



**FIRMAMENTUM**  
CONSULTING

**DevSecOps Minefield  
for High-Growth Teams**

*Pathfinding your way to effective*

***DevSecOps adoption,***

*including AI's impact, compliance, and silent killers.*

# Why DevSecOps & Shift Left?

DevSecOps is a solution to help:

- Unreliable software releases.
- Unstable infrastructure woes.
- Nightmares about cybersecurity.

The basic concept is shifting left both operational and security concerns as early as possible in the software production pipeline.

This is very simple in theory.

It is extremely hard to do well.

There are pitfalls and silent killers that, despite your best wishes, might sabotage your journey.



# Too Many Tools Spoil the Software

DevSecOps is a paradigm, not a set of tools.



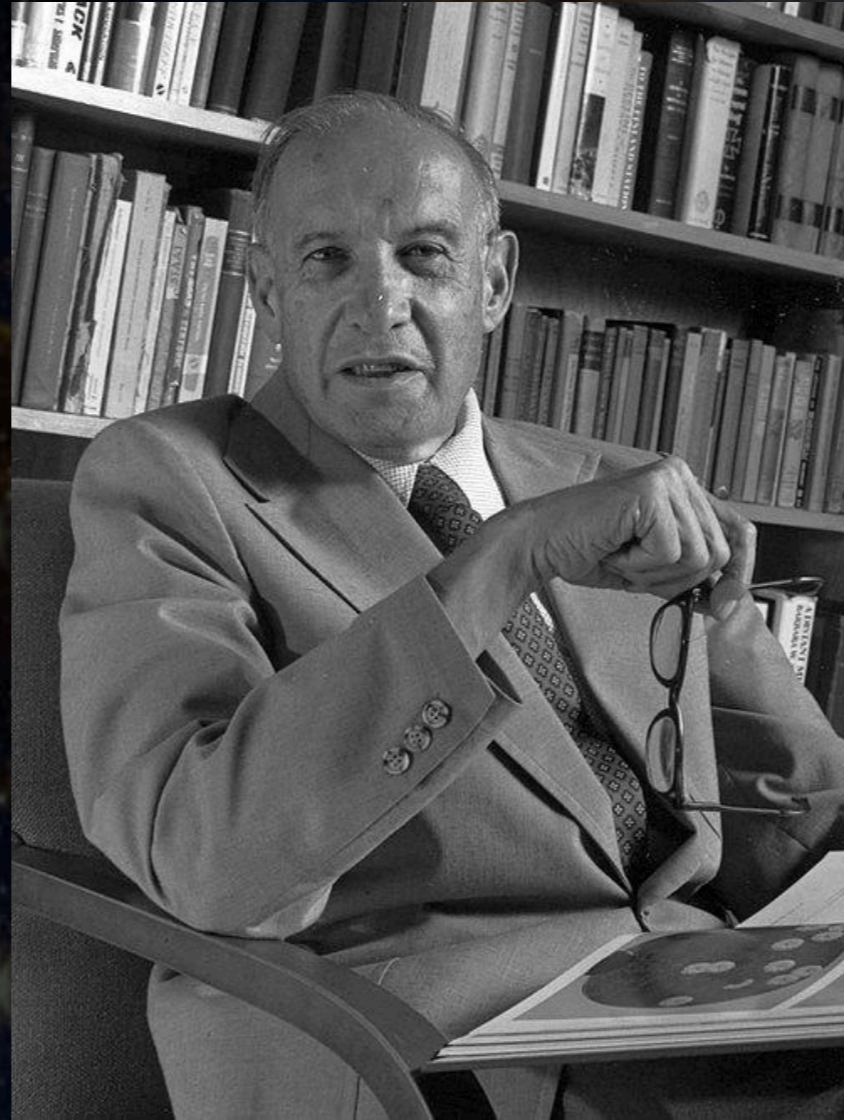
All too common traps are:

- Using multiple tools that conflict, duplicate effort, or miss gaps between them.
- Detection galore, but no resources to act on the findings
- Deploying many AIs without risk framework or impact analysis.
- CI/CD filled with processes that block and impede.
- AI produces more code, is this really an asset?

Tools do help provided they enhance and amplify.

# Culture Eats Strategy for Breakfast

Often erroneously attributed to Peter Drucker, true nonetheless...



Leadership must lead the way:

- Not buying into the mindset change to shift left leads to doing the same thing as before.
- Incentives misaligned: rewarding speed over security
- Adding work load to operations and security, yet they still have to do everything they did before.
- Blame culture ensure no one either risks anything or admits needing help.
- Assuming AI will help in all places/cases.

