Summer school CNRS-ENSTB "mobile learning" 2009

Role of network components in a context aware architecture

Alain OTTAVI

Orange Labs

Table of content

1 Introduction

Network sees all the streams

- All users' streams go through the network. Good !
- Therefore, we have access to network users informations.

How to store them, to analyze them to provide personalized services to mobile and work-out-of-office users ?

Presentation purpose: principles, pros et limits, methods and constraints of streams network capture and user activity traces managment in order to provide personalized services in core network.

/!\ The architectures in this document are general and not limited to mobile and work-out-of-office situations. However, this presentation put the accent on interesting use case for these two types of situation.

/!\ It's not a network course: The architectures in this document demonstrate the interest of core network for applications in mobile and work-out-of-office situations. Therefore the presentation will focus on interesting high-level caracteristics for designing "mobile learning" applications.

2 Streams interception and analyze

Interception: cons and constraints

- Cons:
  - Impossible to "see" encrypted streams between clients and servers in core network.
  - The streams only designed for client-side execution are difficult to understand, often not understandable (*e.g.* Ajax streams).
  - Application behavior could change. Application code modification could be necessary.
  - Applications secured by certificates : possible modification of the configuration of the client and/or the server and/or intermediate equipments.

- Constraints:
  - Streams Interception, analyse and redirection:

    - Matching application pattern ? It's always changing!!!

    - Adding a new machine (configuration, estimated load), with a specific software (configuration, signatures database of applications and protocoles matching patterns updates).

  - Important « privacy » constraints (user information, access and modification right to personal data, conservation and deletion of these data ([CNIL], [G29])).

Interception: principle et pros

- Principle: core network component:
  - Stream rupture (explicit or transparent proxy, probe or DPI in bridge mode),
  - interception/redirection (WCCP, Policy Based Routing, transparent proxy, L4/L7 switch).
- Pros: all users streams are accessibles whatever theirs origin/destination (client/server or server/client).
  - Apply common rules to every stream : authentication, security, all types of filters. Minimal or null client-side configuration.
  - Build a map of users interactions between each others.
  - Add profile informations to every stream.
  - Provide personalized services depending on user profile and/or machines/clients caracteristics used for connection to the different services.
  - Design easily personalized applications by changing services or applications calls order ordering *(e.g.* ACLs).
  - Add "load-balancing" and "fail-over" for different applications servers for having high availability and greater scalability applications.

Principal components for capture in core network

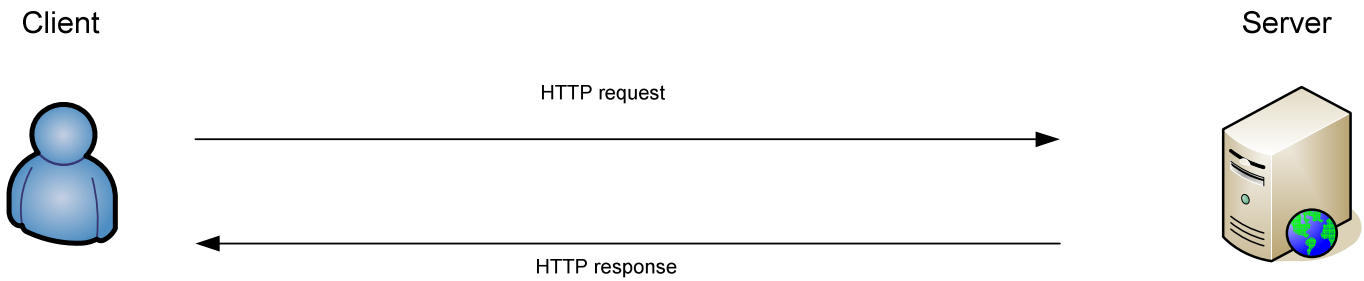
/!\ Firewall, IDS/IPS are dedicated to network protection only and therefore are excluded from this presentation.

Component	Origin roles	Today supplementary roles
Probe	Network traces capture in bridge mode, connection reinitialization.	Application identification ("Deep Packet Inspection"), Shaping (bandwith limitation for each applicatif/protocol, QoS), redirection, map of streams.
Router /(Switch)	Packets routing.	ACL application (bandwith limitation, filter user and time, content filtering and content protection …), conditional redirection (protocol, @IP source and destination, port source and destination), DoS protection.
Proxy	HTTP and FTP bandwith saving, security and acceleration ("reverse-proxy")	- HTTP connections optimization, ACL application on protocols (time, user, protocol nature), add services (*e.g.* ICAP), stream enrichement (add header), redirection to services, TCP protocol tunneling, SSL connection control, SSL terminaison point and SSL acceleration, "on the fly" application compression.
WAF	Web sites protection and integrity against all attacks types.	User behavior detection (basic), SSL acceleration.

All these components are now able to :

- to authenticate a user against a directory, a database or a Radius.
- To apply an ACL (Access Control List) on every stream.

