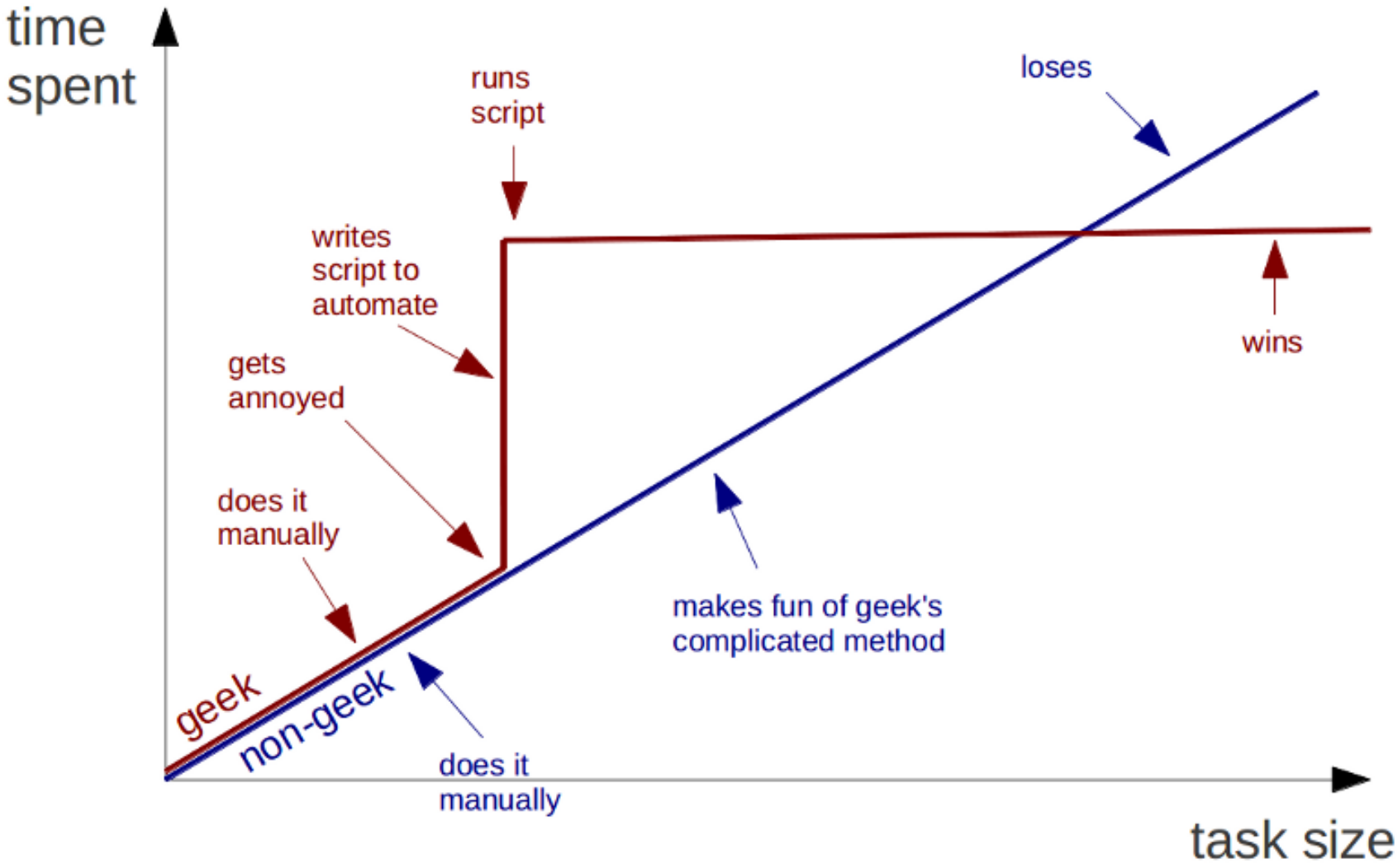
Network Automation Do's & Don'ts

Job Snijders
NTT Communications
job@ntt.net

NANOG66, San Diego

Geeks and repetitive tasks



Source: https://plus.google.com/+BrunoOliveira/posts/MGxauXypb1Y

Research Sample Size

I've asked the brightest minds in the industry about network automation:

- John Heasley
- Troy Boudreau
- Andree Toonk

But also myself and Jared Mauch ;-)

DO #1

Follow UNIX principles (whenever possible).

