lecture

Lecture on

Intermediaries and Services

Security aspects of proxies and internet services

Walter Kriha

Roadmap

Part 1: Firewall Architecture

- The purpose of a firewall
- IP components important for firewalls
- Firewall Types
- Firewall limits

Part 2: Filtering Technology

- IP, TCP, ICMP filtering
- static filtering: ipchains
- dynamic (stateful) filtering: iptables
- Application level filtering: proxies
- Filtering limits

Part 3: Services and Protocols

- frequently needed services and their problems
- dangerous services
- middleware protocols
- New threats (p2p, webservices)

Part 4: Securing Web Applications

- Content Management on the web
- Transactional Services
- Web Application Servers

We will deal with firewall issues rather in detail as they have a lot of impact on software architecture as well.

Goals for today

Learn how to analyse security aspects of internet services

Learn the problems of basic services (smtp, web, dns, news)

Understand security aspects of middleware protocols

Have a look at the new web services infrastructure and how it relates to firewalls

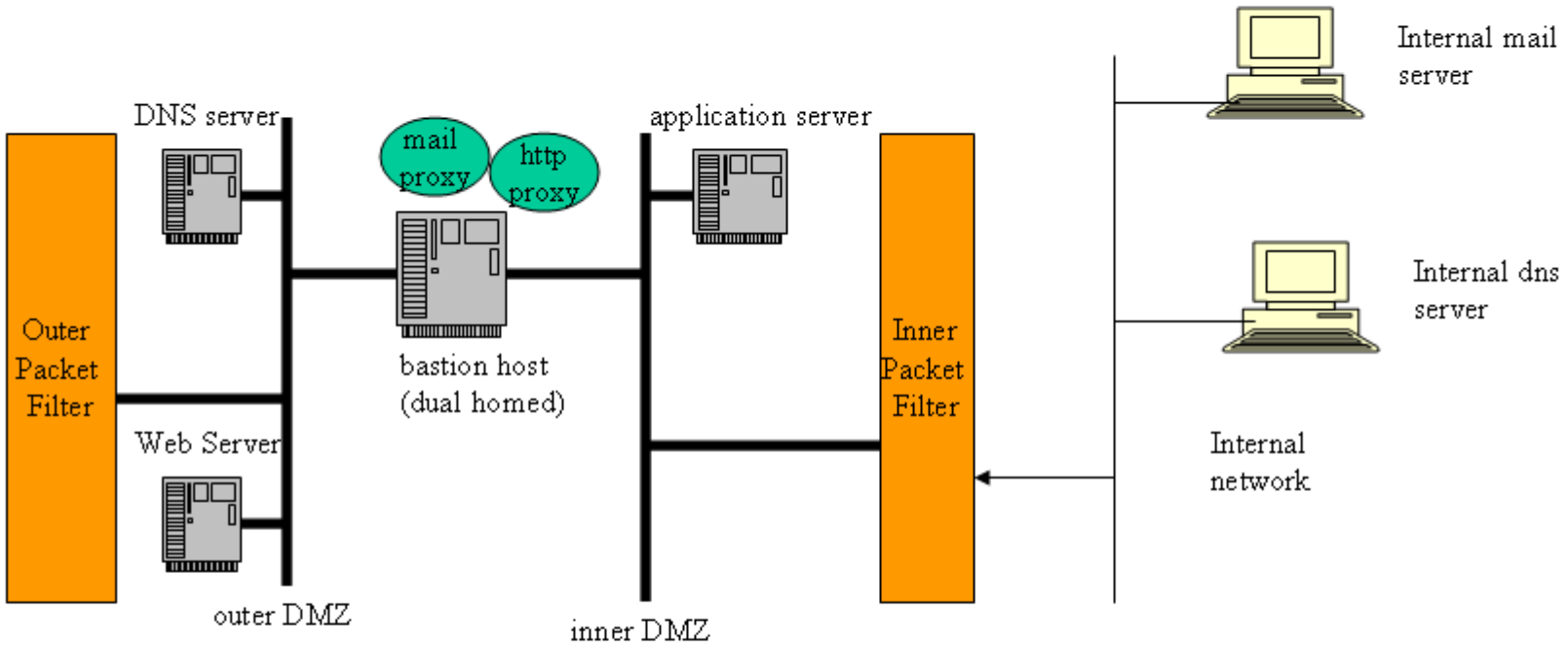
Your job as a security specialist includes knowing how to secure the standard internet services as well as doing an analysis of new and upcoming technology.

Intermediaries

„Intermediary components act as both a client and a server in order to forward, with possible translation, requests and responses. A proxy component is an intermediary selected by a client to provide interface encapsulation of other services, data translation, performance enhancement or security protection. A gateway aka „reverse proxy“ component is an intermediary imposed by the network or origin server to provide an interface encapsulation of other services, for data translation, performance enhancement or security enforcement. Note that the difference between a proxy and a gateway is that a client determines when it will use a proxy“

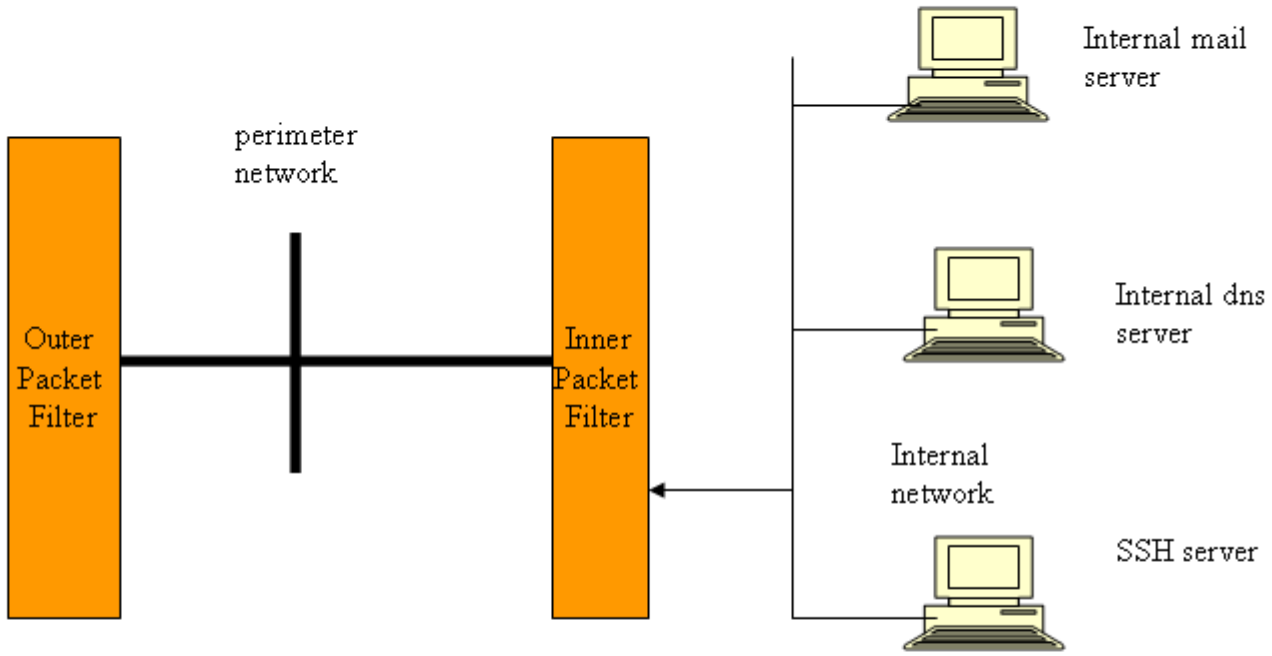
R.Fielding, Taylor, Principled Design of the Modern Web Architecture

Basic proxying firewall architecture



The quality of a firewall today is determined by the speed and quality of its proxy services. (See firewall study II by the Bundesamt für Sicherheit in der Informationstechnik, bsi.de). In this design there is no direct connection between internal and external network. All traffic needs to be proxied.

