Welcome to

4. Web Application Security: Recon

Security in Web Development Elective, KEA

Henrik Kramselund he/him han/ham hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Github

4-web-app-hacking-recon-security-in-web.tex in the repo security-courses

Goals for today

Todays goals:

- Web Application Hacking – more structured knowledge about common web apps
- Do reconnaissance on web applications
- Talk about common patterns

Photo by Thomas Galler on Unsplash

Plan for today

Subjects: overall Web Application Security: Recon

The Structure of a Modern Web Application

REST APIs, JSON

Authentication and Authorization Systems

Finding domains, subdomains and vhosts

- Structured approach to discovery, information gathering, mapping
- 1) Going over my presentation – first 45min
- 2) The book chapters 2-8, a structured method – next 2x 45min, and blends into last part of today
- 3) Use tools, free to choose those mentioned in book or my recommended
- 4) Work with our insecure OWASP JuiceShop

Exercises: a few tools mentioned in exercises, some things we will do in parallel with the book

Kind of a repeat from last time, but this time you will go through the process.

Reading Summary

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

Part I. Recon, chapters 2-8, very short chapters

2. Introduction to Web Application Reconnaissance
3. The Structure of a Modern Web Application
4. Finding Subdomains
5. API Analysis
6. Identifying Third-Party Dependencies
7. Identifying Weak Points in Application Architectur
8. Part I Summary

Goals: Discovery and Reconnaissance

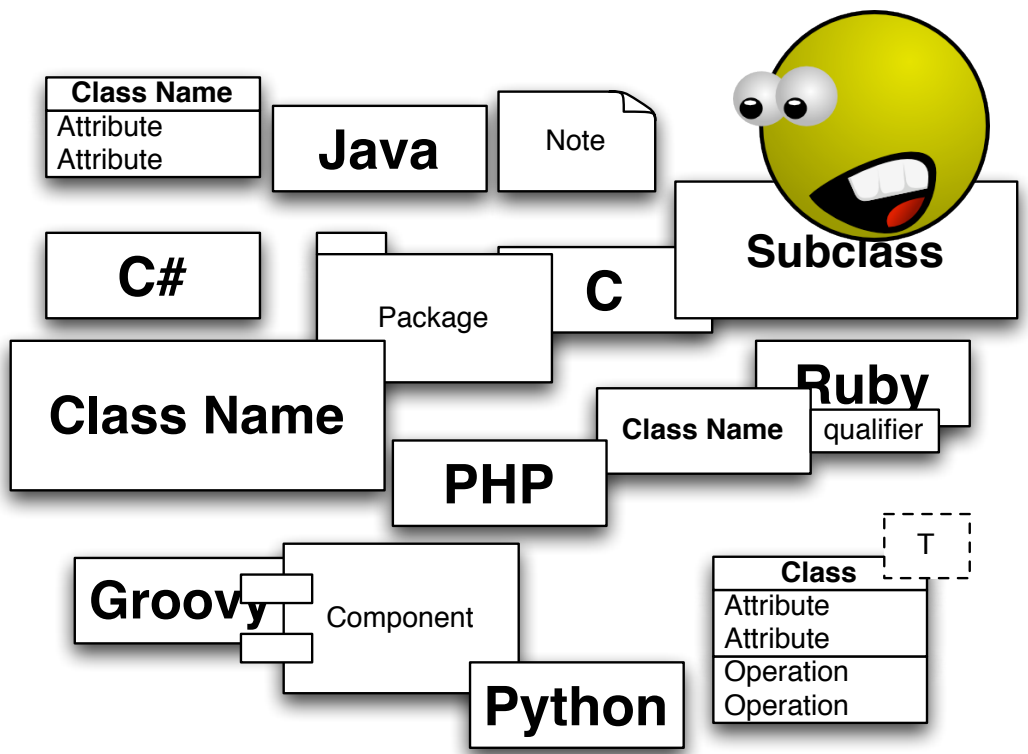
Be able to discover parts of web application environments – Web Application Mapping

Hands-on working with technologies

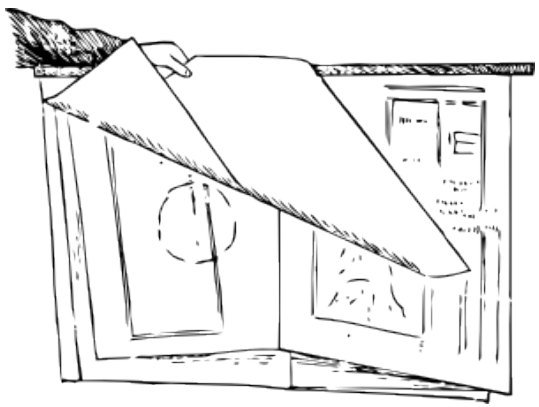
Iron out problems in understanding, catch up. Bring us all up to the same level!

