Benchmarking Orange Forge with CLIF

Bruno Dillenseger - Christian Bayle
Orange Labs

At the middle of this year, Orange Labs switched its internal software forge (so-called Orange Forge) to a completely new hardware infrastructure and a new version of the Tuleap forge software.

Could this new brand new Orange Forge platform cope with current user workload?
If so, what about users' quality of experience?
This is what we tried to know using OW2's load testing project CLIF.
Feedback says it was a good idea to do so.

Orange Forge

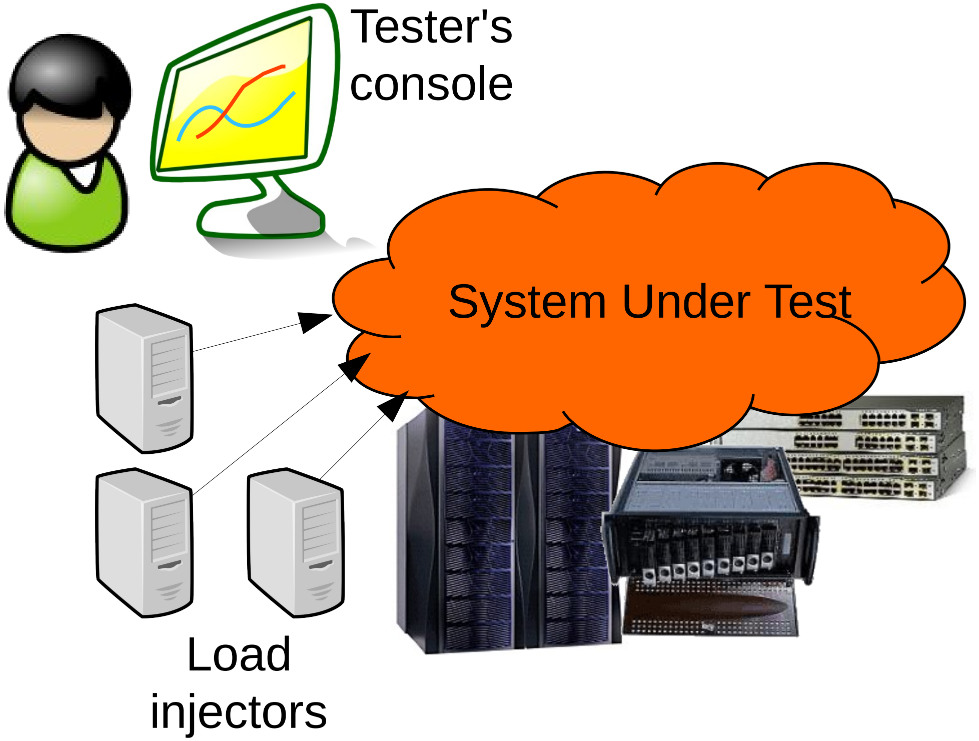
- A software forge for Orange Labs
  - also open to the Orange Group and partners
  - 1000+ connected users daily
  - 16000+ registered users worldwide
  - 5000+ active projects
  - trackers are also widely used by Orange Labs support teams
  - based on Tuleap Application Lifecycle Management open source suite (https://tuleap.net/)
- Integrates a great variety of services:
  - CVS, SVN, GIT, Wikis, trackers, documentation and package management, agile backlog/cardwall/dashboard/roadmap/, mailing lists, continuous integration...

Upgrading Orange Forge

- Towards of completely new platform
  - new servers
  - new network equipment
  - Tuleap and Linux upgrade
- Critical questions:
  - Will the new platform stand the incoming traffic?
  - What about quality of experience?
- A necessity: benchmarking!

What is benchmarking?

- **Performance testing** consists in generating traffic on a system under test and get a performance metric
- A **benchmark** specifies
  - the system under test and its interface (API)
  - the incoming traffic (mix of requests on the API)
  - a metric of interest



Benchmarking enables comparing the performance of technical alternatives

CLIF: OW2's load testing framework

- generic/extensible

  IP, VoIP, database, mobile networks, custom protocols...

