Welcome to

0. Introduction

Security in Web Development Elective, KEA

Henrik Kramselund he/him han/ham xhek@kea.dk @kramse

Slides are available as PDF, kramse@Github

0-Introduction-security-in-web-development.tex in the repo security-courses

Contact information

- Henrik Kramselund , internet samurai mostly networks and infosec
- Network and security consultant Zencurity, teach at KEA and activist
- Master from the Computer Science Department at the University of Copenhagen, DIKU
- Email: xhek@kea.dk Mobile: +45 2026 6000

You are welcome to drop me an email

Goals for today

- Welcome, course goals and expectations
- Prepare Virtual Machines - hope you brought a laptop
- Create a good starting point for learning
- Concrete Expectations
- Prepare tools for the exercises

Photo by Thomas Galler on Unsplash

Plan for today

- Create a good starting point for learning
- Introduce lecturer and students
- Expectations for this course
- Literature list walkthrough
- Prepare tools for the exercises
- Kali and Debian Linux introduction
- Story time Hacking

Exercise theme: Get tools up and running

- Debian Linux installation
- Git tutorials, Python, Ansible

Linux is a toolbox we will use and participants will use virtual machines

Time schedule

Official times:

- 12:00 - 13:30 Session 1
- 15min break
- 13:45 - 15:15 Session 2
- 15min break
- 15:15 - 16:00 45min Session 3

In practice: We will aim at one break at least for every 45min, and we will also aim at doing exercises. Especially if working remote we need for you to be activated and not sit for 4 hours straight listening.

Course Materials

This material is in multiple parts:

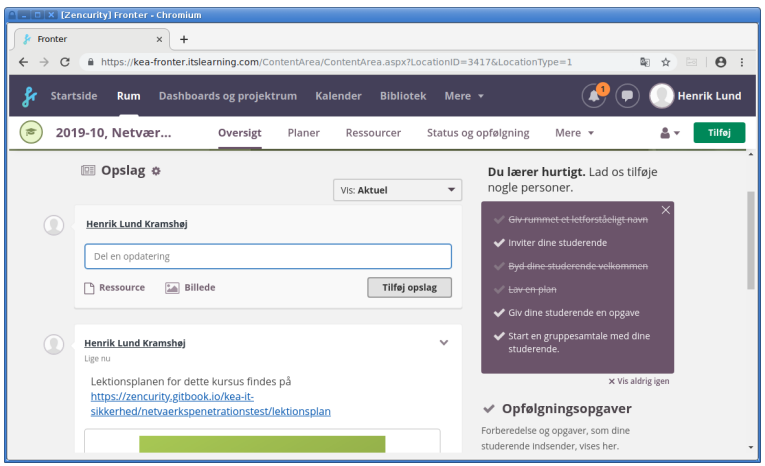
- Slide shows - presentation - this file
- Exercises - PDF which is updated along the way

Books listed in the lecture plan and here

Additional resources from the internet

Note: the presentation slides are not a substitute for reading the books, papers and doing exercises, many details are not shown

Fronter Platform



We will use fronter a lot, both for sharing educational materials and news during the course.

You will also be asked to turn in deliverables through fronter

`https://kea-fronter.itslearning.com/`

If you haven't received login yet, let us know

Course: Security in Web Development elective

Teaching dates: mostly tuesdays and thursdays 17:00 - 20:30
1/2, 8/2, 15/2, 22/2, 1/3, 8/3, 15/3, 22/3, 29/3, 5/4, 19/4, 26/4, 3/5, 10/5, 17/5

Exam: to be found 2022

Photo by Pawel Janiak on Unsplash

Deliverables and Exam

- Exam
- Individual: Oral based on curriculum
- Graded (7 scale)
- Exam is 30 minutes in total, including pulling the question and grading
- Count on being able to present talk for about 10 minutes
- Prepare material (keywords, examples, exercises, wireshark captures) for different topics so that you can use it to help you at the exam

- Deliverables:
- 2 Mandatory assignments
- Mandatory assignments are required in order to be entitled to the exam.

Course Description

Course description

Security in web development is designed to give the students an idea of some of the challenges that web developers face when implementing web applications. It also gives some suggestions on how to handle these challenges, and what to be especially aware of.