Basic direct service architecture



Some services may be provided by a direct connection through the firewall to the server. Some others should always be proxied.

Why use Proxies?

- additional services like authorization
- no direkt network link between networks
- User mode proxies allow advanced protocol filtering
- User mode proxies allow advance content filtering
- Proxies allow caching of frequently used data and can increase efficiency considerable. Clients get data faster and overall bandwidth needs are reduced

Disadvantages of Proxies

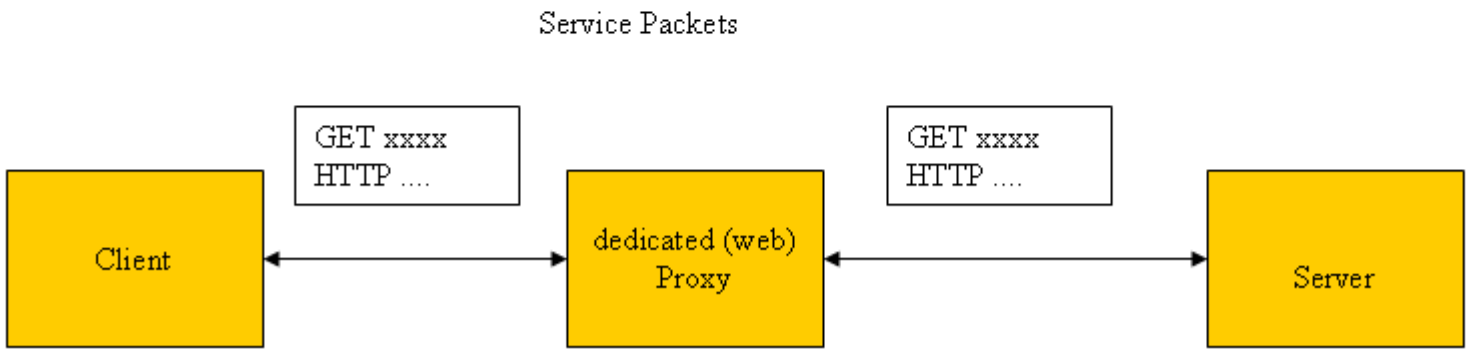
- User mode intermediaries are much slower
- For every service to be allowed through the firewall there needs to be a proxy software available
- If we already allow fairly unrestricted access for our users to e.g. ftp servers in passive mode (to ports above 1023) then restricting some access e.g. to www servers on port 80 does not make much sense.

Types of service proxying

1. Through proxy aware clients (e.g. web browsers and http proxies)
2. Through special proxy enabled clients (e.g. applications linked with a special proxy-enable library)
3. Through special user procedures because the service (client) itself does not understand proxies.
4. Transparent proxying, e.g. by redirecting traffic through destination NAT.

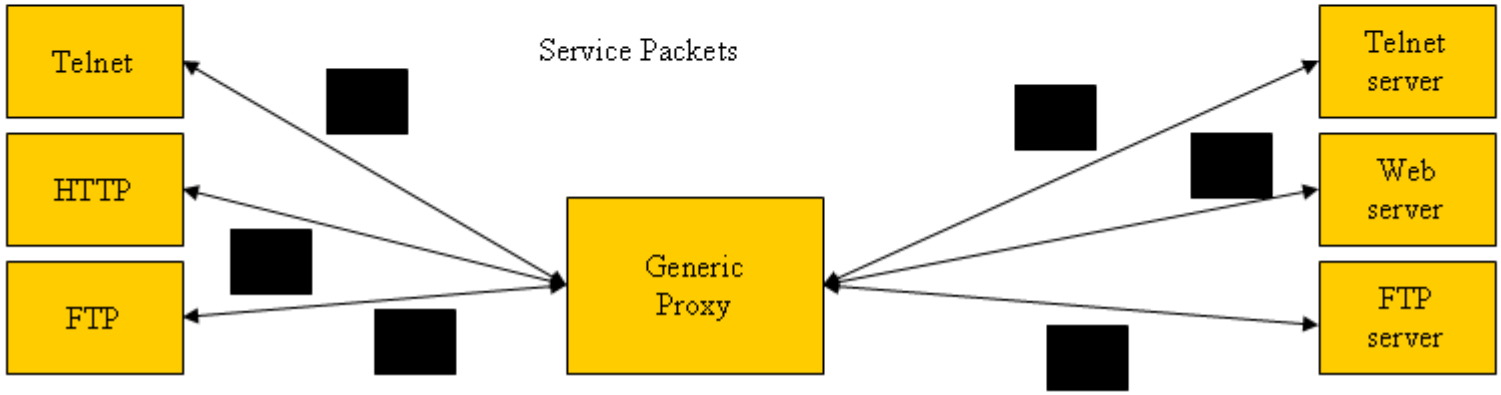
According to the end-to-end argument of networking and communications, proxy aware applications are the most powerful. Most other solutions hide the proxy from the application which can sometimes cause communication failures which are not easily explained.

Application Level Proxies



Application level proxies understand the protocol of the service which is proxied. Typically those proxies support only one service type or protocol and are therefore also called „dedicated proxies“. Application level proxies are necessarily dedicated proxies.

Circuit Level Proxies



Circuit-level proxies just forward packets back and forth without inspecting the application protocol. They can still provide a number of useful security features while being able to support a lot of different applications or services.

Firewall proxy packages

- SOCKS5 (circuit level, user authentication, udp and icmp support, connection logging, notifications, very generic. Clients need to be changed to support socks)
- TIS FWTK (individual intelligent proxies for each service, uses proxy aware user procedures mostly (clients need not be changed)
- SQUID
- „Natural“ proxy services (store-and-forward services like SMTP, NNTP, NTP)

If for a certain unsafe service no proxy package is available, the last ressort is to use a victim host on the internet side of the dmz.

Protocol characteristics good for proxies

- no embedded IP addresses or port numbers
- no checksum of IP header information in other data blocks: proxies would need to change source, destination and port numbers.
- A clear connection establishing phase that declares what is needed but does not give away private information yet. Unlike „connect“ to http proxies which forces them into generic mode but does not tell for which protocol this is requested.

Example: http service without Proxy

```
$IPTABLES – A FORWARD –s $INTERNAL_NET –dport httpport –j accept
$IPTABLES – A FORWARD –s $INTERNAL_NET –dport httpsport –j accept
```

Note that this rule will allow access to external http servers ONLY at the default port 80. All internal machines will have www access to those servers. No user authentication is done.

```
$IPTABLES – A FORWARD –s $INTERNAL_NET  –dport $UNPRIVILEGED –j accept
```

If you want to allow http access to web servers running on non-standard ports you would have to open outgoing connections from all internal unprivileged ports. Remember that you have no guarantee that the target ports are really running http servers.

(We are assuming that the proper incoming rules to accept established and related connections are in place. Can you name some security risks associated with the second approach?)

Security Analysis:

```
$IPTABLES – A FORWARD –s $INTERNAL_NET –dport $UNPRIVILEGED –j accept
```

The consequences of such a rule are rather dire:

-users are completely trusted to not connect to any servers on the Internet running dangerous services

-Any form of malicious code that gets downloaded can talk back freely to master controller on the internet.

-Firewall piercing is easy

Example: http service with transparent Proxy

```
$IPTABLES –t na t–A PREROUTING –s $INTERNAL_NET –p tcp–dport httpport –j REDIRECT –to-ports 8080
```

Here we silently re-route traffic to external web servers running on port 80 to our web proxy running on the firewall machine. We could also forward the request to a proxy running in the inner DMZ. But note that we still catch only traffic to the default port and not to a web-server running e.g. on 4040 on some machine.

