A Fully Trained Jedi, You are Not

Adam Shostack
Secure Our Streets
Sept, 2023

SHOSTACK + ASSOCIATES

How Many Jedi?

How Many Jedi?

We talk a lot about Jedi

How to become a cybersecurity Jedi, Part 4: Three lessons from 'Star Wars: The Last Jedi'

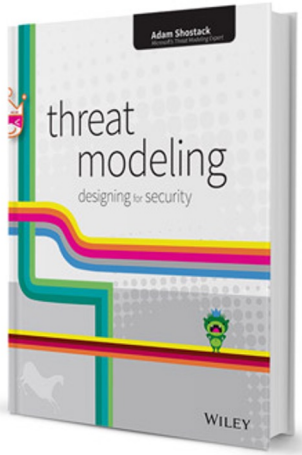
SHOSTACK + ASSOCIATES

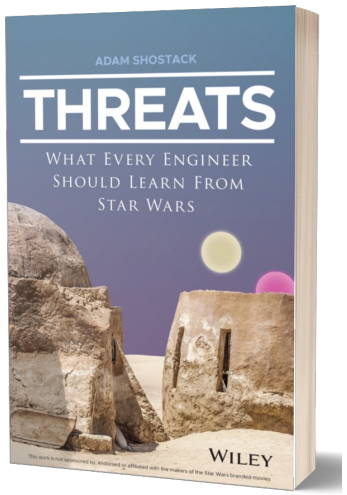
It's a Bad Goal

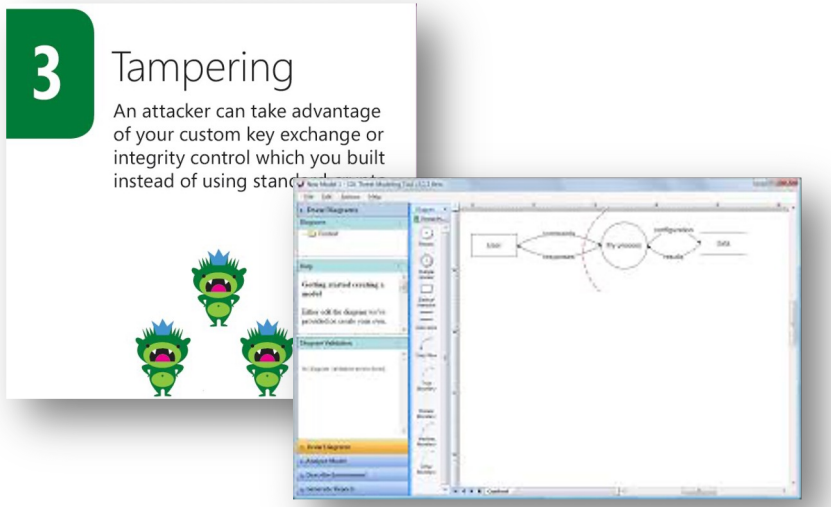
- Expectations of heroism drive burnout
- Tearing children from families…
  - … Forced to live without attachments
- Even so, many people just don’t qualify

SHOSTACK + ASSOCIATES

About Adam Shostack







SHOSTACK + ASSOCIATES

Agenda

- The problem starts with software
- “Shifting left” isn’t working
- Reasonable Expectations
  - Bloom
  - Chunking
  - Frames
  - Threats

We've known for a while…

Axiom 1 (Murphy) *All programs are buggy.*

Theorem 1 (Law of Large Programs) *Large programs are even buggier than their size would indicate.*