Knowledge

The goal is that the student gains knowledge of:

- How hackers exploit web applications
- Basic web application security concepts
- Basic principles of cryptography
- Collecting information about new attack patterns
- Applying basic security assessment Skills

Learning objectives

- Understand basic web application security concepts
- Understand how hackers exploit web applications
- Understand the principle of layered security
- Spot potential security flaws in web applications
- Use best practice on some web security challenges
- Apply risk management with focus on IT-security

Competences

The goal is that the student is able to:

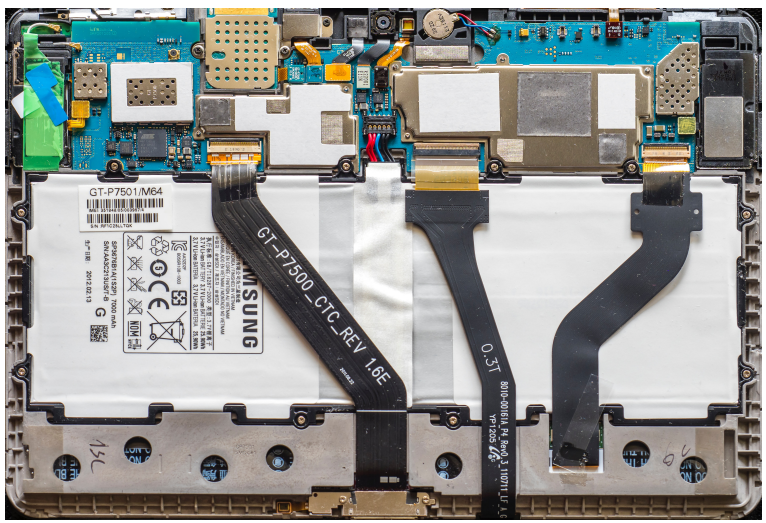
The goal is that the student is able to build a full stack web application designed with security in mind, and by applying secure principles

Content

- Hacking in general
- History, cases, vulnerability info etc
- Basic Applied Cryptography
  Symmetric and asymmetric encryption
  TLS (create certificate and install it on server)
  Hashing and salting
- Security principles, least privilege etc.
- Security touchpoints

- Attack patterns
  SQL injection, XSS, XXE, XSRF, Client side manipulation, Session hijacking, DoS, DDoS
- Linux security
  Basic CLI (folders, privileges), basic firewall (iptables/ufw), basic servers (SSH, Apache, MySQL, Nginx)
  Server security settings (Apache, Nginx)

What is Infrastructure – Software



- Enterprises today have a lot of computing systems supporting the business needs
- These are very diverse and often discrete systems

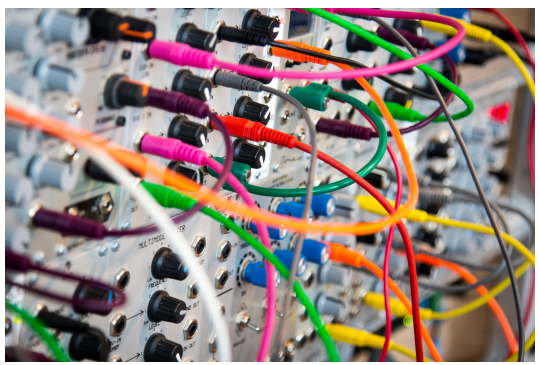
Photo by Alexander Schimmeck on Unsplash

Business Challenges

- Accumulation of software
- Legacy systems
- Partners
- Various types of data
- Employee churn, replacement

Photo by Adam Bignell on Unsplash

Software Challenges



- Complexity
- Various languages
- Various programming paradigms, client server, monolith, Model View Controller
- Conflicting data types and available structures
- Steam train vs electric train

Photo by John Barkiple on Unsplash

Developers Challenges

- Work in teams across organisation - and partners, vendors, sub-contractors
- Work with legacy systems, old technology
- Learn new Technologies

Photo by Kelly Sikkema on Unsplash

Integration Challenges

- Enable communication between components
- Need mediator, interpreter, translator
- Recognize standard patterns

Photo by Thomas Drouault on Unsplash

Exercises

Exercise theme: Virtual Machines allows us play with tech

Hardware

Since we are going to be doing exercises, each team will need virtual machines.

