Introduction to SMTP and e-mail

RES, Lecture 4

Olivier Liechti

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Recap:
What did we put in practice in the roulette lab??

What is an Application-Level Protocol?

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **A set of rules** that specify how the application components (e.g. clients and servers) **communicate with each other**. Typically, a protocol defines at least:
  - **Which transport-layer protocol** is used to exchange application-level messages. (e.g. TCP for HTTP)
  - **Which port number(s)** to use (e.g. 80 for HTTP)
  - **What kind of messages** are exchanged by the application components and the **structure** of these messages.
  - The **actions** that need to be taken when these messages are received and the **effect** that is expected.
  - Whether the protocol is **stateful** or **stateless**. In other words, whether the protocol requires the server to manage a session for every connected client.

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

blocking IO (synchronous)

n employee = n threads
employees are expensive
limited space for employees in the truck
few employees => long queue

non-blocking IO (asynchronous)

there is only 1 employee (1 thread)
customers are called back when the request is fulfilled
no queue

Multi Threaded
Single Process
Blocking

The 'old' Java way

The server uses a first thread to wait for connection requests from clients.

Each time a client arrives, a new thread is created and used to serve the client.

Millions of clients, millions of threads?

Resource hungry.
Not scalable.

The ReceptionistWorker implements a run() method that will execute on its own thread.

```
private class ReceptionistWorker implements Runnable {

   @Override
   public void run() {
      ServerSocket serverSocket;

      try {
         serverSocket = new ServerSocket(port);
      } catch (IOException ex) {
         LOG.log(Level.SEVERE, null, ex);
         return;
      }

      while (true) {
         LOG.log(Level.INFO, "Waiting for a new client”);
         try {
            Socket clientSocket = serverSocket.accept();
            LOG.info("A new client has arrived...");
            new Thread(new ServantWorker(clientSocket)).start();
         } catch (IOException ex) {
            LOG.log(Level.SEVERE, ex.getMessage(), ex);
         }
      }
   }
}
```

As soon as a client is connected, a new thread is created.
The code that manages the interaction with the client executes on this thread.

2 types of workers, n+1 threads

Single Thread
Single Process
Asynchronous Programming

The 'à la Node.js' way

The server uses a single thread, but in a non-blocking, asynchronous way.

Callback functions have to be written, so that they can be invoked when clients arrive, when data is received, etc.

Different programming logic.
Scalable.

We are registering callback functions on the various types of events that can be notified by the server...

```
// let's create a TCP server
var server = net.createServer();

// it reacts to events: 'listening', 'connection', 'close', etc.
// register callback functions, to be invoked when the events
// occur (everything happens on the same thread)

server.on('listening', callbackFunctionToCallWhenSocketIsBound);
server.on('connection',
callbackFunctionToCallWhenNewClientHasArrived);

//Start listening on port 9907
server.listen(9907);

// This callback is called when the socket is bound and is in
// listening mode. We don't need to do anything special.
function callbackFunctionToCallWhenSocketIsBound() {
   console.log("The socket is bound and listening");
   console.log("Socket value: %j", server.address());
}

// This callback is called after a client has connected.
function callbackFunctionToCallWhenNewClientHasArrived(socket) {
  ...
}
```

... and we code these functions, implementing the behavior that is expected when the events occur.

This is a non-blocking call...

So, we need to pass a callback that will be called when connection is established

```
function talkToServer(data) {
   client.connect(PORT, HOST, function() {
      console.log('CONNECTED TO: ' + HOST + ':' + PORT);
      //client.setNoDelay();
      client.write('INFO\n');
      client.write('LOAD\n');
      client.write(data);
      client.write('\n');
      client.write('ENDOFDATA\n');
      client.write('INFO\n');
      client.write('BYE\n');
   });

   // The following code registers a callback function on an event notified by the client.
   // When data is available for the client (i.e. bytes have arrived through the socket),
   // the we print it on the console.

   client.on('data', function(data) {
      console.log(">>> " + data);
   });

   client.on('end', function() {
      console.log("** socket closed");
   });
}
```

