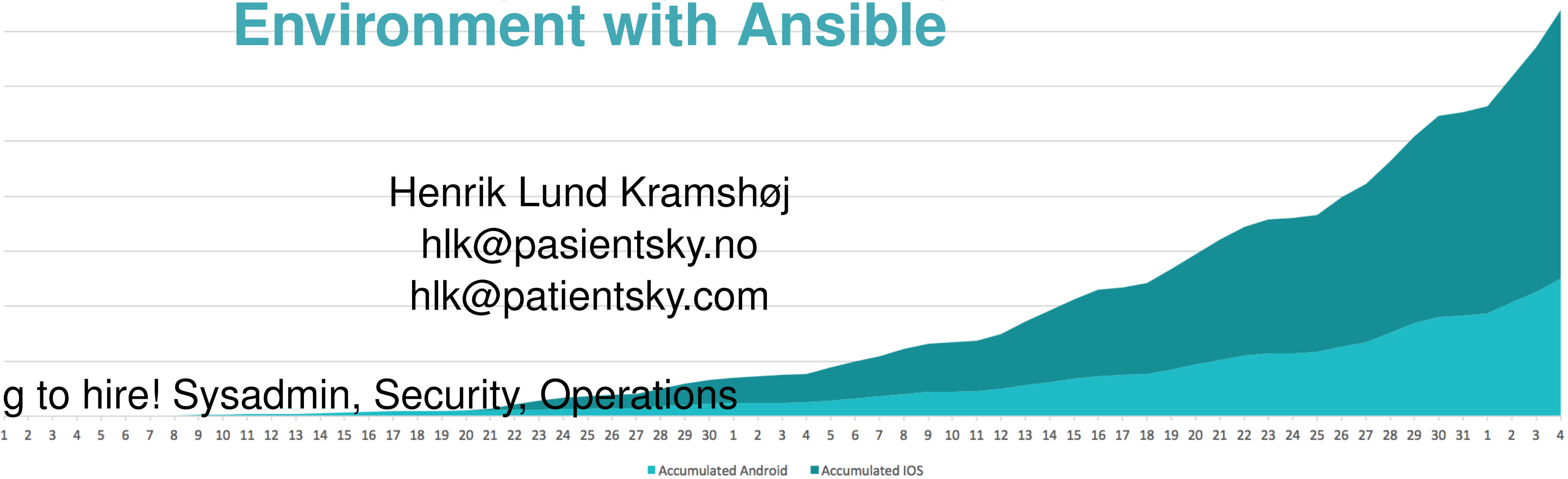
Pasient Sky

Welcome to

Controlling a High Security Environment with Ansible

Henrik Lund Kramshøj
hlk@pasientsky.no
hlk@patientsky.com

BTW we are looking to hire! Sysadmin, Security, Operations



En enklere hverdag med PasientSky

Health data

Doctors appointment

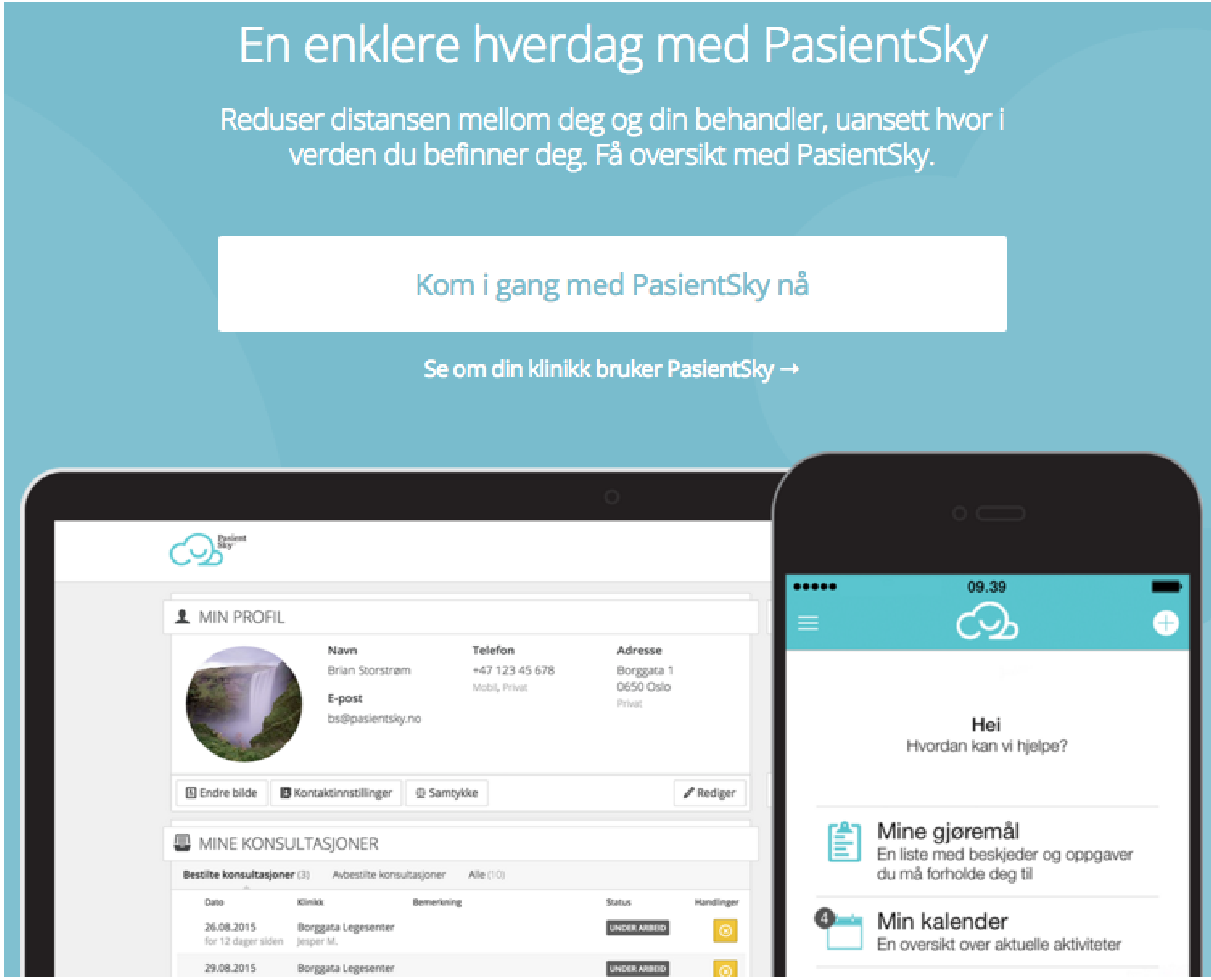
Doctors Journals

Medical data

Prescriptions

...

Obviously this means personal data



En enklere hverdag med PasientSky

Reduser distansen mellom deg og din behandler, uansett hvor i verden du befinner deg. Få oversikt med PasientSky.

Kom i gang med PasientSky nå
Se om din klinikk bruker PasientSky →
Pasient Sky

MIN PROFIL

Navn
Brian Storstrøm

E-post
bs@pasientsky.no

Telefon
+47 123 45 678
Mobil, Privat

Adresse
Borggata 1
0650 Oslo
Privat

Endre bilde
Kontaktinnstillinger
Samtykke
Rediger

MINE KONSULTASJONER

Bestilte konsultasjoner (3) Avbestilte konsultasjoner Alle (10)

Dato	Klinikk	Bemerkning	Status	Handlinger
26.08.2015 for 12 dager siden	Borggata Legesenter Jesper M.		UNDER ARBEID	⊙
29.08.2015	Borggata Legesenter		UNDER ARBEID	⊙

Hei
Hvordan kan vi hjelpe?