Lifeguard training photo by Margarida CSilva on Unsplash

Selecting Technologies for your enterprise



Why talk about selecting technologies



- A big part of systems integration it to make sure applications can work together
- Data interchange
- Running systems require skills, many different technologies, many humans needed
- Managing complexity with many systems become harder

Later today we will discuss this subject more with the hand-in assigment

Exercise: Start selecting technologies – 30min

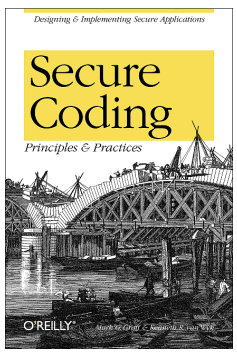
Think about your own choices, previously

Then think about creating a new template application

- Which operating system, and why
- Which language – PHP, Python, Perl, Powershell, JavaScript, ...
- Which supporting frameworks – CSS, HTML, looks and compatability Bootstrap, Material, ...
- Do you need databases and storage? SQL or No-SQL? why / why not
- Other stuff to add? Git, Github, virtual machines, ...

Prepare a short presentation (no slides) explaining why you would select those specific technologies. I will ask some of you, if you like to share with the class.

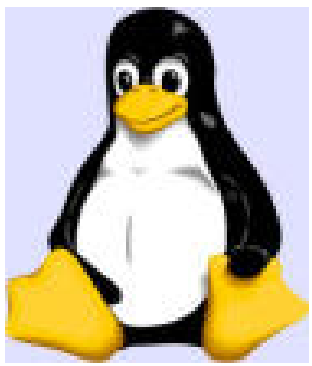
Secure Infrastructures starts with architecture and design



Secure Coding: Principles and Practices af Mark G. Graff, Kenneth R. Van Wyk 2003

- Architecture and design while you are thinking about the application
- Implementation while you are writing the application
- Operations After the application is in production
- Approx. 200 pages, but very dense with information.

Operating Systems



- Applications need to run within some controlled system
- What is an operating system today?
- Is Docker an operating system? What is Docker?

Use the Modern Operating Systems

Newer versions of Microsoft Windows, Mac OS X and Linux

- Buffer overflow protection
- Stack protection, non-executable stack
- Heap protection, non-executable heap
- *Randomization of parameters* stack gap m.v.

Note: these still have errors and bugs, but are better than older versions

Check end-of-life and when updates will stop for each version

OpenBSD has shown the way in many cases
http://www.openbsd.org/papers/

Always try to make life worse and more costly for attackers

Technologies used in enterprises

The following tools and environments are examples that are used in enterprises today:

- Programming languages and frameworks Java, Spring, Python
- Development environments IDE NetBeans / Eclipse / IntelliJ, Atom
- Systems for running Java: TomCat / GlassFish
- Networking and network protocols: TCP/IP, HTTP, DNS
- Formats XML, JSON, WSDL, GRPC, msgpack, protobuf, apache thrift
- Web technologies and services: REST, API, HTML5, CSS
- Tools like cURL, Git and Github
- Integration tools Camel
- Message queueing systems: MQ
- Aggregated example platforms: Elastic stack, logstash, elasticsearch, kibana, grafana
- Cloud and virtualisation Docker, Kubernetes, Azure, AWS, microservices

This list is not complete, but what was discussed in System Integration course!

What about dependencies

- Are you using some special software, or hardware
- Does your application depend on some tools, library that needs help

Data overview XML data, JSON

Photo by Chris Lawton on Unsplash

XML data

> Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification[2] of 1998[3] and several other related specifications[4]—all of them free open standards—define XML.[5]
>
> The design goals of XML emphasize simplicity, generality, and usability across the Internet.[6] It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures[7] such as those used in web services.