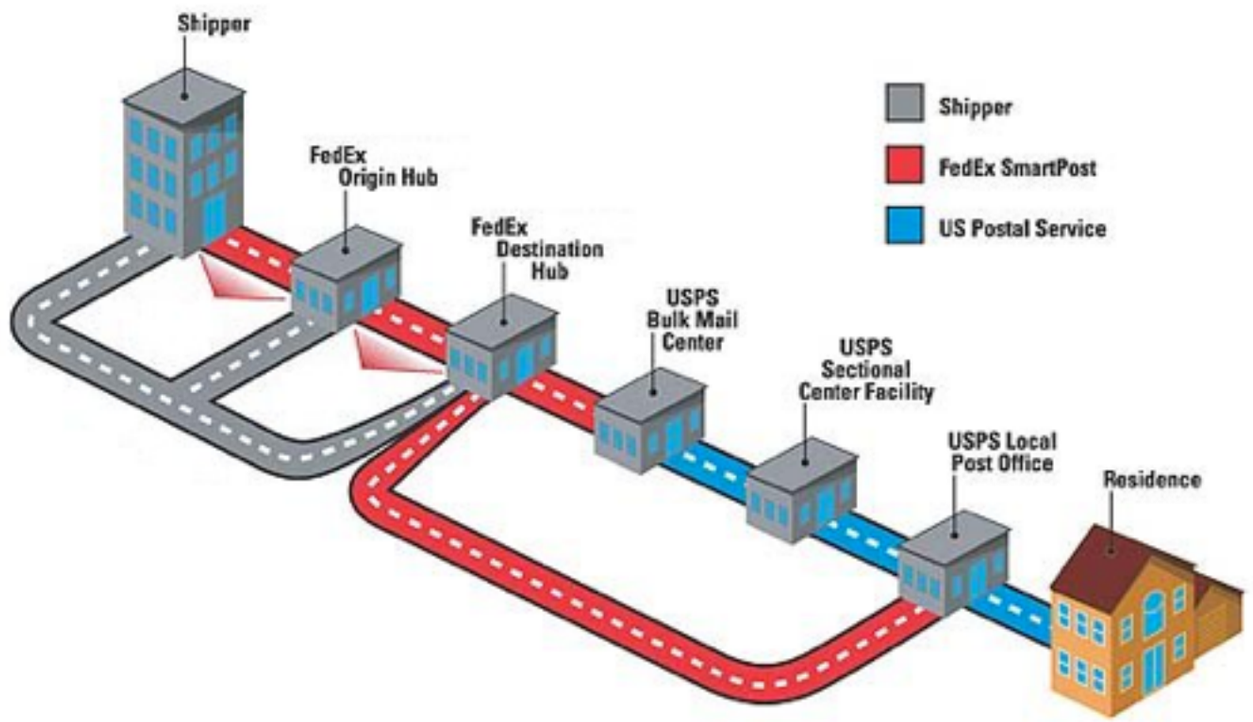
non-blocking I/Os are event-based

What else?

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- What we built is a **library**, which could be used from the user interface layer (whether a GUI, a CLI, etc.).
- Using TDD to specify and build the library was helpful to **avoid coupling** between the the UI and the network layers (compare with a quick-hack where the network code would be in the event handlers of buttons).
- We have seen the importance of **stable interfaces** (and seen that changing published interfaces can negatively impact a large number of people). Creating an interface is easy. Managing its lifecycle is hard.
- We have seen that unit tests can be used to specify the intended behaviour, but that this can be a bit challenging (and require more than a single test). **Having more tests reduces ambiguity**.
- **Is it a bad idea to count the number of issued commands on the client side**, if the server keeps track of them? (building the right product + building the product right).

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



Warning 1

The slides and the webcasts contain examples and demos with **real SMTP servers**.

The behaviour of these servers may change over time. It may also change depending on the network you are connected to (internal, ISP, other ISP).

The main reason why a server might behave differently is the fight between mail administrators and **spammers**.

Warning 2

It is a good thing to experiment with real SMTP servers.

But remember that they are real servers and act responsibly.

Please avoid launching a **surprise denial of service attack** with your accidental infinite loop.

13 

Labo SMTP, part 1

Olivier Liechti

14 

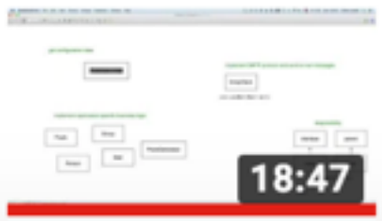
Labo SMTP, part 2

Olivier Liechti

15 

Labo SMTP, part 3

Olivier Liechti

16 

Labo SMTP, part 4

Olivier Liechti

- SMTP demo & hints
- SMTP protocol
- Mock server
- Implementation walk-through

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Tuesday	Wednesday	Friday
10.04 SMTP	11.04 M'sieur, j'ai loupé mon train. See you next year.	13.04 M'sieur, docker marche pas. See you next year.
17.04 Labo SMTP	18.04 Labo SMTP Démos "option rapide"	20.04 Labo SMTP Démos "option rapide"
24.04 Travail écrit Tout, y compris SMTP	25.04 Labo SMTP Démos "option standard"	27.04 Labo SMTP Démos "option standard"

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Démo (**5 minutes MAX**)	
Le labo est terminé et la démo est faite le 18 ou le 24 avril.	1 pt
Le groupe arrive à démarrer un serveur mock dans un container Docker et à expliquer à quoi il sert.	1 pt
Le groupe montre comment configurer la campagne de "pranks" et lance son programme. Le groupe explique les résultats dans le serveur mock. La démo ne marche pas: 0 pt!	1.5 pt
Le groupe montre son repo GitHub. En regardant les commits, on voit que tout le monde a participé et qu'il n'y a pas seulement un gros commit à la fin.	0.5 pt
Une documentation de qualité et conforme aux exigences est fournie dans le repo GitHub.	1 pt

https://docs.google.com/spreadsheets/d/1KuuhFDZ85_CEbqpb51mL8zuOnvbP65Nd0qCEmYHNAQc/edit?usp=sharing

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

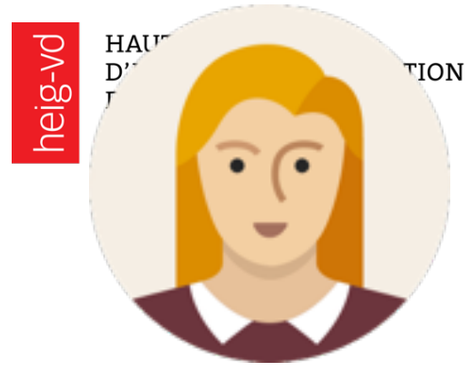
What happens when Bob wants to **send an e-mail** to Alice?