- flexible

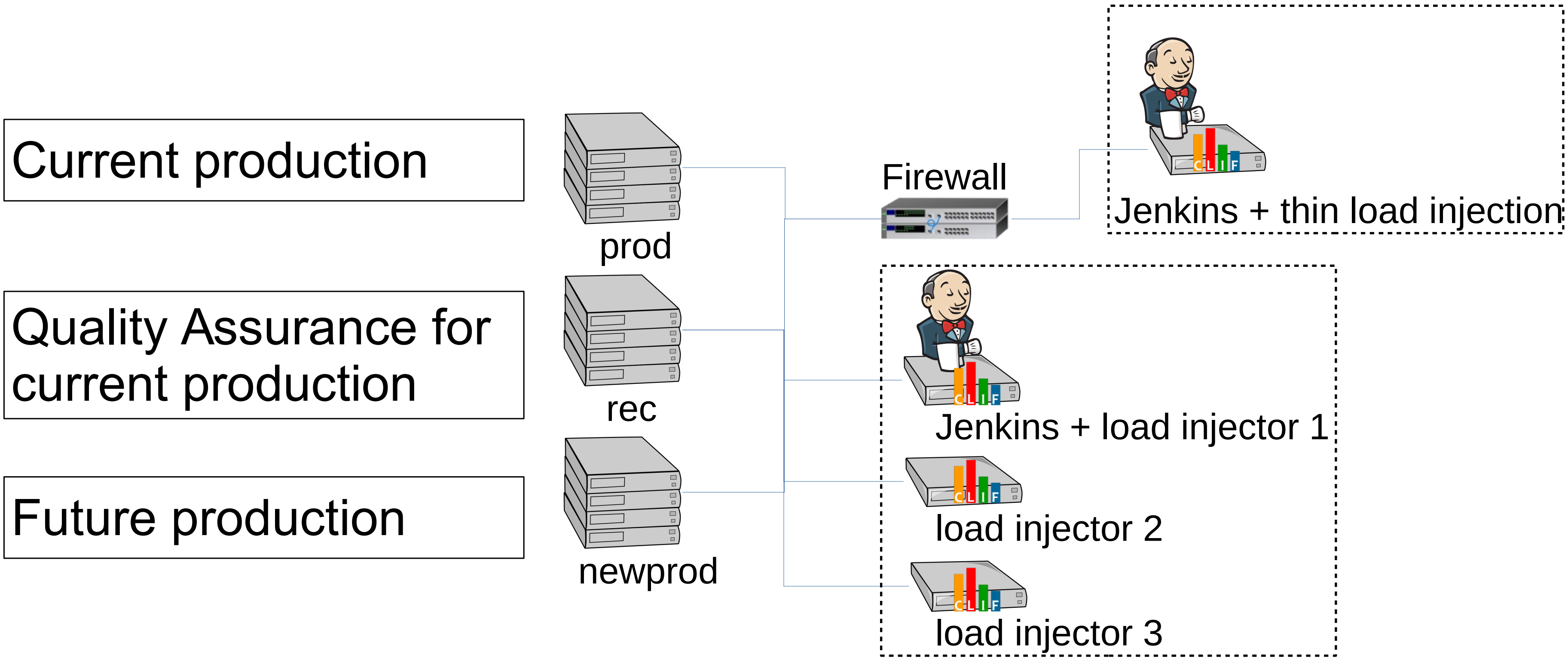
  Eclipse, Java Swing, command line, Maven, Jenkins

- advanced
  - Millions of virtual users
  - 1000+ distributed load injectors
  - miscellaneous facilities
- mature
  - more than 12 years feedback

http://clif.ow2.org

- CLIF plug-in for Jenkins
  - performance testing in continuous integration
  - web interface for running CLIF tests
  - automatic performance reports and performance trends

The Orange Forge platforms



for Quality of Experience monitoring

Current production

Quality Assurance for current production

Future production

prod

rec

newprod

Firewall

Jenkins + thin load injection

Jenkins + load injector 1

load injector 2

load injector 3

for benchmarking campaign

Benchmarking Orange Forge

- **Best practices**
  - analyze logs from the production platform to define a realistic traffic for each forge service
  - run the benchmark on both the new platform and an exact copy of the production platform
  - change only one element at a time to help troubleshooting

- **Reality**
  - protocols are too complex, logs are too low-level, usage is too variable
  - no exact copy of the production platform is available (Q-A "rec" has less resources)
  - all brand new hardware and software (OS and software stack)

