

Introduction to SMTP

RES, Lecture 3

Olivier Liechti
Juergen Ehrensberger



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.



May changing your
Facebook relationship
status as an April Fool's
joke not cause
the end of your
relationship.



som^{ee}cards



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



Démonstration (5 minutes MAX)	
Le labo est terminé et la démonstration est faite dans les délais.	
Le groupe arrive à démarrer un serveur mock dans un container Docker et à expliquer à quoi il sert. Le groupe a aussi configuré le service mailtrap.io	
Le groupe montre comment configurer la campagne de "pranks" et lance son programme dans un environnement de test (mock mock, mailtrap ou autre). Le groupe explique les résultats. La démonstration ne marche pas: 0 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.	
Une documentation de qualité et conforme aux exigences est fournie dans le repo GitHub.	





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



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



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



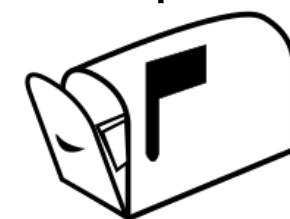
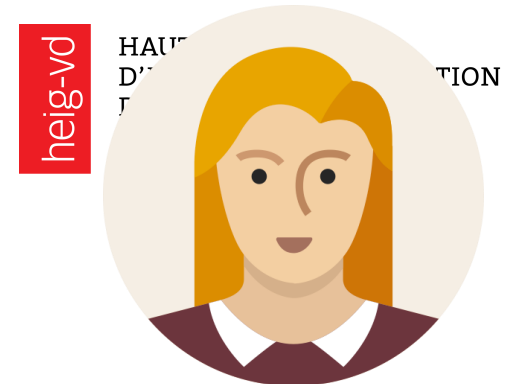
In the technical specs (RFCs),
these programs are called
Mail User Agents (MUA)





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

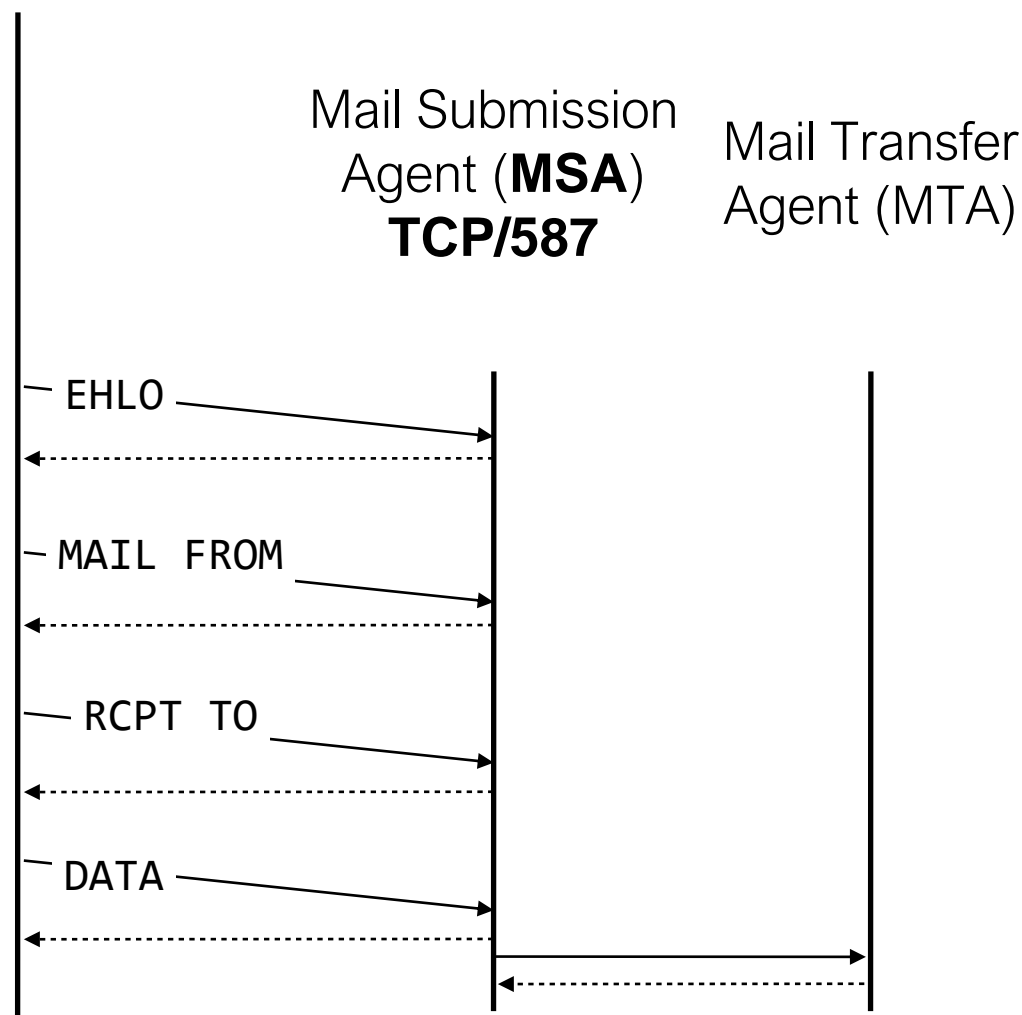
Alice uses her private address. She has an account (and a **mailbox**) on the **Google gmail** infrastructure.