But we already gain the advantage of being able to authenticate internal users and to restrict access based on user or requested resource. We can also do advanced content filtering, e.g. look for keywords. The target server will need to be a http server otherwise the proxy won‘t work.

(We cannot route ALL traffic to our web proxy because only some requests are really http requests)

Example: http service with visible Proxy

```
$IPTABLES –A FORWARD –s $INTERNAL_WEB_PROXY –p tcp –sport $UNPRIVILEGED –j ACCEPT
```

We accept all traffic coming from our internal web proxy on unprivileged ports no matter where it goes on the internet. We know that the proxy will only let through regular http traffic. We gain the same advantages as with the transparent proxy except for clients not needing to know about the proxy.

Most http clients are already proxy aware and can be centrally administrated so that knowing about the proxy is not a big deal.

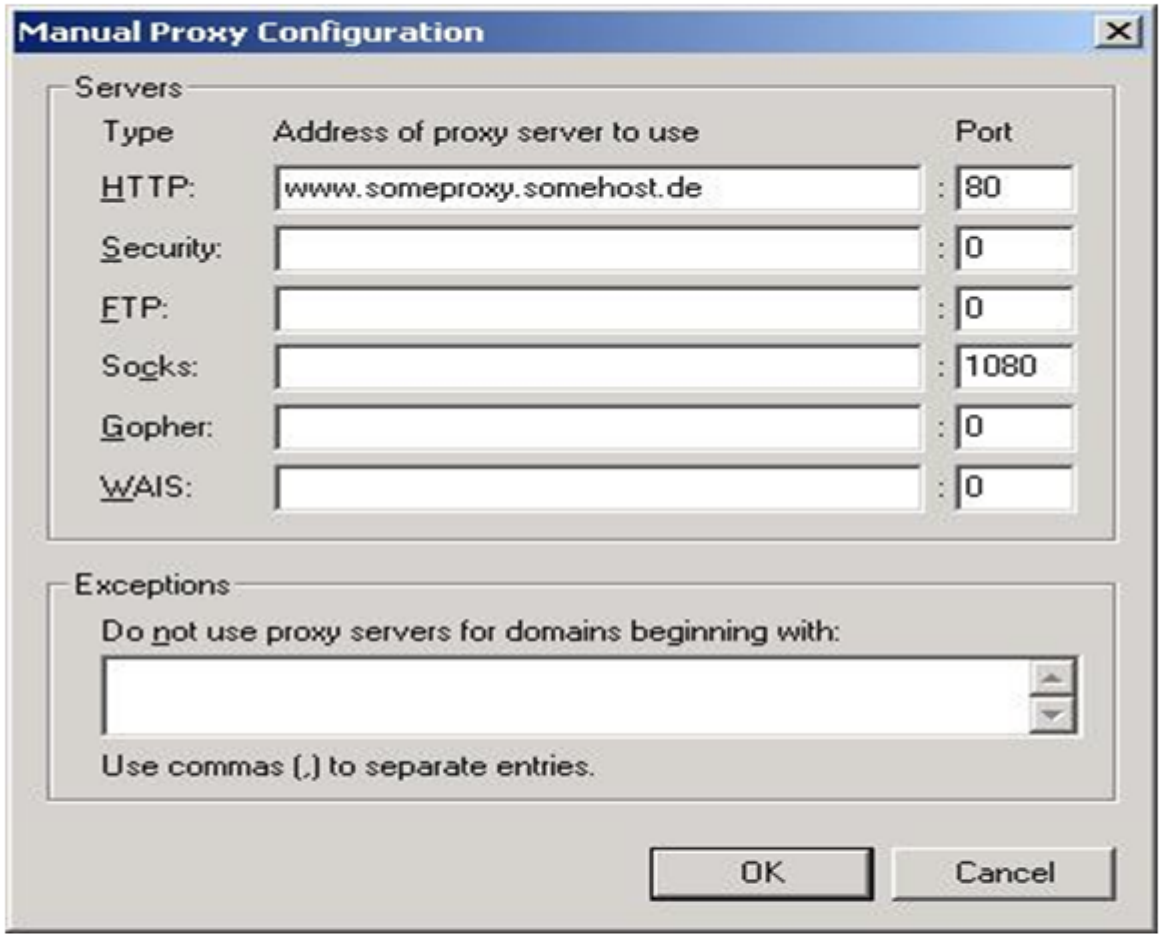
Please note: To force all web traffic through the proxy the following policies need to be in place:

-no direct connection from internal net to internet is allowed (no unspecific „allow-all“ policy from internal network

-every service is proxied through special proxy software (e.g. ftp, mail etc.)

What we cannot avoid is the possibility of some internal user running a protocol ON TOP OF HTTP (http tunneling) to pierce the firewall.

Visible proxy configuration in Netscape



A client browser would request a page from a proxy server by providing the fully qualified URL, e.g. GET www.somehost.de/index.html HTTP/1.0 instead of just providing the relative path information (/index.html). The proxy can then parse the full URL and extract the target host easily. Frequently attackers scan for proxy services which are also available from the outside. Fully qualified URLs serve to detect http proxies.

An evaluation framework for services

- authentication, non-repudiation
- confidentiality/integrity
- proxying capabilities
- NAT capabilities
- Protocol semantics (separation of concerns etc.)

The RFC –draft from Rescorla/Corver covers the security details that need to be defined for an internet service (see resources)

Service Risk Analysis

- What does a protocol allow/do? The more powerful a protocol is, the more dangerous it is. Example: „turn“ in SMTP.
- What information is exposed through a protocol?
- What can be changed through a protocol?
- Is it bi-directional?
- Will the service open the door for other attacks? (e.g. ftp support)
- Does the service need user accounts on DMZ machines (SSH?)
- Does the service require an interactive connection between parties or can the task be performed in batch mode – this reduces the chances of an intruder to cause damage. E.g. instead of interactive ftp a batch command using secure copy.

Useful Service Properties

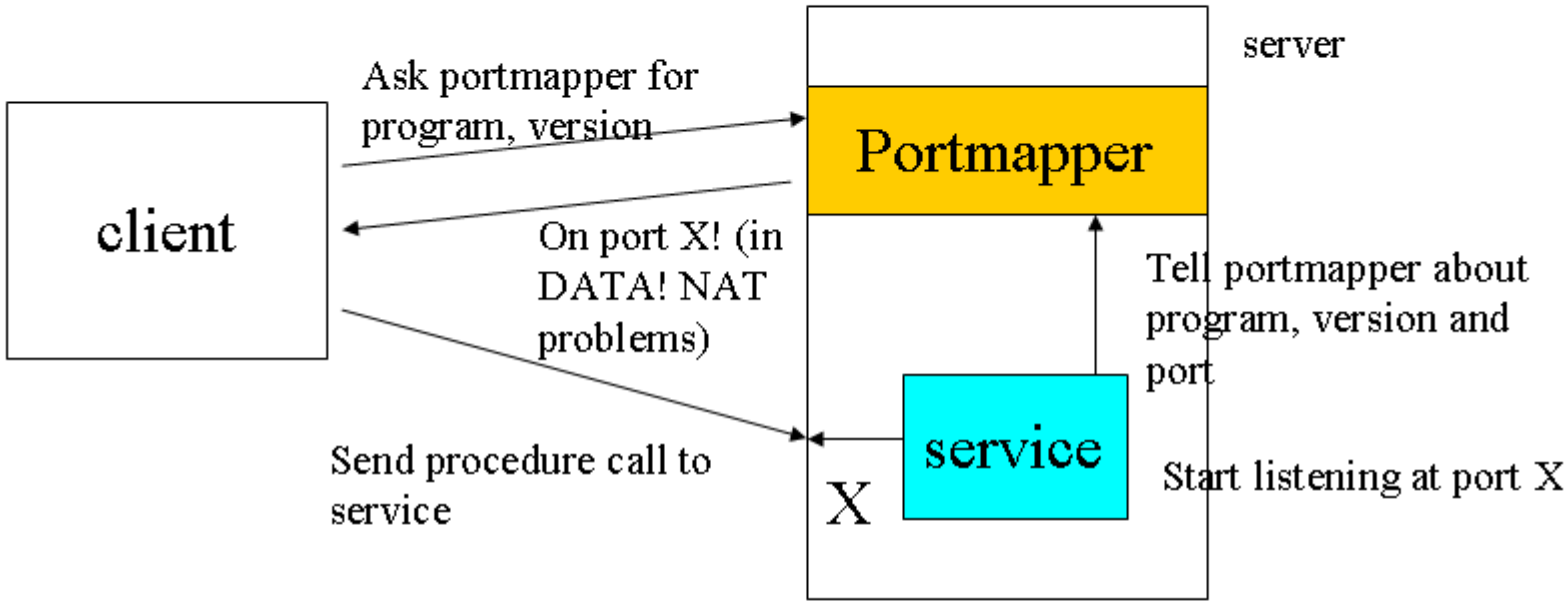
- TCP protocol used instead of UDP or ICMP. (Connection tracking possible)
- Only one connection per service use: Return connections always require holes in the firewall to be opened, even if only for a short time.
- Only a single session running over each connection. No connection re-use for different service types.
- Single and fixed port assignments