The following are recommended systems:

- One VM based on Debian, running software servers and web applications
- One VM based on Kali Linux, running hacking tools – primary tool is Burp Suite and browser
- Setup instructions and help `https://github.com/kramse/kramse-labs`

Linux is a toolbox we will use and participants will use virtual machines

PS We will from time to time have exercises, groups dont need to be the same each time.

Goals and plans

> "A goal without a plan is just a wish."
> Antoine de Saint-Exupéry

I want this course to

- Include everything listed in contents above
- Be practical – you can do something useful
- Kickstart your journey into Web Security
  Getting a practical book with pointers about the subject
- Present a lot of useful sources and tools
- Prepare you for production use of the knowledge

We have a lot of flexibility.

Prerequisites

This course includes exercises and getting the most of the course requires the participants to carry out these practical exercises

We will use Linux for some exercises but previous Linux and Unix knowledge is not needed

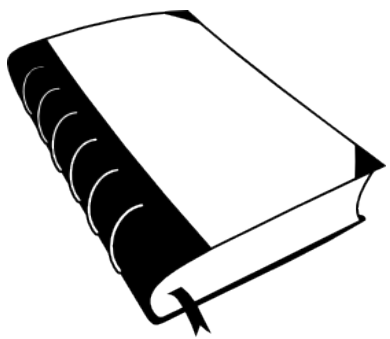
It is recommended to use virtual machines for the exercises

Security and most internet related security work has the following requirements:

- Network experience
- Server experience
- TCP/IP principles - often in more detail than a common user
- Programming is an advantage, for automating things
- Some Linux and Unix knowledge is in my opinion a **necessary skill** for infosec work
  – too many new tools to ignore, and lots found at sites like Github and Open Source written for Linux

Primary literature

Primary literature:



Free graphics by Lumen Design Studio

- *Web Application Security*, Andrew Hoffman, 2020, ISBN: 9781492053118 called WAS
  This book is currently available for "free"if you give your email address:
  `https://www.nginx.com/resources/library/web-application-security/`

It is recommended to buy the *Pwning OWASP Juice Shop* Official companion guide to the OWASP Juice Shop, but it is also available online for free.

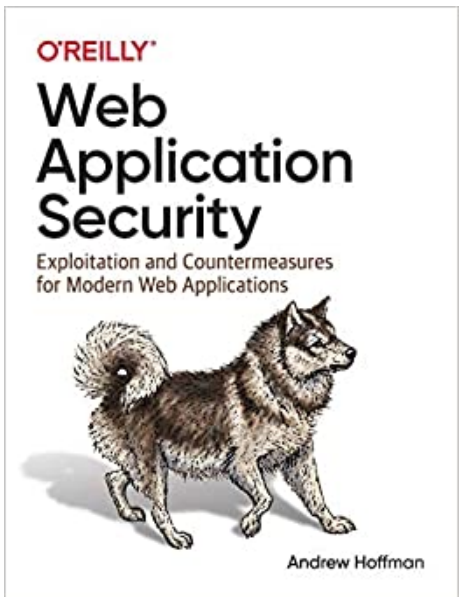
From `https://leanpub.com/juice-shop](https://leanpub.com/juice-shop` - suggested price USD 5.99

Course overview

We will now go through a little from the Table of Contents in the books.

and the lecture plan in Fronter
(Source is also in Git `https://github.com/kramse/kea-it-sikkerhed`)

Web Application Security

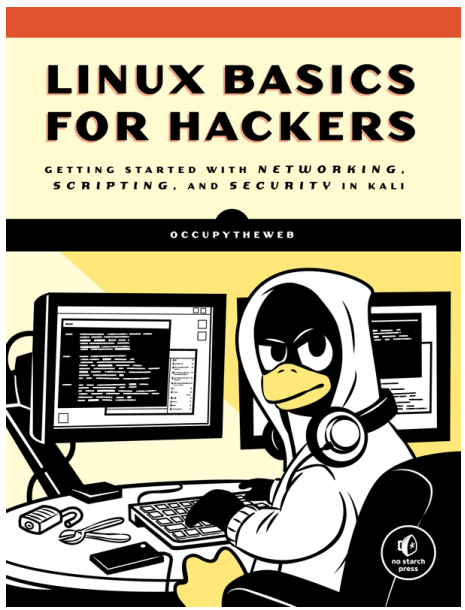


Web Application Security, Andrew Hoffman, 2020, ISBN: 9781492053118 called WAS

Supporting literature books

- *Linux Basics for Hackers Getting Started with Networking, Scripting, and Security in Kali*
  OccupyTheWeb, December 2018, 248 pp. ISBN-13: 978-1-59327-855-7 - shortened LBfH
- *The Debian Administrator's Handbook*, Raphaël Hertzog and Roland Mas
  `https://debian-handbook.info/` - shortened DEB
- *Kali Linux Revealed Mastering the Penetration Testing Distribution*
  Raphaël Hertzog, Jim O'Gorman - shortened KLR