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.





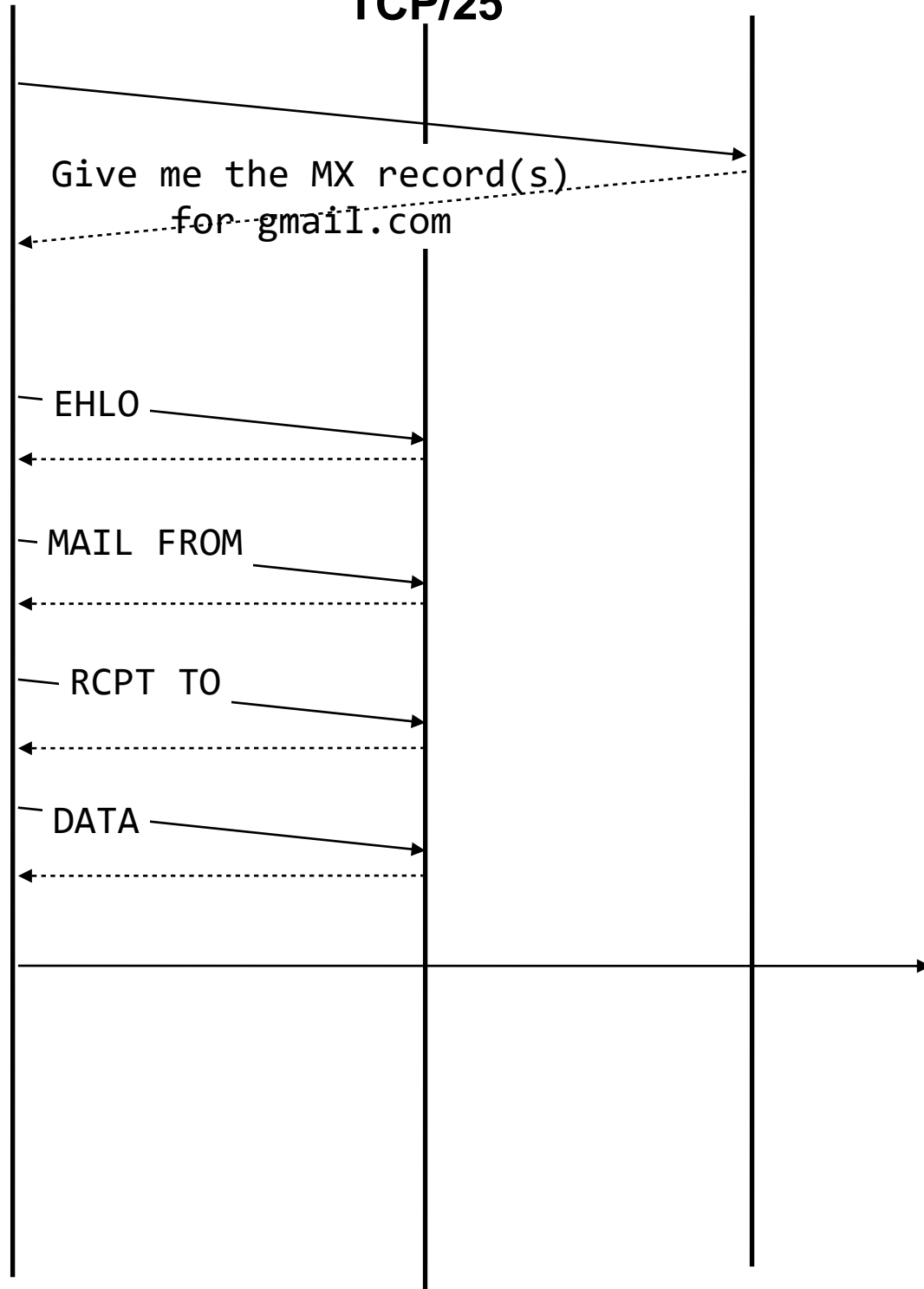
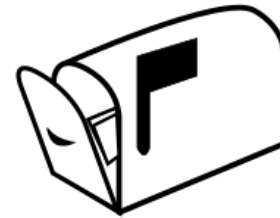
Mail Transfer Agent (MTA)



Mail Transfer Agent (MTA)