Source: https://en.wikipedia.org/wiki/XML

- We have seen XML used for configuration in Apache Tomcat and Camel
- Perfect for computers, less for humans

XML data example - Nmap output

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 7.70 scan initiated Sat Feb 22 23:35:53 2020 as: nmap -oA router -sP 10.0.42.1 -->
<nmaprun scanner="nmap" args="nmap -oA router -sP 10.0.42.1" start="1582410953"
 startstr="Sat Feb 22 23:35:53 2020" version="7.70" xmloutputversion="1.04">
<verbose level="0"/>
<debugging level="0"/>
<host><status state="up" reason="echo-reply" reason_ttl="62"/>
<address addr="10.0.42.1" addrtype="ipv4"/>
<hostnames>
</hostnames>
<times srtt="2235" rttvar="5000" to="100000"/>
</host>
<runstats><finished time="1582410953" timestr="Sat Feb 22 23:35:53 2020" elapsed="0.32"
 summary="Nmap done at Sat Feb 22 23:35:53 2020; 1 IP address (1 host up)
 scanned in 0.32 seconds" exit="success"/><hosts up="1" down="0" total="1"/>
</runstats>
</nmaprun>
```

XML data - documents

> Hundreds of document formats using XML syntax have been developed,[8] including RSS, Atom, SOAP, SVG, and XHTML. XML-based formats have become the default for many office-productivity tools, including Microsoft Office (Office Open XML), OpenOffice.org and LibreOffice (OpenDocument), and Apple's iWork[citation needed]. XML has also provided the base language for communication protocols such as XMPP. Applications for the Microsoft .NET Framework use XML files for configuration, and property lists are an implementation of configuration storage built on XML.[9]

Source: https://en.wikipedia.org/wiki/XML

- Document formats using XML may still be proprietary!
- Documents using XML can be validated, are they well-formed according to the Document Type Definition (DTD)

Transforming XML using XSLT

> XSLT (Extensible Stylesheet Language Transformations) is a language for transforming XML documents into other XML documents,[1] or other formats such as HTML for web pages, plain text or XSL Formatting Objects, which may subsequently be converted to other formats, such as PDF, PostScript and PNG.[2] XSLT 1.0 is widely supported in modern web browsers.[3]

Source: https://en.wikipedia.org/wiki/XSLT

- Can be seen as a mapping between formats, different XML schemas
- Also is Turing complete, is a programming language

XSLT example

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" indent="yes"/>
  <xsl:template match="/persons">
    <root> <xsl:apply-templates select="person"/> </root>
  </xsl:template>
  <xsl:template match="person">
    <name username="{@username}"> <xsl:value-of select="name" /> </name>
  </xsl:template>
</xsl:stylesheet>
```

- XSLT uses XPath to identify subsets of the source document tree and perform calculations. XPath also provides a range of functions
- XSLT functionalities overlap with those of XQuery, which was initially conceived as a query language for large collections of XML documents
  Source: https://en.wikipedia.org/wiki/XSLT

xsltproc example using Nmap

```
$ su -
# apt install nmap xsltproc
# nmap -sP -oA /tmp/router 91.102.91.18
# exit
$ xsltproc /tmp/router.xml > /tmp/router.html
$ firefox /tmp/router.html
```

- We can use the command line tool `xlstproc` for transforming documents
- `apt install xsltproc`
- Its part of the package Libxslt https://en.wikipedia.org/wiki/Libxslt

- Try installing the tools Nmap and `xsltproc` and reproduce the above
- This is an easy tool to try, and very useful too

Data overview JSON

> JavaScript Object Notation (JSON, pronounced / d e sən/; also / d e s n/[note 1]) is an open-standard file format or data interchange format that uses **human-readable text** to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). It is a very common data format, with a diverse range of applications, such as serving as replacement for XML in AJAX systems.[6]

Source: https://en.wikipedia.org/wiki/JSON

- I like JSON much better than XML
- Many web services can supply data in JSON format

JSON example

```
{
  "first name": "John",
  "last name": "Smith",
  "age": 25,
  "address": {
    "street address": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postal code": "10021"
  },
  "phone numbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
  ],
}
```