Linux Basics for Hackers (LBhf)



Linux Basics for Hackers Getting Started with Networking, Scripting, and Security in Kali by OccupyTheWeb December 2018, 248 pp. ISBN-13: 9781593278557

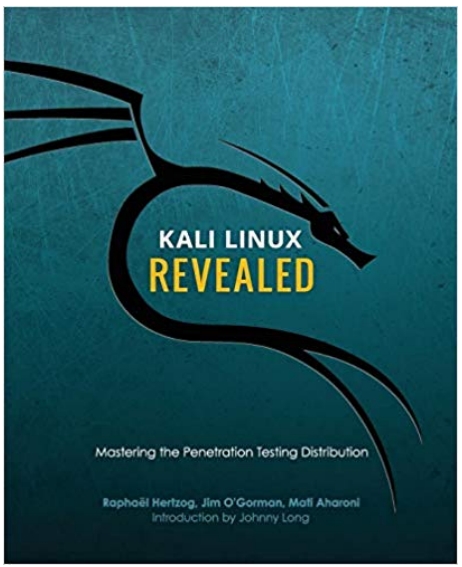
`https://nostarch.com/linuxbasicsforhackers` Not curriculum but explains how to use Linux

The Debian Administrator's Handbook (DEB)

The Debian Administrator's Handbook, Raphaël Hertzog and Roland Mas
`https://debian-handbook.info/` - shortened DEB

Not curriculum but explains how to use Debian Linux

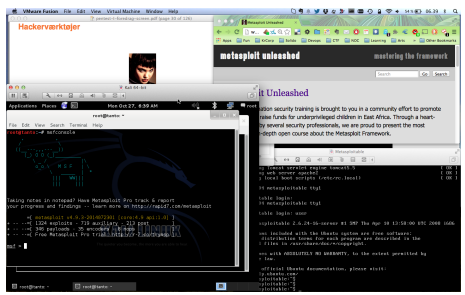
Kali Linux Revealed (KLR)



Kali Linux Revealed Mastering the Penetration Testing Distribution

`https://www.kali.org/download-kali-linux-revealed-book/`
Not curriculum but explains how to install Kali Linux

Hackerlab Setup



- Hardware: modern laptop CPU with virtualisation
  Dont forget to enable hardware virtualisation in the BIOS
- Virtualisation software: VMware, Virtual box, HyperV pick your poison
- Linux server system: Debian amd64 64-bit `https://www.debian.org/`
- Setup instructions can be found at `https://github.com/kramse/kramse-labs`

It is enough if these VMs are pr team

Technologies used in this course

The following tools and environments are examples that may be introduced in this course:

- Programming languages and frameworks Java, Python, regular expressions
- Development environments – choose your own IDE / Editor – I use **Atom**
- Networking and network protocols: TCP/IP, HTTP, DNS
- Formats XML, JSON, CSV, raw text, web scraping
- Web technologies and services: REST, API, HTML5, CSS, JavaScript
- Tools like cURL, Git and Github
- Hacking tools Nikto, sslscan, Burp Suite
- Optional - but demoed aggregated example platforms: Elastic stack, logstash, elasticsearch, kibana, Filebeat
- Cloud and virtualisation **Docker** – are similar and can be added

This list is not complete or a promise

OWASP Juice Shop Project

We will also use the OWASP Juice Shop Tool Project as a running example. This is an application which is modern AND designed to have security flaws.

Read more about this project at: `https://www.owasp.org/index.php/OWASP_Juice_Shop_Project`
`https://github.com/bkimminich/juice-shop`

It is recommended to buy the Pwning OWASP Juice Shop Official companion guide to the OWASP Juice Shop from `https://leanpub.com/juice-shop` - suggested price USD 5.99

Mixed exercises

Then we will do a mixed bag of exercises to introduce technologies, find your current knowledge level with regards to:

- Linux
- Linux command line
- Git, Python and Ansible
- Elasticsearch – how to run a *service*
- Running Java on Linux – environment variables?!
- Ansible provisioning – installing and configuring software for production

Note: today we will consider all these optional, we wont be able to do them all

Later we will return to them!