Services which have those properties are usually easy to proxy (tcp based) and have a clear purpose which can be tracked by proxies. Those types of services serve typically one purpose and are therefore less generic than e.g. middleware protocols.

Security Properties of Middleware

1. RPC type protocols
2. Distributed Object Middleware: DCOM, CORBA
3. SMB (Netbios based services)
4. SSL/TLS
5. Network layer tunnels (IPSEC, PPTP, L2TP)
6. Web Services

RPC based services (Sun RPC, DCE)



One major problem for firewalls (both packet filter as well as proxy based) is that ports for rpc services can change frequently. There is no fixed mapping of service to port because the client first contacts the portmapper to find the real port. A packet filter needs to open most ports (unprivileged). Other problems are missing authentication and the possibility of two-way control flow. A proxy would need to proxy both locator service (portmapper) as well as the services themselves. And last but not least: most rpc services are UDP based! Do not let NIS or NFS etc. through your firewall!

Distributed Objects (DCOM, CORBA)

-generally not designed for internet use

-no or weak authentication, authorization

-bi-directional connection use

-the protocols use callbacks (reverse connections)

-the protocols embed ip addresses and ports in data sections and cannot be NAT‘ed easily

-in the DCOM case: services are tightly integrated into the operating system and offering them through the firewall basically opens up complete machines to outside use.

In the case of IIOP every application needs its own special proxy. Another possibility is to run IIOP over SSL/TLS for authentication and integrity. The consequences are: do not run DCOM over your firewall. Use an IPSec tunnel if windows machines need to be connected across firewalls. Use special proxies or SSL for CORBA/IIOP services.

Netbios over TCP/IP, SMB

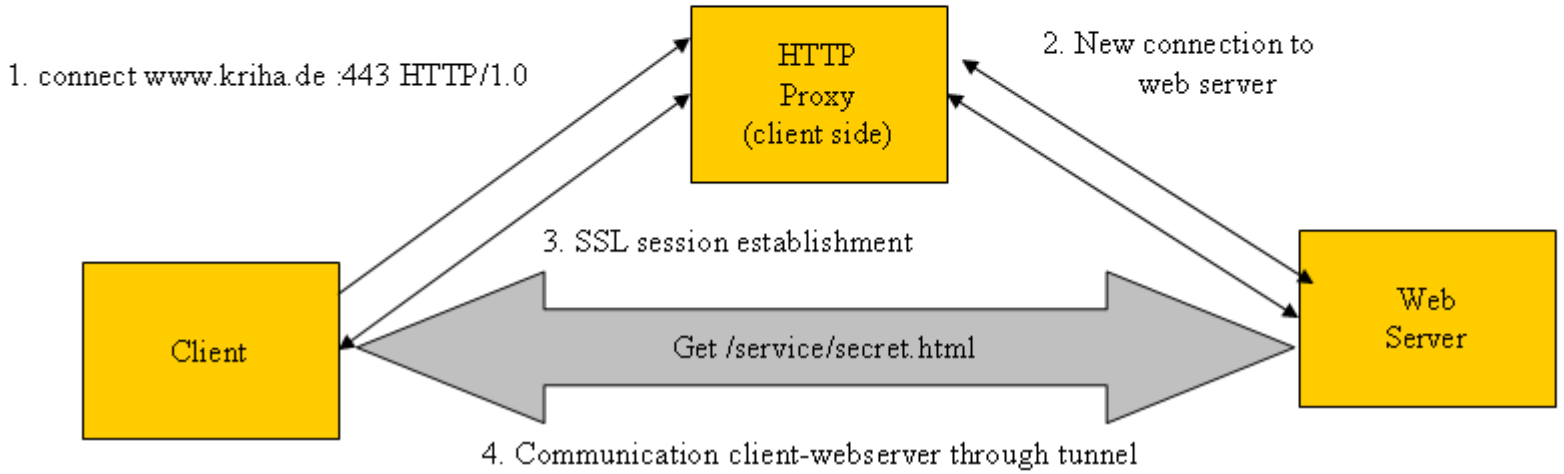
Firewall complications of the microsoft file and printer sharing services:

-if SMB access is allowed, access to ALL SMB based transaction services on this machine is provided. Each service needs to be secured individually.

-Multiple protocols can use the same SMB port, bi-directionally, and on top of this: they can use the same SMB connection. This means that every packet needs to be checked individually because it can belong to different applications or services.

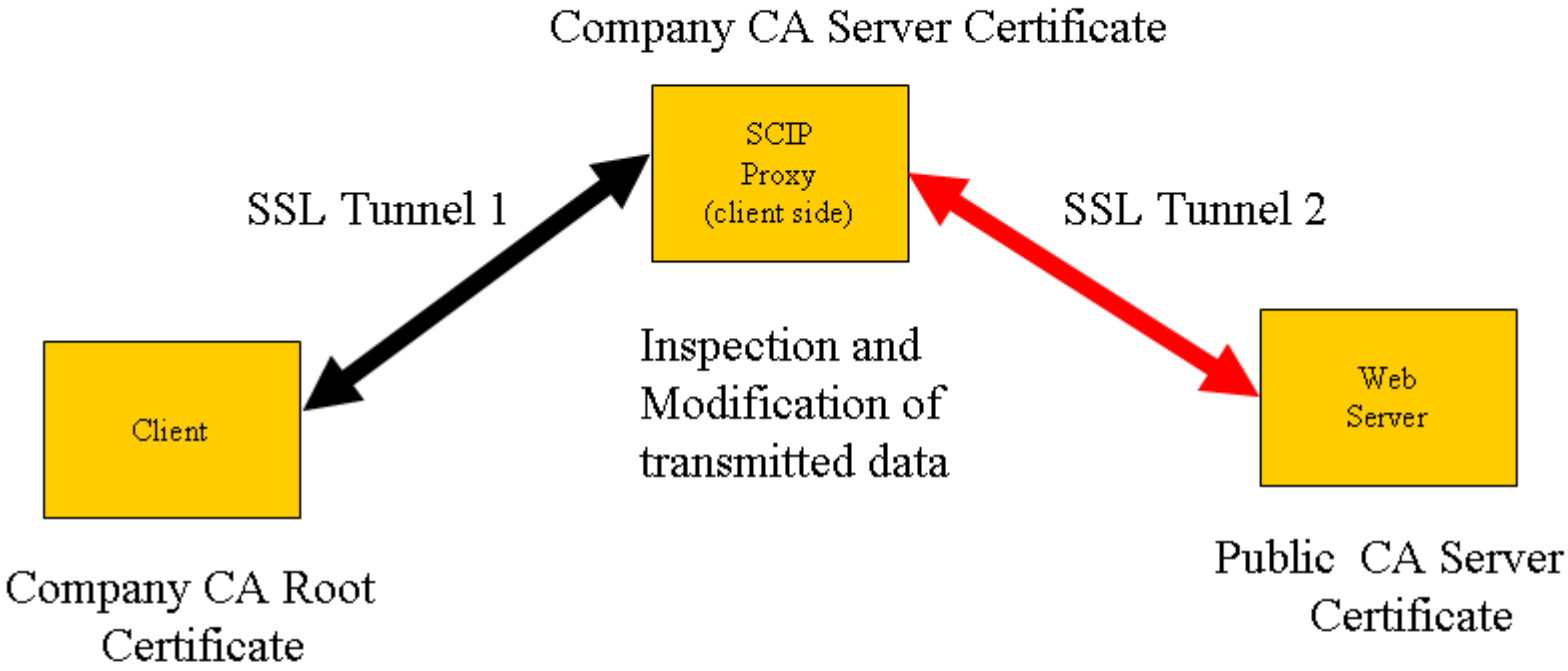
The consequence is: Do not run Netbios over TCP/IP or SMB across your firewall. Use some form of tunneling (e.g. IPSec) if necessary. For details see: Zwicky et.al. pages 350 ff.)