- This is a basic JSON document, new data attribute-value pairs can be added
  Source: https://en.wikipedia.org/wiki/JSON

Note about frameworks and libraries

```
import xml.etree.ElementTree as ET
tree = ET.parse('testfile.xml')
root = tree.getroot()

print(root.tag)
print('Nmap version: \t\t{:s} '.format(root.attrib['version']))
print('Nmap started: \t\t{:s} '.format(root.attrib['startstr']))
print('Nmap command line: \t{:s} '.format(root.attrib['args']))

hosts = tree.findall('./host')
for host in hosts:
    print(host.tag)
    print(host.attrib)
    for hostvalues in host:
        print(hostvalues.tag)
        print(hostvalues.attrib)
```

- Dont import JSON or XML using home made programs
- Example uses xml.etree.ElementTree from Python https://docs.python.org/3.7/library/xml.etree.elementtree.html

Convert XML to JSON

```
import xml.etree.ElementTree as ET
import json
def etree_to_dict(t):
    d = {t.tag : map(etree_to_dict, t.getchildren())}
    d.update(('@' + k, v) for k, v in t.attrib.items())
    d['text'] = t.text
    return d

tree = ET.parse('testfile.xml')
root = tree.getroot()
mydict = etree_to_dict(root)
print(type(tree))
print(type(root))
print(type(mydict))

print(mydict)

with open('testfile.json', 'w') as json_file:
  json.dump(mydict, json_file)
```

Converting using Python is easy

REST Service

Figure 9.1
An entity service based on the Invoice business entity that defines a functional scope that limits the service capabilities to performing invoice-related processing. This agnostic Invoice service will be reused and composed by other services within the same service inventory in support of different automated business processes that need to process invoice-related data. This particular invoice service contract displays two service capabilities based on primitive methods and two service capabilities based on complex methods.

- Very typical REST URL/method `GET /invoice/{invoice-id}`
- See also https://en.wikipedia.org/wiki/Representational_state_transfer

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

10.1 RESTful services

Table 10.1 The RESTful API for the Rider Auto Parts order web service

Verb	http://rider.com/orders	http://rider.com/orders/{id}
GET	Retrieves a list of all the orders	Retrieves the order with the given ID
PUT	N/A	Updates the order with the given ID
POST	Creates a new order	N/A
DELETE	Cancels all the orders	Cancels the order with the given ID

RESTful services, also known as REST services, has become a popular architectural style used in modern enterprise projects. REST was defined by Roy Fielding in 2000 when he published his paper, and today REST is a foundation that drives the modern APIs on the web. You can also think of it as a modern web service, in which the APIs are RESTful and HTTP based so they're easily consumable on the web.

Source:

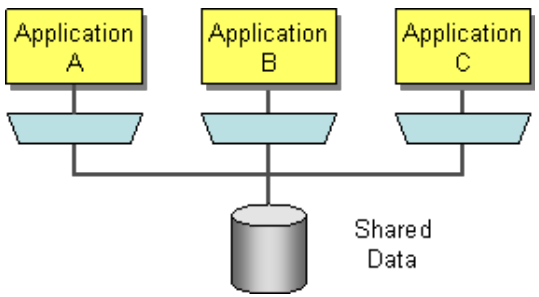
Camel in action, Claus Ibsen and Jonathan Anstey, 2018

Python and REST

```
#!/usr/bin/env python
import requests
r = requests.get('https://api.github.com/events')
print (r.json());
```

- Lets try to use some Python to access a REST service.
- We will use the JSONPlaceholder which is a free online REST API: https://jsonplaceholder.typicode.com/
- Start at the site: https://jsonplaceholder.typicode.com/guide.html and try running a few of the examples with your browser
- Then try using the same URLS in the Requests HTTP library from Python, https://requests.readthedocs.io/en/master/