TCP/25

DNS



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

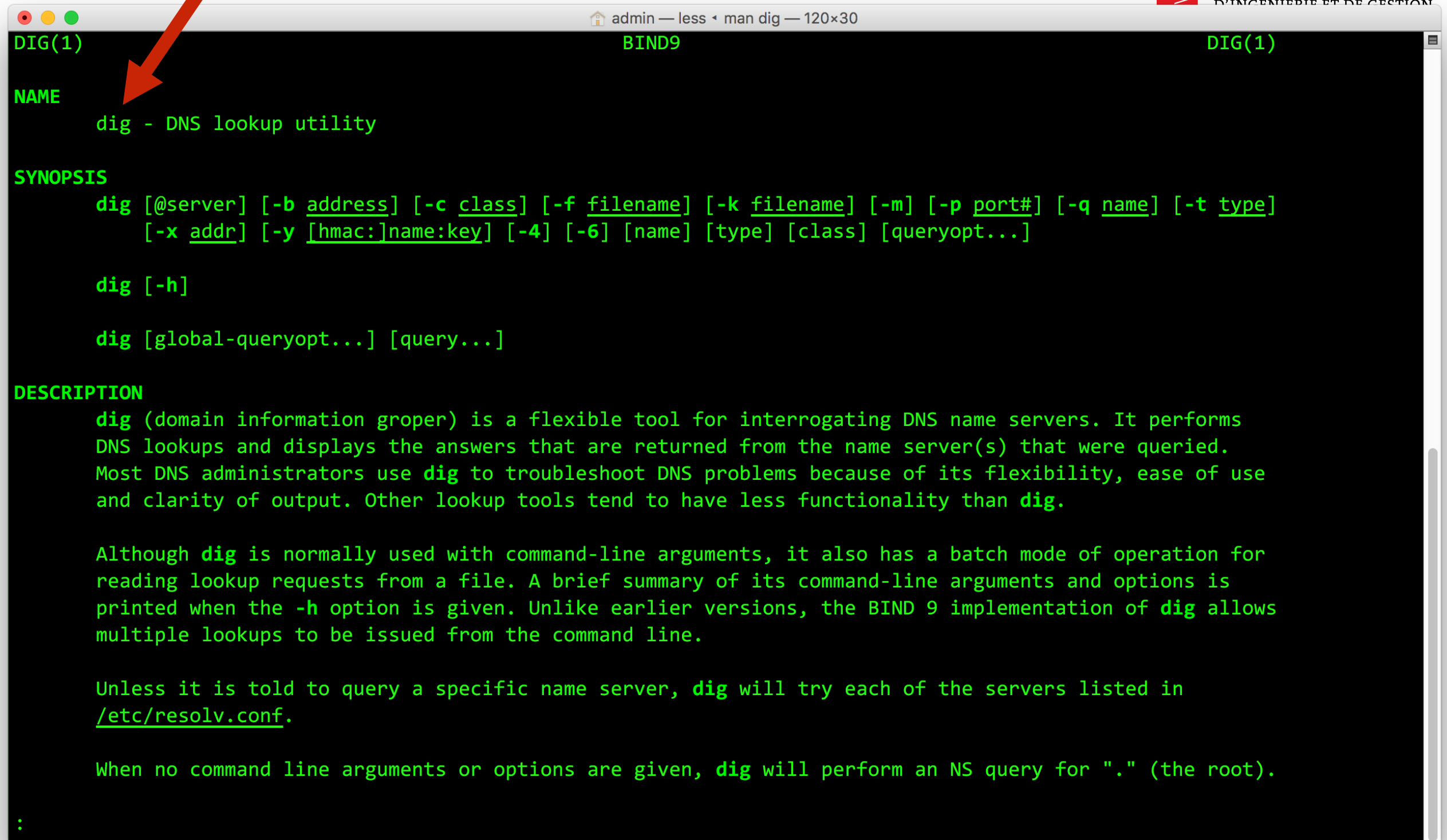
It issues a **DNS** query to get a list of **MX records** for Alice's domain (gmail.com).