A pragmatic approach

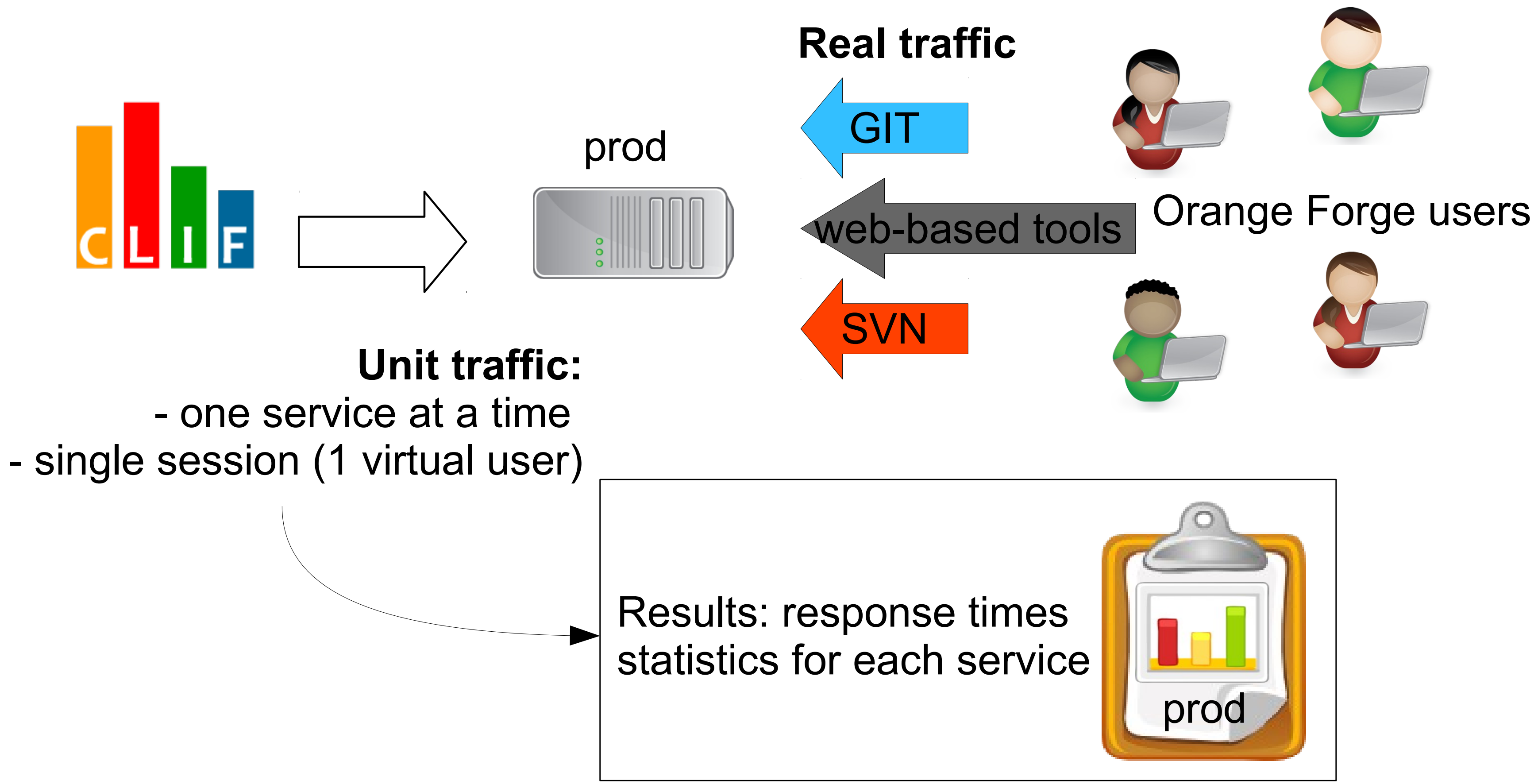
- Choose a number of services and define user-scenarios related to these services
  - SVN checkout/commit, GIT push/fetch, web portal, agile tools, SVN Tree, GIT tree, tracker, forum
  - Luckily, APIs don't change with the new Tuleap version!
- Create datasets
  - projects, user, repositories and database contents
- For each service, compare performance of current production with the new platform
  - taking network throughput as a traffic reference

Taking advantage on CLIF features

- Used provided HTTP injector for all web-based tools
- Developed SVNInjector and GitInjector plug-ins
  - using the CLIF console itself, based on an Eclipse environment including Java SE tools and a dedicated wizard
  - wrapping existing open source Java clients
  - advantage: checked-out/pulled data are discarded, committed data are generated (no need for keeping a sample set of files)
- Used probes for examining resources usage
  - at the server side (web front-end, database) for tuning and measuring network throughput
  - at the load injection side for checking its health and network throughput
- Scenarios are used in different CLIF test plans
  - one scenario alone or combined with the others
  - with different numbers of virtual users and load injectors

