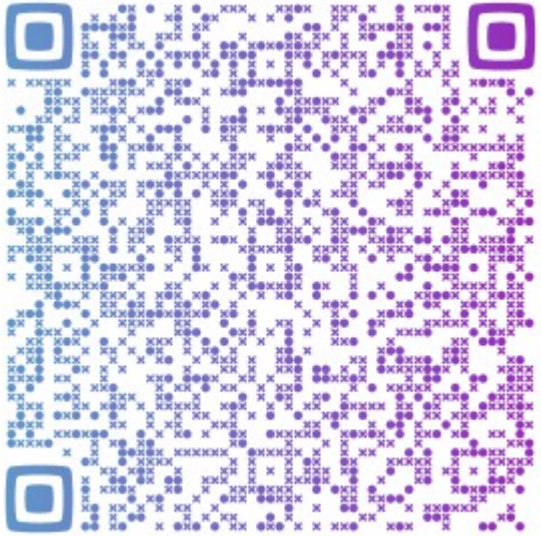


Practical Limitations of the Formal Systems

Kamil J. Dudek

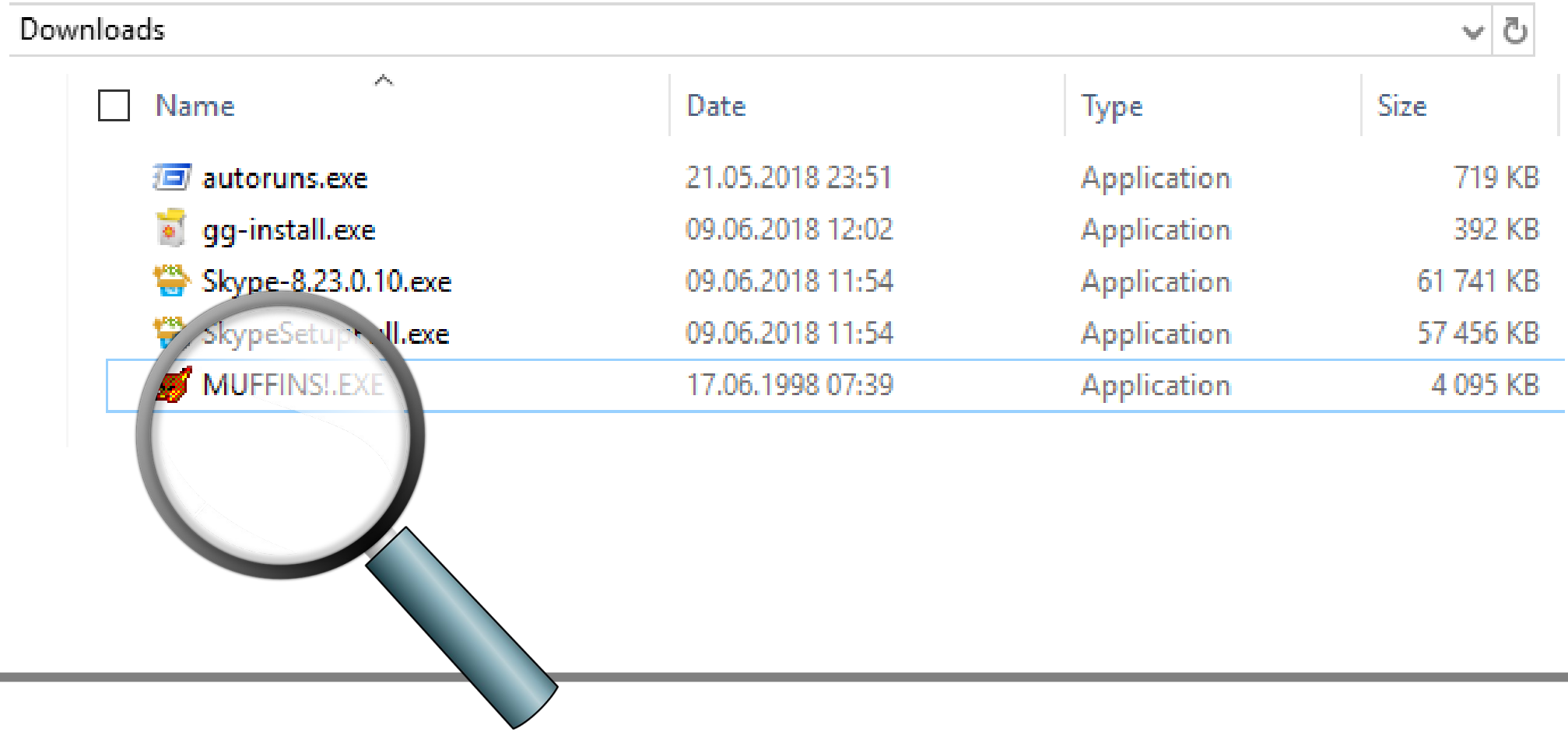
a2FtaWxqZHVkZWsk@ノoヮoノ.みんな
kamiljdudek@gmail.com
@kamiljdudek

When is it okay not to know something – few words about testing and security

How do we know what a program does?