SSL/TLS over proxies



A proxied SSL connection first needs to establish a tcp connection from the client to the proxy. Then a special proxy request „connect“ is used to put the proxy into transparent tunneling mode. After establishing the connection between proxy and web server target, the proxy goes into tunnel mode and just forwards data back and forth. No intelligent filtering or caching is possible, even if the connection does not use integrity protection. Please note: the connect command can be used by ANY protocol, not just https. It punches a whole into your firewall, see Eric Rescorla, SSL and TLS, page 316. The use of wildcard certificates would allow intelligent filtering at the proxy at the price of creating a huge security risk.

Secure Content Inspection Proxies (SCIP)



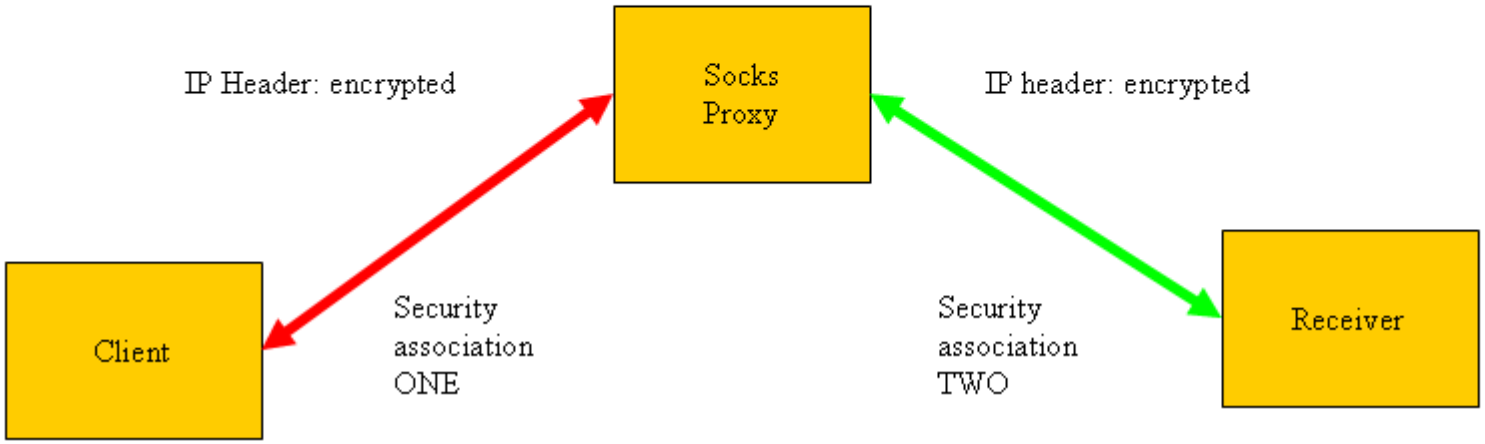
SCIP proxies terminate the Client SSL session and start a different one the the real target. The client accepts the SCIP proxy as a server because of the Root Cert. in its trust store. Companies use this design to inspect, modify or archive communication between employees and external servers (e.g. anonymizers). But the SCIP proxy now presents a huge danger for all trusted communication because on the SCIP proxy data are not protected. Some SCIP Proxies even map different clients onto one SSL SessionID for the same target. This easily confuses servers who have a hard time distinguishing the clients.

Protocol Analysis

1. Reference Integrity: The protocol must allow the client to identify a server as the one the client wanted to connect to. And it must allow a clear indication about the security properties of the connection, e.g. that SSL is required for a certain request.
2. Protocol selection: secure and non-secure versions of a service can be distinguished through differnt ports or through upward negotiation. The latter requires that the protocol has an extension mechanism in place – something many existing protocols do not have.
3. Protocol Semantics: Does the protocol have a clear means to end a connection? Otherwise truncation attacks can happen. Does it define when a connection can be resumed and when not? Does it specify when and how key material needs to be renewed or how security properties can be re-negotiated?
4. Client Authentication: Does the protocol allow or enforce certificates? Is client authentication mandatory or optional? How can certificate based authentication be mixed with e.g. password based authentication?

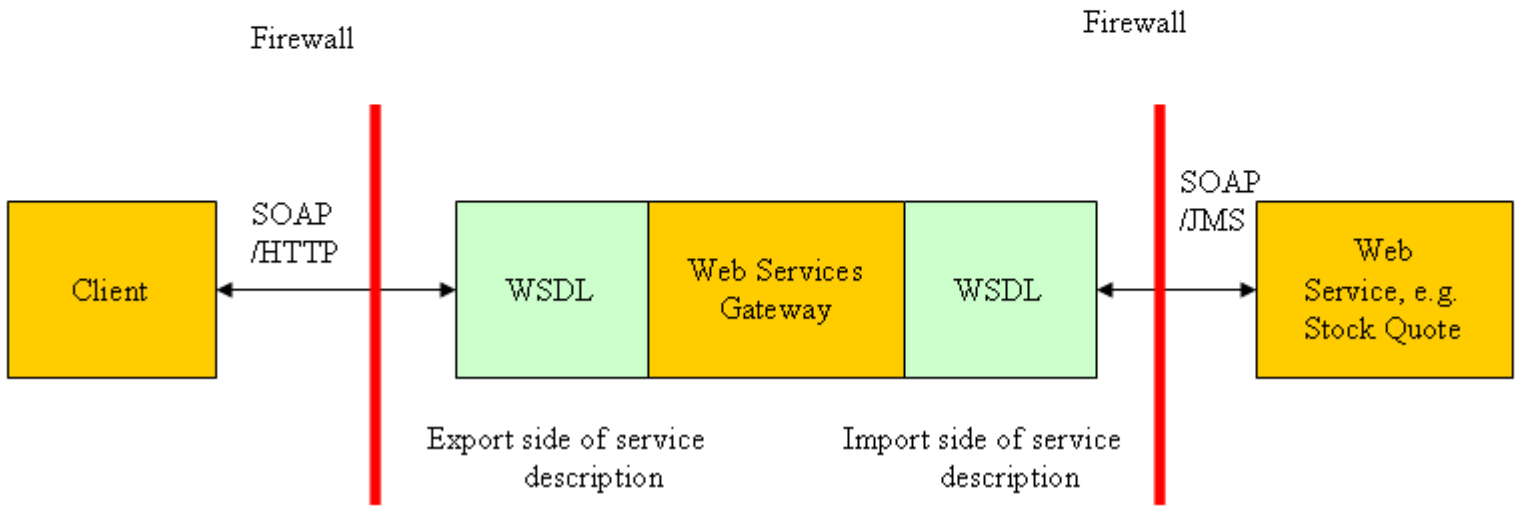
see: Eric Rescorla, SSL and TLS, pg. 230ff.

