Welcome to

Git/Github Basics

2020

Henrik Kramselund Jereminsen hkj@zencurity.com @kramse

Slides are available as PDF, kramse@Github
git-github.tex in the repo security-courses

slides are available on Github

Goal for today

- Plan:
- Approx 2h including break
- Inspiration for using Git in your life
- Not an expert in Git

My daily job – Security engineering a job role

```
On any given day, you may be challenged to:
        Create new ways to solve existing production security issues
        Configure and install firewalls and intrusion detection systems
        Perform vulnerability testing, risk analyses and security assessments
        Develop automation scripts to handle and track incidents
        Investigate intrusion incidents, conduct forensic investigations and incident responses
        Collaborate with colleagues on authentication, authorization and encryption solutions
        Evaluate new technologies and processes that enhance security capabilities
        Test security solutions using industry standard analysis criteria
        Deliver technical reports and formal papers on test findings
        Respond to information security issues during each stage of a project's lifecycle
        Supervise changes in software, hardware, facilities, telecommunications and user needs
        Define, implement and maintain corporate security policies
        Analyze and advise on new security technologies and program conformance
        Recommend modifications in legal, technical and regulatory areas that affect IT security
```

Source: `https://www.cyberdegrees.org/jobs/security-engineer/`
also `https://en.wikipedia.org/wiki/Security_engineering`

So I am *not a developer*

What is asset management

> CIS Control 1:
> Inventory and Control of Hardware Assets Actively manage (inventory, track, and correct) all hardware devices on the network so that only authorized devices are given access, and unauthorized and unmanaged devices are found and prevented from gaining access.

Source: `https://www.cisecurity.org/`

- Hardware - bought, connected, loaners, stolen ...
- Software - licenses, procurement, upgrades are cheaper if you have earlier versions
- Virtual assets - business critical data
- ...

Configuration: TLS and VPN settings

```
  # Input from https://github.com/tykling/ansible-roles/blob/master/nginx_server/templates/tls.conf.j2#L6
  ssl_protocols                   TLSv1.2 TLSv1.3;
  ssl_ciphers                     ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-
AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA;
  ssl_prefer_server_ciphers       on;
  ssl_session_cache               shared:SSL:10m;     ssl_session_tickets       off;   ssl_session_timeout    4h;
  ssl_stapling                    on;                 ssl_stapling_verify       on;
  resolver                        105.238.53.1;
  ssl_ecdh_curve secp384r1;                           ssl_dhparam /etc/nginx/dh4096.pem;
  add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
  add_header Referrer-Policy "no-referrer";  add_header X-Content-Type-Options "nosniff";
  add_header X-Frame-Options "DENY";  add_header X-XSS-Protection "1; mode=block";
  add_header Content-Security-Policy "default-src 'self'; script-src 'self; img-src 'self'; object-src 'none'; font-src 'self'; fram
ancestors 'none' https:";
```

- Most have web sites with TLS/HTTPS – how is it configured
- Recommend centralizing and going over settings regularly
- Create a policy for your organisation

Where do we store this information?

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
```

"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

Alternative

Download and install the public signing key:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-k
```

Installing from the APT repository

edit

You may need to install the `apt-transport-https` package on Debian before proceeding:

```
sudo apt-get install apt-transport-https
```

Save the repository definition to `/etc/apt/sources.list.d/elastic-7.x.list`:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | suc
```

My playbooks allow installation of a whole Elastic stack in less then 10 minutes,

compare to:

Getting started with the Elastic Stack

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

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/`
- Lets explore while we talk
- Hint; getting the presentation from today might be a task ☺

Demo: output from running a git clone

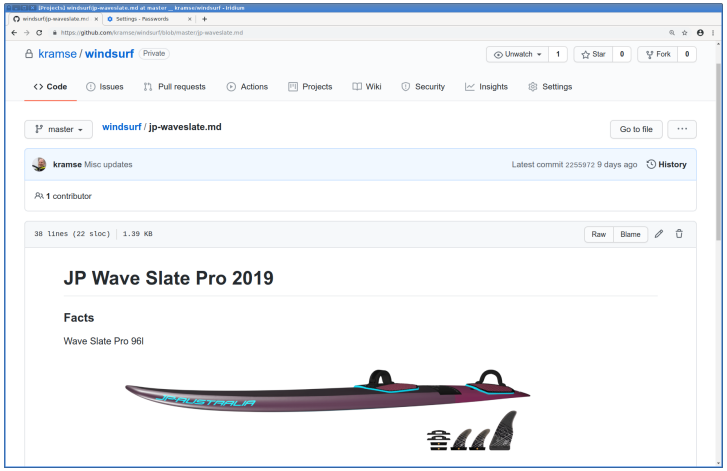
```
user@Projects:tt$  git clone https://github.com/kramse/kramse-labs.git
Cloning into 'kramse-labs'...
remote: Enumerating objects: 283, done.
remote: Total 283 (delta 0), reused 0 (delta 0), pack-reused 283
Receiving objects: 100% (283/283), 215.04 KiB | 898.00 KiB/s, done.
Resolving deltas: 100%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