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Bob uses **Thunderbird** to
write his mail.

Alice uses **MS Outlook** to
check and read her mails.

In the technical specs (RFCs),
these programs are called
Mail User Agents (MUA)



Bob uses his professional e-mail address. His company runs a **MS Exchange Server**.

Alice uses her private address. She has an account (and a **mailbox**) on the **Google gmail** infrastructure.

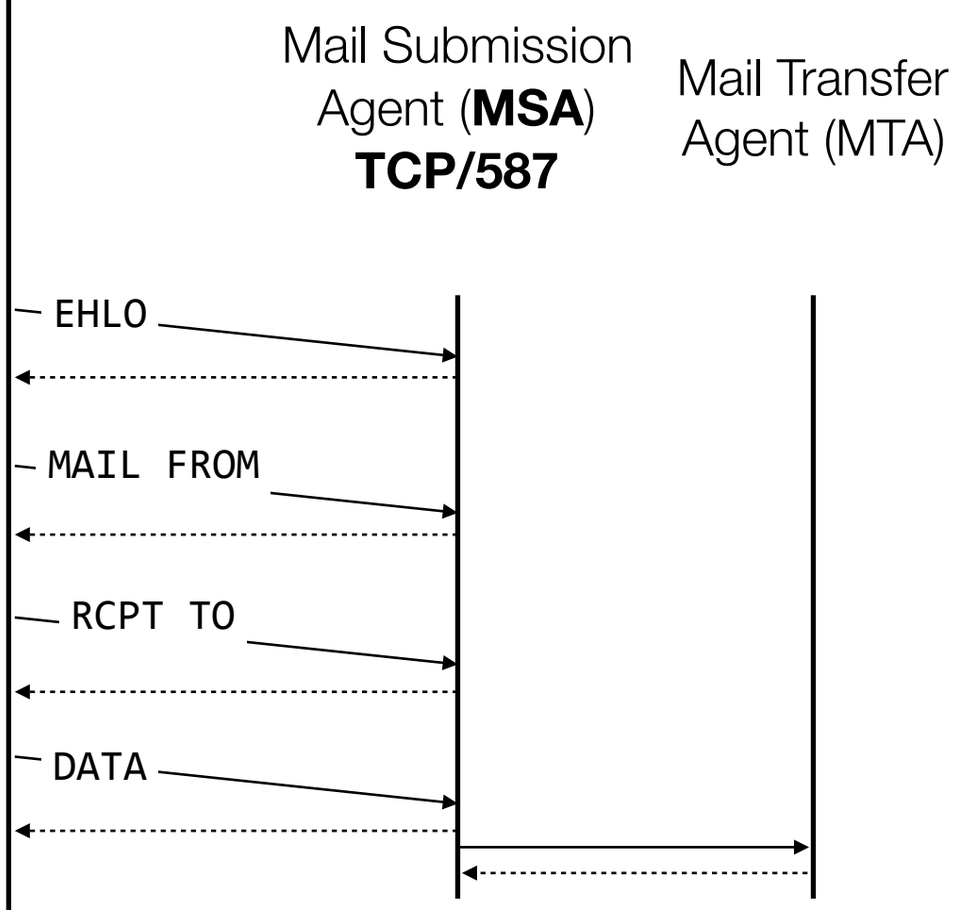
Microsoft Exchange

Gmail by Google

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Bob writes a message to "**alice.res@gmail.com**". He pushes on the "Send" button.