Command prompts in Unix

Shells :

- sh - Bourne Shell
- bash - Bourne Again Shell, often the default in Linux
- ksh - Korn shell, original by David Korn, but often the public domain version used
- csh - C shell, syntax similar to C language
- Multiple others available, zsh is very popular

Windows have `command.com`, `cmd.exe` but PowerShell is more similar to the Unix shells

Used for scripting, automation and programs

Command prompts

```
[hlk@debian hlk]$ id
uid=6000(hlk) gid=20(staff) groups=20(staff), 0(wheel), 80(admin), 160(cvs)
[hlk@debian hlk]$ sudo -s
[root@debian hlk]#
[root@debian hlk]# id
uid=0(root) gid=0(wheel) groups=0(wheel), 1(daemon), 20(staff), 80(admin)
[root@debian hlk]#
```

Note the difference between running as root and normal user. Usually books and instructions will use a prompt of hash mark # when the root user is assumed and dollar sign $ when a normal user prompt.

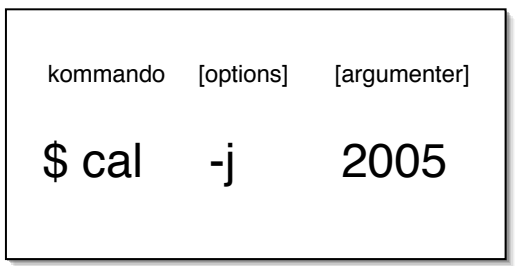
Command syntax

```
echo [-n] [string ...]
```

Commands are written like this:

- Always begin with the command to execute, like `echo` above
- Options typically short form with single dash `-n`
- or long options `--version`
- Some commands allow grouing options, `tar -c -v -f` becomes `tar -cvf`
  NOTE: some options require parameters, so `tar -c -f filename.tar` not equal to `tar -fc filename.tar`
- Optional options are in brackets `[ ]`
- Output can be saved using redirection, into new file/overwrite `echo hello > file.txt` or append `echo hello >> file.txt`
- Read from files `wc -l file.txt` or pipe output into input `cat file.txt | wc -l`
  `wc` is word count, and option l is count lines

Unix Manual system



It is a book about a Spanish guy called Manual. You should read it. – Dilbert

Manual system in Unix is always there!

Key word search `man -k` see also `apropos`

Different sections, can be chosen

See `man crontab` the command vs the file format in section 5 `man 5 crontab`

A manual page

```
NAME
     cal - displays a calendar
SYNOPSIS
     cal [-jy] [[month]  year]
DESCRIPTION
   cal displays a simple calendar.  If arguments are not specified, the cur-
   rent month is displayed.  The options are as follows:
   -j      Display julian dates (days one-based, numbered from January 1).
   -y      Display a calendar for the current year.

The Gregorian Reformation is assumed to have occurred in 1752 on the 3rd
of September.  By this time, most countries had recognized the reforma-
tion (although a few did not recognize it until the early 1900's.)  Ten
days following that date were eliminated by the reformation, so the cal-
endar for that month is a bit unusual.
```