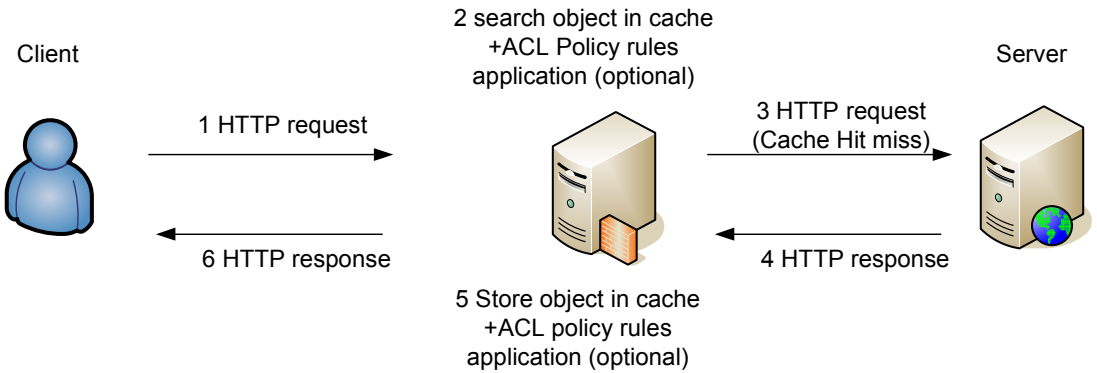
HTTP: connection without interception



HTTP Level	GET http://server:port/url Host : server
TCP Level	(Client @IP, Client source port) >> (Server @IP, Server Port)

HTTP: interception with "explicit" proxy

"End-user" level



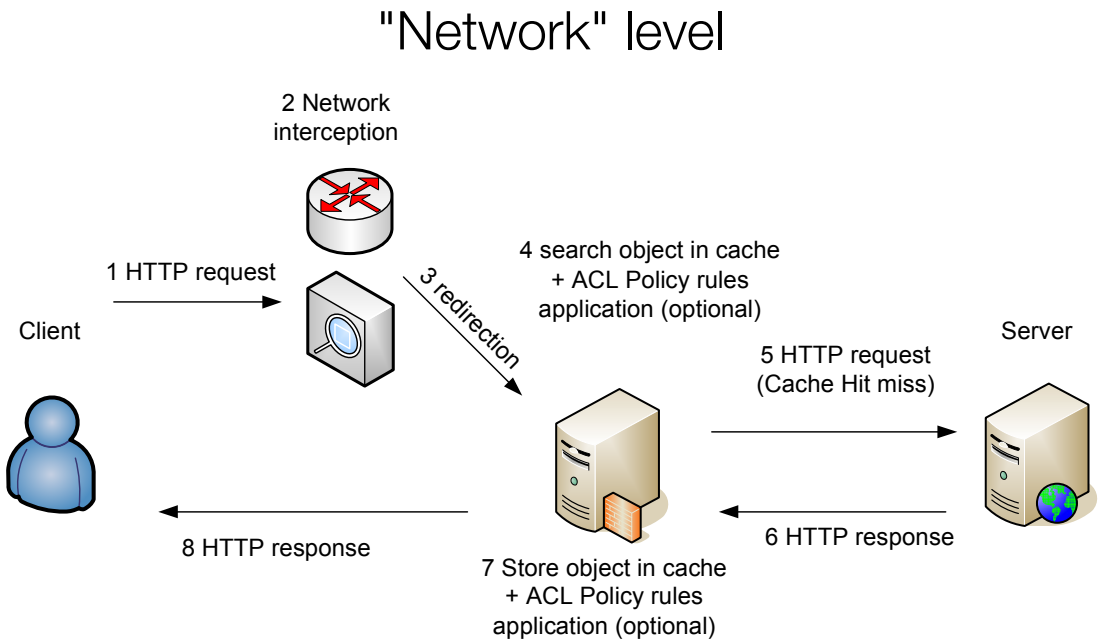
HTTP Level	GET http://server:port/url HTTP/1.x Host : server	GET /url HTTP/1.x Host : server
TCP Level	(Client @IP, Client source port) >> (Proxy @IP, Proxy Port)	(Proxy @IP, Proxy source port) >> (Server @IP, Server Port)

The client is "proxy aware" and makes an explicit connection to the proxy for every request.

Two ways for "client-side" configuration:

- By hand, giving proxy URL, or script .pac (Proxy Auto-Config) URL.
- Loaded by a startup script *(e.g.* during the user authentication).