Step #1: getting a reference benchmark from current production

Performance testing of the Orange Forge production platform



prod

Real traffic

GIT

web-based tools

SVN

Orange Forge users

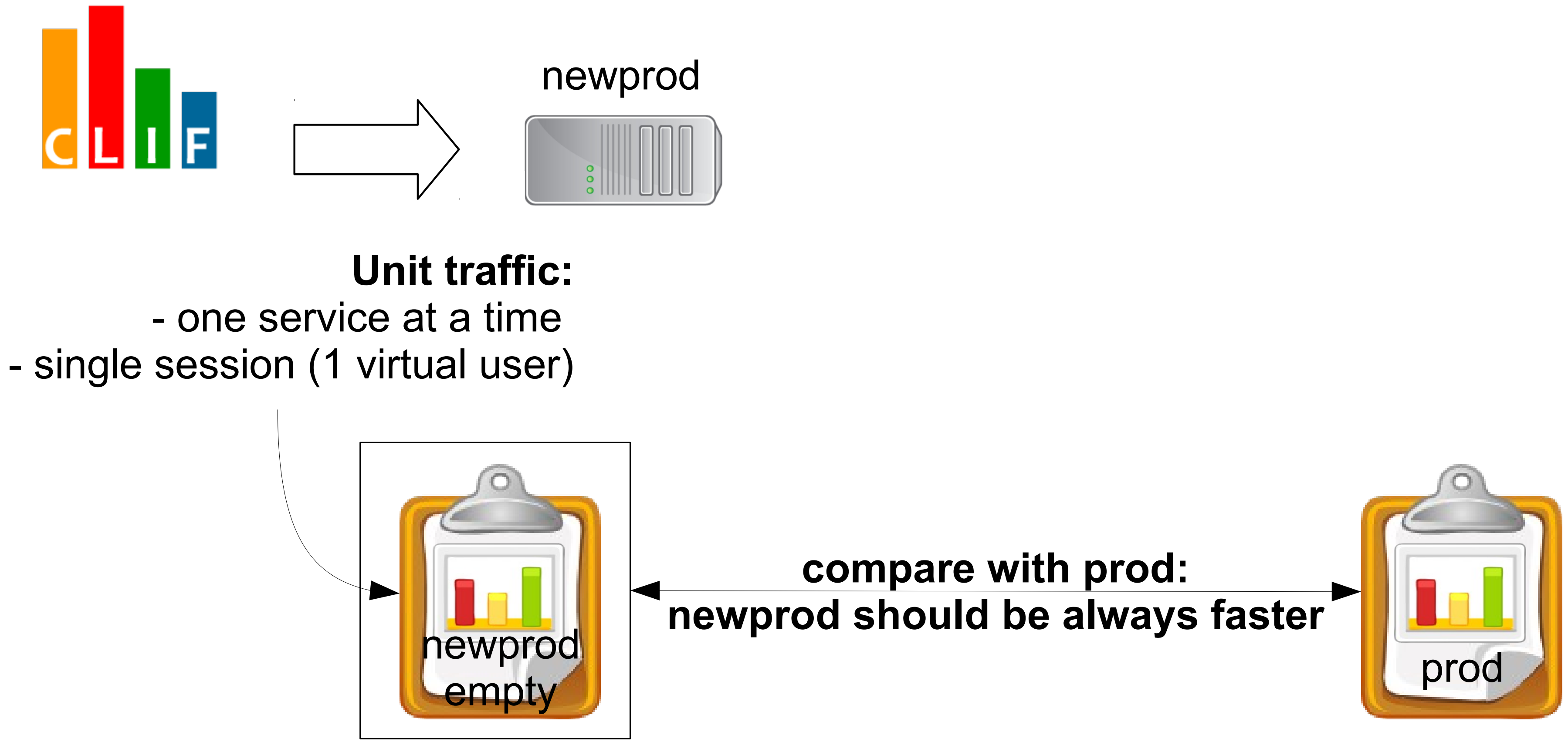
Unit traffic:
- one service at a time
- single session (1 virtual user)

Results: response times statistics for each service

prod

Step #2: comparing prod with newprod (unit traffic only)

