WARP Core
OpenChange Server

Julien Kerihuel
<j.kerihuel@openchange.org>
Contents

1 OpenChange 2008-2009

2 OpenChange Proxy

3 OpenChange Server
1 OpenChange 2008-2009
Provides a portable Open Source implementation of Microsoft Exchange Server and Exchange protocols.

Exchange is a groupware server designed to work with Microsoft Outlook providing a messaging server, shared calendars, contact databases, public folders, notes and tasks.

OpenChange is working on two different aspects:

- Interoperability with Exchange protocols
- Transparent replacement to Microsoft Exchange Server with native Exchange protocols support and direct communication with Microsoft Outlook
Why OpenChange at Samba XP?

- OpenChange client framework is relying on several Samba components and libraries:
  - exchange.idl processed by pidl (Perl IDL compiler)
  - dcerpc, ndr for MSRPC stack
  - ldb and tdb for databases
  - talloc for memory allocation

- OpenChange server side is plugged into Samba4:
  - Extensively use Samba Active Directory
  - Developed as endpoint servers for Samba4
The « not so secret » plan revealed
Classical Microsoft Exchange environment

Windows workstation
Microsoft Outlook client

ExchangeRPC / MAPI

Microsoft Windows Server
Microsoft Exchange Server
Client Migration

No modifications required on Exchange server
(Neither WebDAV nor specific connector/provider)
No modifications required on Outlook clients - works out of the box

(Neither WebDAV nor specific connector to install)
OpenChange 2008-2009

Heterogeneous environment & complete interoperability

AND / OR

ExchangeRPC / MAPI

Microsoft Windows Server
Microsoft Exchange Server

OR

Linux Server
Samba4 + OpenChange
• OpenChange and Samba4 client libraries have now been integrated into various Linux distributions

• Part of **Fedora Core 11 Leonidas** core distribution

• Also available for:
  - **Ubuntu 9.04 Jaunty**
  - **Debian Etch 4.0 (experimental)**
  - **Suse Linux**

• Distributed with **evolution-mapi** package for **Gnome 2.26**

• Portage effort has been provided for:
  - **FreeBSD 7.0**
  - **Windows port of OpenChange and Samba4 client libs at some point?**
OpenChange MAPI library is maturing well
- Consistent return type
- Memory leak and gcc warnings fixed
- API coverage (mapitest unit tests) and documentation
- We are now running a buildbot

New features
- New ROPs integrated
- Multisession code and multiple Mailbox Logon
- Clustered Exchange support
- FreeBusy support
- Preliminary Python bindings
- Provisioning scripts moved from EJS to Python
OpenChange 2008-2009

- 3 slots allocated for promising projects:
  - Proposal: Thunderbird Integration with OpenChange
    - Student: Andrey Yakubovich
    - Mentor: Jelmer Vernooij + Mozilla co-mentoring
  - Proposal: Graphical front-end for OpenChange
    - Student: Billy Okal
    - Mentor: Julien Kerihuel
  - Proposal: Exchange2ICAL tool
    - Student: Ryan Lepinsky
    - Mentor: Brad Hards
OpenChange Proxy
OpenChange Proxy

- The French Waiter technique
The French Waiter technique

Salt coffee:
- How does the customer react?
  - Silently drink it (French and their Chef's speciality …)
  - Ask for another one (politely or not)
  - Swear and leave
The French Waiter technique

Salt coffee:
- How does the customer react?
  - Silently drink it (French and their Chef's speciality ...)
  - Ask for another one (politely or not)
  - Swear and leave

Ignore customer calls:
- How long does he wait? (indefinitely, 5 minutes etc.)
- Does he call the manager or another waiter?
- Does he move to the cafe next door?
• **The French Waiter technique**

  • **Salt coffee:**
    • How does the customer react?
      • Silently drink it (French and their Chef's speciality …)
      • Ask for another one (politely or not)
      • Swear and leave

  • **Ignore customer calls:**
    • How long does he wait? (indefinitely, 5 minutes etc.)
    • Does he call the manager or another waiter?
    • Does he move to the cafe next door?

  • **Pretend you don’t understand English:**
    • How many tries before he leave?
    • Does the client try to order in another language?
OpenChange Proxy

- What is OpenChange Proxy?
  - Proxy server for ExchangeRPC traffic

  - **Transparent/Intercepting proxy:**
    - Does not modify request/responses beyond what is required for authentication and identification

  - **Non-Transparent proxy:**
    - modifies the request or response in order to provide some added service to the user agent

  - **Forwarding proxy:**
    - Forward inbound/outbound traffic
    - Cache results
OpenChange Proxy

Why was OpenChangeProxy developed?
- Writing a server is not trivial
- While openchange MAPI library can test Exchange Server behavior, we had no similar tool to test Outlook behavior
- Helps figuring out what is required/mandatory and what is optional (for preliminary OpenChange Server implementation)

How is it developed?
- Endpoint server for Samba4
- `dcerpc endpoint servers = epmapper, mapiproxy`
- Initial server skeleton based on `dcerpc_remote endpoint from Stefan Metzemacher`
- Register all (used) Exchange endpoints only once for all:
  - `exchange_ds_rfr` (NSPI Referral Service)
  - `exchange_nsp` (NSPI protocol)
  - `exchange_emsmdb` (EMSMDB protocol)
- Microsoft Exchange Servers/Protocols Hub for Samba4
Some « French Waiter techniques » involved in OpenChange proxy:

- EcDoConnect downgrade process
- NSPI bindings replacement
- EcDoConnect downgrade process
OpenChange Proxy