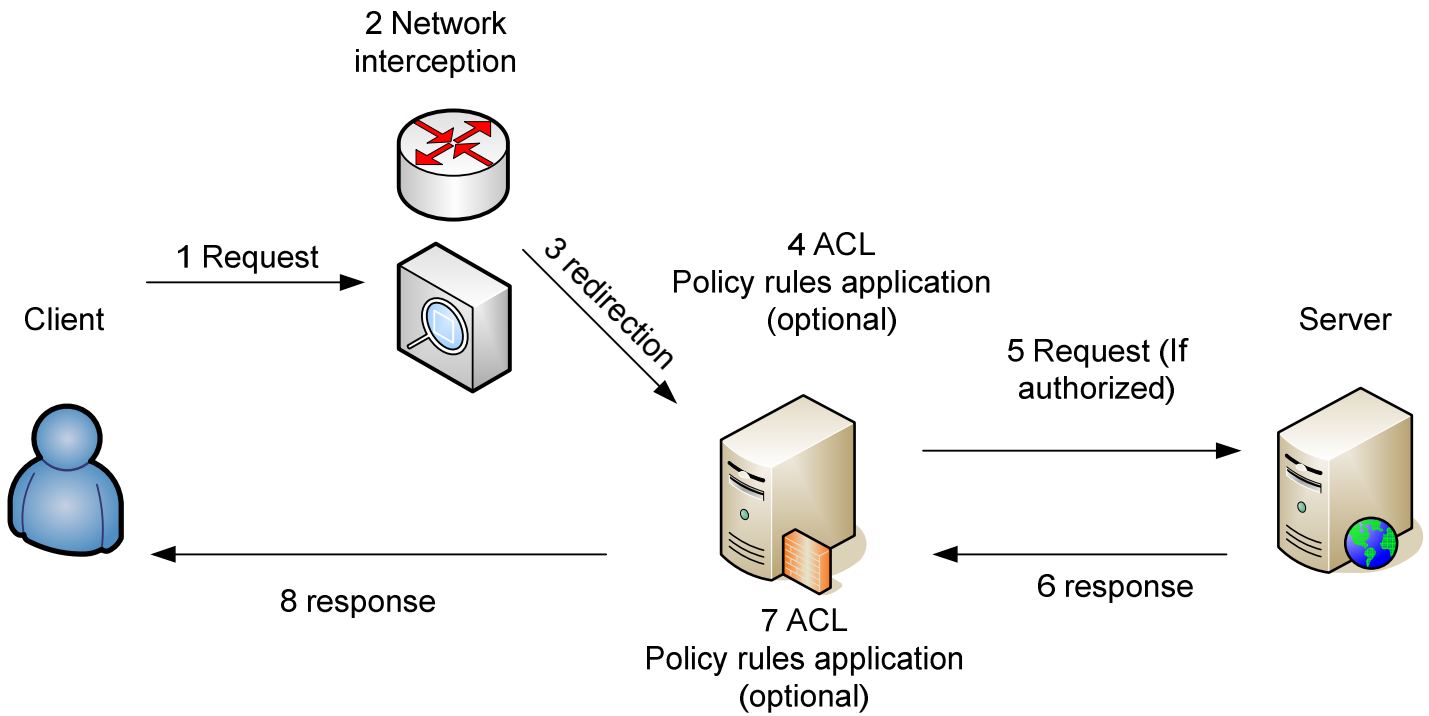
HTTP: interception with "transparent" proxy



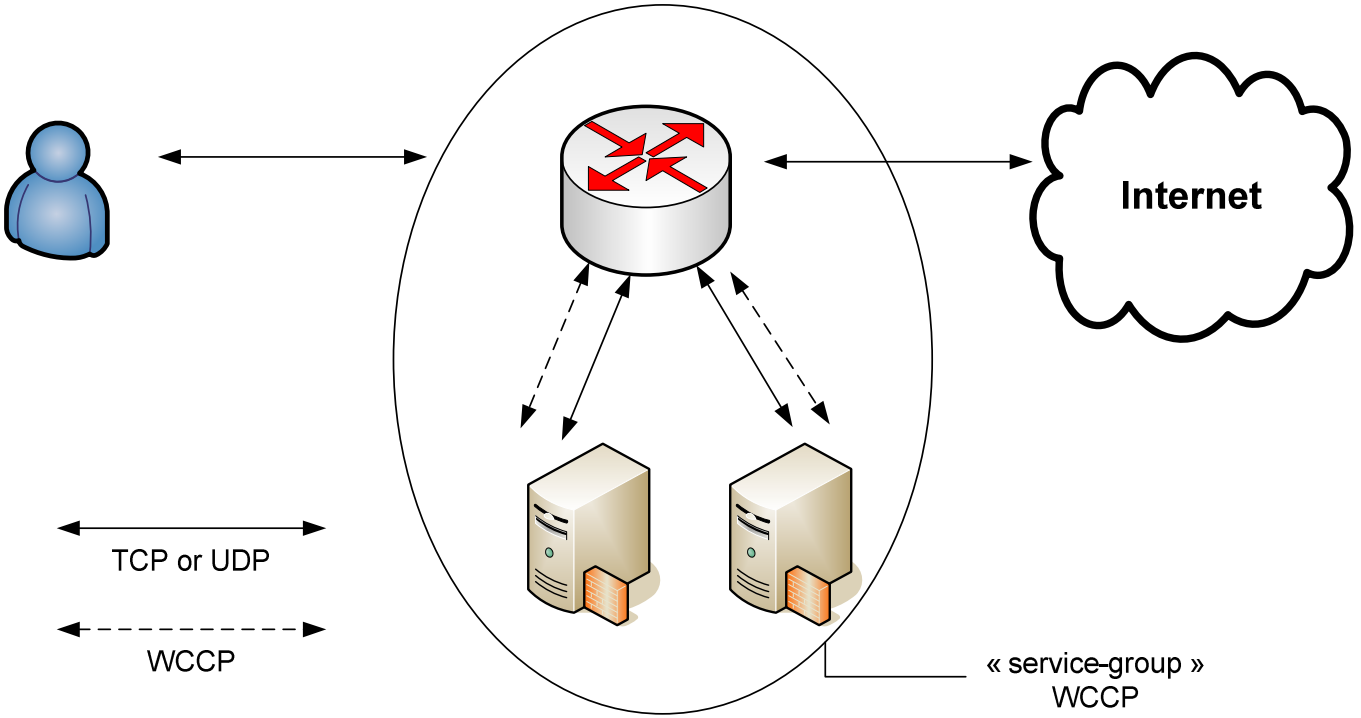
HTTP Level	GET http://server:port/url HTTP/1.x Host : server	GET /url HTTP/1.x Host : server
TCP Level	(Client @IP, Client source port) >> (Server @IP, Server Port)	(Proxy @IP, Proxy source port) >> (Server @IP, Server Port)

- The client is not "proxy aware" and thinks its request go directly to the Web Server.
- No "client-side" configuration.
- The stream is intercepted and redirected to an other network component (*e.g.* proxy).
- The network interceptor could be a probe or a proxy in bridge mode, a router configured with PBR (Policy Based Routing) or WCCP, a L4/L7 switch.