Performance testing of Orange Forge new production with no background workload

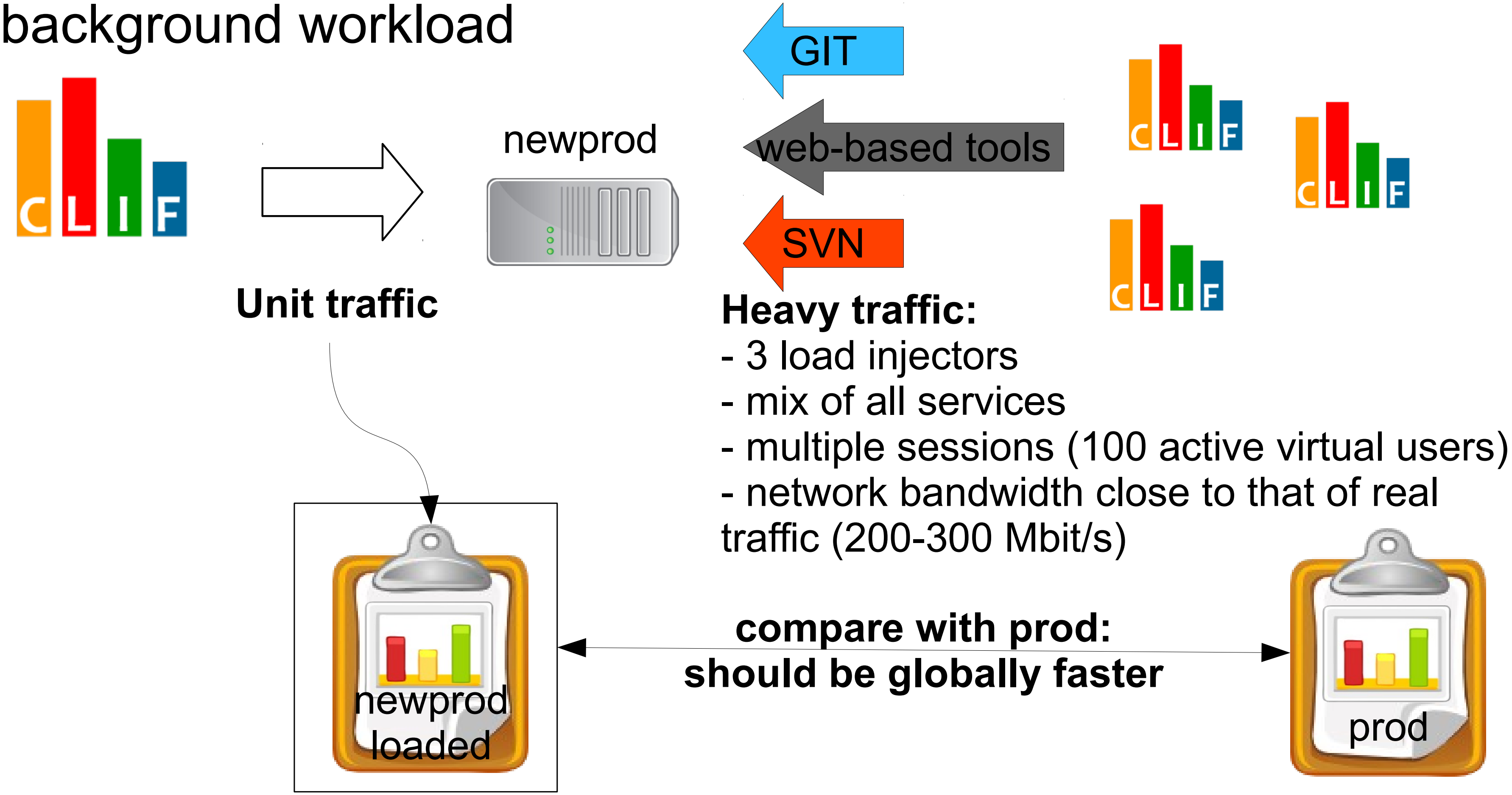


Unit traffic:
- one service at a time
- single session (1 virtual user)