Leadership support and understanding is vital.

# Supply Chain

Your foundations need to be strong.

Your software is built on many third party modules, libraries, and dependences. Each one impacts your security.

How do you track them?

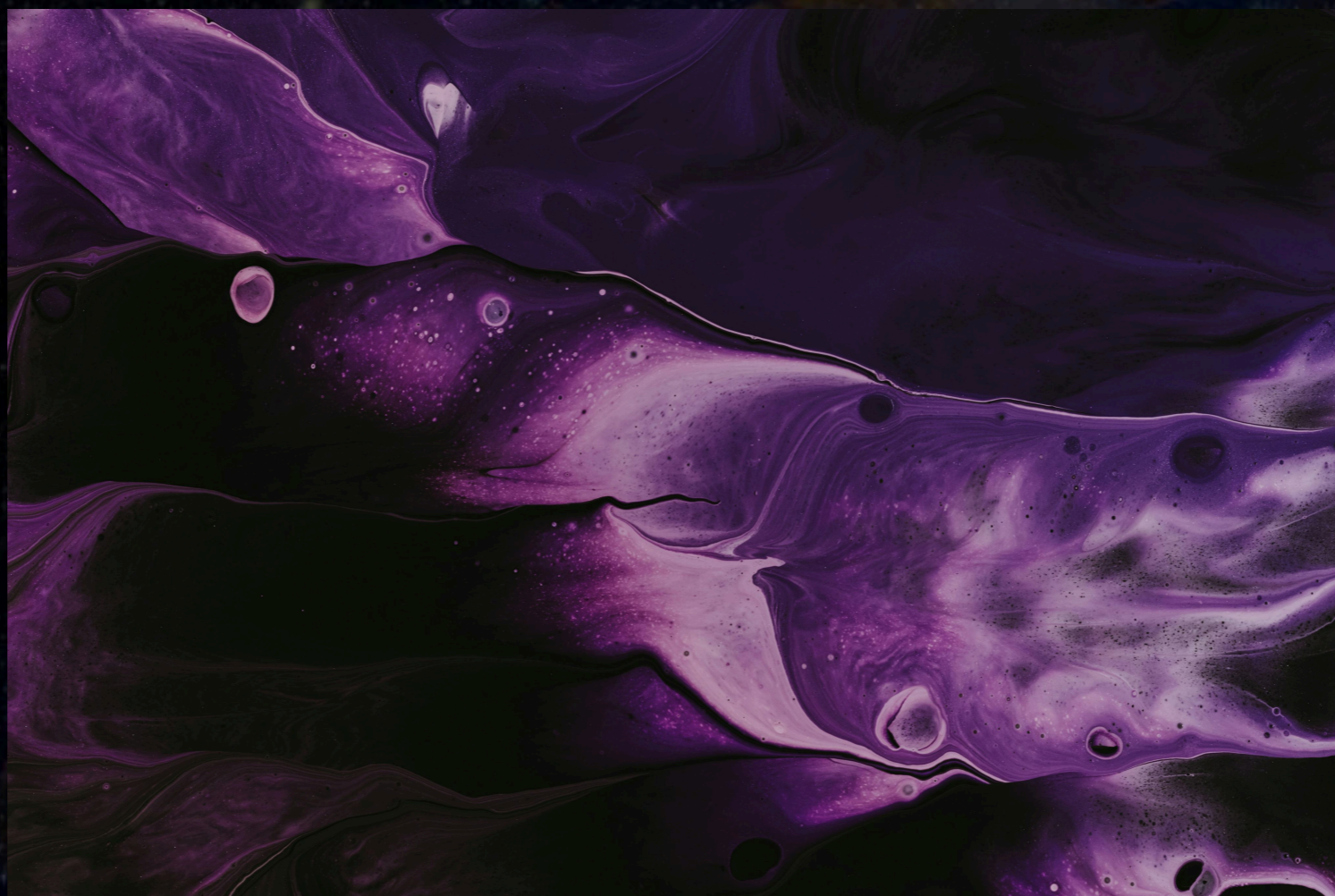
- No SBOM, no clear supplies, varied versions.
- No patching cycle.
- AI let loose to download whatever dependencies it fancies.
- System hardening and task separation.

