **Internet Standard** | 63, 65 |
| **Standardisation body** | 58 |
| **SURFnet** | 96 |
| **Switch** | 83 |
| **TagReference** | 35 |
| **Ticket** | See Certificate |
| **TLS** | 20 |
| **ToS** | 18 |
| **Traffic Class** | 18 |
| **Triggering** | See Event Triggering |
| **Trust-relation** | 70, 73, 77 |
| **Tunnel** | 94 |
| **UMTS** | 60 |
| **Working group** | 60, 64, 65 |
### Table of Figures

Figure I-1. The pull model, with a user who wants to access a service from the PEP (Policy Enforcement Point). The decision to allow or deny the access request is outsourced by the PEP to the PDP (Policy Decision Point) 9

Figure I-2. The Access Bind model (a pull model) 11

Figure I-3. Example of DiffServ elements inside a PEP. (Figure based on source by Walter Weiss). 16

Figure I-4. Negotiation of the Authentication protocol for PPP connections. 19

Figure I-5. Example of IPCP negotiation. IPCP negotiates the configuration of IP traffic. 20

Figure I-6. Authentication using PAP. 22

Figure I-7. Authentication using CHAP, Challenge Authentication Protocol. 22

Figure I-8. Sequence of messages for the EAP-MD5 authentication protocol. 24

Figure I-9. EAP Message types defined in RFC 2284. 25

Figure I-10. Sequence of actions and events that may occur in this order at a PEP. 28

Figure I-11 Message sequence at boot time of the PEP. The PDP configures the PEP by installing policies. 28

Figure I-12. The messages that may be sent between a PEP, PDP and User when an unauthenticated user connects to a PEP, and EAP is used for authentication. 29

Figure I-13. Combining messages. 31

Figure I-14. Legend for the data structure figures. 32

Figure I-15. Event classes and eventHandler classes. 33

Figure I-16. A few Filter classes. 34

Figure I-17. contextData classes 39

Figure I-18. Authentication classes 40

Figure I-19. Internals of an AAA server, with five identified types of protocols or API’s. 45

Figure I-20. A typical edge device. The network access server is located at the edge of the network, and acts as a portal for the user to access the network. 48

Figure I-21. Example of an Access Server which is not located between the User and the network. 48

Figure II-1. The OSI reference model for networks 54

Figure II-2. The IETF logo 60

Figure II-3. The AAAarch logo 64

Figure II-4. Trust relations when buying in a shop using a bankpas. 67

Figure II-5. Model for dialing in using a Network Access Server and an authentication server. 68

Figure II-6. Dial-in roaming service 69

Figure II-7. A possible model for a bandwidth broker. 70

Figure II-8. The AAA pull model. 75

Figure II-9. The AAA push model. 76

Figure II-10. The AAA Agent model. 76

Figure II-11. Internal structure of an AAA-server, with five types of different interfaces or protocols identified 78

Figure III-1. Typical embedding of PRI (instances) in a COPS-PR message. 85

Figure III-2. Encoding of an event and a ctxtL3Hdr instance in a COPS-PR request message. 86

Figure III-3. Example of a tunnel: PPP over IP 88

Figure III-4. SSHUnet topology and the networks it is connected to. 91

Figure III-5. Possible model for authenticating all students and university personnel. 92
REFERENCES

The numbers refer to the pages in this thesis where the reference was mentioned.