General case: interception with "transparent" proxy



General case: WCCP with compatibles router and proxys



Principle : redirection to a proxy set of a TCP or UDP streams which comes on a router interface.

Important pros:

- Works for every TCP or UDP stream (**with known port number**).
- Load balancing between proxys, with "heart-beat" mechanism,

Important cons:

-- Works only on a little number of simultaneous ports to define before the "service-group" settings.
-- Router load.
-- Works only with compatibles router and proxys.

3 Value Added Services in core network

Value Added Service in core network

After interception, the stream could be redirected on others equipments (depending from the interception component capabilities). The following cases will be introduced:

- DPI →Specific component.
- (DPI/Router/[L4/L7 Switch]/WAF/Proxy) →Proxy
- Proxy →Specific component or service.
- Proxy →Web Server with URL rewriting for HTTP protocol.
- Proxy with stream computing by local specific application for HTTP protocol.
- Proxy →iCAP Server for HTTP/FTP protocols.

Value Added Service in core network

- DPI →Specific component

Ex : Allot (DPI) et Aladdin (security specialized editor) partnership for content protection / content filtering on HTTP/FTP/SMTP/POP contents. The DPI and the component are directly connected through the MAC Layer.

- (DPI/Router/[L4/L7 Switch]/WAF/Proxy) →Proxy

- DPI → Proxy: example by Optenet [OPTENET] and it's CCOTTA component. Also studied by others DPI editor.

- Router →Proxy: WCCP (compatible Router and Proxys) or PBR (Policy Based Routing).

- Switch L4/L7 → Proxy: PBR.

- WAF → Proxy or Proxy → Proxy obvious: the two network components were created to understand HTTP and to be linked together.

- Proxy →Specific component or service

- Content protection / content filtering applications. Often a specific partnership between a equipment maker and an editor, with frequent usage of non standardized protocols or extensions. Ex : BlueCoat (Proxy Leader) partnerships with AntiVirus or parental control editors.

Value Added Service in core network

- Proxy → Web Server with URL rewriting for HTTP protocol

- A proxy in "reverse-proxy" mode (the Client thinks to be connected to the Web Server. In fact he is connected to the proxy which do all the needed URL translations).

- An Apache Server with mod_rewrite module for URL rewriting. Others Web Server have similar capabilities.

- Proxy with stream computing by local specific application for HTTP protocol.

- Use of "squid redirector": SQUID sends on its "standard output" le 4 uplet (URI, Client/FQDN, user, method) which could be computed by a program on the computer. This program sends on it's "standard output" the final URI.

- Proxy →iCAP Server for HTTP/FTP protocols

iCAP: "Internet Content Adaptation Protocol": lightweight protocol for executing a RPC ("Remote Procedure Call") on HTTP messages".

- iCAP Clients: BlueCoat, NetCache (BlueCoat), Squid (GPL),

- iCAP Servers: Webwasher (McAfee), BlueCoat AV, IBM Datapower, Netasq, Orange iCAP Server and differents services (see below).

Value Added Service in core network: iCAP protocol

- Proxy →iCAP Server for HTTP/FTP protocols

- "Idea behind iCAP": following general procedure for adding services in core network:

1 – Stream interception (*e.g.* with redirection on a proxy) and redirection to the specific component with the desired service after user authentication (against a directory, a Radius database, a database, a basic HTTP authentication…) for service personalization.

2 – Add a load-balancer in front of these components to ensure service scalability.

3 – Add the others components with the needed services.

4 – Define rules (ACL and specific call order) for service access for each user.

It's an iCAP architecture (RFC 3507) !!

This architecture is based on:

- A iCAP Client (a proxy in general) which manages user authentication, call order of the different iCAP services with ACLs (service composition), stores Servers iCAP responses and communication with the internet Client.

- A iCAP Server with the desired application.

There are two modes on a iCAP service:

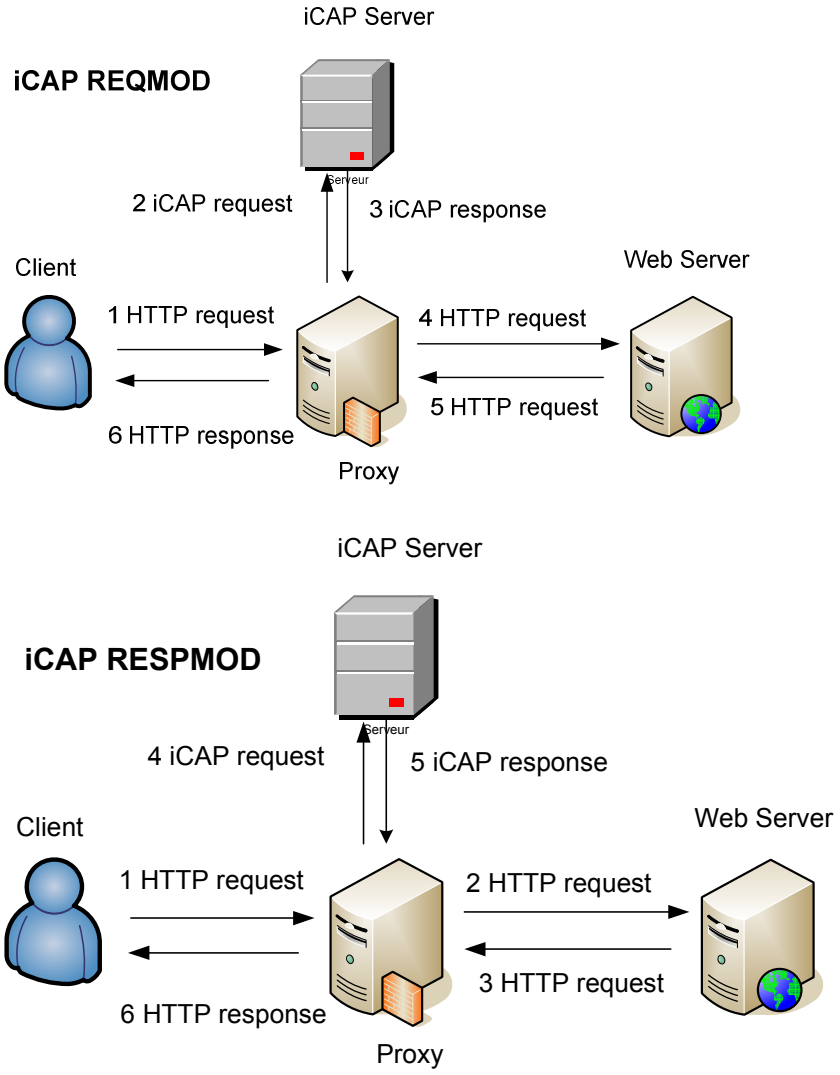
1 – **request** modification by the iCAP Server: "reqmod",