Small programs that do one thing well and can be part of a pipeline. So don’t hack rancid to do tricky email shit, do that as a post processing step.

DO #2

No shortcuts – check for all failure possibilities and only allow precise coherence to expected results, else recover and exit.

Example: “This code will work as long as nobody manually touches the config on the device”

Example: “The RIPE IRR database will always produce correct results formatted as valid RPSL”

DO #3

Don’t buy crappy equipment without demoing and testing it. You are not saving money or time. Test SNMP, CLI, NETCONF.

DO #4

Don’t be liberal in user inputs; check all user input rigidly. Not only syntactic, but siphon the data through **business rules** as well!

Bad example in middleware or IOS:

- User inputting a router2router linknet and mistakenly configuring an IPv6 /30
- 2000::/6 in DFZ (should’ve been a /64)
- 2000::/12 in DFZ (should’ve been a /126)

DO #5

Less opportunity for users/managers to customize is a good thing.

Through network automation you can exercise a degree of control to enforce healthy network design.

DO NOT #1

Don’t let users use your tools without training.

“a user ran a script used to reboot the entire network when they had no clue what they were doing.”

DON’T #2

Don’t allow other groups to take control of any part of your domain (ie: devices, DNS, IRR, etc) unless they are under you. You will regret it eventually.

DON’T #3

Not every new tech trend is useful. Consider changes thoroughly.

Example: docker on routers

DON'T #4

Load your production database in a test environment without locking down the test environment.

"Where a test box doing tests sent email to users about change BGP ACL changes"

DON’T #5

Don’t agree to change management. Managers are rarely engineers and should not be making technical decisions. (nor should sales)

Change Advisory Boards will be the first to go against the wall when the revolution comes.

DO #6

Make automation a top priority, change your way of thinking, everything should be automated, especially greenfield deployments.

Example: "Automation Friday" as expression of institutional buy-in

DO #7

Done is better than perfect.

Think in terms of minimal viable product (MVP). Most of us are not hardcore programmers.

We learn while doing, so start delivering small deliverables. Learn and build on that.

Don’t refactor when adding 1 new feature.

DO #8

“Decrufting” - Deleting unused code saves you money!

Example: by reducing the amount of peer-group templates, the test surface is shrunken considerably because there are less permutations to cycle through.

DO #9

Don’t be too religious regarding what language to use. What ever does the job, and works for your team.

This is counter-intuitive because arguing about what is the best language is awesome.

DO #10

Steal ideas!

Net-eng teams are typically pretty new to this automation stuff. Other teams have done this for a while and learned lessons along the way. Go talk to your sysadmin/SRE/devops team. It's likely they solved similar problems in the past and learned some lessons along the way. Borrow their expertise for a day to get you started.

DO #11

Have tests, simple ones are fine!

Pre deployment: Are the ACLs of non-zero length?

Post deployment: Simple validation tests that greps in the RANCID repository for stuff, like making sure the DENY-ALL rule is there.

Future: This can evolve into a database consistency check script that validates stuff like interface descriptions.

DO #12

Never delete logfiles.

When you have software making autonomous changes to the network, have logging and keep track of why and what it's changing. It's likely at some point it will do something you didn't expect or understand.

DO #14

Version control systems everywhere!

Everybody has RANCID – which is **after the fact**.

Consider putting all config snippets you plan to push to routers in a VCS too. To cover "**planned to push**".

DO #13

Throttle!

If you have software that makes changes to the network, like, optimizing latency by injecting BGP announcements or shutting down sessions, consider a max number of operations allowed per hour. Use this as a safe guard to prevent unexpected massive changes or runaway code

DON’T #6

Don’t buy IOS-style shit!

Get router equipment that supports loading and checking configurations prior to committing them.

DON’T #7

Don’t allow your staff to do “conf t” – bypassing the automation is like joining the antichrist. Punish people who disable the automation.

DO #15

TEST TEST TEST TEST TEST TEST

- Invest in a luxury lab (with at least all your currently deployed hardware)
- Get actual test equipment/software

DON’T #8

SMOKER’S COUGH

- JAGER
- WARM MAYO

DON’T #8

Don’t mix **data and logic** too much!

Avoid conditionals based on hardcoded values in your templates.

If you hardcode a hostname, a migration will bite you. If you hardcode a peer’s ASN, a change in the relationship will cause issues.

DONE