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



```
admin — less ◀ man dig — 120x30
DIG(1)                                BIND9                                DIG(1)

NAME
    dig - DNS lookup utility

SYNOPSIS
    dig [@server] [-b address] [-c class] [-f filename] [-k filename] [-m] [-p port#] [-q name] [-t type]
    [-x addr] [-y [hmac:]name:key] [-4] [-6] [name] [type] [class] [queryopt...]

    dig [-h]

    dig [global-queryopt...] [query...]

DESCRIPTION
    dig (domain information groper) is a flexible tool for interrogating DNS name servers. It performs
    DNS lookups and displays the answers that are returned from the name server(s) that were queried.
    Most DNS administrators use dig to troubleshoot DNS problems because of its flexibility, ease of use
    and clarity of output. Other lookup tools tend to have less functionality than dig.

    Although dig is normally used with command-line arguments, it also has a batch mode of operation for
    reading lookup requests from a file. A brief summary of its command-line arguments and options is
    printed when the -h option is given. Unlike earlier versions, the BIND 9 implementation of dig allows
    multiple lookups to be issued from the command line.

    Unless it is told to query a specific name server, dig will try each of the servers listed in
    /etc/resolv.conf.

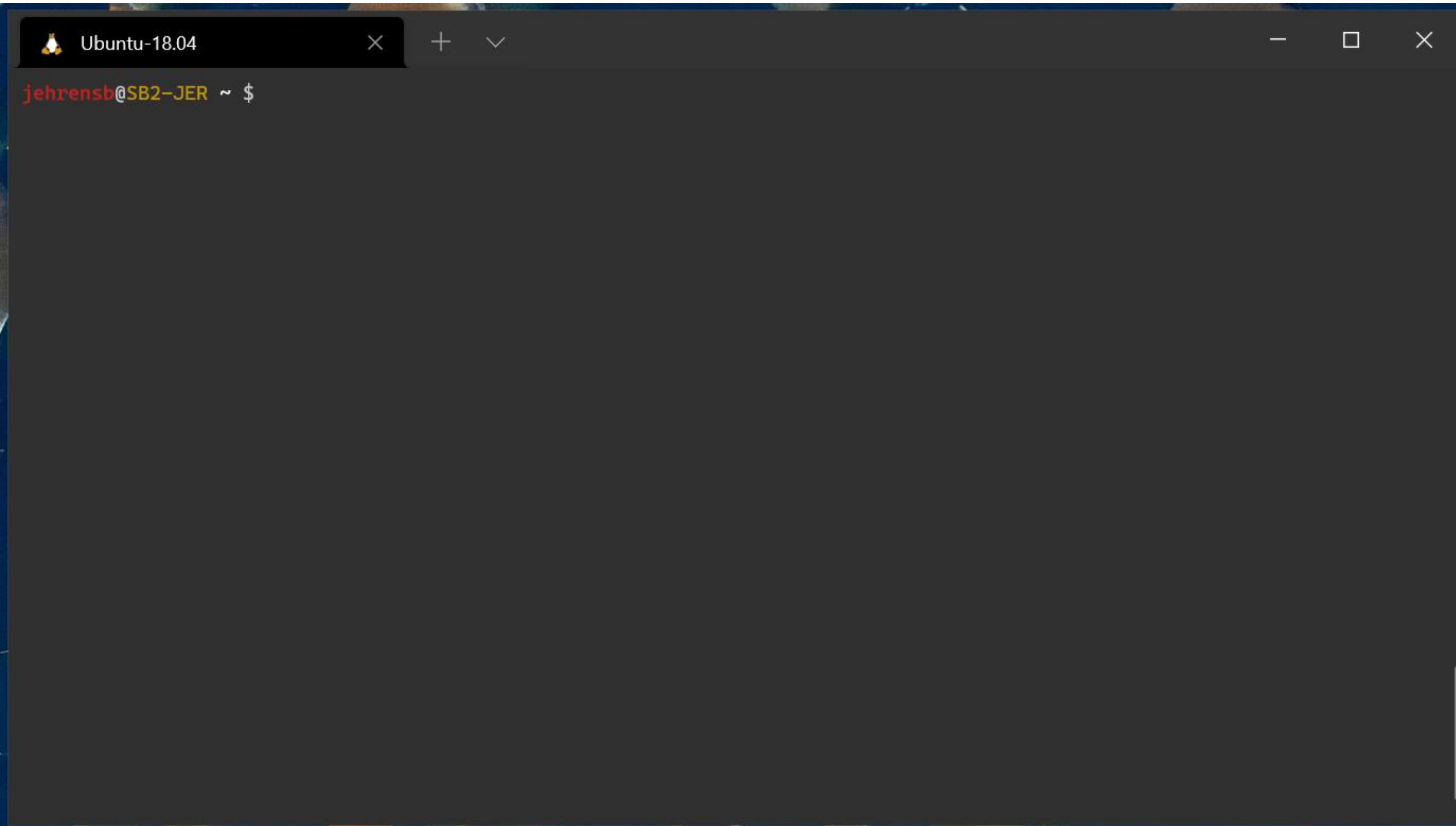
    When no command line arguments or options are given, dig will perform an NS query for "." (the root).

:
```

nslookup is another command for querying DNS



Demo dig and telnet



Demo dig and telnet

```
Ubuntu-18.04 × + ∨ — □ ×

heig-vg.ch.      600    IN      TXT      "v=spf1 ip4:193.134.216.180/30 ip4:193.134.218.105 ip4:193.134.216.126
ip4:193.134.216.121 mx ~all"
heig-vg.ch.      600    IN      TXT      "MS=ms50694826"
heig-vg.ch.      3600   IN      A        193.134.222.245
heig-vg.ch.      3600   IN      NS       ns02.heig-vg.ch.
heig-vg.ch.      3600   IN      NS       ns01.heig-vg.ch.
heig-vg.ch.      3600   IN      MX       10 mail01.heig-vg.ch.
heig-vg.ch.      3600   IN      MX       20 gwsmtpl.avdtec.ch.

;; Query time: 24 msec
;; SERVER: 10.192.22.5#53(10.192.22.5)
;; WHEN: Tue Apr 20 09:16:09 CEST 2021
;; MSG SIZE rcvd: 580

jehrensb@SB2-JER ~ $ telnet mail01.heig-vg.ch 25
Trying 193.134.218.124...
Connected to mail01.heig-vg.ch.
Escape character is '^]'.
220 HEIG-VD Antispam solution v1.2
ehlo gmail.com
250-mail01.heig-vg.ch Hello gmail.com [10.193.156.177], pleased to meet you
250-SIZE 35000000
250-STARTTLS
250-PIPELINING
250-8BITMIME
250 HELP
quit
221 mail01.heig-vg.ch Goodbye gmail.com, closing connection
Connection closed by foreign host.
jehrensb@SB2-JER ~ $
```