(but isn't that obvious.?)

How do we know what a program does?



How do we know what a program does?

- Run it and see what happens (because $why\ \ not\ !$)
- If we have the source code, we can look there
- Connect the linker or a code analysis tool

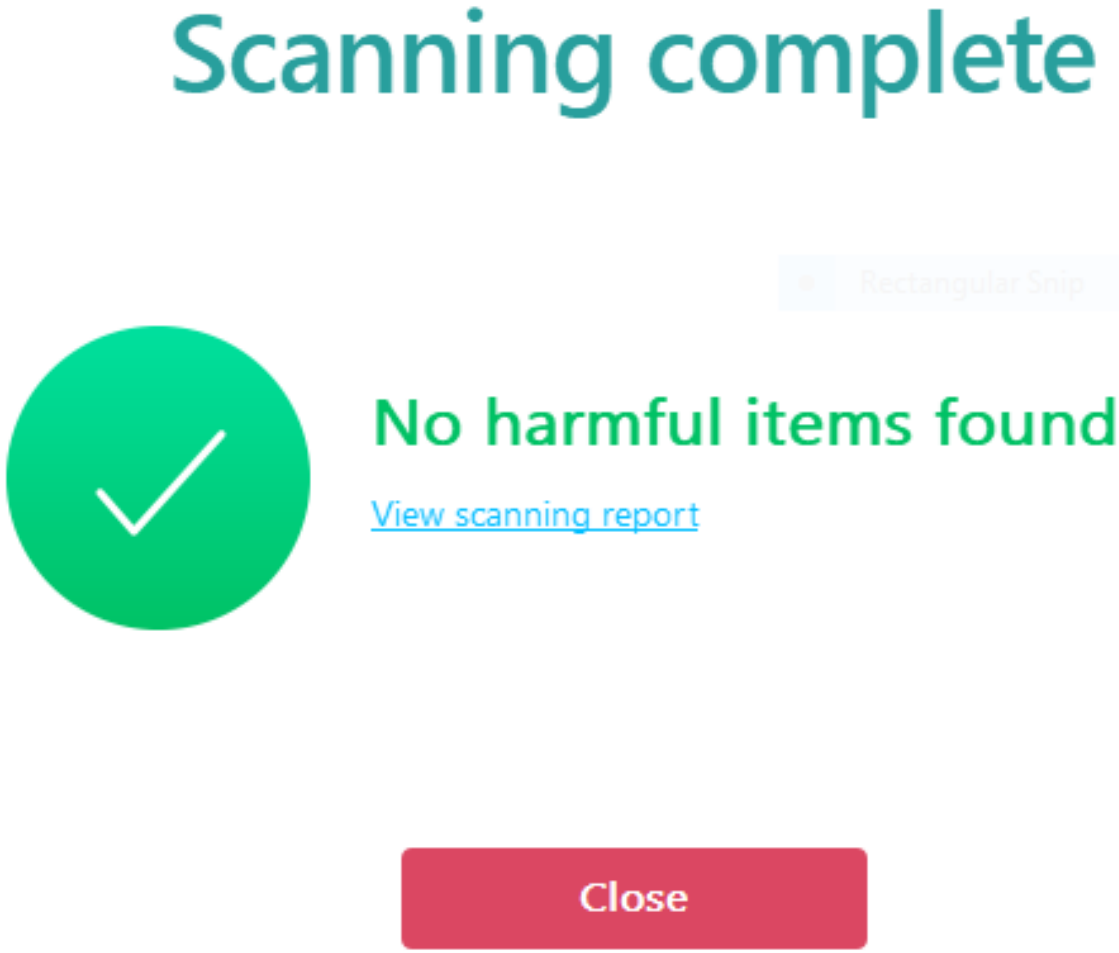
How do we know what a program does?