Mine gjøremål
En liste med beskjeder og oppgaver du må forholde deg til

Min kalender
En oversikt over aktuelle aktiviteter

Ansible: provisioning, configuration management, security

Open Source ☺

Simple playbooks and ad-hoc commands

Well supported on mainstream OSs

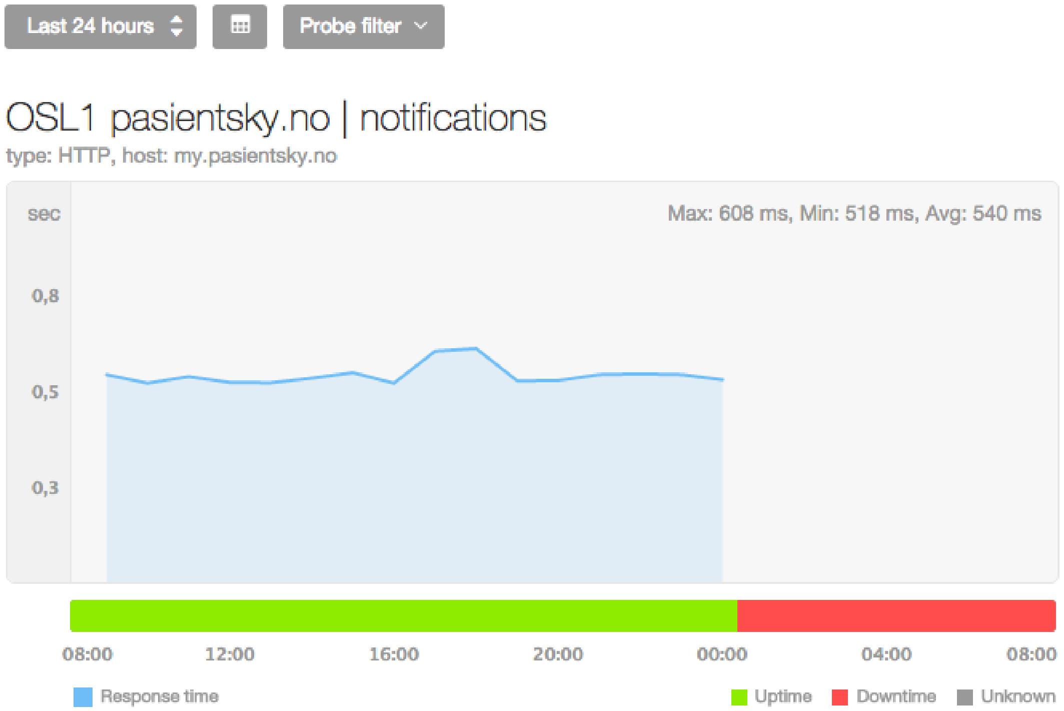
over 200 modules in the core

Supports almost anything which has SSH+Python

Currently 100s of servers

```
http://www.ansible.com/
```

Note: we dont use Tower



OSL1 pasientsky.no | notifications

type: HTTP, host: my.pasientsky.no

Max: 608 ms, Min: 518 ms, Avg: 540 ms

What we learnt about Ansible

.