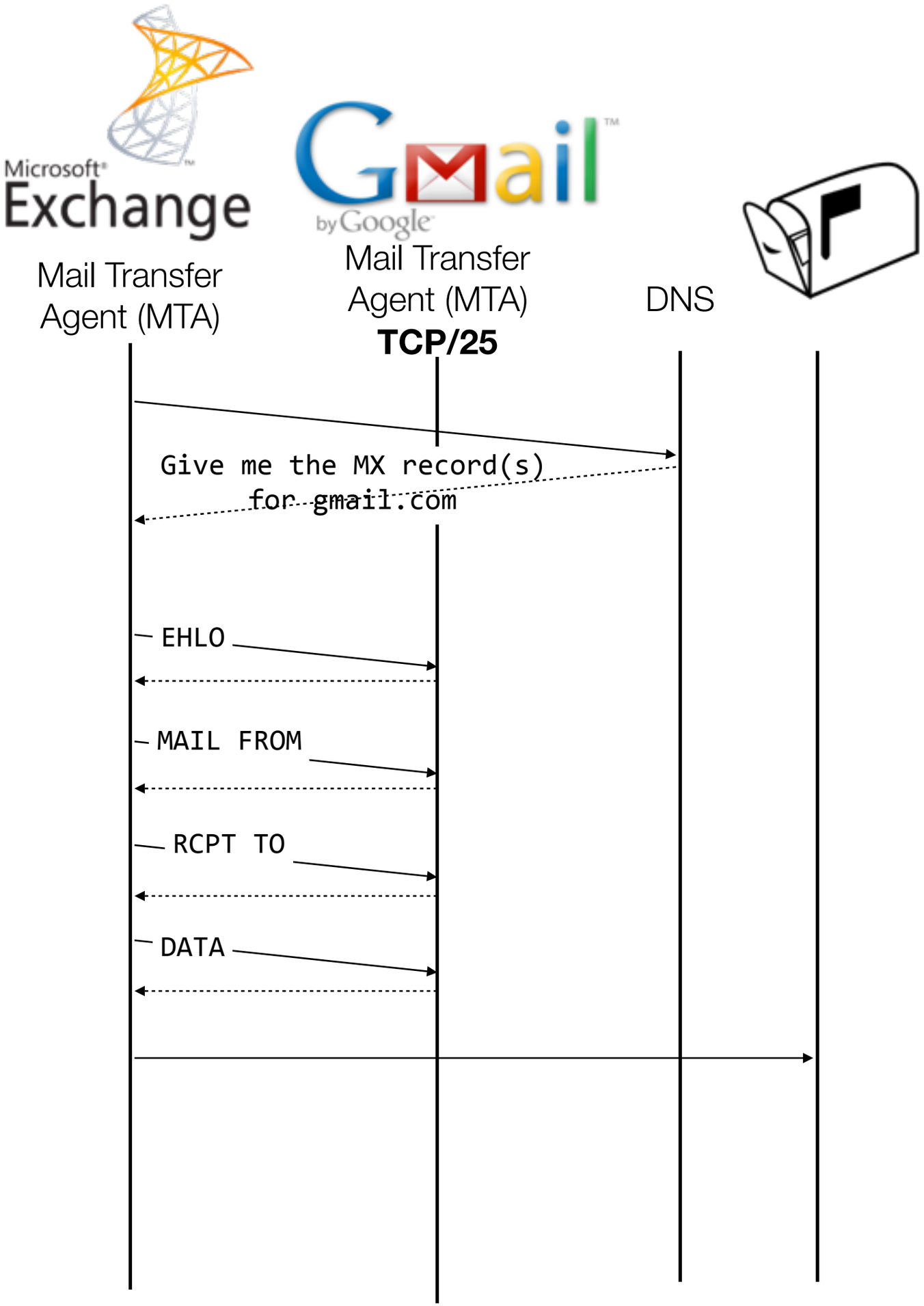
The Exchange Server is made of **2 logical components:** the **MSA** and the **MTA**.



Bob's MUA asks Bob's MSA to deliver the mail. It uses the **SMTP** protocol for that purpose and (should) use TCP port 587.

After enforcing **usage policies**, the MSA delegates the work to the MTA. We don't know how.

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



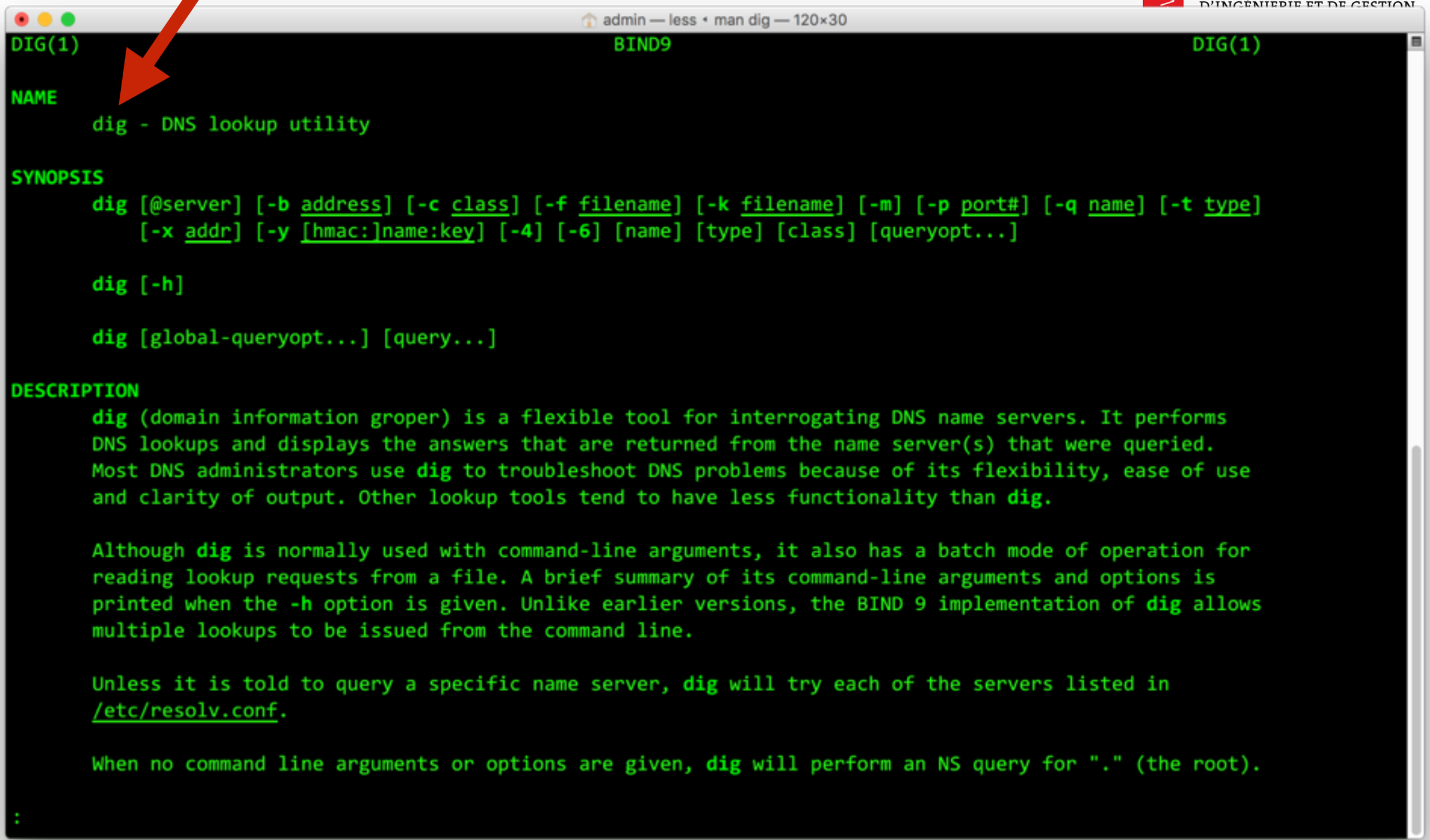
Bob's MTA initially does not know where to forward the mail...