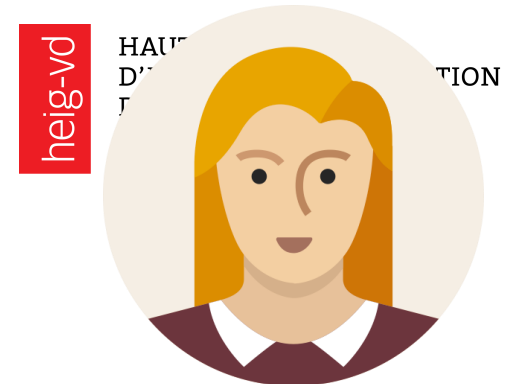
SMTP
587



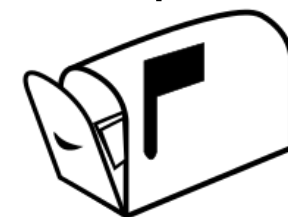
Microsoft
Exchange

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

SMTP
25

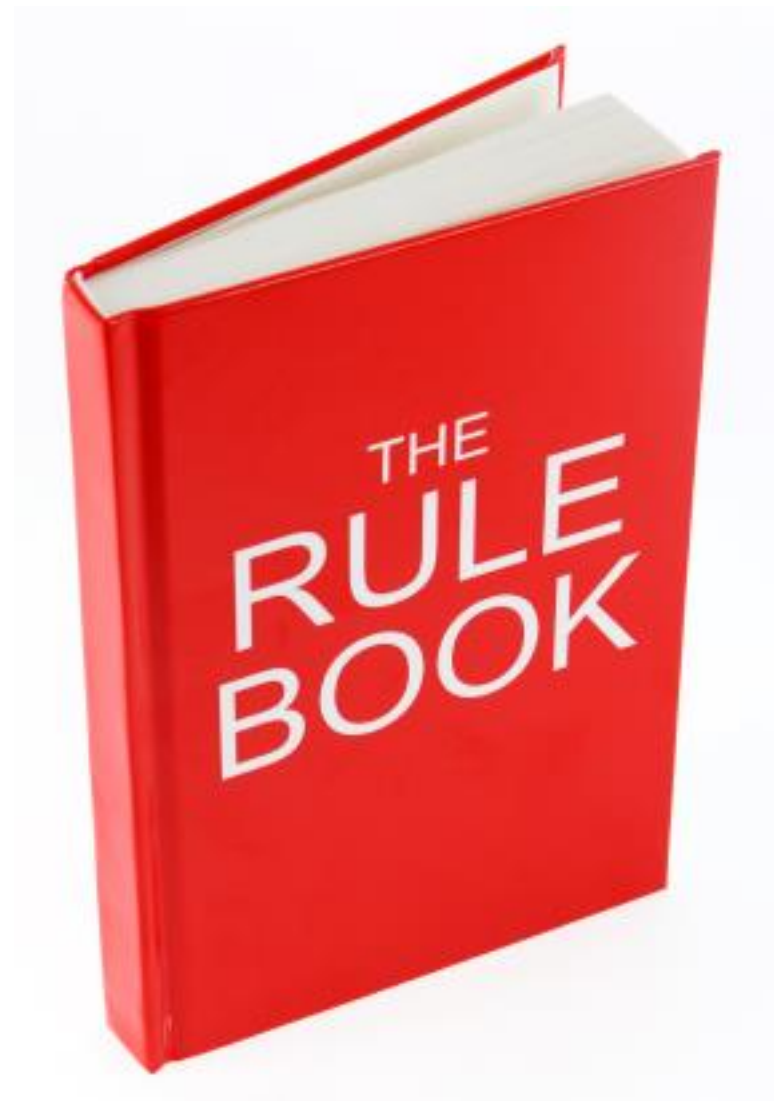


IMAP/POP3



Gmail
by Google





The Specs



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

Table of Contents

RFC 5321

SMTP

October 2008

1.	Introduction	5			
1.1.	Transport of Electronic Mail	5			
1.2.	History and Context for This Document	5			
1.3.	Document Conventions	6			
2.	The SMTP Model	7			
2.1.	Basic Structure	7			
2.2.	The Extension Model	9			
2.2.1.	Background	9			
2.2.2.	Definition and Registration of Extensions	10			
2.2.3.	Special Issues with Extensions	11			
2.3.	SMTP Terminology	11			
2.3.1.	Mail Objects	11			
2.3.2.	Senders and Receivers	12			
2.3.3.	Mail Agents and Message Stores	12			
2.3.4.	Host	13			
2.3.5.	Domain Names	13			
2.3.6.	Buffer and State Table	14			
2.3.7.	Commands and Replies	14			
2.3.8.	Lines	14			
2.3.9.	Message Content and Mail Data	15			
2.3.10.	Originator, Delivery, Relay, and Gateway Systems	15			
2.3.11.	Mailbox and Address	15			
2.4.	General Syntax Principles and Transaction Model	16			
3.	The SMTP Procedures: An Overview	17			
3.1.	Session Initiation	18			
3.2.	Client Initiation	18			
3.3.	Mail Transactions	19			
3.4.	Forwarding for Address Correction or Updating	21			
3.5.	Commands for Debugging Addresses	22			
3.5.1.	Overview	22			
3.5.2.	VRIFY Normal Response	24			
3.5.3.	Meaning of VRIFY or EXPN Success Response	25			
3.5.4.	Semantics and Applications of EXPN	26			
3.6.	Relaying and Mail Routing	26			
3.6.1.	Source Routes and Relaying	26			
3.6.2.	Mail eXchange Records and Relaying	26			
3.6.3.	Message Submission Servers as Relays	27			
3.7.	Mail Gatewaying	28			
3.7.1.	Header Fields in Gatewaying	28			
3.7.2.	Received Lines in Gatewaying	29			
3.7.3.	Addresses in Gatewaying	29			