**compare with prod:
newprod should be always faster**

Step #3: compare with newprod under heavy traffic

Performance testing of Orange Forge new production with background workload

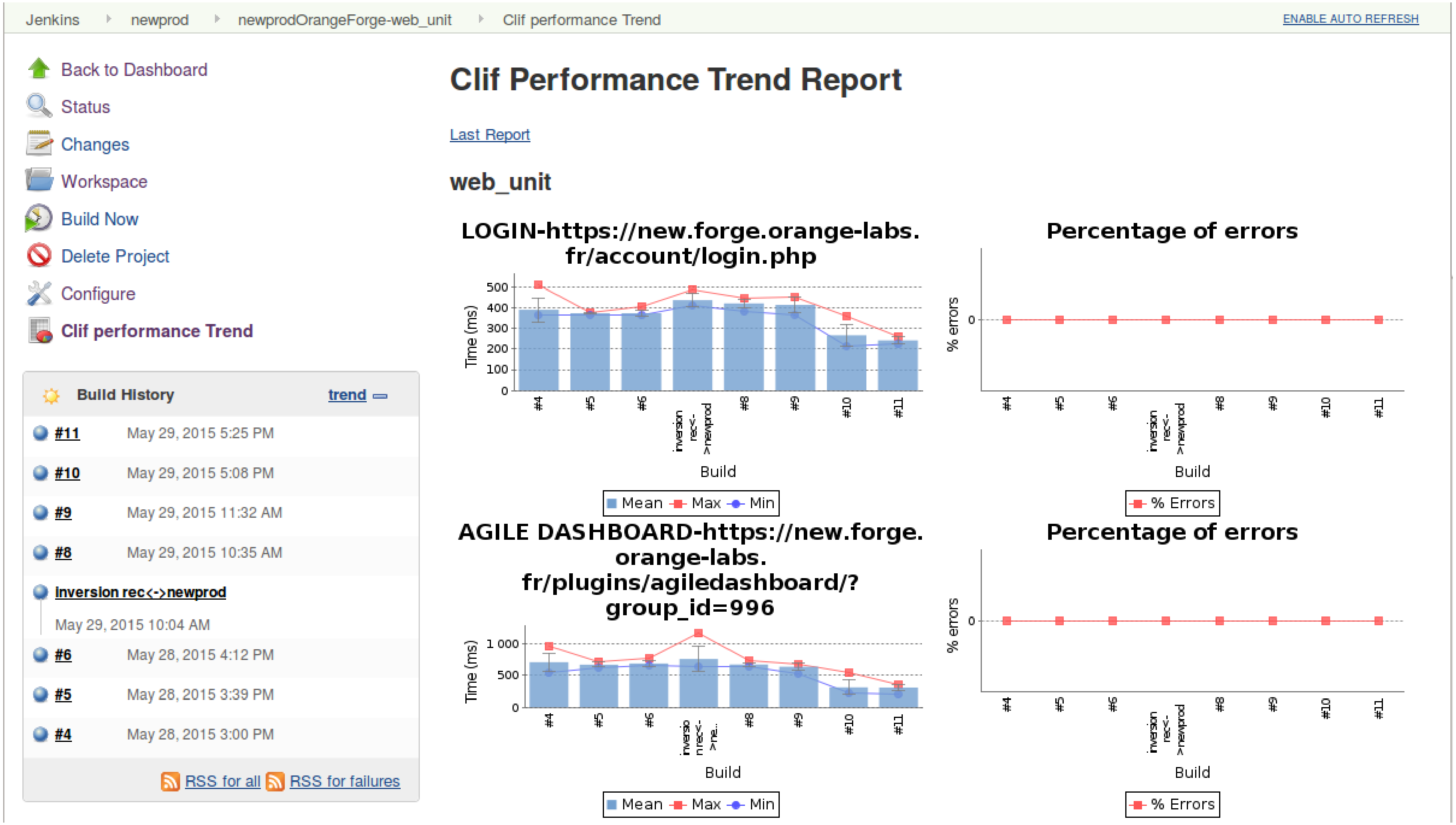


Unit traffic

Heavy traffic:
- 3 load injectors
- mix of all services
- multiple sessions (100 active virtual users)
- network bandwidth close to that of real traffic (200-300 Mbit/s)

**compare with prod:
should be globally faster**

Jenkins+CLIF plug-in: Performance trend report



Jenkins ▸ newprod ▸ newprodOrangeForge-web_unit ▸ Clif performance Trend

ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Delete Project
- Configure
- **Clif performance Trend**

Build History trend

#11	May 29, 2015 5:25 PM
#10	May 29, 2015 5:08 PM
#9	May 29, 2015 11:32 AM
#8	May 29, 2015 10:35 AM
Inversion rec<->newprod	May 29, 2015 10:04 AM
#6	May 28, 2015 4:12 PM
#5	May 28, 2015 3:39 PM
#4	May 28, 2015 3:00 PM