Shared Database



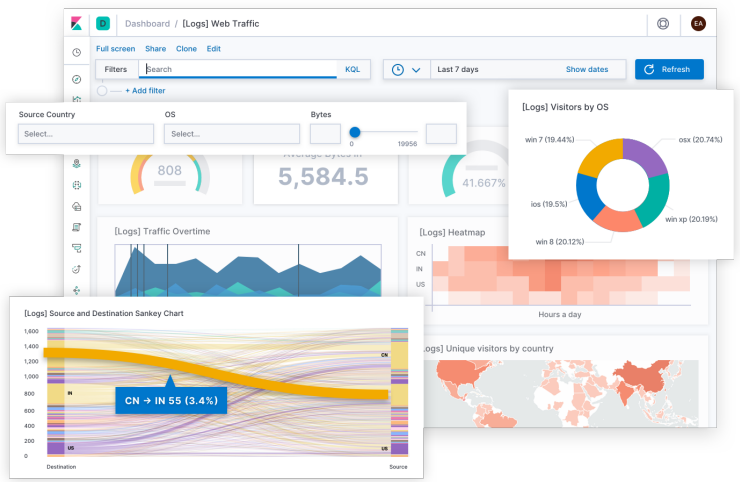
Shared Database — Have the applications store the data they wish to share in a common database.

Common systems and technologies used:

- database management system (DBMS) using Structured Query Language (SQL), relational database examples:

- PostgresSQL, Oracle DM, Microsoft SQL, MySQL https://en.wikipedia.org/wiki/SQL
- NoSQL databases has been a new input with examples like: MongoDB, CouchDB, Redis, RIAK https://en.wikipedia.org/wiki/NoSQL

Elastic stack Kibana



Screenshot from https://www.elastic.co/kibana

Elasticsearch is a search engine and ocument store used in a lot of different systems, allowing cross application integration.

Getting started with Elastic Stack

Easy to get started using the tutorial *Getting started with the Elastic Stack* :

https://www.elastic.co/guide/en/elastic-stack-get-started/current/get-started-elastic-stack.html

Today Elastic Stack contains lots of different parts.

- Elasticsearch - the core engine
- Logstash - a tool for parsing logs and other data.
  https://www.elastic.co/logstash

  "Logstash dynamically ingests, transforms, and ships your data regardless of format or complexity. Derive structure from unstructured data with grok, decipher geo coordinates from IP addresses, anonymize or exclude sensitive fields, and ease overall processing."
- Kibana - a web application for accessing and working with data in Elasticsearch
  https://www.elastic.co/kibana

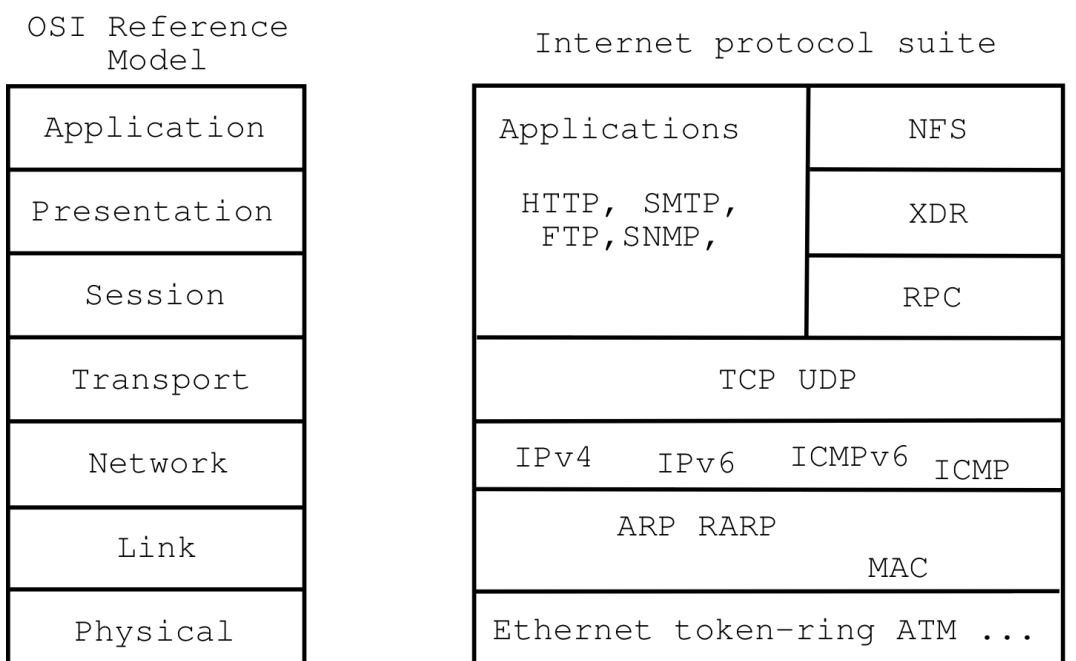
TCP/IP and External Data Representation 1987

External Data Representation (XDR) is a standard data serialization format, for uses such as computer network protocols. It allows data to be transferred between different kinds of computer systems. Converting from the local representation to XDR is called encoding. Converting from XDR to the local representation is called decoding.