Easy to get started - YAML playbooks

Easy to configure services

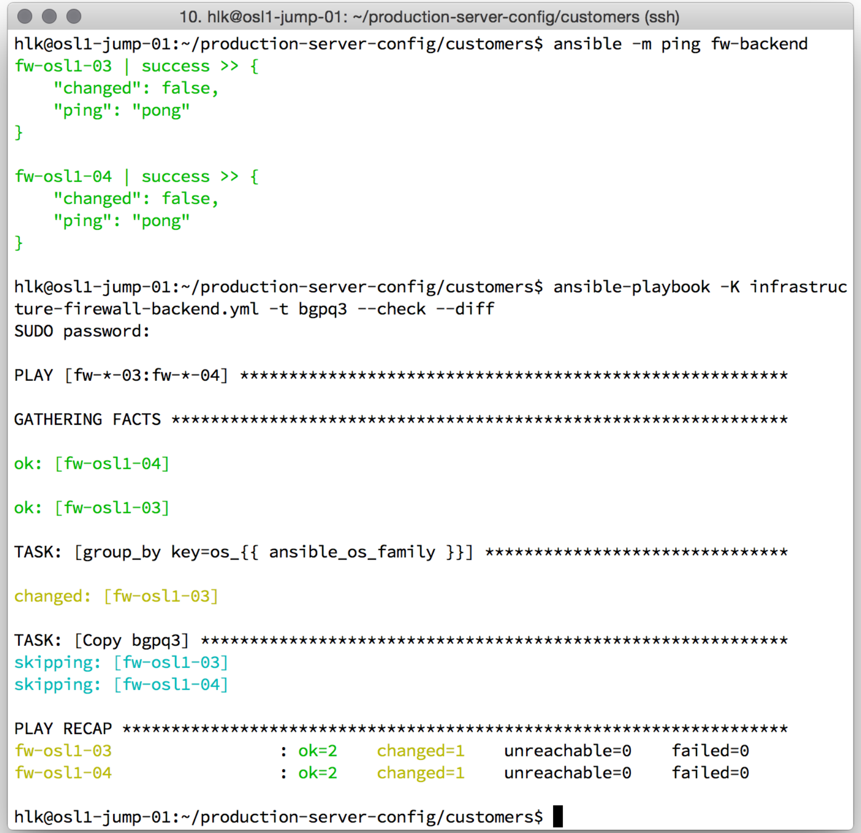
Roles sometimes suck

- too many files in too many directories

Using more flat playbooks nice

Long lists of settings like sysctl

We will continue with Ansible


10. hlk@osl1-jump-01: ~/production-server-config/customers (ssh)

```
hlk@osl1-jump-01:~/production-server-config/customers$ ansible -m ping fw-backend
fw-osl1-03 | success >> {
    "changed": false,
    "ping": "pong"
}

fw-osl1-04 | success >> {
    "changed": false,
    "ping": "pong"
}

hlk@osl1-jump-01:~/production-server-config/customers$ ansible-playbook -K infrastruc
ture-firewall-backend.yml -t bgpq3 --check --diff
SUDO password:

PLAY [fw-*-03:fw-*-04] ********************************************************

GATHERING FACTS ***************************************************************

ok: [fw-osl1-04]

ok: [fw-osl1-03]

TASK: [group_by key=os_{{ ansible_os_family }}] *******************************

changed: [fw-osl1-03]

TASK: [Copy bgpq3] ************************************************************
skipping: [fw-osl1-03]
skipping: [fw-osl1-04]

PLAY RECAP ********************************************************************
fw-osl1-03                 : ok=2    changed=1    unreachable=0    failed=0
fw-osl1-04                 : ok=2    changed=1    unreachable=0    failed=0

hlk@osl1-jump-01:~/production-server-config/customers$ 
```

Why Ansible brings Higher Security

```
# VPN tunnels via customer VPN server
pass quick proto { esp, ah } from any to {{ public_ip_prefix }}.59
pass quick proto { esp, ah } from {{ public_ip_prefix }}.59 to any
```

We can rebuild advanced servers easily

Example complete Log environment from single playbook:

- Syslog servers, PostgreSQL database, Logstash parser, software and rules, Elasticsearch indexing servers
- Nginx with Kibana frontend - in about 150-200 lines of playbook!
- From a base Ubuntu install with no manual steps, other than starting Ansible

Settings are saved in playbooks - documented and readable

Config files are templated and

Across testing, staging and production use **the exact same playbooks/configs**

What Ansible brings in a High Security Environment

We can deploy a complete IDS solution in 15 minutes

A complete Suricata IDS environment from a single playbook,

- Suricata IDS Intrusion Detection System
- Rulesets - configuration files the same across environments
- Cron - jobs for updating rules
- Elasticsearch indexing servers
- Kibana front end

Consistency and tracking, when combined with Git

Audit servers? Run Ansible - anything changed manually? `--check --diff`

Plan-Do-Check-Act process - very ISO 27001 compatible

Templates

```
jdbc \{
  # Postgres jdbc connection string to our database
  jdbc_driver_library => "/usr/share/java/postgresql-jdbc4-9.2.jar"
  jdbc_driver_class => "org.postgresql.Driver"
  jdbc_connection_string => "jdbc:postgresql://{{ private_ip_prefix }}.22.100:5432/Syslog"
```

We can test the SAME CONFIGS in multiple environments

Using variable group vars, host vars, templates

- Site specific data,
- RFC1918 subnets, IPs, port numbers, DNS, NTP
- Domain names, update servers,
- environment: development, staging, production
- Passwords, S3 access keys, administrative users - using Ansible Vault
- Service names: ssh (Debian), sshd (OpenBSD)
- ...