IPSec firewall properties



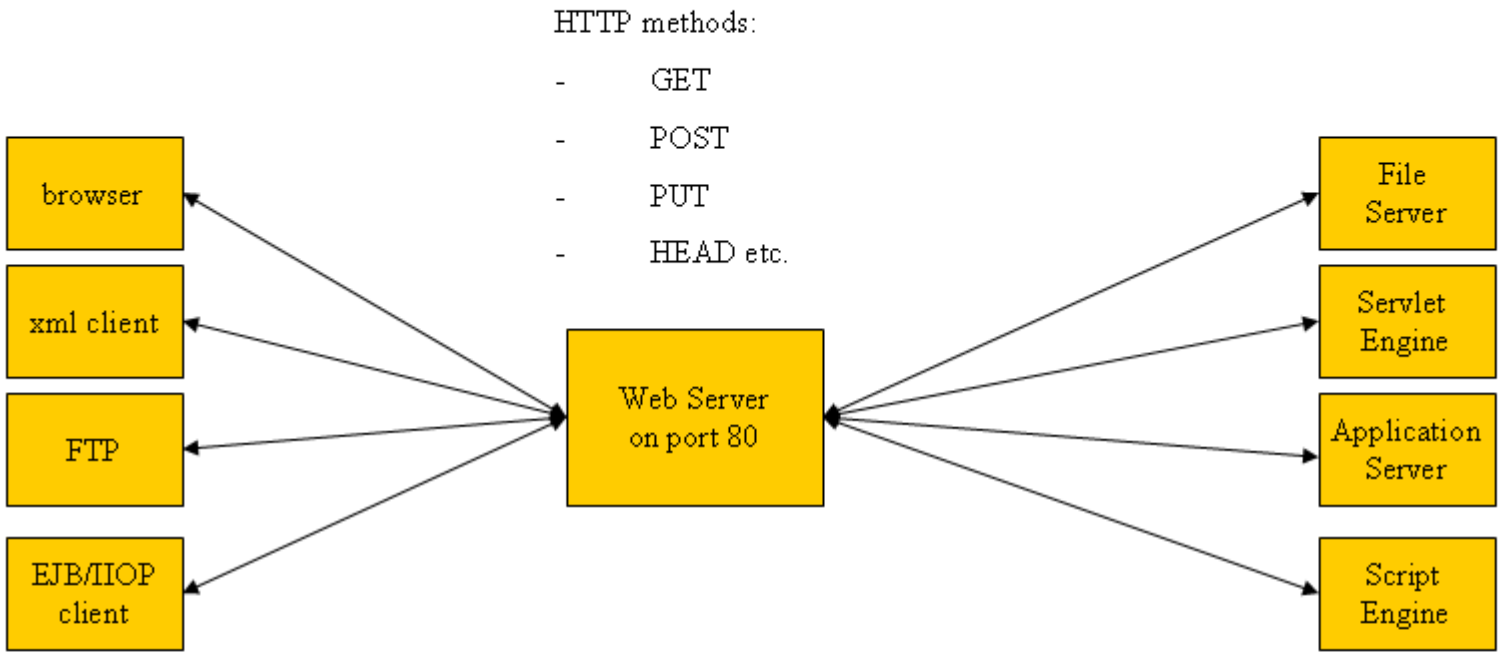
IPSec works directly as an IP level protocol. It encrypts IP header information. A proxy needs to change the header and would therefore break the integrity protection. The only alternative is to establish two security associations between client and proxy and proxy and target receiver. But this breaks end-to-end security and adds a lot of computing load. Therefore IPSec cannot reasonably be proxied. A packet filter needs to let the IPSec protocol numbers pass through. Intelligent filtering is not possible. Do not use PPTP or L2TP protocols because they are unsafe.

Web Services



Web Services over SOAP are currently seen like RPC type services. This means that arbitrary service semantics are possible, posing a problem for firewalls because all applications are mapped on port 80 http. A possible advantage of web services could be the explicit service description in form of WSDL (Web Services Description language) which would allow intelligent filtering based on the service requests. A prototype of such a gateway has been described by IBM (see resources section).

Is http also middleware?



Recently the concept of REST (Representational State Transfer Architecture has raised a lot of attention. The REST afficionados claim that the http basic methods are more in-line with the spirit of the web than the SOAP based web services which work more like RPC systems. Does the web really need an extra SOAP like protocol if every client can GET or POST any kind of information to servers already? From a security point of view http as it is used nowadays is really extremely generic middleware. With http security is tied to a resource mapping. (see Roy Fielding and Mark Baker on the RESTWiki discussion forum)

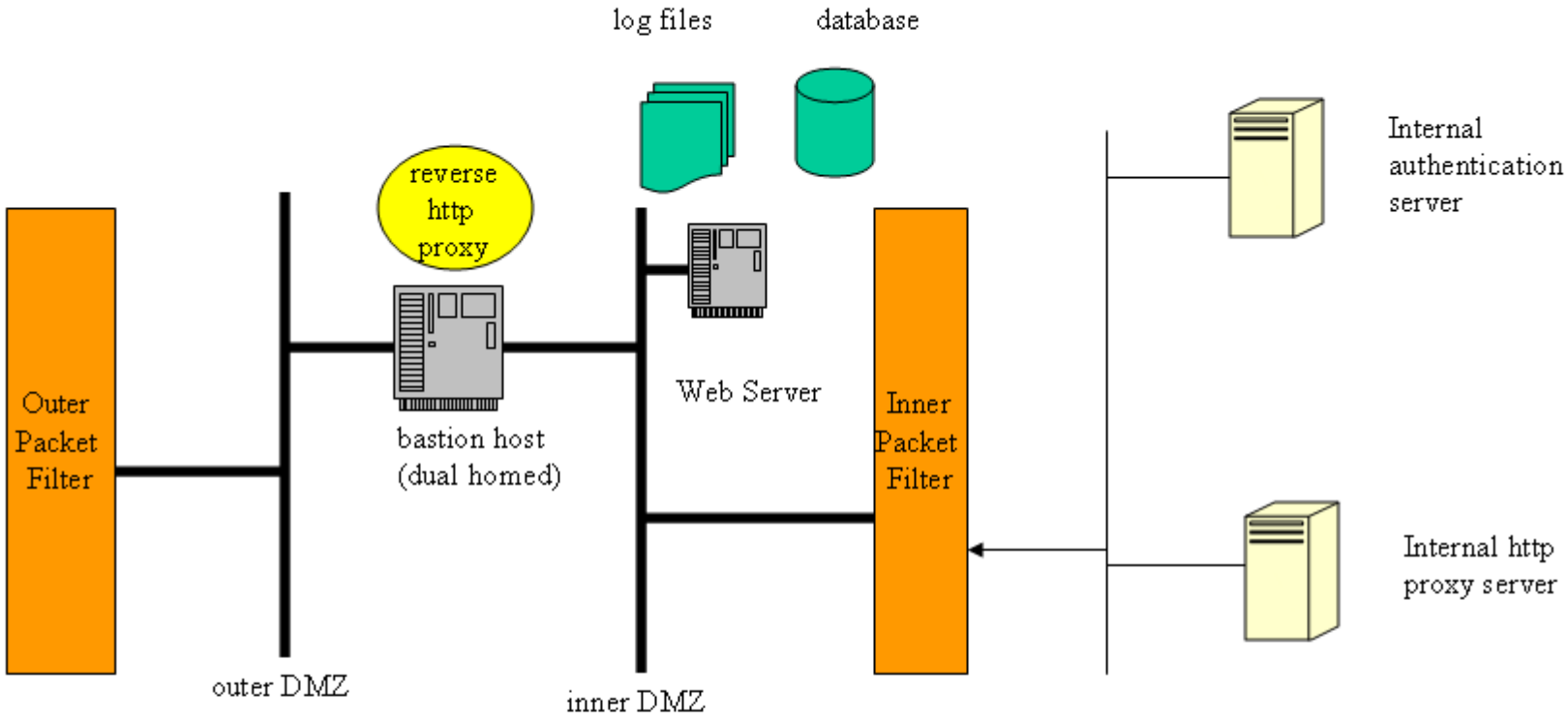
Internet Services

- WWW services
- Terminal access
- File transfer
- mail
- news
- DNS
- Multi-media applications