- NSPI bindings replacement

![Diagram showing the MAPI client request to Exchange Server with NSPI bindings replacement.]

```
NspiGetProps (0x9) request
mapiproxy

NspiGetProps (0x9) reply:
[...]  
PR_EMS_AB_NETWORK_ADDRESS:
 ncacn_ip_tcp:mapiproxy.openchange2003.local
 ncacn_np:mapiproxy.openchange2003.local

NspiGetProps (0x9) request
mapiproxy

NspiGetProps (0x9) reply:
[...]  
PR_EMS_AB_NETWORK_ADDRESS:
 ncacn_ip_tcp:exch2k3.openchange2003.local
 ncacn_np:exch2k3.openchange2003.local
```
OpenChange Proxy

- **Provide a stackable modules system**
  - Development framework to add new features
  - Let developers focus on ExchangeRPC traffic rather than transport
  - Russian Dolls

- **Stackable:**
  - Modules are added to a list
  - Each of these modules can have a specific scope
  - Modifications from one module **transparently relayed** to the next one
  - **They have a limited set of hooks**

- **Mapiproxy modules are DSO** (dynamic shared object):
  - Install in a specific location (dcerpc_mapiproxy folder)
  - Enabled or not in smb.conf:
    - `dcerpc_mapiproxy:modules = downgrade,dummy`
    - Modules are sequentially processed
    - **Module ordering matters**
OpenChange Proxy

- Stackable modules figure

- Endpoints and modules figure
OpenChange Proxy

- mapiproxy structure
  - Sometimes a module may want to **bypass the module stack**
  - Want to **impact the overall openchange proxy** behavior
  - Modules can alter the default behavior in their dispatch routine

- **norelay:**
  - Boolean variable
  - Do not relay the incoming request to the remote server
  - directly jump to the push (response) openchange proxy code

- **ahead:**
  - Boolean variable
  - Do not relay the incoming response to the client through the push and dcerpc_ndr_request routine
  - loop over the dispatch routine
The cache module study case
OpenChange Proxy

- mapisession API
  - Server creates the session context using dcerpcHandle_new()
  - Mapiproxy relays the traffic and relies on handle returned by Exchange and used by Outlook
  - If a module is performing complex operations with a lifetime > 1 call, you need to save the server policy_handle for further calls

- Mapisession API makes this tracking easier to handle:
  - create a context
  - can store private data
  - can set a destructor
  - compare current session with saved one
  - release the context
OpenChange Proxy

- OK ... looks great, but give me a study case where it may be useful

Upcoming features:
- full credentials support (delegated credentials)
- clustered Exchange environment support
OpenChange Proxy

- EPITECH Security Project

- Gcalendar and bogofilter module
3 OpenChange Server
Once upon a time ... SambaXP 2006 and the NSPI (Address Book) server:

- Outlook able to setup Exchange server account using OpenChange server only
- Exchange username(s) lookup
- Active Directory Wrapper
- Outlook was happy with the information the server returned

- `dcesrv_exchange_nsp` dynamic shared object
- The NSPI server was registering the `exchange_nsp` endpoint itself
- Used a lot of *French Cafe techniques* and a bit of waiter's ones.
OpenChange Server

- However openchange proxy already registers NSPI endpoint

- This meant we could only have openchange proxy or NSPI server registered at once.

- This was a very restrictive limitation for packaging and distributions.

**Solution:**
1. Turn openchange proxy into a hub for servers AND modules
2. Turn the existing NSPI server into a server module
3. Improve NSPI server behavior using MS-NSPI.pdf specifications
OpenChange Server

- Disabled by default, servers can be enabled through a single smb.conf option

- Case 1:
  - Enable NSPI server only
  - Use modules and remote server for other endpoints

- Case 2:
  - Let 3\textsuperscript{rd} party vendors write their own provider
OpenChange Server

```
dcerpc_mapiproxy: server = true
```

- NSPI server (exchange_nsp)
- EMSMDB server (exchange_emsmdb)
- RFR Server (exchange_ds_rfr)

MAPIProxy Instance
OpenChange Server

dcerpc_mapiproxy:server = false
dcerpc_mapiproxy:nsi_server = exchange_nsp

client

NSPI server (exchange_nsp)

EMSMDB traffic

RFR traffic

MAPIProxy Instance

Exchange Server
OpenChange Server

dcerpc_mapiproxy:server = true
dcerpc_mapiproxy:nspi_server = my_nspi_server

client

NSPI server (my_nspi_server)

EMSMDB server (exchange_emsmdb)

RFR Server (exchange_ds_rfr)

MAPIProxy Instance
OpenChange Server

- EMSMDB/MAPI Store: The WARP core
  - This is the server users are mostly referring to when they talk about Exchange Server.
  - Message store

**Main question:**

What are the requirements to write a Proof of Concept server that **does** something?
How to write a working/demo POC?

- We need to know **how fault tolerant Outlook is:**
  - Minimum required of MAPI calls to launch properly
    - Alinto got through this step – internship/documentation
    - Which data does Outlook fundamentally ask for?

- **What are the Exchange semantics needed** to write the POC?

- What about the **complete customizable storage backend system**?
OpenChange Server

- Won't go much into details ... because the technology is pretty immature/draft

- However:
  - **OpenChange Dispatcher database**
    - LDB database referencing root/special mailbox folders and including a storage namespace attribute
    - **We can have one different storage backend for each root/special folder**

- **MAPIStore abstraction layer**
  - Modular storage abstraction layer
  - Abstract the MAPI semantics from the storage layer
We obviously failed showing up a server serving basic/faked e-mails for SambaXP 2009

So what is the next step?
- Consolidate the client-side
- Improve the user desktop experience with Exchange
- Production environment for openchangeproxy
- September objective:
  - A preliminary server doing mail and calendaring
Questions?