2 – **response** modification by the iCAP Server: "respmod".

In these two modes, the iCAP Server can be called before searching in the cache of the proxy or after. These four possibilities are called "vector-point". Two of them allow service personalization function of the user identity.

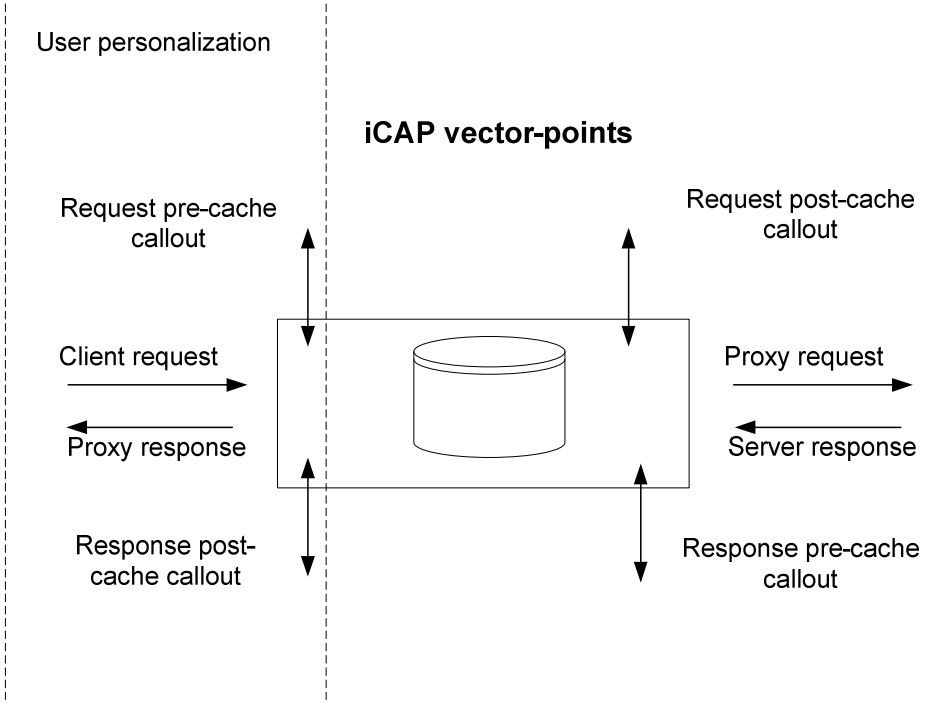
Value Added Service in core network: iCAP protocol

The two modes :



Value Added Service in core network: iCAP protocol

The four "vector-points" :



Value Added Service in core network: iCAP protocol

- Some typical reqmod services:
  - – Parental/enterprise control by authorized URLs liste ("white list") and/or forbidden one ("black list").
  - – Services list authorized by user, function of his identity.
  - – LEP (Liberty Enabled Proxy) SSO (Single Sign On) (Orange iCAP service)
- Some typical respmod services:
  - - Parental control by content page filtering.
  - – Web Page translation "on the fly".
  - – Web Server Response compression for bandwith saving (important in the mobile world) (Orange iCAP service).
  - –Dangerous,forbidden or not "editor-certified" components deletion,in a Web Page.
  - – Stylesheet (CSS) application according to specific rules (Orange iCAP service).
  - – Add a frame with different items at the top of every page. This frame can be personalized for each user (Orange iCAP service).
  - – LEP (Liberty Enabled Proxy) SSO (Single Sign On) (Orange iCAP service).
- Some ideas for iCAP services for "mobile learning"
  - – Selection of the right course perzonalised by user profile (language, previous courses): REQMOD.
  - – Page adaptation for mobile caracteristics and network capabilities: RESPMOD.
  - – Get a user' mobile or WiFi geolocalization: REQMOD.
  - – List of the students connected to the different learning applications, with their status: RESPMOD.