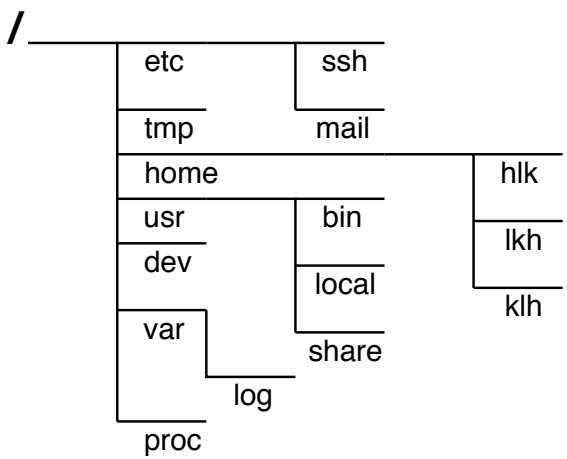
The year 1752

```
  user@Projects:$ cal 1752
...
        April                 May                   June
  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
              1  2  3   4                 1  2      1  2  3  4  5  6
   5  6  7  8  9 10 11   3  4  5  6  7  8  9   7  8  9 10 11 12 13
  12 13 14 15 16 17 18  10 11 12 13 14 15 16  14 15 16 17 18 19 20
  19 20 21 22 23 24 25  17 18 19 20 21 22 23  21 22 23 24 25 26 27
  26 27 28 29 30        24 25 26 27 28 29 30  28 29 30
                        31
         July                  August                September
  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
              1  2  3   4                    1        1  2 14 15 16
   5  6  7  8  9 10 11   2  3  4  5  6  7  8  17 18 19 20 21 22 23
  12 13 14 15 16 17 18   9 10 11 12 13 14 15  24 25 26 27 28 29 30
  19 20 21 22 23 24 25  16 17 18 19 20 21 22
  26 27 28 29 30 31     23 24 25 26 27 28 29
                        30 31
...
```

Linux configuration in /etc

.

- Command line is a requirement in the *studieordningen* ☺
- Linux and Unix uses a single virtual file system
  `https://en.wikipedia.org/wiki/Unix_filesystem`
- No drive letters like the ones in MS-DOS and Microsoft Windows
- Everything starts at the root of the file system tree / - NOTE: *forward slash*
- One special directory is `/etc/` and sub directories which usually contain a lot of configuration files



Installing software in Debian – apt

```
DESCRIPTION
apt provides a high-level commandline interface for the package management system. It is intended as an end user interface
and enables some options better suited for interactive usage by default compared to more specialized APT tools like apt-
get(8)
and apt-cache(8).

update (apt-get(8))
  update is used to download package information from all configured sources. Other commands operate on this data to e.g.
  perform package upgrades or search in and display details about all packages available for installation.

upgrade (apt-get(8))
  upgrade is used to install available upgrades of all packages currently installed on the system from the sources configured
  via sources.list(5). New packages will be installed if required to satisfy dependencies, but existing packages will never
  be removed. If an upgrade for a package requires the removal of an installed package the upgrade for this package isn't performed.

full-upgrade (apt-get(8))
  full-upgrade performs the function of upgrade but will remove currently installed packages if this is needed to upgrade the
  system as a whole.
```

- Install a program using apt, for example `apt install nmap`

Ansible

From my course materials:
Ansible is great for automating stuff, so by running the playbooks we can get a whole lot of programs installed, files modified - avoiding the Vi editor.

- Easy to read, even if you don't know much about YAML
- `https://www.ansible.com/` and `https://en.wikipedia.org/wiki/Ansible_(software)`
- Great documentation
  `https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html`

Ansible Dependencies

- Ansible based on Python, only need Python installed
  `https://www.python.org/`
- Often you use Secure Shell for connecting to servers
  `https://www.openssh.com/`
- Easy to configure SSH keys, for secure connections

Ansible playbooks

Example playbook content, installing software using APT:

```
apt:
    name: "{{ packages }}"
    vars:
      packages:
        - nmap
        - curl
        - iperf
        ...
```

Running it:

```
cd kramse-labs/suricatazeek
ansible-playbook -v 1-dependencies.yml 2-suricatazeek.yml 3-elasticstack.yml 4-configuration.yml
```

"YAML (a recursive acronym for "YAML Ain't Markup Language") is a human-readable data-serialization language."
https://en.wikipedia.org/wiki/YAML

Python and YAML – Git

- We need to store configurations
- Run playbooks
- Problem: Remember what we did, when, how
- Solution: use git for the playbooks
- Not the only version control system, but my preferred one

Git getting started

Hints:
Browse the Git tutorials on `https://git-scm.com/docs/gittutorial`
and `https://guides.github.com/activities/hello-world/`

- What is git
- Terminology

Note: you don't need an account on Github to download/clone repositories, but having an acccount allows you to save repositories yourself and is recommended.

Demo: Ansible, Python, Git!

Running Git will allow you to clone repositories from others easily. This is a great way to get new software packages, and share your own.

Git is the name of the tool, and Github is a popular site for hosting git repositories.