- Run it and see what happens (because *why not!*)
  - *that doesn't sound secure, does it?*
  - *also, it does not provide a complete answer*
- If we have the source code, we can look there
  - *we can only assume that we understand it*
  - *impossible to run all the code paths entirely in the head*
- Connect the linker or a code analysis tool
  - *if it walks like an executable linked to the crypto API library…*
  - *but this is only a meta-analysis, gives hints but not answers*
  - *Maybe we should try and pick some more advanced tools, like…*

Heuristics, right? That should work

- Anti-virus software has some advanced mechanisms to detect the class of threats known as “potentially harmful”.
- It uses the signature-backed, heuristic, sandboxed and behavioral set of assessment tools
- *But what if...*

What if:


Scanning complete
No harmful items found
View scanning report
Close

What if:

virustotal

SHA256:	b1a810f94a02c88536ce6d9b88f765b8cea62d2e6ac3d3c08ada19c599060de8
File name:	MUFFINS!.EXE
Detection ratio:	0 / 67
Analysis date:	2018-06-27 18:47:54 UTC (0 minutes ago)

Are we back at the square one?

“run it and see what happens”?

Running programs considered harmful

- The controlled demolition scenario: run the code in a dedicated VM that simulates the ordinary work environment
  - *shhhh! Hide that fact from the program!*
- We can cause, detect and observe the harmful behavior that way
- However...

So it’s impossible to know. Ever.

how does the world still exist, then?

Fundamental limits of knowledge and research

"Creating an infallible anti-virus is impossible!"

-- frustrated sysadmins everywhere

"No algorithm exists that always correctly decides whether, for a given arbitrary program and input, the program halts when run with that input"

-- paraphrased, Alan Turing, 1936 [1][2]

[1] https://en.wikipedia.org/wiki/Halting_problem#Sketch_of_proof
[2] Alan Turing, *On computable numbers, with an application to the Entscheidungsproblem*, Proceedings of the London Mathematical Society, Series 2, Volume 42 (1937)

Fundamental limits of knowledge and research

"There are statements of the language of F which can neither be proved nor disproved in F.

-- Kurt Gödel, Incompleteness Theorem

"Arithmetical truth cannot be defined in arithmetic."

-- paraphrased, Alfred Tarski, 1936 [1][2]

[1] https://en.wikipedia.org/wiki/Tarski%27s_undefinability_theorem#General_form_of_the_theorem
[2] A. Tarski (1936). „*Der Wahrheitsbegriff in den formalisierten Sprachen*"

How to deal with not knowing?

This is not poetry, this is engineering!

How to deal with not knowing

Truism (?):

It's normal and painfully common ***not to know*** *an essential piece of information (which is by design)*

- This is not a contest of who knows more (deal with it)

- „Strong assumption" is not knowledge and is never „good enough" to be called knowledge

- Cheating yourself into believing that you know something is more dangerous than a glaring, but honest omission

Examples!

Temet Nosce

In search of knowledge – Group Policy

- I was hired as an administrator of the Active Directory
- There was a fleet of 200 computers manually recovered using an old version of Norton Ghost
- After establishing a domain, the unattended network installations using WDS and PXE had to be implemented
- When it was finally done, I decided to make it **indestructible** and heavily secure it using the Group Policy lockdown

Outsmarting the evil – Group Policy

- Locking down a **developer machine** is futile by definition
- So I decided to „get creative”
- I hunted down a perfect set of GPOs
- Not only did I remove all the unnecessary stuff, but also quite a numer of clever, rare and bizarre things, like...

Lockdown Showdown – Group Policy

- HTA application framework
- JS and VBS file associations
- Deprecated OLE and DDE extensions
- Entire PowerShell
- Administrative shares and SMB1
- Macros in Excel (LOL)
- SCT caching
- **Windows Calculator!**

- I removed a bunch of possibilities of lateral movement
- *I felt pretty good about myself…*

The peace through Group Policy

- Anti-virus dashboard showed no detections in the network
- Net traffic was acceptably low, nothing extraordinary
- Users encountered no issues, no reports were coming
- However...

- All the computers (almost) were mining cryptocurrencies
- CPUs were thrashing, but the high loads were ignored because it was an HPC lab...

The pride cometh before a fall

I was truly devastated.

In time, you will know what it's like to lose. To feel so desperately that you're right, yet to fail nonetheless. Dread it. Run from it. Destiny still comes.

What happened?

- Many students tried to pass the course using a ready-made code
- The code was uploaded to some shady website
- It tricked the users to install a malicious Chrome Extension
- At the same time, it used the JavaScript code to mine Monero
- Chrome worked in the background, even after closing

What happened?

- I was technically right
- I prevented much more things than the default configuration and much more things than *that other guy out there*

- **I won the contest of who knows the most**

- Nobody paid me for overtime, but that time for once, they shouldn't have anyway

Testing!

Do not test the entire world.

Don’t test everything!