Value Added Service in core network: conclusions

- "Binary" protocols have fewer possibilities to add or modify theirs streams than "text" protocols. "Binary" protocols additions / modifications are limited in general to:

- Available ACLs for every protocol (bandwith and object size limitation, filtrage @IP source et @IP destination, port source, port destination, … depending on the user' identity)
- content protection / content filtering applications (AntiVirus, certified components). These applications are often the result of a specific partnership between a equipment maker and an editor, with frequent usage of non standardized protocols or extensions.

- HTTP is the protocol which allows the best possibilities for addition / modifications:

- headers (request and response ones),
- request,
- Object send back in response

In addition to the available ACLs for every protocol.

- ACLs allow services composition by ordering theirs callings.

4 Log creation and automatic centralization

Log file creation

- All Application Servers and many network components (proxy, WAF, directory) create log files with respect to "Common Log" ([LOGW3C]) et "extended" W3C log format.
- Creation and centralization tools were created to facilitate log files managment:
  - –Syslogd ([SYSLOG]) for C/C++ applications mainly.
  - – Log4J ([LOG4J]) for Java applications (org.apache.log4j package).

  These two toolkits, because of their power:

  - were ported under others languages,
  - have many companions programs for adding new functionalities.

Log files creation and managment: Syslogd ([SYSLOG])

Syslog uses UNIX syslogd daemon to define for each program the log files with the following format:

<date> <Machine DNS Name> <Service.criticity level> <Message>

With the following parameters in syslogd file (syslog.conf):

- services and criticity levels to log.

- Log files name.

- The machines (DNS name or IP addresses) which receives the log files.

- Log files rotation parameters: by size and time mainly, compression, number of log files before deletion.

Syslogd allows the centralization of numerous big log files on a dedicated computer.

Syslogd allows a great liberty to organize log files and a "clean" system administration.

Log files creation and managment: Log4J ([LOG4J])

Hierarchical and versatile log file creation thanks to "loggers".

- Six levels of log : trace < debug < info < warn < error < fatal (by ascending order of criticity/priority).

- Hierarchical "logger" organisation: all the application "loggers" inherit from the "root" "logger". All the "loggers" are defined as a tree, with a syntax similar to the DNS ones.

- Definition of a default priority level p : only "log" messages whose priority q>=p.

- Definition of the traces' destination ("Appenders"): type console (standard output or standard error), files, graphic component, distant socket Server, JMS message or syslog daemon on a remote UNIX system.

- Definition of the presentation formats ("Layers"): log line fields definition.

- Definition of the parameters in a configuration file –text format- with parameter=valeur syntax or XML format .

- A "logger" inherits by default from a parent "logger" its default priority if its own default priority is not defined and "additives" "appenders" (children's "appenders" are added to parents' "appenders"). These two inheritances can be desactivated for each child.

5 Example of platform for external services calls, user activity traces centralization and computing

Platform: needed functionalities

- Needed functionalities of the centralized platform' log files managment

  – Openess: standard and normalized protocols, widespread in the industry. Source code platform must be freely available for Orange.

  – Mobile geolocalization by Orange mobile network: first external service to implement*.

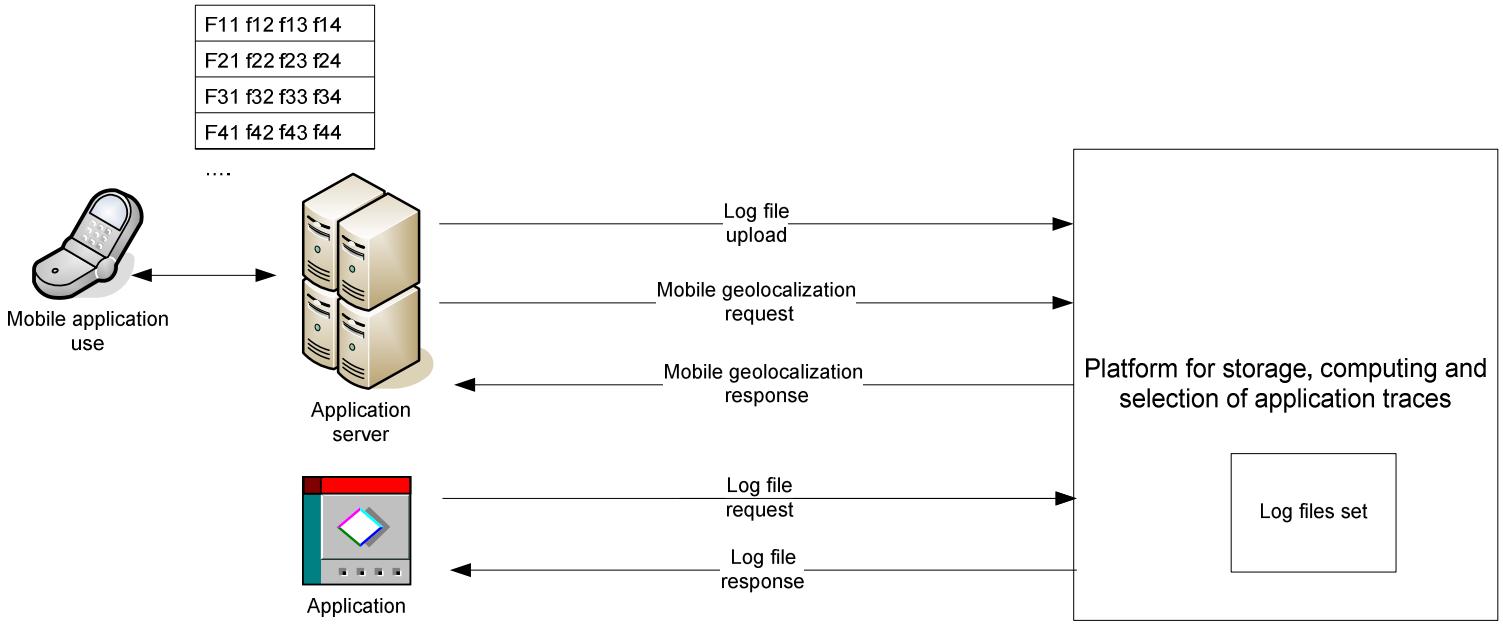
  – Respect of the "Informatique et Libertés" law: log files will often include personal data ("données à caractère personnel"). ([CNIL], [G29])

  – The platform will be accessed from Internet: the connection to the platform is made through public APIs. This should facilitate a coherent access rights policy to the platform.