It issues a **DNS** query to get a list of **MX records** for Alice's domain (gmail.com).

When Bob's MTA knows the IP address of Alice's MTA, it uses the **SMTP** protocol once more to forward the message. TCP **port 25** is used in this case.

When Alice's MTA receives the mail, it stores it in Alice's **mailbox** (for later retrieval).

dig



`nslookup` is another command for querying DNS

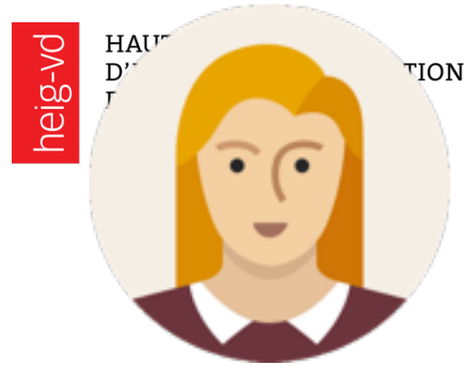
dig -t ANY heig-vd.ch

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION

admin — -bash --login — 120×30

```
$ dig -t ANY heig-vd.ch

; <<>> DiG 9.8.3-P1 <<>> -t ANY heig-vd.ch
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62138
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;heig-vd.ch.                    IN      ANY

;; ANSWER SECTION:
heig-vd.ch.             3600    IN      NS      ns01.heig-vd.ch.
heig-vd.ch.             3600    IN      NS      ns02.heig-vd.ch.
heig-vd.ch.             3600    IN      A       193.134.220.23
heig-vd.ch.             3600    IN      TXT     "MS=ms50694826"
heig-vd.ch.             3600    IN      TXT     "v=spf1 ip4:193.134.216.180/30 mx ~all"
heig-vd.ch.             3600    IN      MX      10 mailcl2.heig-vd.ch.
heig-vd.ch.             3600    IN      MX      10 mailcl1.heig-vd.ch.
heig-vd.ch.             3600    IN      MX      10 mailcl0.heig-vd.ch.
heig-vd.ch.             3600    IN      SOA     ns01.heig-vd.ch. domain.heig-vd.ch. 2014141923 10800 3600 2419200 900

;; Query time: 2 msec
;; SERVER: 10.192.22.5#53(10.192.22.5)
;; WHEN: Tue Apr  5 13:20:45 2016
;; MSG SIZE  rcvd: 273

$
```

`MX records` point to the SMTP servers for the domain

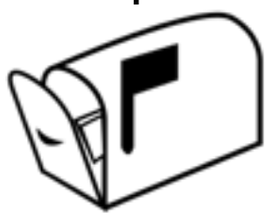


In the last step, Alice's MUA uses another protocol (e.g. IMAP, POP3) to fetch mails from the mailbox.

SMTP
587

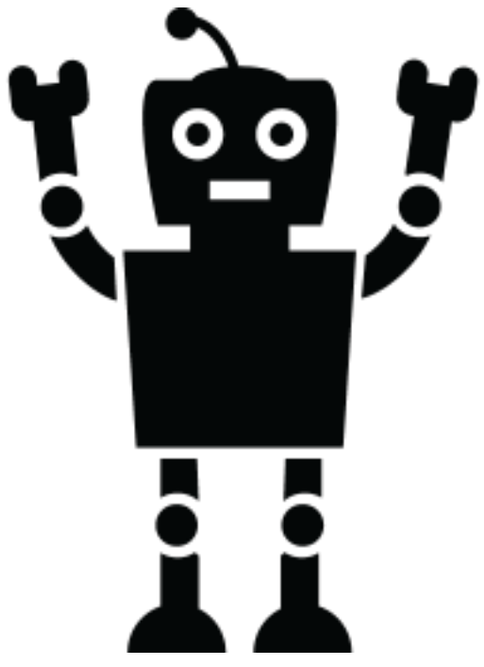
IMAP/POP3

SMTP
25



Let's be human Exchange Servers
(and play the role of Bob's MTA).