3.7.4.	Other Header Fields in Gatewaying	29			
3.7.5.	Envelopes in Gatewaying	30			
3.8.	Terminating Sessions and Connections	30			
3.9.	Mailing Lists and Aliases	31			
3.9.1.	Alias	31			
			3.9.2.	List	31
			4.	The SMTP Specifications	32
			4.1.	SMTP Commands	32
			4.1.1.	Command Semantics and Syntax	32
			4.1.2.	Command Argument Syntax	41
			4.1.3.	Address Literals	43
			4.1.4.	Order of Commands	44
			4.1.5.	Private-Use Commands	46
			4.2.	SMTP Replies	46
			4.2.1.	Reply Code Severities and Theory	48
			4.2.2.	Reply Codes by Function Groups	50
			4.2.3.	Reply Codes in Numeric Order	52
			4.2.4.	Reply Code 502	53
			4.2.5.	Reply Codes after DATA and the Subsequent <CRLF>.<CRLF>	53
			4.3.	Sequencing of Commands and Replies	54
			4.3.1.	Sequencing Overview	54
			4.3.2.	Command-Reply Sequences	55
			4.4.	Trace Information	57
			4.5.	Additional Implementation Issues	61
			4.5.1.	Minimum Implementation	61
			4.5.2.	Transparency	62
			4.5.3.	Sizes and Timeouts	62
			4.5.3.1.	Size Limits and Minimums	62
			4.5.3.1.1.	Local-part	63
			4.5.3.1.2.	Domain	63
			4.5.3.1.3.	Path	63
			4.5.3.1.4.	Command Line	63
			4.5.3.1.5.	Reply Line	63
			4.5.3.1.6.	Text Line	63
			4.5.3.1.7.	Message Content	63
			4.5.3.1.8.	Recipients Buffer	64
			4.5.3.1.9.	Treatment When Limits Exceeded	64
			4.5.3.1.10.	Too Many Recipients Code	64
			4.5.3.2.	Timeouts	65
			4.5.3.2.1.	Initial 220 Message: 5 Minutes	65
			4.5.3.2.2.	MAIL Command: 5 Minutes	65
			4.5.3.2.3.	RCPT Command: 5 Minutes	65
			4.5.3.2.4.	DATA Initiation: 2 Minutes	66
			4.5.3.2.5.	Data Block: 3 Minutes	66
			4.5.3.2.6.	DATA Termination: 10 Minutes	66
			4.5.3.2.7.	Server Timeout: 5 Minutes	66
			4.5.4.	Retry Strategies	66
			4.5.5.	Messages with a Null Reverse-Path	68
			5.	Address Resolution and Mail Handling	69
			5.1.	Locating the Target Host	69
			5.2.	IPv6 and MX Records	71
			6.	Problem Detection and Handling	71