- There was a need to test if the app can add a record to the database
- The database was provided by another team an integrated on our CI

- It was present in our environment and was being installed by us, we just used one table fromit

- A new hire was put in charge of writing a test for it, as an entry-level task

Mic check – one two.

- The work started with a simple test: assert against the event of adding a row in the specific table

```
def add_single_sql_query_app3( self, targetenv ):
```

- It failed if anything went wrong. Cool.
- But what if the connectivity was broken because the service was down?
- Okay, we're installing *their* config package, but the OS is *ours*, right? Let's check the service

Test all the things!

- Check the service first, *then* run the query:

```
def sqld_systemd_unit_sane( self, targetenv ):
def add_single_sql_query_app3( self, targetenv ):
```

- This way, we'd know if the service management interface has changed or some new unit garbled the dependencies!

- But what if we can't connect even though the service is running?
- Other tests cover the firewall and port assignement

Test until your feet bleed and the keep testing

- There are other things that might prevent it! Therefore:

```
def avc_denials( self, svcname, targetenv ):
def sqld_systemd_unit_sane( self, targetenv ):
def add_single_sql_query_app3( self, targetenv ):
```

- The DB guys have the exact same environment, but maybe they...didn't install the newest patches! Now that makes sense, right?
- Or maybe they softened the config too much for CI?

Test until your feet bleed and the keep testing

- Yeah, let's check that as well!

```
def poll_auditd_halts( self, svcname, targetenv ):
def avc_denials( self, svcname, targetenv ):
def sqld_systemd_unit_sane( self, targetenv ):
def add_single_sql_query_app3( self, targetenv ):
```

- They'll thank us someday, when it finally fails!
- Also, while we're at it, maybe check if the test is run in the proper VLAN, so the names are correct?

Indiana Jones and the Temple of Tests

- Sure, people tend to run tests as a hobby, so better be sure:

```
def supported_vlan( self, ip_plan, svcnm, tenv, tm ):
def poll_auditd_halts( self, svcname, targetenv ):
def avc_denials( self, svcname, targetenv ):
def sqld_systemd_unit_sane( self, targetenv ):
def add_single_sql_query_app3( self, targetenv ):
```

- Better safe than sorry.

Finally, it failed!

How to be glad about other people's mistakes.

Underwater chainsaw backgammon

- New enforcing rules cut the access to the DB, located at the custom path:

```
def supported_vlan( self, ip_plan, svcnm, tenv, tm ):
def poll_auditd_halts( self, svcname, targetenv ):
def avc_denials( self, svcname, targetenv ):
def sqld_systemd_unit_sane( self, targetenv ):
def add_single_sql_query_app3( self, targetenv ):
```

- .and those horribly over- engineered tests have detected it!

Sharing is caring

- Since the tests detected the database failure, the database team guys were called about that so finally the tester had a chance to make themself useful.
- Described the problem with needlessly extreme detail, without being asked asked to.
- After being done, eagerly awaited the appraisal, even some kind of a joke- like one.

- It didn’t come.

Sharing is caring

- They knew the issue and were working on it.
- They knew, because it was *their job* to adapt their piece of software.
- He could not sensibly explain why he was actually telling them all that and what I expected.

- The tests took three times as much to write
- ..and they worked three times slower than anticipated.
- Hundreds of lines of the new code brought nothing meaningful to the table, but at least it could tell *exactly* what was wrong.

But isn't that..obvious?

Why are we discussing a poorly done job?

Because this is about psychology

- The notion of 'unknown' is unavoidable, natural and infused into the very core of defining any system
- It comes natural when discussing things like a $void()$ subroutine or passing a $NULL$ type parameter

- It somehow becomes much $less$ obvious when confronted with tasks that simulate a closed-ended issue but aren't one
- It gets even worse if ambition gets to play some role

How is this a problem again?

- Let's wrap it up a bit:
  - Cheating yourself into believing that all the loose ends are tied is dangerous and poses a security threat
  - Deciding not to chase all the clues is a bold decision and should be made with reason
  - It's not about not being *skilled enough* to do something
  - The point is to realize the extent at which some tasks are downright *impossible* to achieve
  - A **healthy balance here** is the ability to assess the following: *either I don't understand how that works or I am attempting to do the impossible*
  - Overdoing any of those opposites is a problem of ambition
- We must not fail at educating our younger peers