But instead of forwarding the mail
to gmail, let's forward the mail via
the **HEIG-VD's SMTP** server.



```
dig -t MX heig-vd.ch

heig-vd.ch. 3600 INMX 10 mailcl0.heig-vd.ch.
```

~~telnet mailcl0.heig-vd.ch 25~~

```
openssl s_client -starttls smtp -crlf -connect
mailcl0.heig-vd.ch:25
```

```
EHLO mycompany.com
```

```
$ telnet mailcl10.heig-vd.ch 25
mailcl10.heig-vd.ch: nodename nor servname provided, or not known
$ telnet mailcl0.heig-vd.ch 25
Trying 193.134.216.181...
Connected to mailcl0.heig-vd.ch.
Escape character is '^]'.
220 heig-vd.ch ESMTP MailCleaner (Enterprise Edition 2016.01) Tue, 05 Apr 2016 14:18:24
+0200
EHLO mycompany.com
250-heig-vd.ch Hello mbp-de-admin.einet.ad.eivd.ch [10.192.116.92]
250-SIZE 20480000
250-8BITMIME
250-PIPELINING
250-AUTH PLAIN LOGIN
250-STARTTLS
250 HELP
MAIL FROM: bob@bob.com
250 OK
RCPT TO: olivier.liechti@wasabi-tech.com
250 Accepted
DATA
354 Enter message, ending with "." on a line by itself
From: bob@areyousure.com
To: olivier.liechti@wasabi-tech.com
Subject: demo

Ok. Cool. Bye.
.
250 OK id=1anPx9-0003KC-BC
quit
221 heig-vd.ch closing connection
Connection closed by foreign host.
```

SMTP command

!=

Message header

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

The Specs

https://tools.ietf.org/html/rfc5321

Table of Contents

https://tools.ietf.org/html/rfc5321

D.1. A Typical SMTP Transaction Scenario

This SMTP example shows mail sent by Smith at host bar.com, and to Jones, Green, and Brown at host foo.com. Here we assume that host bar.com contacts host foo.com directly. The mail is accepted for Jones and Brown. Green does not have a mailbox at host foo.com.

```
S: 220 foo.com Simple Mail Transfer Service Ready
C: EHLO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250 HELP
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
C: RCPT TO:<Brown@foo.com>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: Blah blah blah...
C: ...etc. etc. etc.
C: .
S: 250 OK
C: QUIT
S: 221 foo.com Service closing transmission channel
```