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
```



D.3. Relayed Mail Scenario

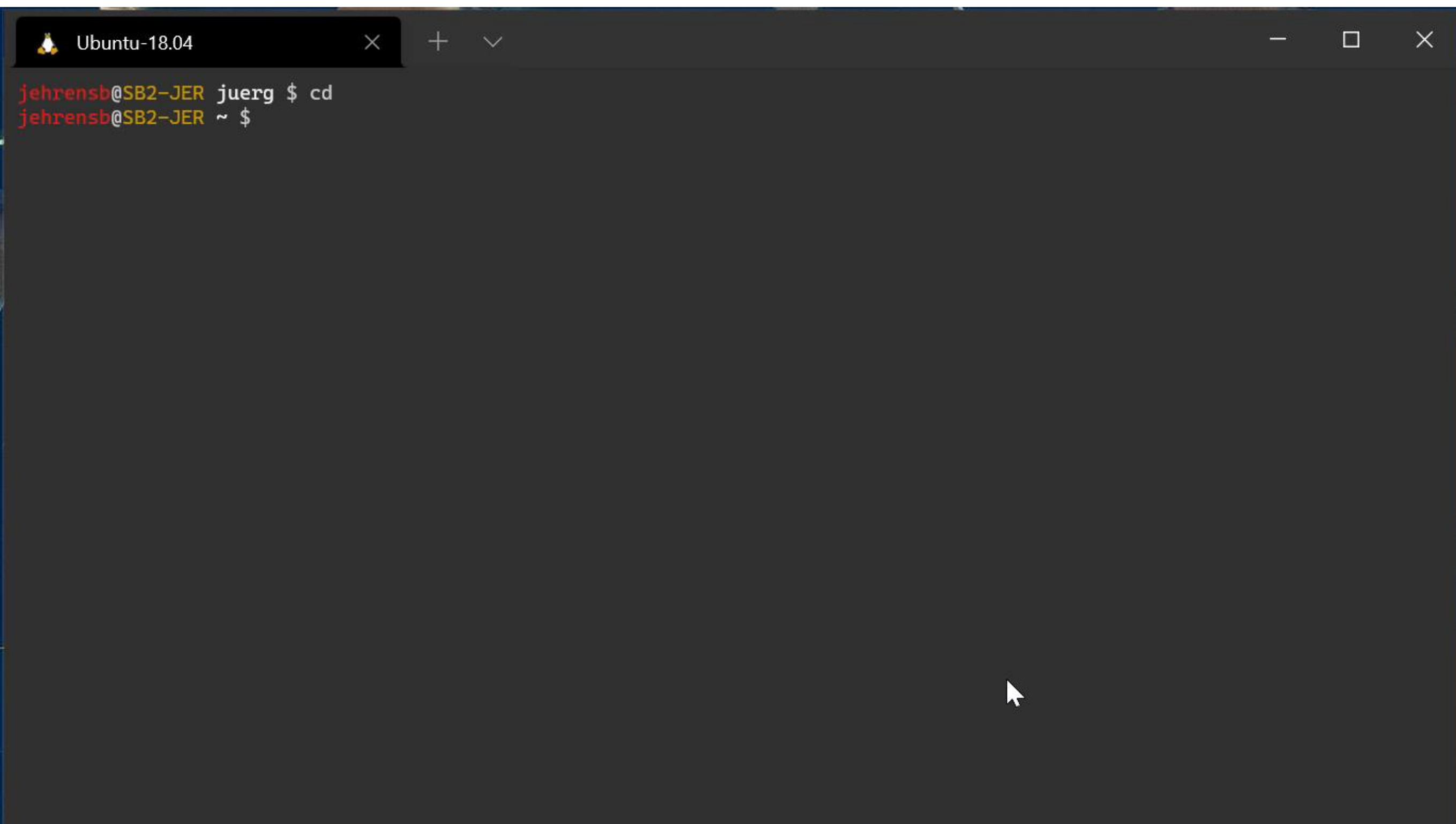
Step 1 -- Source Host to Relay Host

The source host performs a DNS lookup on XYZ.COM (the destination address) and finds DNS MX records specifying xyz.com as the best preference and foo.com as a lower preference. It attempts to open a connection to xyz.com and fails. It then opens a connection to foo.com, with the following dialogue:

```
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:<JQP@bar.com>
S: 250 OK
C: RCPT TO:<Jones@XYZ.COM>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: Date: Thu, 21 May 1998 05:33:29 -0700
C: From: John Q. Public <JQP@bar.com>
C: Subject: The Next Meeting of the Board
C: To: Jones@xyz.com
C:
C: Bill:
C: The next meeting of the board of directors will be
C: on Tuesday.
C: John.
C: .
S: 250 OK
C: QUIT
S: 221 foo.com Service closing transmission channel
```



Demo send email



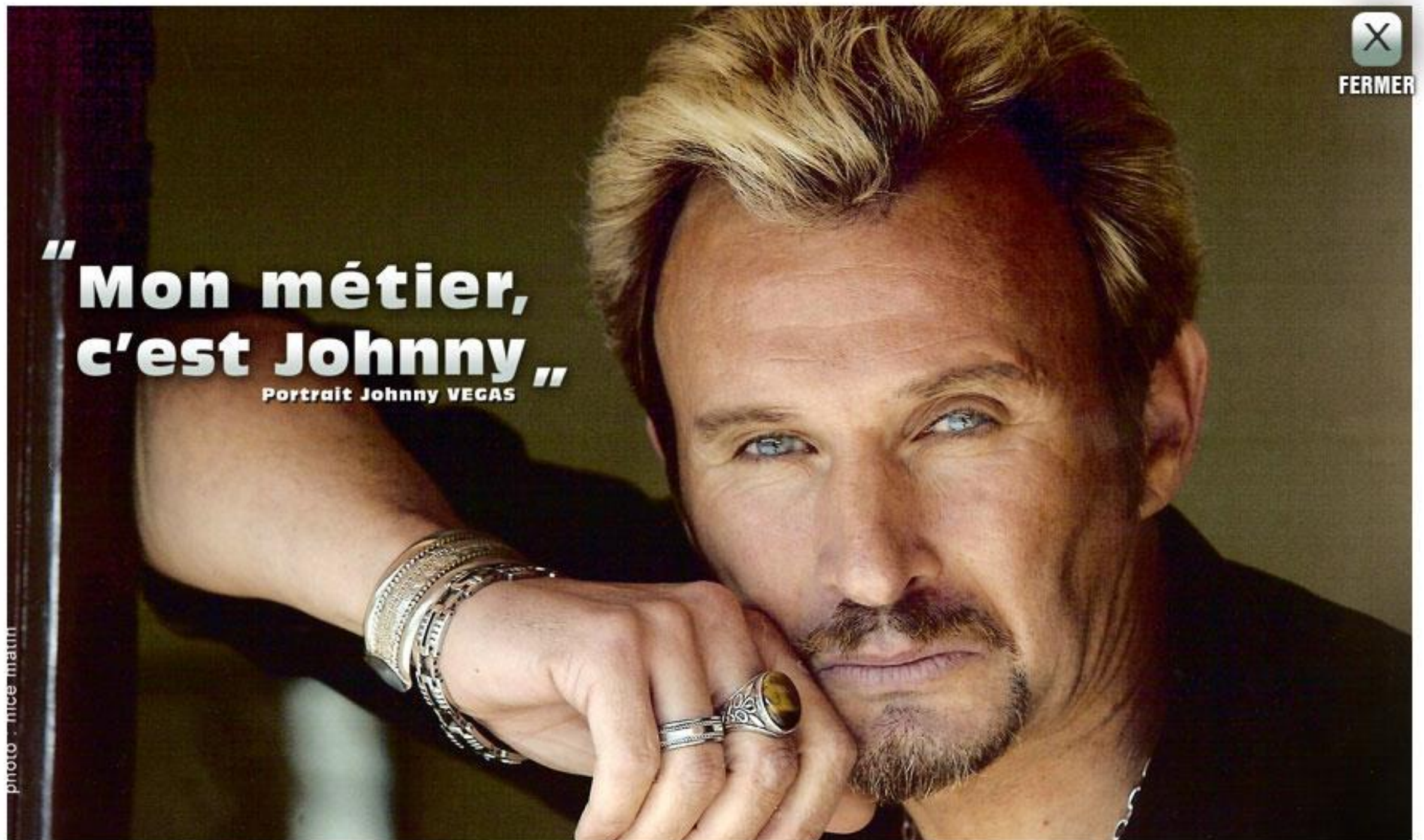
```
Ubuntu-18.04 × + ∨  
jehrensb@SB2-JER juerg $ cd  
jehrensb@SB2-JER ~ $
```

A terminal window titled "Ubuntu-18.04" with standard window controls. The terminal shows a user "jehrensb" at host "SB2-JER" in the directory "~". The user has executed the command "cd" and the prompt has changed to "juerg \$".



SMTP Servers for experiments






Mock Servers



https://github.com/tweakers/MockMock

 **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?



Teaching-HEIGVD-RES-2021-Labo-SMTP

☰ README.md

- **A brief description of your project:** if people exploring GitHub find your repo, without a prior knowledge of the RES course, they should be able to understand what your repo is all about and whether they should look at it more closely.
- **Instructions for setting up a mock SMTP server (with Docker - which you will learn all about in the next 2 weeks).** The user who wants to experiment with your tool but does not really want to send pranks immediately should be able to use a mock SMTP server. For people who are not familiar with this concept, explain it to them in simple terms. Explain which mock server you have used and how you have set it up.
- **Clear and simple instructions for configuring your tool and running a prank campaign.** If you do a good job, an external user should be able to clone your repo, edit a couple of files and send a batch of e-mails in less than 10 minutes.
- **A description of your implementation:** document the key aspects of your code. It is probably a good idea to start with a class diagram. Decide which classes you want to show (focus on the important ones) and describe their responsibilities in text. It is also certainly a good idea to include examples of dialogues between your client and an SMTP server (maybe you also want to include some screenshots here).

References

- [MockMock server](#) on GitHub. Pay attention to this [pull request](#). While it has not been merged, it will give you the solution to compile the project on your machine.
- The [mailtrap](#) online service for testing SMTP
- The [SMTP RFC](#), and in particular the [example scenario](#)
- Testing SMTP with TLS: `openssl s_client -connect smtp.mailtrap.io:2525 -starttls smtp -crlf`





Ubuntu-18.04



jehrensb@SB2-JER MockMock \$



HOW IT WORKS

PRICING

API

BLOG

FAQ

HELP

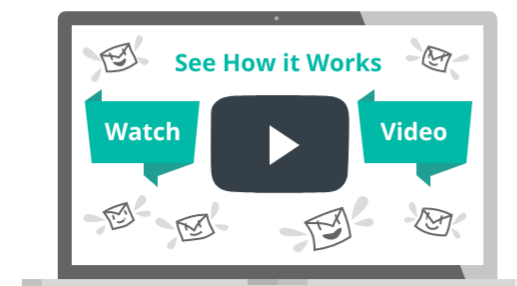
Log in

Sign up



SAFE

*email testing
for dev teams*



End of chapter

☰ README.md

2	Java IO - part 1	Java IO
3	Java IO - part 2	Java IO (grade, weight 1)
4	TCP programming	Protocol design exercise (no grade)
5	TCP programming	Protocol implementation exercise (no grade)
6	Test 1	SMTP lab
Eastern break		
7	SMTP	SMTP lab
8	Web casts: HTTP Protocol + intro to Docker	SMTP lab
9	Web casts: HTTP Protocol + intro to Docker	SMTP lab (grade, weight 1)
10	Live: HTTP infrastructure	HTTP infra lab
11	HTTP infra lab (grade)	HTTP infra lab
12	Test 2	HTTP infra lab
13	HTTP infra lab (grade)	HTTP infra lab (grade, weight 3)
14	Live: UDP programming	UDP Lab (orchestra)
15	UDP Lab (orchestra)	UDP Lab (orchestra)
16	Semester review & exam prep	UDP Lab (orchestra) (grade, weight 1)