- Go to `https://github.com/kramse/kramse-labs`
- Lets explore while we talk
- Install Git, Ansible and Python manually using apt
- Then I will use Git to clone a repository
- Using Ansible I can then run a playbook to install the Atom editor

Demo: output from running a git clone

```
user@Projects:tt$ git clone https://github.com/kramse/kramse-labs.git
Cloning into 'kramse-labs'...
remote: Enumerating objects: 283, done.
remote: Total 283 (delta 0), reused 0 (delta 0), pack-reused 283
Receiving objects: 100% (283/283), 215.04 KiB | 898.00 KiB/s, done.
Resolving deltas: 100% (145/145), done.

user@Projects:tt$ cd kramse-labs/

user@Projects:kramse-labs$ ls
LICENSE  README.md  core-net-lab  lab-network  suricatazeek  work-station
user@Projects:kramse-labs$ git pull
Already up to date.
```

for reference at home later

Exercise CHAOS: Don't Panic – have fun learning

> "It is said that despite its many glaring (and occasionally fatal) inaccuracies, the Hitchhiker's Guide to the Galaxy itself has outsold the Encyclopedia Galactica because it is slightly cheaper, and because it has the words 'DON'T PANIC' in large, friendly letters on the cover."

Hitchhiker's Guide to the Galaxy, Douglas Adams

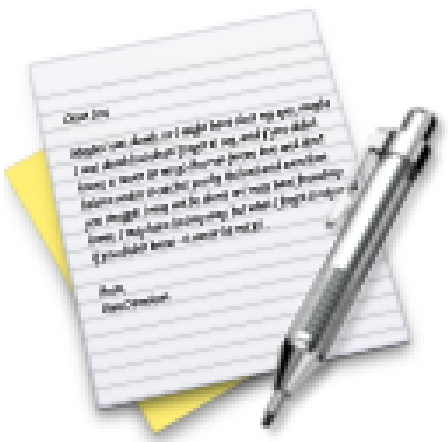
Your lab setup

- Go to GitHub, Find user Kramse, click through kramse-labs
- Look into the instructions for the Virtual Machine – Debian only
- Get the lab instructions, from

  `https://github.com/kramse/kramse-labs/`

TODAY ALL EXERCISES after installing the basic VMs ARE OPTIONAL!

Exercise

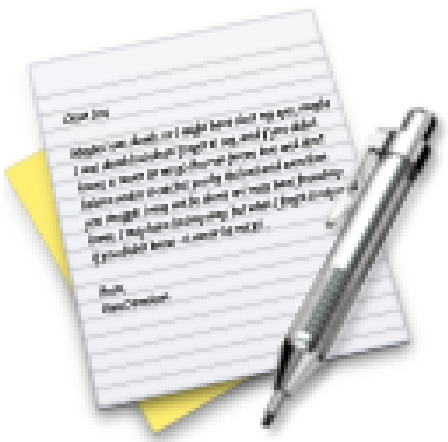


Now lets do the exercise

Download Debian Administrator's Handbook (DEB) Book 10 min

which is number 1 in the exercise PDF.

Exercise

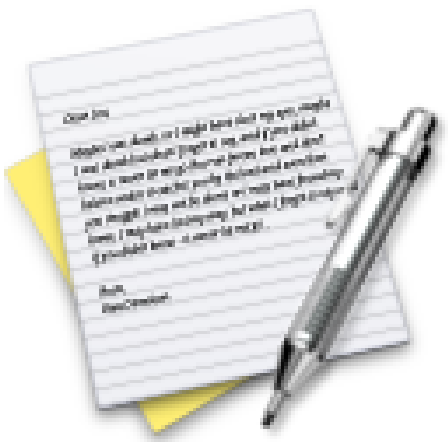


Now lets do the exercise

Download Kali Linux Revealed (KLR) Book 10 min

which is number 2 in the exercise PDF.

Exercise

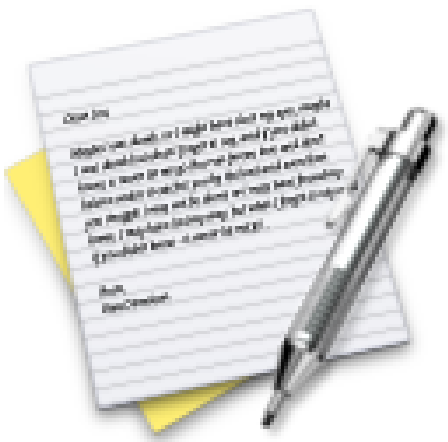


Now lets do the exercise

Check your Debian VM 10 min

which is number **3** in the exercise PDF.