Organisations

[3GPP] Third Generation Partnership Project (3GPP), website; http://www.3gpp.org/ .................................................................58
[AAAnr -charter] Authentication Authorisation Accounting ARCHitecture Research Group, website; http://www.irtf.org/charters/AAAnr.html .10, 64
[ECMA] ECMA, website; http://www.ecma.ch/ ..................59
[IBB-SSHU] Huisvestingscomplex IBB Utrecht, website: http://www.ibb.sshunet.nl/ ..............................................90
[IEC] IEČ, website; http://www.iec.ch/ ................81
[IEEE] Institute of Electrical and Electronics Engineers, website; http://www.ieee.org/ ..............................................................58
[IETF] Internet Engineering Task Force, website; http://www.ietf.org/ ....58
[IPNRG-charter] Interplanetary Internet Research Group Charter, website; http://www.irtf.org/charters/ipnrg.html .....................63
[ISO] International Organisation for Standardisation, website; http://www.iso.org/ .................................................................57
[ISOCJ Internet Society, website; http://www.isoc.org/ ...........60
[ITU] International Telecommunication Union, website; http://www.itu.int/ ....58
[Merriam-Webster] Merriam-Webster dictionary, website; http://www.m-w.com/ ......................................................56
[SSHU] Stichting Sociale Huisvesting Utrecht, website: http://www.sshu.nl/ ...90
[W3C] World Wide Web Consortium (W3C), website; http://www.w3.org/ ....59

Internet documents

A special type of Internet Documents are RFC’s. You can find RFC’s at http://www.rfc-editor.org/ or at http://www.ietf.org/

[IANA PPP-numbers] IANA Database, Assigned PPP numbers; http://www.iana.org/assignments/ppp-numbers ................19, 25


[RFC 1661] W. Simpson (Editor), The Point-to-Point Protocol (PPP), STD 51, RFC 1661, Jul 1994..............................................................19


[RFC 2205] R. Braden (Editor), Resource ReSerVation Protocol (RSVP), RFC 2205, Sep 1997 ..............................................................81


[RFC 2555] RFC Editor, et al., 30 Years of RFCs, RFC 2555, April 1999 ..............................................................61


[RFC 2661] W. Townsley, et al., Layer Two Tunneling Protocol (L2TP), RFC 2661, Aug 1999 ..............................................................88


[RFC 2748] D. Durham (Editor), The COPS (Common Open Policy Service) Protocol, RFC 2748, Jan 2000 ..............................................................13, 37, 84

[RFC 2749] S. Herzog (Editor), COPS usage for RSVP, RFC 2749, Jan 2000 ..............................................................13

[RFC 2873] X. Xiao, TCP Processing of the IPv4 Precedence Field, RFC 2873, Jun 2000 ..............................................................17


[RFC 3233] P. Hoffman, S. Bradner, Defining the IETF, BCP 58, RFC 3233, Feb 2002 ..................................................................................60

Internet drafts
You can find Internet Drafts at http://www.ietf.org/. Note that Internet Drafts expire after six months and removed from the website. You can find copies at the website of this thesis, http://www.macfreek.nl/pubs/

[draft-engelstad-pana-eap-over-udp] P. Engelstad, EAP over UDP (EAPoUDP), draft-engelstad-pana-eap-over-udp-00.txt (Work In Progress), Feb 2002 89
[draft-tsirtsis-eap-over-icmp] G. Tsirtsis, EAP over ICMP, draft-tsirtsis-eap-over-icmp-00.txt (Work In Progress), Jan 2002 89

Books and articles
[UU FRW (2001)] Onderzoek naar de persoonskenmerken van SSH-huurders, waardering voor SSH-complexen en woonwensen van SSH-huurders, Onderzoek in opdracht van de SSH, uitgevoerd door studenten van de Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht, pag. 14, maart 2001 ............................................................................................................................................ 90

[Van Yperen (2000)] P. van Yperen, M. Engelbarts, Website meten over het internet; 2000; http://hst3791.phys.uu.nl/ ........................................................................................................... 48

Meetings and lectures

[De Laat (2001)] Cees de Laat, Intelligence to the edge, speech during the symposium Twee gezichten van Fysiche Informatica, held to commemorate the retirement of Prof. Dr. Wim Lourens, 15 june 2002................................................................. 65

[De Laat (2002)] Cees de Laat, Private Communications, 2002 .................................................. 77


[EAP-BOF] EAP BOF (Birds Of a Feather, a first meeting aiming to establish a working group) during 53rd IETF, Minneapolis, Mar 2002; http://www.ietf.org/proceedings/02mar/206.htm (agenda) and http://www.drizzle.com/~aboba/EAP/presentations.zip (slides of this meeting).......................................................................................................................... 87