WWW Services

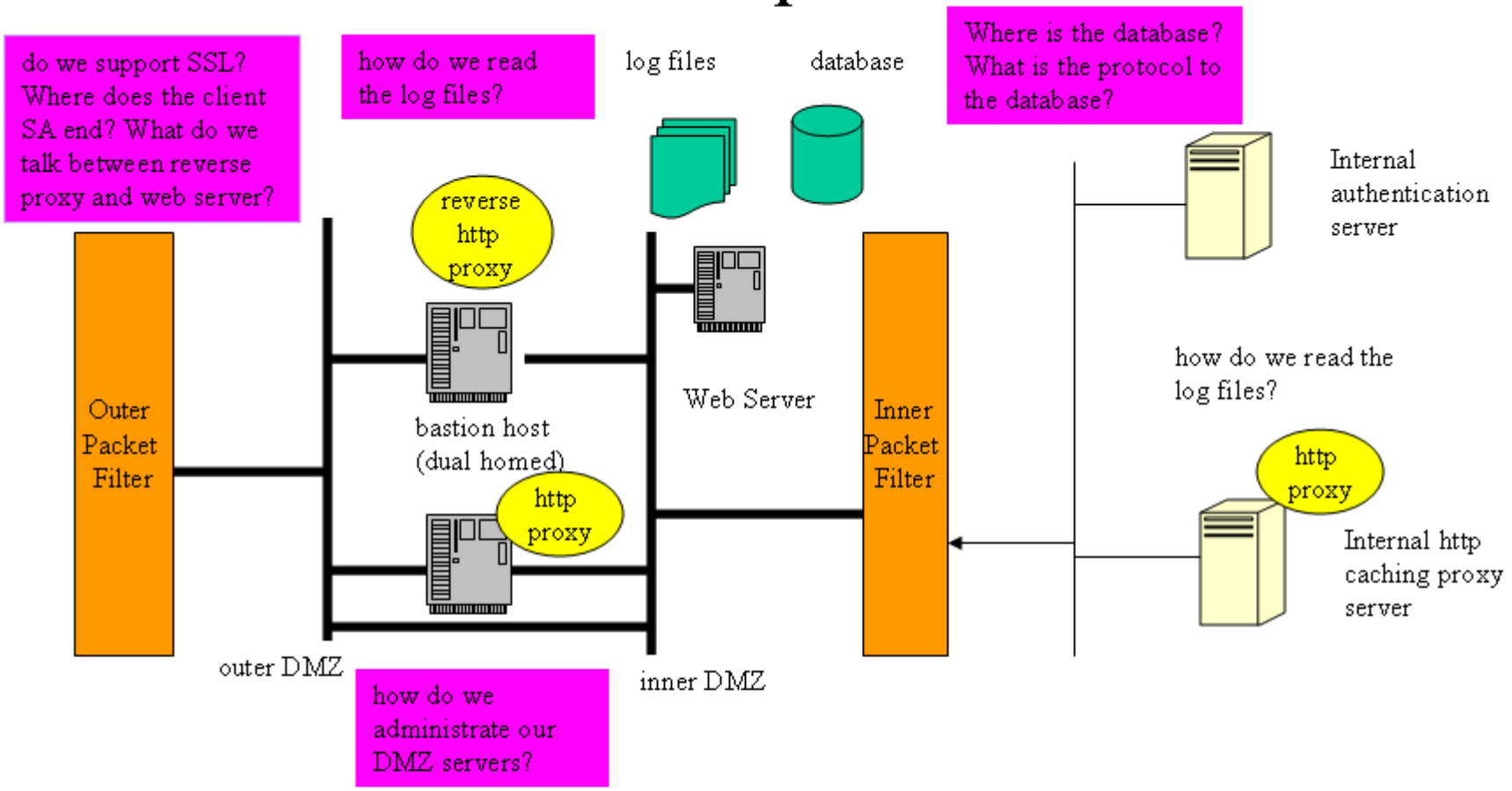
- Proxying www access
- Reverse proxies
- Caching
- HTTPS/SSL problems
- Resource control
- Web server security (semantic gap: file permissions and their consequences)

http proxy and reverse proxy



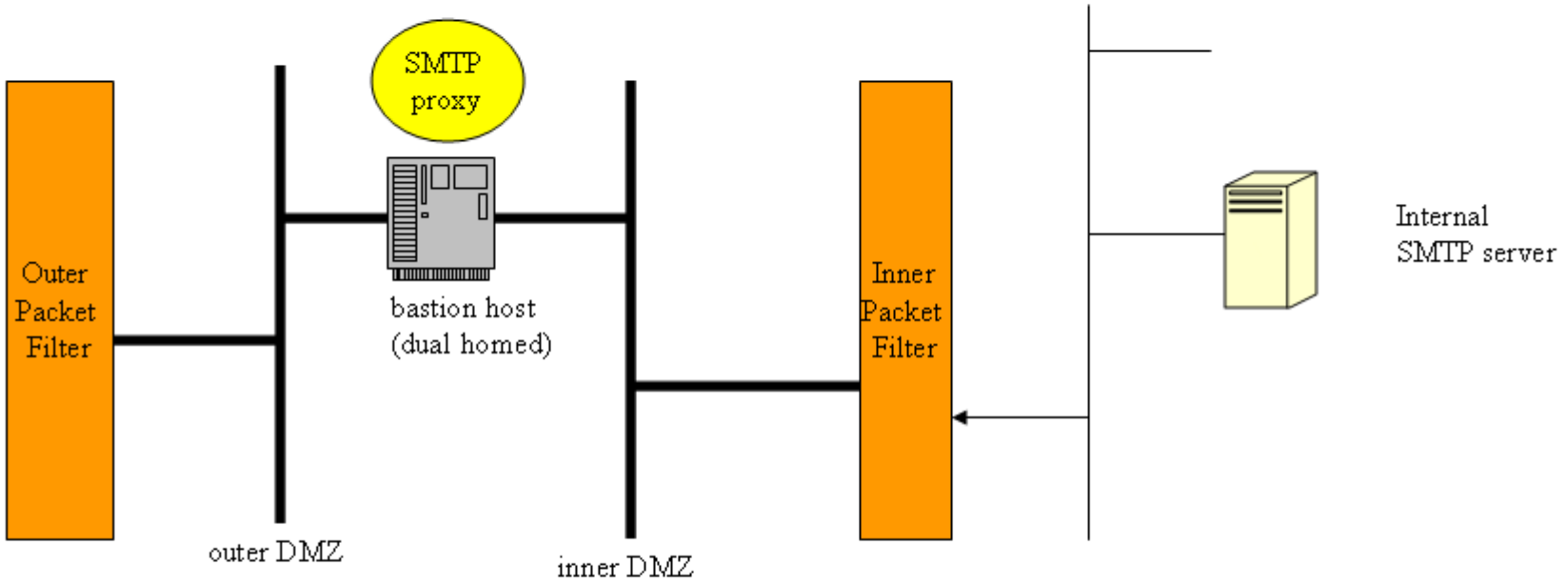
It is usefull to separate incoming http service requests for our own web server (publishing) from outgoing user requests against other web servers (browsing). Our own outgoing requests are configured for the internal proxy server which will contact servers on the internet. Incoming publishing requests are again proxied by a so called reverse proxy which looks like the real web server to external clients but which only forwards requests – after successful authentication – to our web server in the inner DMZ.