Proof: By inspection. ■

Corollary 1.1 *A security-relevant program has security bugs.*

- *Firewalls and Internet Security (Cheswick and Bellovin, 1994)*

SHOSTACK + ASSOCIATES

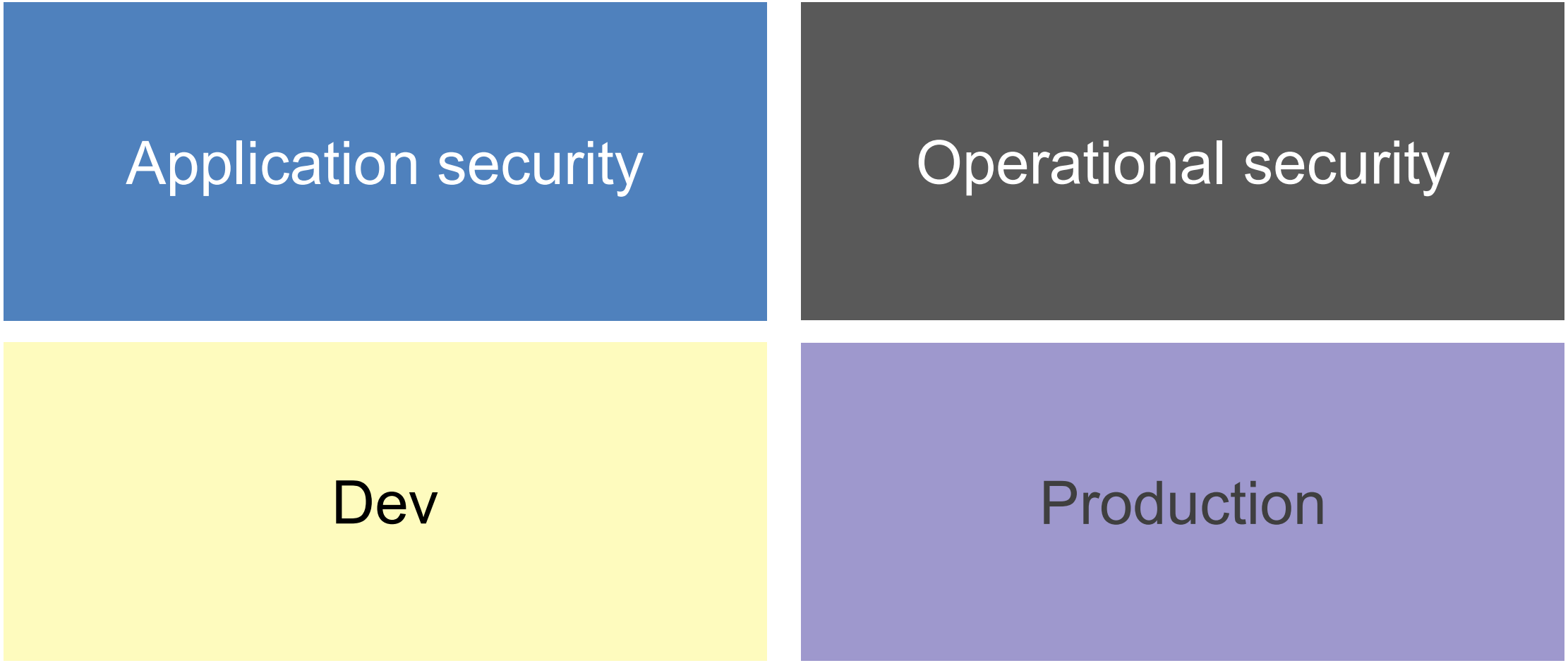
Where do bugs come from?

Software Developers

Developers introduce problems

- Code with security bugs
- Missing security features
- Unusable security features

Software

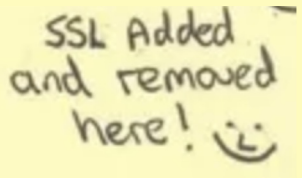


SHOSTACK + ASSOCIATES

Software

Application security

Operational security



Dev

Production

Firehose of CVEs

SHOSTACK + ASSOCIATES

Software

Pen Test

Language improvements

Fuzzing

Application security

Static analysis

Reverse engineering

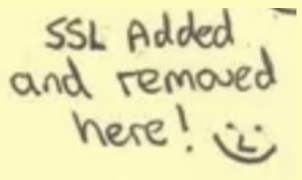
Firewalls!