No untested changes brought into production

Operational benefits: Updating daemons and security parameters

Updating a Secure Shell daemon config:

```
- lineinfile:
    dest=/etc/ssh/sshd_config state=present
    regexp='PasswordAuthentication'
    line='PasswordAuthentication no'
  notify: restart sshd
  tags:
    - sshd
```

Combined with:

```
- name: restart sshd
    service: name= service_sshd  state=restarted
```

Never forget to restart a service after changing config

Cluster firewalls always consistent

Updating multiple files on multiple systems:

```
- name: copy PF tables
  template:
    src=.../pf-tables/ item | basename
    dest=/etc/pf/ item | basename
  with_fileglob:
      - .../pf-tables/*.list
  notify:
    - reload pf
```

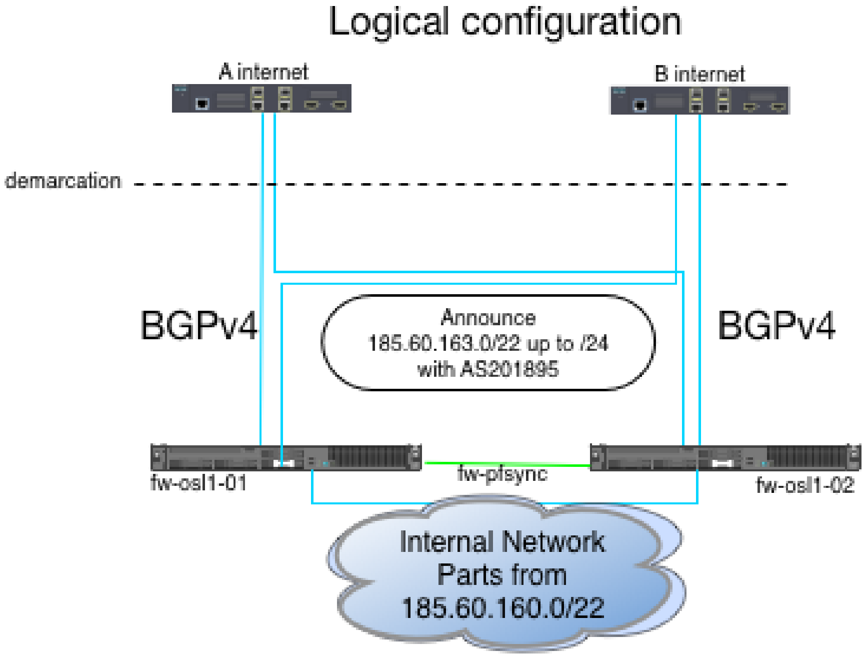
Updating firewall tables and rules

Updating BGP import filters

Roll-out made easy

Less skills required to update

Less manual steps => more reliable



Problems with Ansible / configuration management

Major problem

1) Insert configuration option with playbook
2) Remove that task from playbook
3) Some servers still have the option

Removing stuff from playbooks does not remove it from servers!

End up having tasks that do cleanup

So always test lineinfile - they sometimes surprise

Also when not logged into systems, do you miss problems?

Golden rules

Some things we picked up using Ansible:

- Always use descriptive name: so people know why/what is being done
- Dont use lineinfile, if changing more than a few lines, use a template
- Dont use copy, always use a template (if syntax permits)
- Manual changes should be banned and monitored
- Use tags liberally, tags: pf.conf, only update THIS thing
- Try to gather a project/feature/setup in single playbook
- Example: logging setup with both PostgreSQL and Elasticsearch in same playbook
- Use versioning for your playbooks, we use Git
- Run playbooks often, like maybe daily

And learn your $EDITOR - search/replace in lots of files/dirs ☺

Questions?

BTW we are looking to hire! Sysadmin, Security, Operations

Henrik Lund Kramshøj
hlk@pasientsky.no
hlk@patientsky.com

`https://pasientsky.no/`

You are always welcome to send me questions later via email