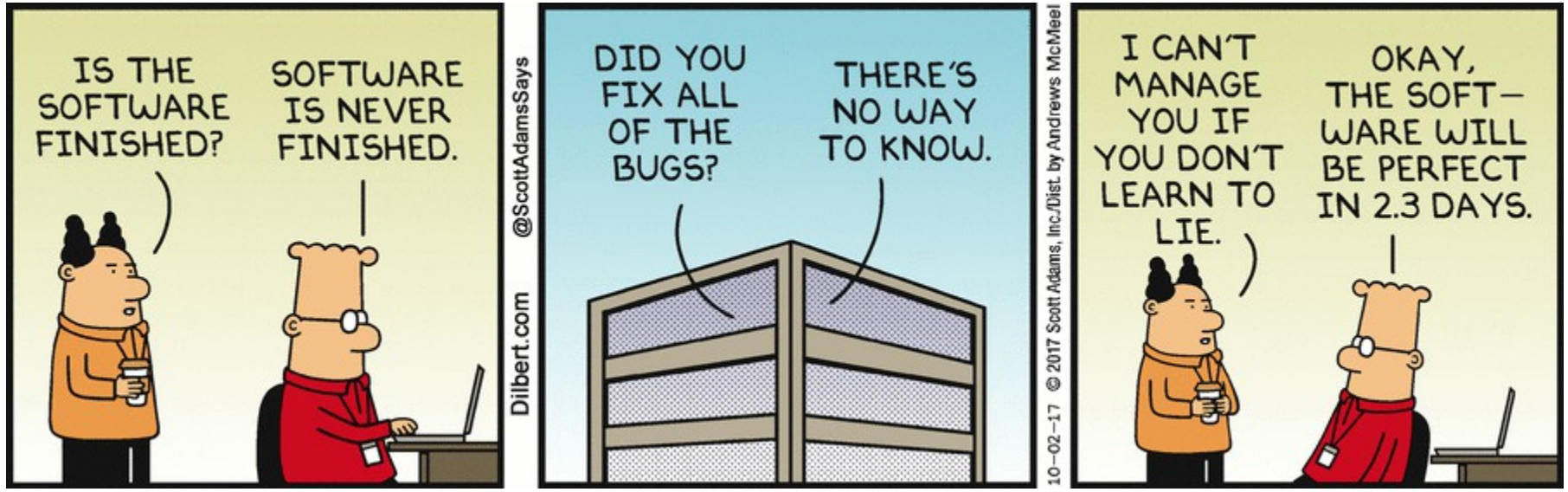
How is this a problem again?

- It creates work to leads to nowhere
  - Engineering teaches to avoid the indefinite states, but those can be easily mistaken with „safely unknown"
  - Trying to define all forms of a problem can drive us $away$ from properly defining a $class$ of problem we're dealing with.

- We must not fail at educating our younger peers
  - Corporate „rules" of work division and all the „obvious" mechanisms that create the communication workflow are not science, nor skills. They're $culture.$

What to do about it?

- Remember about the limitations of the formal systems!



- The software is never $actually$ done!

What to do about it?

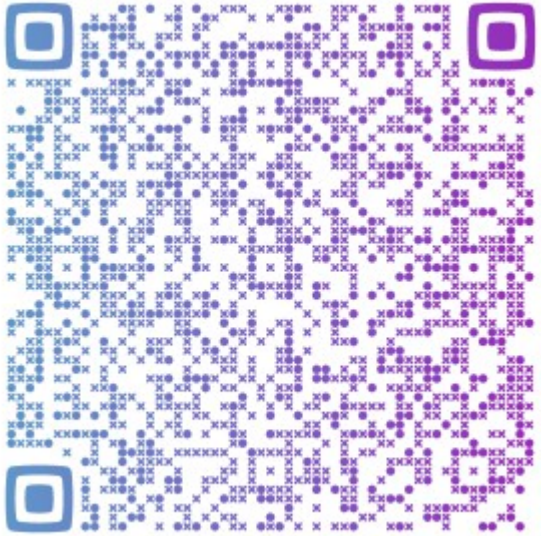
- Try to answer the questions:
  - Can I ever be really done?
    - Is it feasible by definition?
    - Can I even tell?
  - Do I need to educate myself more or am I trying to work myself to death?
  - Why am I reaching out of scope?
    - Is it helping me in any way?
    - Or maybe I didn't even notice?

This is still obvious!

„Listen, I've been working here for years, okay?"

Maybe. But consider this:

- Talents, skills, practice and culture are different things: do not mistake your routine with proper training.
- Failing to educate new team members about the risks of false knowledge can be just as harmful as leaving them untrained.
- Soft skills are not a meme. They're knowledge.



..and that's my time
Thank you!

a2FtaWxqZHVkZWsk@ノoヮoノ.みんな
kamiljdudek@gmail.com
@kamiljdudek

Let's talk about it.
I encourage to share the experience now.
What were your issues with unreachable knowledge?
Questions are welcome as well!