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

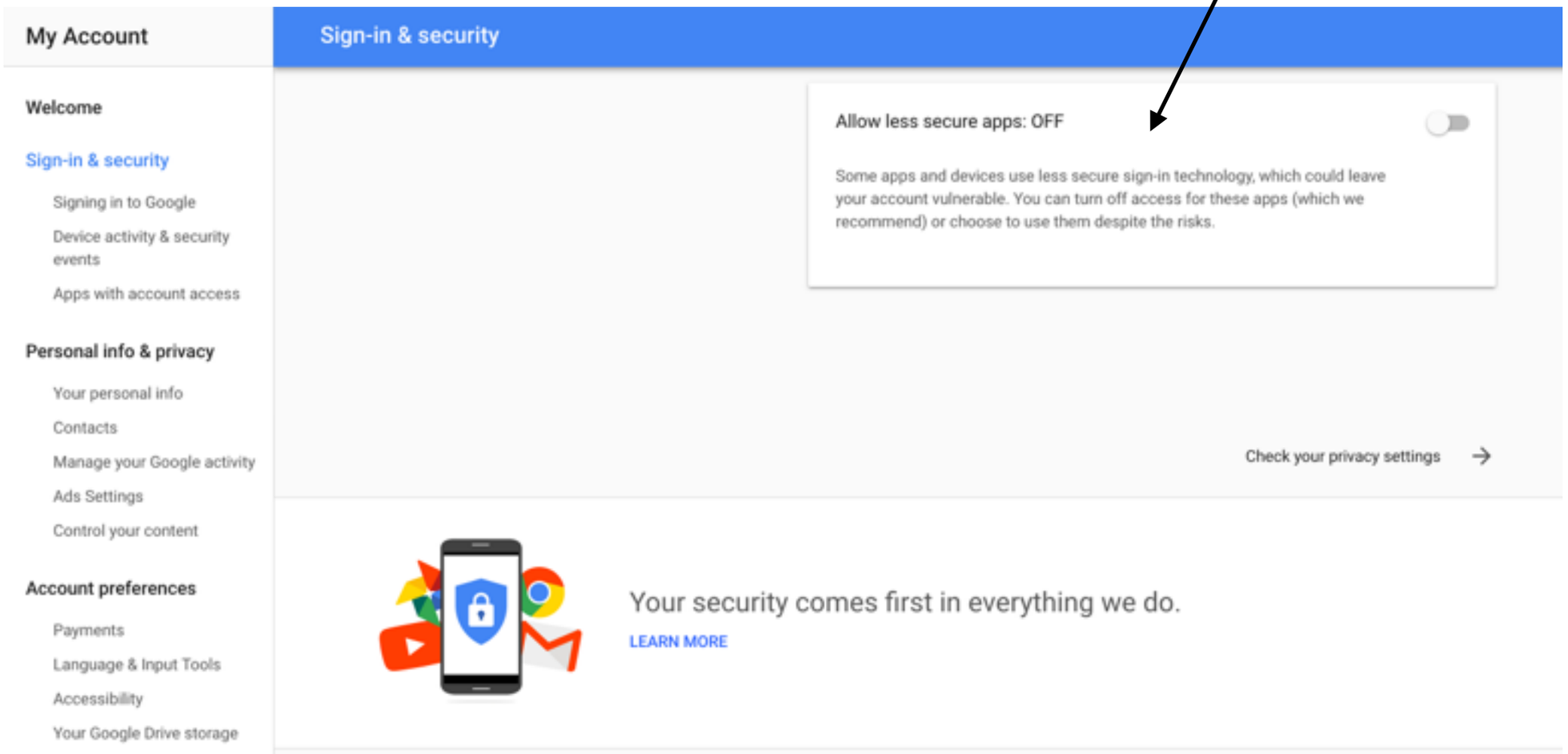
SMTP Servers for experiments

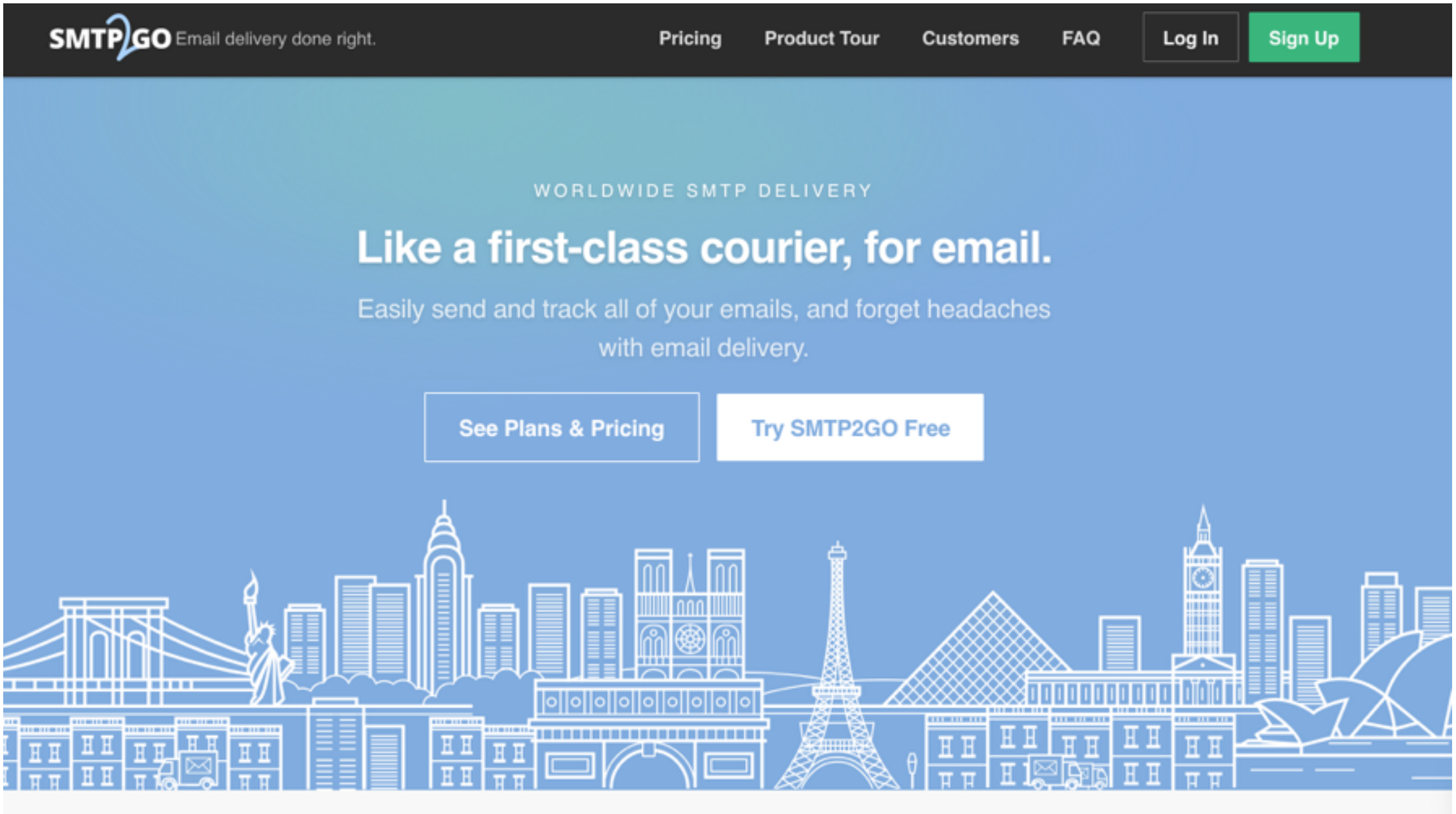

Mon site | Mes liens | Bienvenue LIECHTI Olivier
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch
Quicklinks
La boîte à outils des services HEIG-VD
Services Départements Ra&D Académique Campus Infrastructure Liens Conseil représentatif
Rechercher
Intranet HEIG-VD > Services > Informatique > Poste de travail > Messagerie > Configuration > SMTP
Webmail
Outlook
Configuration
Exchange
IMAP
POP
SMTP
LDAP
Archivage
Spam
Mailing
Réservation salles
Le protocole SMTP est utilisé lors de l'envoi des mails. Il est complémentaire aux protocoles POP et IMAP qui ne s'occupent que de la réception.
L'adresse de notre serveur SMTP est :
smtp.heig-vd.ch
Attention, por lutter contre le spam ainsi que pour des raisons de sécurité, notre serveur smtp, comme bien d'autres, n'autorisent l'envoi d' e-mails que depuis l'intérieur de notre réseau (ou via vpn). Depuis chez vous, il faut utiliser le serveur smtp de votre fournisseur d'accès internet.
Pour en savoir plus : wikipedia
ATTENTION !!
Pour le moment, la connexion IMAP impose d'activer le SLL.