Some open issues



Even this little example shows us a lot of problems with maintaining such a web infrastructure. Only the outgoing proxy server is fairly simple because it is completely in the intranet and can use all intranet services to authenticate users etc. Another question is if we want to provide advanced filtering for viruses or illegal content on the proxy server. Finally: should we run the proxy server in the DMZ on a bastion host or should we use packet filtering – requiring a direct connection.

SMTP server and proxy



This configuration ensures that a) internal mail does not leave the intranet, b) all clients need to use the internal server which we can use for filtering/auditing etc. c) the connection to the outside will be configured correctly. The SMTP proxy on the bastion host will forward mails from outside to our internal mail server. Outgoing mail will be sent directly from the proxy on the bastion host to the destinations.

Open Issues and Discussion

1. How do you prevent internal mail from reaching the SMTP proxy? (see split DNS later)
2. Internal clients will have to be configured correctly to access the internal SMTP server
3. Do not allow relay functions in your SMTP proxy
4. What about confidentiality and integrity? SMTP does not guarantee anything here. Your users must be aware of this problem. Alternatives are PGP or S/Mime
5. This configuration make POP connections unnecessary and you should not allow it. This will make users angry which would like to access external mailboxes from behind the firewall.
6. Instead of a full SMTP server serving as a proxy, a simple SMAP/SMAPD proxy could be used which does not allow a direct connection to a full-blown SMTP server from outside.

Due to its store-and-forware architecture SMTP is easy to proxy. Packet filtering is therefor not recommended.

Remote Terminal Access

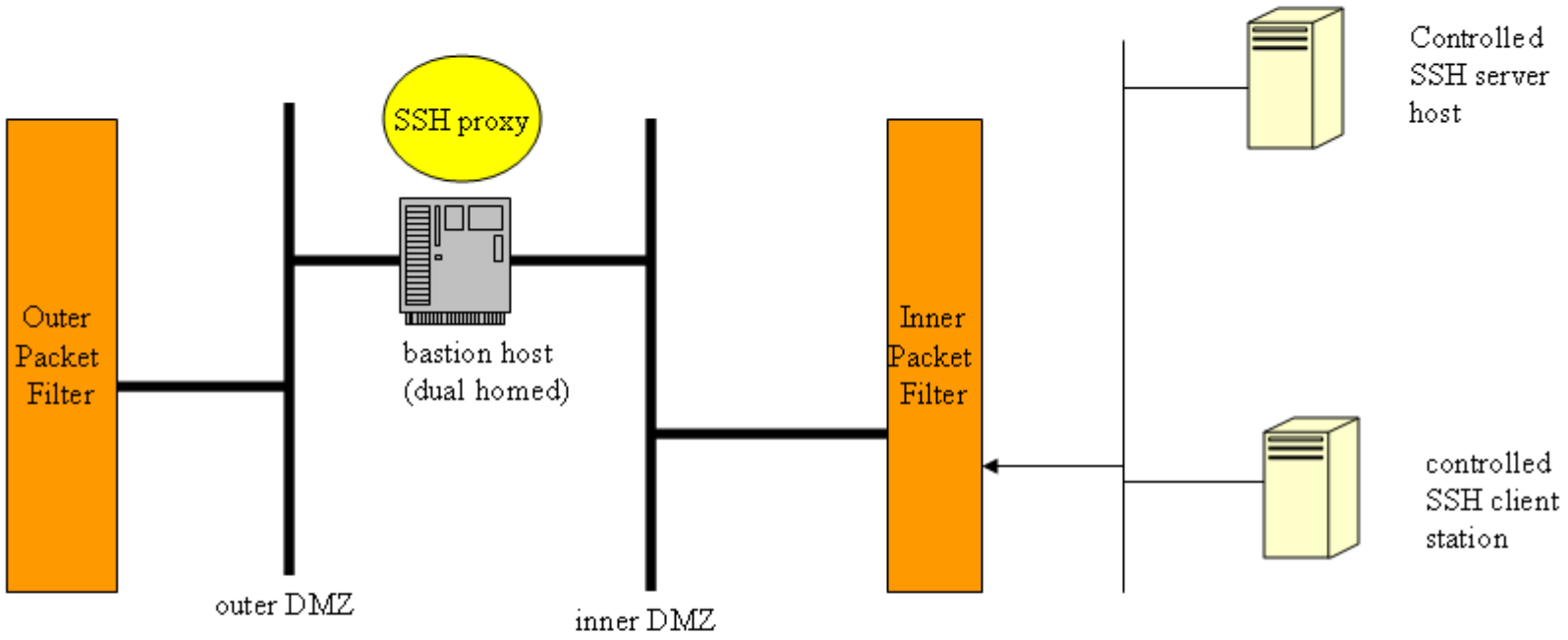
We will need to provide remote terminal access for two reasons:

a) to allow internal users a remote login on external machines

b) to be able to administrate our own DMZ machines securely (login, file copy etc.)

If we decide to use SSH we can just outlaw regular telnet (no security) and also ftp from the intranet to our DMZ machines. We will probably allow ftp outgoing to the internet, either through a proxy ftp kit or by using passive ftp from the browsers.

SSH support



Because of the dangerous port forwarding feature outgoing SSH connections will only be allowed from controlled client stations (clients cannot install their own SSH clients wich might have port forwarding enabled). Incoming SSH will only be allowed (if at all) to a small number of controlled servers. Client and server authentication will use RSA public keys.

Security Analysis of previous architecture

- Where are the weakest links?
- Is the principle of „least privilege“ enforced?
- In which places is a positive user behavior necessary, required? What are the consequences of misconduct?
- Is „defense in depth“ implemented?
- Is the architecture still simple, understandable and maintainable?
- Does the architecture provide a choke point or are there ways around it?
- Is the fail-save-stance implemented?
- Is a diversity of defense implemented?

These are the core questions from Zwicky/Cooper/Chapman about firewall architecture.(pg. 701 ff.)

Next Session:

- How to secure web applications (content management, portals etc.)
- Application server security principles and mechanisms
- Web Services Security

You can find a good introduction to application server security on www.redbooks.ibm.com (search for „websphere“ and „security“.)

Resources (1)

- Elisabeth Zwicky et.al, Building Internet Firewalls. Invaluable book which covers a lot of services and their problems, besides firewall architectures.
- Webservices specification and roadmap: http://www.verisign.com/wss/wss.pdf http://www.verisign.com/wss/architectureRoadmap.pdf
- Chandra Venkatapathy, Simon Holdsworth, A proxy for Web Services, www-106.ibm.com/developerworks/library/ws-gateway
- Roy Fielding et.al, Principled Design of the modern Web Architecture

Resources (2)

- Guidelines for Writing RFC Text on Security Considerations, Author(s) : E. Rescorla, B. Korver, http://www.ietf.org/internet-drafts/draft-rescorla-sec-cons-05.txt Covers most security issues related to internet services and protocols. A must read for everybody who needs to write or judge an internet service.
- David Mertz, Lotus Domino SMTP Server in DMZ (developerworks)