Exercise

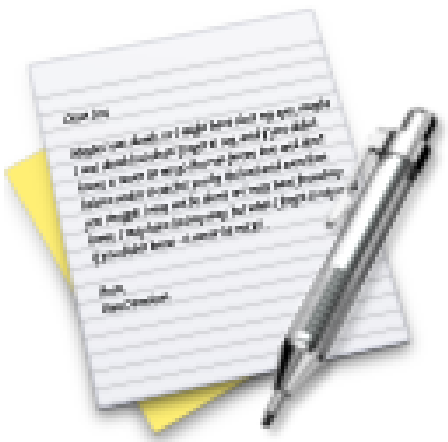


Now lets do the exercise

Check your Kali VM, run Kali Linux 30 min

which is number **4** in the exercise PDF.

Exercise

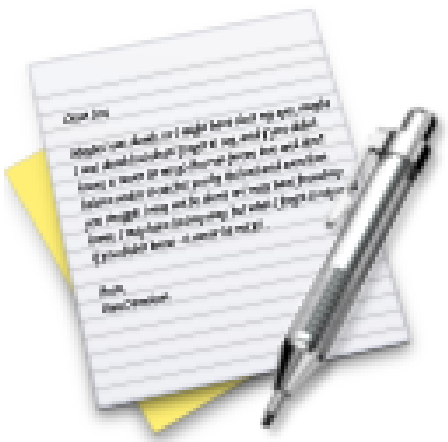


Now lets do the exercise

Investigate /etc 10 min

which is number **5** in the exercise PDF.

Exercise

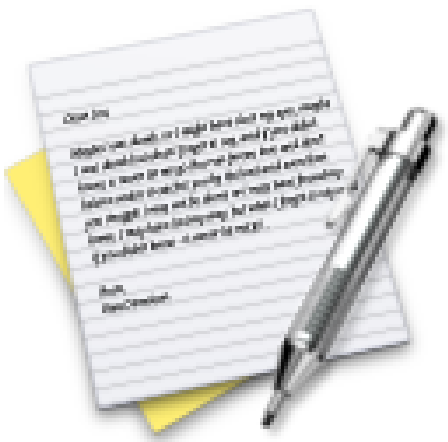


Now lets do the exercise

Enable UFW firewall - 10 min

which is number **6** in the exercise PDF.

Exercise

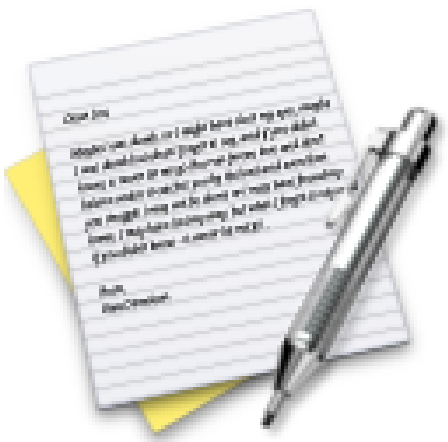


Now lets do the exercise

Git tutorials - 15min

which is number **7** in the exercise PDF.

Exercise

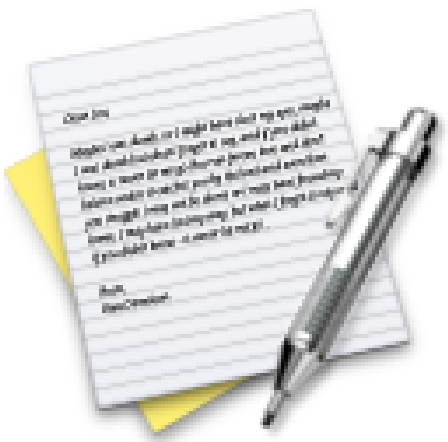


Now lets do the exercise

Install JupyterLab – up to 30min

which is number **8** in the exercise PDF.

Exercise

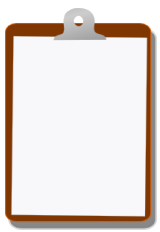


Now lets do the exercise

Optional: Postman API Client 20 min

which is number 11 in the exercise PDF.

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools

Buy the books! Create your VMs