  *For this experimentation, mobile geolocalization is supplied by a platform dedicated to experimentations. This platform supports many geolocalization modes (IP, Orange mobile network, A-GPS, Orange WiFi …) and is accessible by XML/HTTPS request.

  /!\ This platform is a "PoC" (Proof of Concept) to validate the pros and the limits of a centralized platform' log files managment.

Platform: input/ouput streams



The platform will receive:

- Application' text format log files,

- Mobile geolocalization with request and response storage,

- Requests on different log files or geolocalization files.

The platform will send responses in text, CVS or XML format to a request on the different log files it has:

- Log files send by applications,

- mobiles geolocalization coordinates in response to applications' requests.

Platform: software and network protocol choices

- First choice : a RDBM for the storage of log files, mobile geolocalization requests and responses. This allows great performances and rich functionalities (inheritance, trigger, …).
- Second choice : HTTP becomes **THE** universal protocol. To send files and select a parameter set, the association **HTTP form + basic method POST** is very efficient. This choice removes method GET limitations ("clear text" informations in the Web Server' log files, request size limit …) and avoid WebServices defaults ("heavy" protocol, precise interface definition of the services…).
- Third choice : only GPL softwares for the platform with a good reputation for stability, normalization and openess . The final choice for the platform was:

  - **Linux**, **Tomcat** and **PostgreSQL**

Platform: log files choice and privacy rules

- Log file = (who, when, what) generally in a text file. In order to simplify, we have choosen "Common Log" format from W3C consortium ([LOGW3C]).
  - **remotehost, rfc931, authuser, [date], "request", status, bytes.**

Simple, but powerful and quasi-universal.

- Privacy policy choosen:

"**Applications** create log files or mobile geolocalization if they have the adequate access right. Others applications (eventually the same) can read these log files or mobile geolocalization if they have the adequate access right".

These access right are set by the platform administrator. Two access right are defined:

READ: **the application** can **only read** log files records from the database tables.

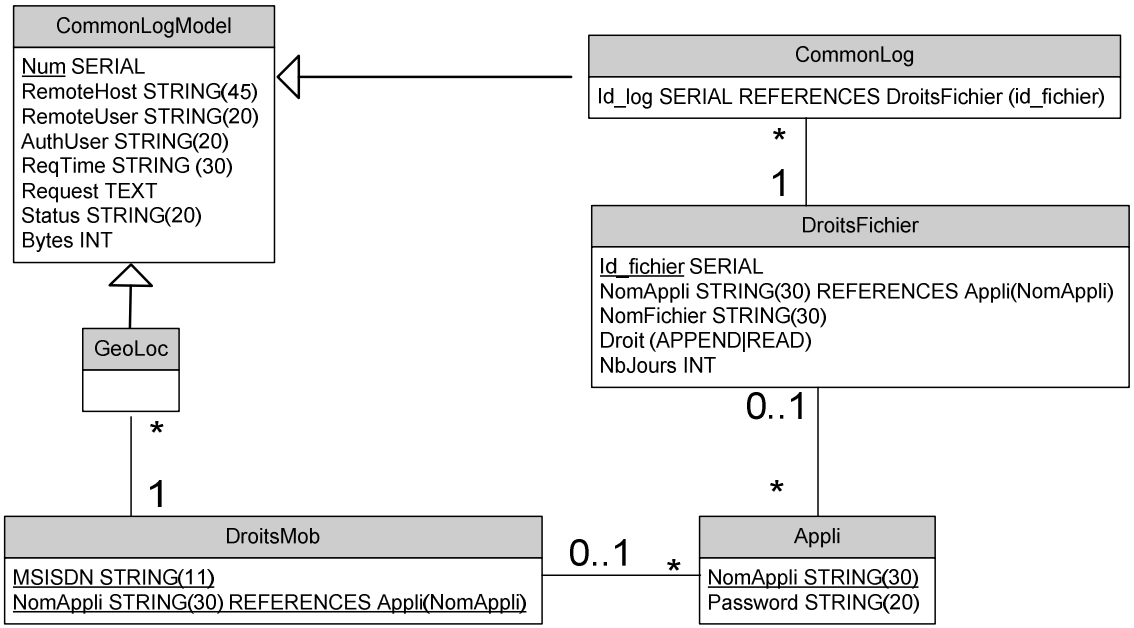
APPEND: **the application** can **read** from the database tables or **add** log files records to the database tables.

No **modification** or **suppression** access right are defined for the applications. It's the administrator privilege.

READ access right doesn't exist for mobile geolocalization. Only **applications** with the right to access to the geolocalization of a particular mobile (*i.e.* access "APPEND") can access to these informations.

No rights, no access

Platform: database scheme



- Remarks :

  –CommonLog and GeoLoc tables inherit from CommonLogModel: queries can select these three tables the way they want.