Sample Project: Windsurf Inventory



- Simple example of using Git, saving some files, and update them
- Problem: Windsurfing equipment and technical details
- Many sources, JP Board, Niel Pryde Sails, Unifiber masts, ...
- Solution: Markdown and Git
- Sorry this repo is closed, you cannot browse it

Sample Project: Presentations

- Problem: Create reusable presentations, allow students to get latest version
- Multiple files, many updates over the years
- Wants: Get PDF output
- Solution: LaTeX and Git
- `https://en.wikipedia.org/wiki/LaTeX`
- Look at the file: `first-presentation.pdf` from `https://github.com/kramse/security-courses`

Git remote

- Git repositories are basically files
- Git repos allow you to clone, download files – `git clone`
- Work locally, including making brances
- Integrate/upload/merge into remote repositories – `git commit` and `git push`
- Git doesn't care if you are one person with multiple laptops, or multiple persons

Git configuration: user information

File: `.gitconfig`

```
[user]
 name = Henrik Kramselund Jereminsen
 email = hkj@kramse.org

[push]
 default = simple
```

- There is a small amount of configuration, user information
- Each repository also has a gitignore

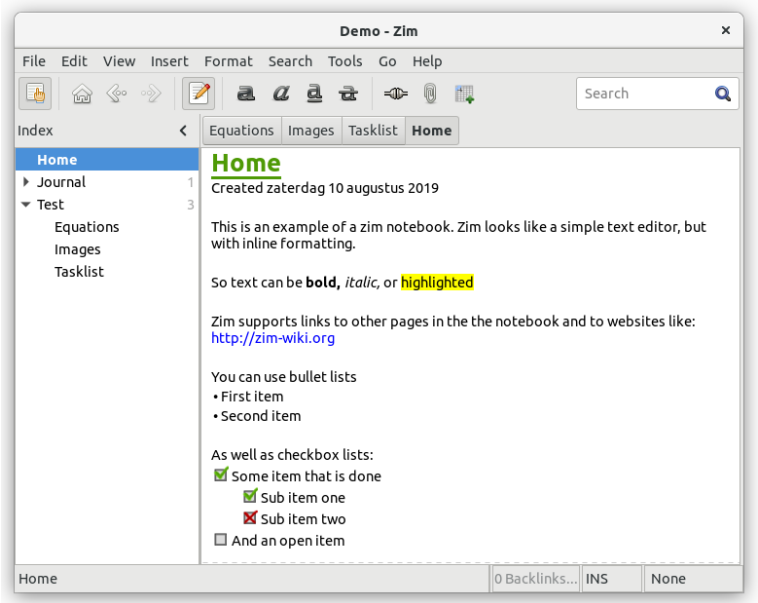
Git configuration: ignore files

File: `.gitignore` top of repository

```
# OS generated files
.DS_Store
.Spotlight-V100
.Trashes
# Latex files
*.aux
*.idx
*.log
*.toc
*.ind
*.out
*.synctex.gz
*.fls
*.fdb_latexmk
```

- Each repository also has a .gitignore, part shown above

Example: Zim Personal Wiki



- My personal ToDo list is using Zim backed by Git repository
- `https://zim-wiki.org/`

Github bragging

Suricata » Suricata Developers Guide »

Contributing This guide describes what steps to take if you want to contribute a patch or patchset to Suricata. Before you start, please review and sign our Contribution Agreement

...

Create your own branch It's useful to create descriptive branch names. If you're working on ticket 123 to improve GeoIP? Name your branch "geoip-feature-123-v1". The -v1"addition for feedback. When incorporating feedback you will have to create a new branch for each pull request. So when you address the first feedback, you will work in "geoip-feature-123-v2"and so on.

Github: Creating a branch with Github Git: Creating a branch with Git

Source: `https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Contributing`

- You can contribute to open source!
- Pull requests - `https://github.com/OISF/suricata/pull/3847` and `https://github.com/zeek/zeek/pull/193`

Questions

"On the Internet, nobody knows you're a dog."

Henrik Kramselund Jereminsen hkj@zencurity.com @kramse