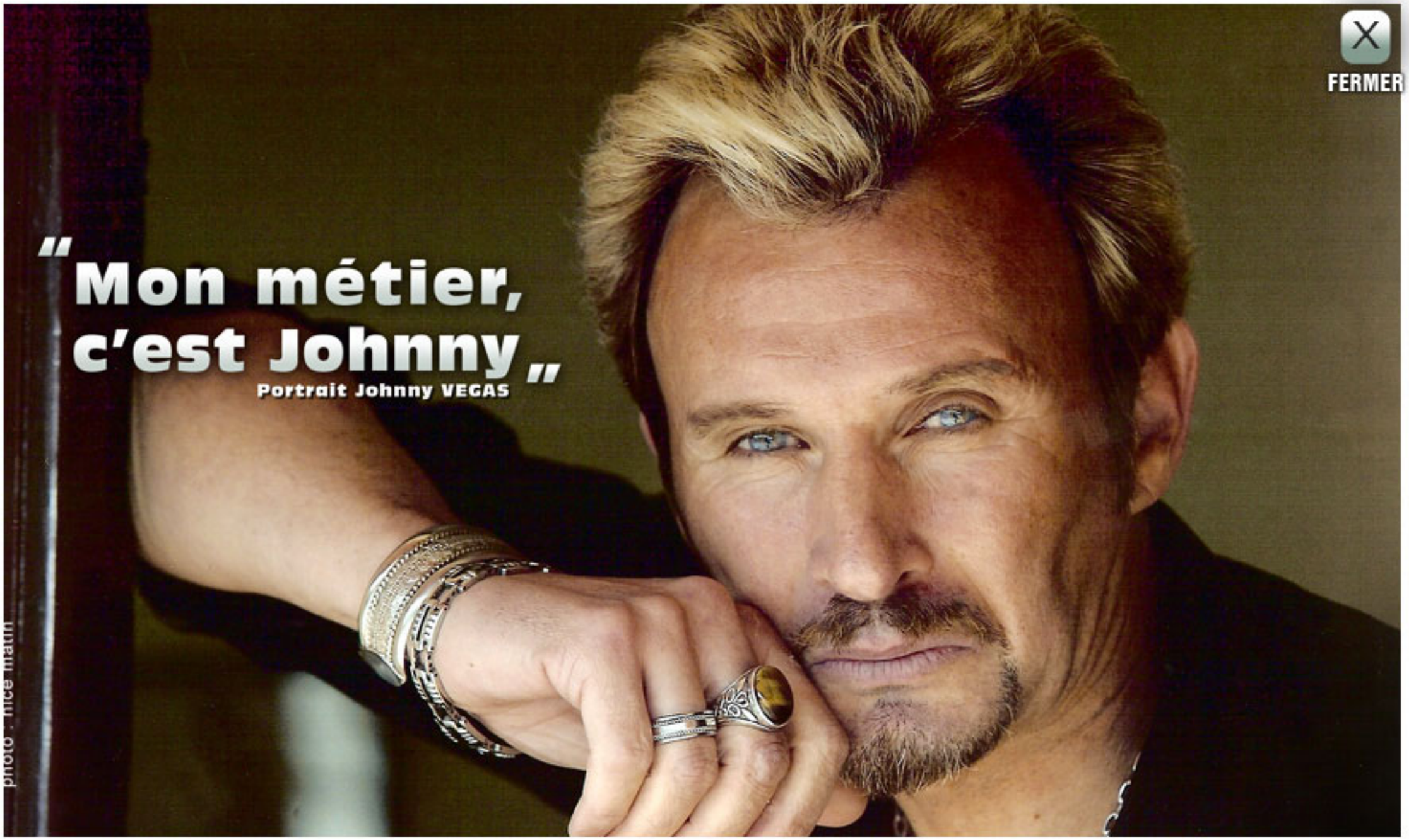
heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Gmail

With this default setup, you will not be able to login with your user id / password.




SMTP2GO Email delivery done right.
Pricing
Product Tour
Customers
FAQ
Log In
Sign Up
WORLDWIDE SMTP DELIVERY
Like a first-class courier, for email.
Easily send and track all of your emails, and forget headaches with email delivery.
See Plans & Pricing
Try SMTP2GO Free



Mock Servers

tweakers-dev / **MockMock**

Watch 10 | Unstar 39 | Fork 24

Code | Issues 2 | Pull requests 4 | Projects 0 | Wiki | Insights

A mock SMTP server built with Java

MockMock | Home | MockMock on Github

I've got 24 mails for you. Nice! Delete all

From	To	Subject
John Doe <someone@example.org>	Some Dude <dude@examp...	Well, this is a nice subject...
John Doe <someone@example.org>	Some Dude <dude@examp...	LOL omg!
John Doe <someone@example.org>	Some Dude <dude@examp...	The iPhone 5 is huge!
John Doe <someone@example.org>	Some Dude <dude@examp...	Did you see the new MockMock version already?
John Doe <someone@example.org>	Some Dude <dude@examp...	Well, this is a nice subject...
John Doe <someone@example.org>	Some Dude <dude@examp...	Well, this is a nice subject...
John Doe <someone@example.org>	Some Dude <dude@examp...	Did you see the new MockMock version already?


mailtrap
HOW IT WORKS
PRICING
API
BLOG
FAQ
HELP
Log in
Sign up
railsware
SAFE
email testing
for dev teams
See How it Works
Watch
Video