  – PostgreSQL TEXT function to store "Request" field whatever the size of the request.

  – Almost all the fields are "string": this allow to store any type of informations. The signification of each field is under control of the applications which use them (They are **theirs** user activity traces after all and they have a **signification only for these applications**).

Platform: some conclusions

This centralized plaform has demonstrated many strong points for Privacy managment. However Privacy legal obligations requires a global and coherent response by the enterprise. This could be only done by coordinating all the involved people: CIL ("Correspond Informatique et Libertés"), Lawyer, CTO.

This sort of platform avoid typical problems of a misdesigned platform:

- Defunct security = defunct confidentiality.

- Defunct scalability = bottleneck under heavy load.

- Defunct monitoring/administration = defunct services for the applications which use this platform.

Indeed, Apache Web Servers, Tomcat, RDBM PostgreSQL support SSL/TLS protocol and are able to do "load balancing" application. Thus allows an effective platform scalability.

Moreover :

- HTTP/HTTPS protocol = **integration capability** in a Information System already there + security access .

- **Input/Ouput platform streams enrichment** when using external services (*e.g.* mobile geolocalization, IP address user geolocalization…) for supplying functionalities for every platform' client applications with respect to a strong access policy to these functionalities.

- Application centered access Policy = The **applications** "ayant-droit" have access to **all appropriate informations for them**.

- "object" RDBM = inheritance between tables for **enrich easily platform' capabilities and policy Privacy**.

6 Conclusions

Conclusions

- By construction the network is:

  - A privileged place for user or group user "context" informations knowledge.
  - A privileged place "enablers" construction available for every application.
  - A important context information itself because of it's physical topology itself: geolocalization by telco mobile network *(e.g.* "cell-id"), @IP (*e.g.* library geoIP), or physical distance calculus between client and server.

- Information interception in core network doesn't allow everything but… it allows a lot of things!!

  – Apply common rules to every stream: authentication, security, all types of filters with minimal or null client-side configuration.
  – Design easily personalized applications by changing services or applications calls order.
  – Enrich/modify streams by adding headers for HTTP based applications.

Conclusions

- Evolution of network components: frontiers become blurred:
  - Routers do port analyze, redirection to proxy.
  - Probes become DPI ("Deep Packet Inspection") and do redirection.
  - Proxy become application security gateway and enrich streams.

  With more and more network components which have a lot of these capabilities.

  This allows to easily design applications by composition of modular services from core network components

  /!\ Privacy !! Power ➔ accountability to the users.

Thanks

Glossary

ACL: Access Control List: access policy rule to a resource.

DoS: Denial of Service: Server shutdown by heavy load generation.

DPI: "Deep Packet Inspection": applicative probe (ISO Layer 7) able to identify applications into TCP packet or UDP frame.

Failover: equipment redundancy with "heart-beat" mechanism for automatic bypassing the defunct equipment.

ICAP: Internet Content Adaptation Protocol: HTTP request or response modification protocol. Defined by IETF RFC3507.

Load-balancing: allows request distribution through an application Server farm in order to avoid server shutdown under heavy load.

POC: Proof Of Concept: proof to demonstrate that a service or architecture is possible. In general demonstrated by a prototype.

Privacy: private data protection: private life protection.

Proxy: "mandatory Server": application gateway.

(L4/L7) Switch: switch with the ability to identify port number in TCP packet or UDP frame and to do some switching decisions on it.

WAF: Web Application Firewall: software or/and hardware to protect Web applications from attacks forged to shutdown Web sites.

WCCP: Web Cache Communication Protocol: communication protocol between Routers and Proxys.

References

General view

[EIAH07] Alain Ottavi, Sylvain Baron, Luigi Lancieri. Capture et exploitation de traces dans un contexte de mobilité. EIAH 2007, Lausanne, June 2007.

[OTT08] Alain Ottavi. "Plate-forme d'interrogation de services externes, de collecte, traitement et mise à disposition de traces applicatives", Mémoire Ingénieur CNAM, 2008.

Log File

[LOG4J] Log 4 Java, http://logging.apache.org/log4j, June 2009.

[LOGW3C] Logging Control In W3C HTTPD, http://www.w3.org/Daemon/User/Config/Logging.html, June 2009.

[SYSLOG] Log file creation and centralization: http://www.syslog.org, Juin 2009.

HTTP, Proxy et WAF

[BC] Blue Coat: http://www.bluecoat.com, June 2009.

[OPTENET] Optenet: http://www.optenet.com, June 2009.

[RFC2616] IETF RFC 2616: definition of HTTP/1.1 protocol.

[RFC3507] IETF RFC 3507: definition of iCAP protocol.

[SQUID] GPL software proxy-cache: http://www.squid-cache.org, June 2009.

[WAF] Web Application Security Consortium : http://www.webappsec.org, June 2009.

References

Platform' Softwares

[PGPOOLII] PGPOOL-II: replication, fail-over and load-balancing between PostgreSQL servers, http://pgpool.projects.postgresql.org/, June 2009.

[PGSQL] PostgreSQL: RDBM, http://www.postgresql.org, June 2009.

[TOMCAT] http://tomcat.apache.org, June 2009.

Privacy

[CNIL] Commission Nationale Informatiques et Libertés, http://www.cnil.fr, June 2009.

[G29] European Commission Workgroup Article 29 on personal data, http://ec.europa.eu/justice_home/fsj/privacy/index_fr.htm, June 2009.