RSS for all RSS for failures

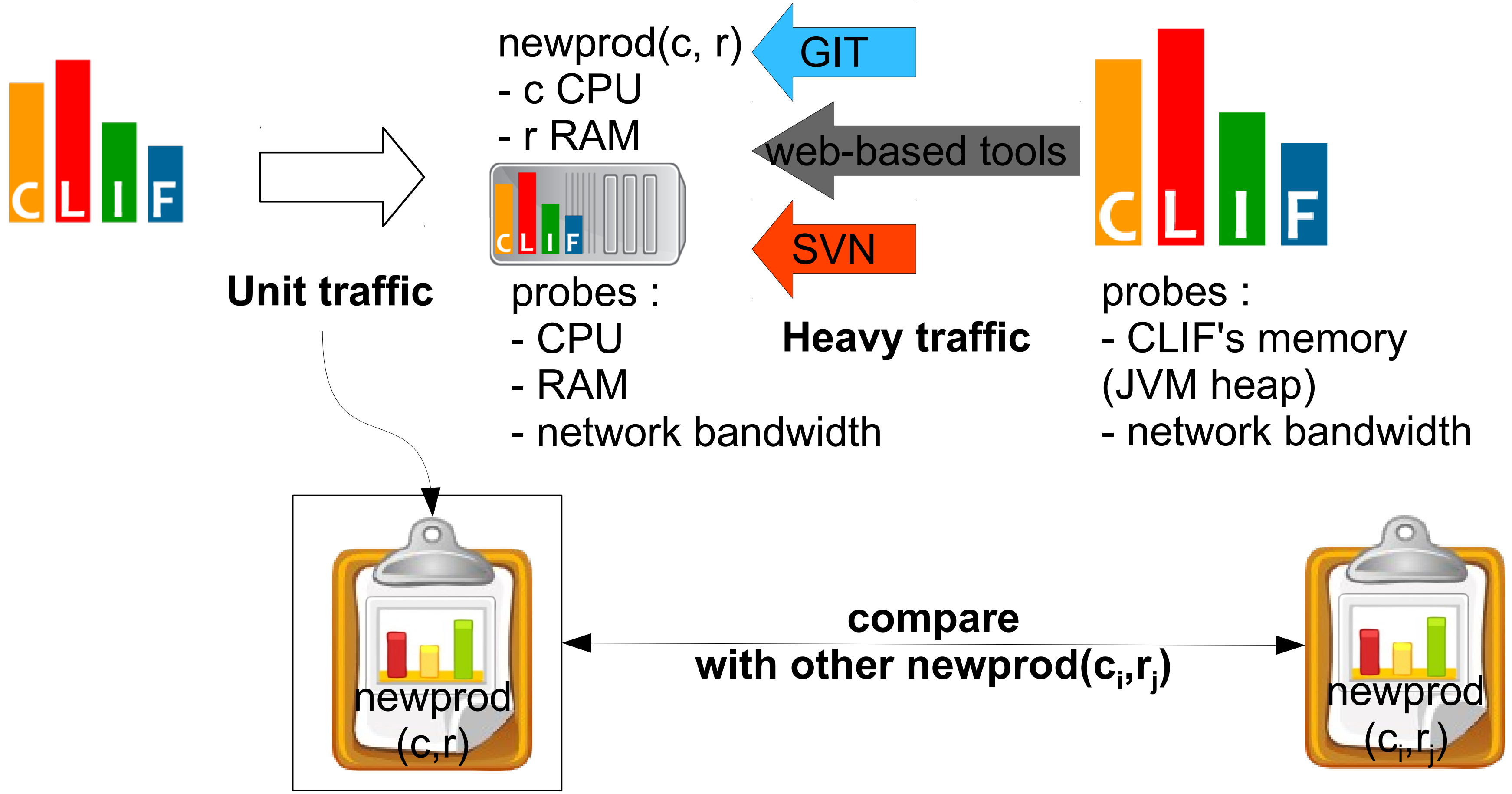
Clif Performance Trend Report

Last Report

web_unit

Step #4: tuning newprod

Load testing of Orange Forge new production with probes



newprod(c, r)
- c CPU
- r RAM

GIT

web-based tools

SVN

Unit traffic

probes :
- CPU
- RAM
- network bandwidth

Heavy traffic

probes :
- CLIF's memory (JVM heap)
- network bandwidth

newprod (c,r)

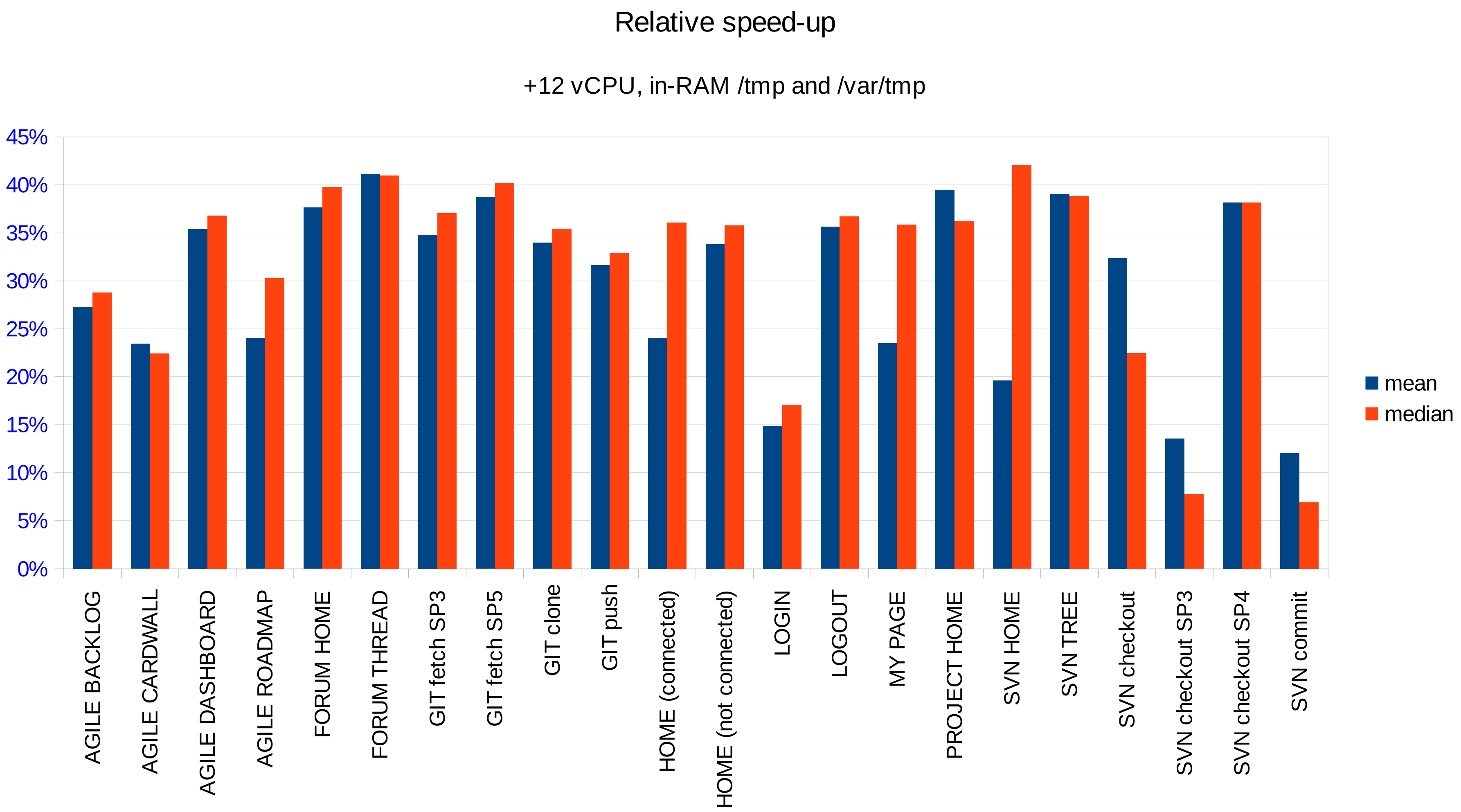
compare with other newprod(c_i,r_j)

newprod (c_i,r_j)