Threat actors now target dependencies to steal data, no matter how small you are.

```
-----
markdown v3.10.2
markupsafe v3.0.3
mergedeep v1.3.4
mkdocs-get-deps v0.2.2
mergedeep v1.3.4
platformdirs v4.9.6
pyyaml v6.0.3
packaging v26.2
pathspec v1.1.1
pyyaml v6.0.3
pyyaml-env-tag v1.1
pyyaml v6.0.3
watchdog v3.0.0
mkdocs-material v9.7.0 (group: dev)
babel v2.18.0
backrefs v7.0
colorama v0.4.6
 Jinja2 v3.1.8 (*)
markdown v3.10.2
mkdocs v1.6.1 (*)
mkdocs-material-extensions v1.3.1
paginate v0.5.7
pygments v2.20.0
pyyaml v6.0.3
requests v2.34.2 (*)
mkdocstrings v1.0.4 (group: dev)
 Jinja2 v3.1.8 (*)
markdown v3.10.2
markupsafe v3.0.3
mkdocs v1.6.1 (*)
mkdocs-autorefs v1.4.4
markdown v3.10.2
markupsafe v3.0.3
mkdocs v1.6.1 (*)
pyyaml v6.0.3
pyyaml-env-tag v1.1
pyyaml v6.0.3
st-serialize v0.5.0
librt v0.11.0
mypy v2.19.0 (group: dev)
mypy-extensions v1.1.0
pathspec v1.1.1
typing-extensions v4.15.0
nox v2026.4.10 (group: dev)
sphinx v8.1.1
attr v26.1.0
colorlog v8.10.1
dependency-groups v1.3.1
packaging v26.2
humanize v4.15.0
packaging v26.2
virtualenv v21.3.3
distlib v0.4.0
filelock v3.29.0
platformdirs v4.9.6
python-discovery v1.3.1
filelock v3.29.0
platformdirs v4.9.6
pdbpp v0.12.1 (group: dev)
fancycompleter v0.11.1
pygments v2.20.0
pre-commit v4.8.0 (group: dev)
cfig v3.5.0
identify v2.8.19
nodeenv v1.10.0
pyyaml v6.0.3
virtualenv v21.3.3 (*)
ptpython v3.0.32 (group: dev)
aspid v1.4.4
hdi v0.20.0
pargo v0.8.7
prompt-toolkit v3.0.52
scandir v1.10.7
pygments v2.20.0
pydocstyle v6.3.0 (group: dev)
snowballstemmer v2.1.0
pytest v8.0.3 (group: dev)
iniconfig v2.3.0
packaging v26.2
pluggy v1.6.0
pygments v2.20.0
pytest-cov v7.1.0 (group: dev)
coverage v7.14.1
pluggy v1.6.0
pytest v8.0.3 (*)
pytest-skip-slow v0.8.5 (group: dev)
pytest v8.0.3 (*)
pytest-sugar v1.1.1 (group: dev)
pytest v8.0.3 (*)
termcolor v3.3.0
requests-mock v1.12.1 (group: dev)
requests v2.34.2 (*)
ruff v0.15.15 (group: dev)
ty v0.0.39 (group: dev)
types-requests v2.33.0.20260518 (group: dev)
urllib3 v2.7.0
```

# Paint vs Dye

Operations and Security should be a dye,  
not painted afterwards



All too often, operations and security work as paint, applied after the software is written.

- User story is done and tested, let's release!
- Operations runs that code, they are the customer but not treated as such.
- Security comes in and denies because they are just too busy doing other things.
- Adding AI agents will increase complexity, not reduce it.
- Compliance is a by product of security.

Dye gets applied on cloth, before the attire is made.

Operations and security should be there before the software is made...

# Why Slow Down Now?

We slow down now to avoid panic and stress later.



No one will do good work in a rush and under pressure...

- AI vibe engineering pushes on bottlenecks elsewhere.
- Tasks will take longer to be finished: velocity goes down.
- More Planning, less firefighting.
- Compliance is a side effect of security. *Déjà vu?*
- If Operations and Security are maxed out, give them time to help.

Like driving, slow down now and see the dangers before you stumble on them.

# Software Is A Social Activity

Good software (that sells) is about synergy  
between many teams



Everyone is responsible for quality software. **Everyone.**

- Development, SQA, Operations, and Security teams are obvious.
- Marketing, Sales, Admin, and C-suite teams are not. But, their impact is massive!
- Leadership's role is to create a strategy that everyone aligns with
- Everyone needs to understand everyone else's struggles, needs, and objectives
- AI is a good enhancer, not a replacement.

It's a team game, with the odd heroine,  
where any gear that locks takes the whole team down.

# Not sure where to start?

I am running a **free workshop**  
on **DevSecOps** in 2026  
on **Monday 29th of June**.  
**12:30 to 13:00 GMT**



[Book your place here](#)