Source:

https://en.wikipedia.org/wiki/External_Data_Representation

https://tools.ietf.org/html/rfc1014



Designing and Standardizing HTTP Response Codes

- 100-199 are informational codes used as low-level signaling mechanisms, such as a confirmation of a request to change protocols
- 200-299 are general success codes used to describe various kinds of success conditions
- 300-399 are redirection codes used to request that the consumer retry a request to a different resource identifier, or via a different intermediary
- 400-499 represent consumer-side error codes that indicate that the consumer has produced a request that is invalid for some reason, example 404 file not found
- 500-599 represent service-side error codes that indicate that the consumer's request may have been valid but that the service has been unable to process it for internal reasElasticsearch exposes REST APIs that are used by the UI components and can be called directly to configure and access Elasticsearch features.ons.

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

Exercise: Communicate with HTTP

Try this - use netcat/ncat, available in Nmap package from Nmap.org:

```
$ netcat www.zencurity.com 80
GET / HTTP/1.0

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 01 Feb 2020 20:30:06 GMT
Content-Type: text/html
Content-Length: 0
Last-Modified: Thu, 04 Jan 2018 15:03:08 GMT
Connection: close
ETag: "5a4e422c-0"
Referrer-Policy: no-referrer
Accept-Ranges: bytes
...
```

Basic test tools TCP - Telnet and OpenSSL

Netcat used for terminal connections over TCP, but is clear-text

Netcat can be used for testing connections to many older protocols which uses text commands

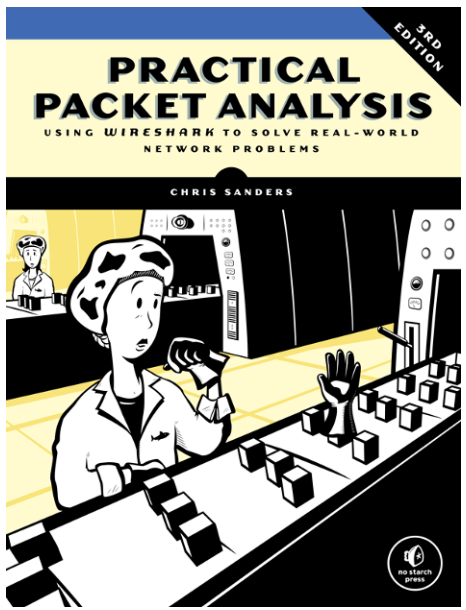
- Create connection to port 25/tcp: `nc mail.kramse.dk 25`
- Create connection to port 80/tcp: `nc www.kramse.dk 80`

Encrypted connections often use TLS and can be tested using OpenSSL command line tool `openssl`

- `openssl s_client -host www.kramse.dk -port 443`
  create connection to port 443/tcp with TLS
- `openssl s_client -host mail.kramse.dk -port 993`
  create connection to port 993/tcp with TLS

Using OpenSSL in client-mode allows the use of the same commands like Telnet after connection

Book: Practical Packet Analysis (PPA)



Practical Packet Analysis, Using Wireshark to Solve Real-World Network Problems by Chris Sanders, 3rd Edition April 2017, 368 pp. ISBN-13: 978-1-59327-802-1

https://nostarch.com/packetanalysis3 highly recommended for basic network skills

Exercise

Now lets do the exercise

Nikto Web Scanner 15 min

which is number **28** in the exercise PDF.

Exercise

Now lets do the exercise

Whatweb Scanner 15 min

which is number 29 in the exercise PDF.

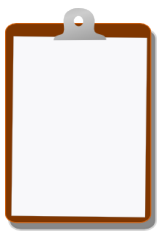
Exercise

Now lets do the exercise

Postman API Client 20 min

which is number 9 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