Found and fixed

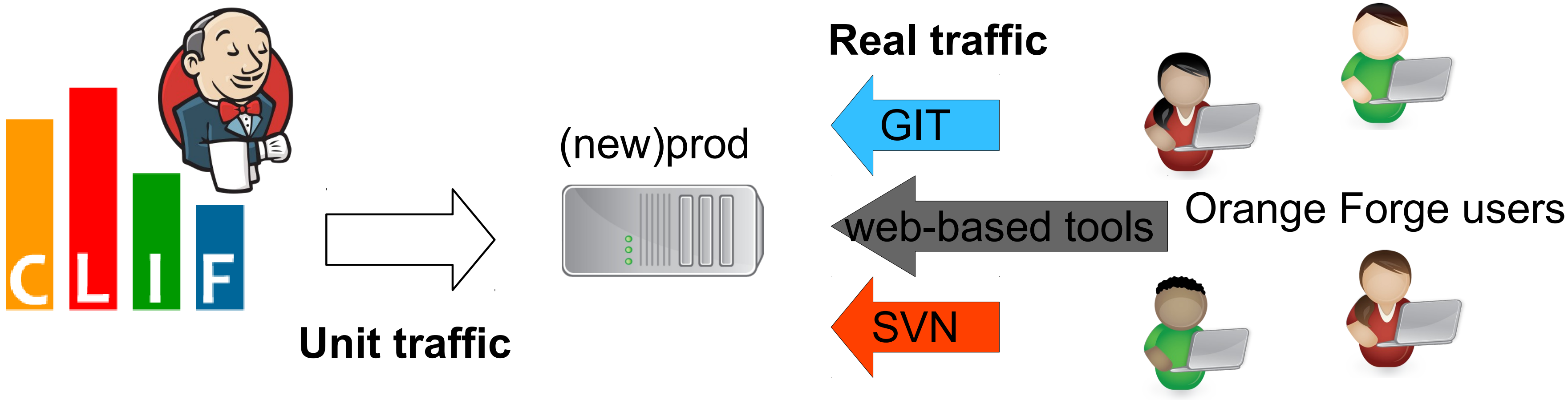
- Steps #2 and #3 revealed lower performance
  - most problems related to caches
    - mainly the PHP cache not working
    - Tuleap's built-in administration tools helped troubleshooting
  - issues with load injection limited to 100 Mbit/s because of one network equipment
  - Linux kernels, NFS vs local disk, in-memory tmp... a lot of "details"!
- Step #4 revealed that it was better to add CPU but memory was oversized (reaffected to tmp)

Tuning: +30% overall speed-up

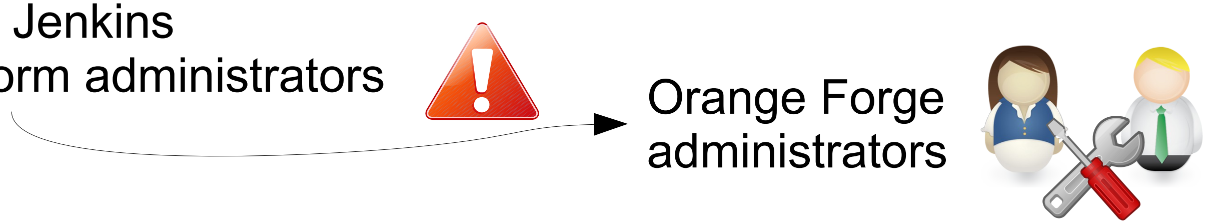


Step #5: monitor Orange Forge quality of Experience

Periodic health check and quality of user experience



Periodic performance test
- scheduled and launched by Jenkins
- detailed reports and performance trends generated by Jenkins
- notification of platform administrators



Conclusion

- The complexity of benchmarking
  - identifying and defining "realistic" scenarios and datasets
  - beware of load injection defaults
  - use complementary monitoring tools to troubleshoot
- The high efficiency of CLIF+Jenkins
  - CLIF's versatility (multi-protocols, scenarios used in different test plans, distributed load injection and monitoring)
  - click-to-test with automatic reporting allow for testing lots of configurations within short cycles
- It was definitely a good idea to benchmark...
  - successful switch to the new platform 15th June
    - we avoided big troubles!
  - scenarios are now used for monitoring QoE