Operational security

Vuln Scan

SIEM

XDR



Dev

Production

Firehose of CVEs

SHOSTACK + ASSOCIATES

Software

Pen Test

Language improvements

Fuzzing

Application security

Static analysis

Reverse engineering

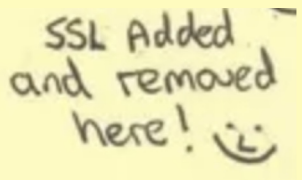
Firewalls!

Operational security

Vuln Scan

SIEM

XDR



Dev

Production

Firehose of CVEs

“Shift Security Left”

“Shift Left”

- Build security in
- Changes to how we/they design, develop, deploy
  - Requires new skills
  - Less pen testing
    - More software engineering
- Growing popularity

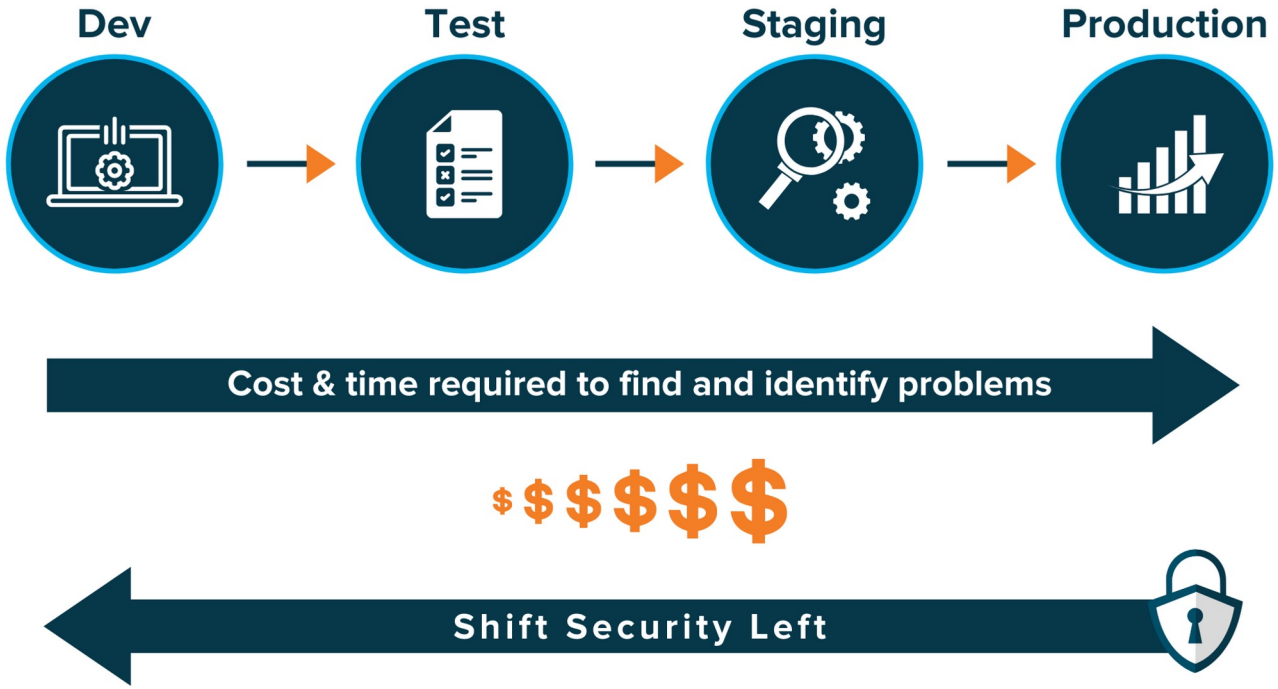


Image: Klogix

Shifting left?

Shift left implies: Change the development process

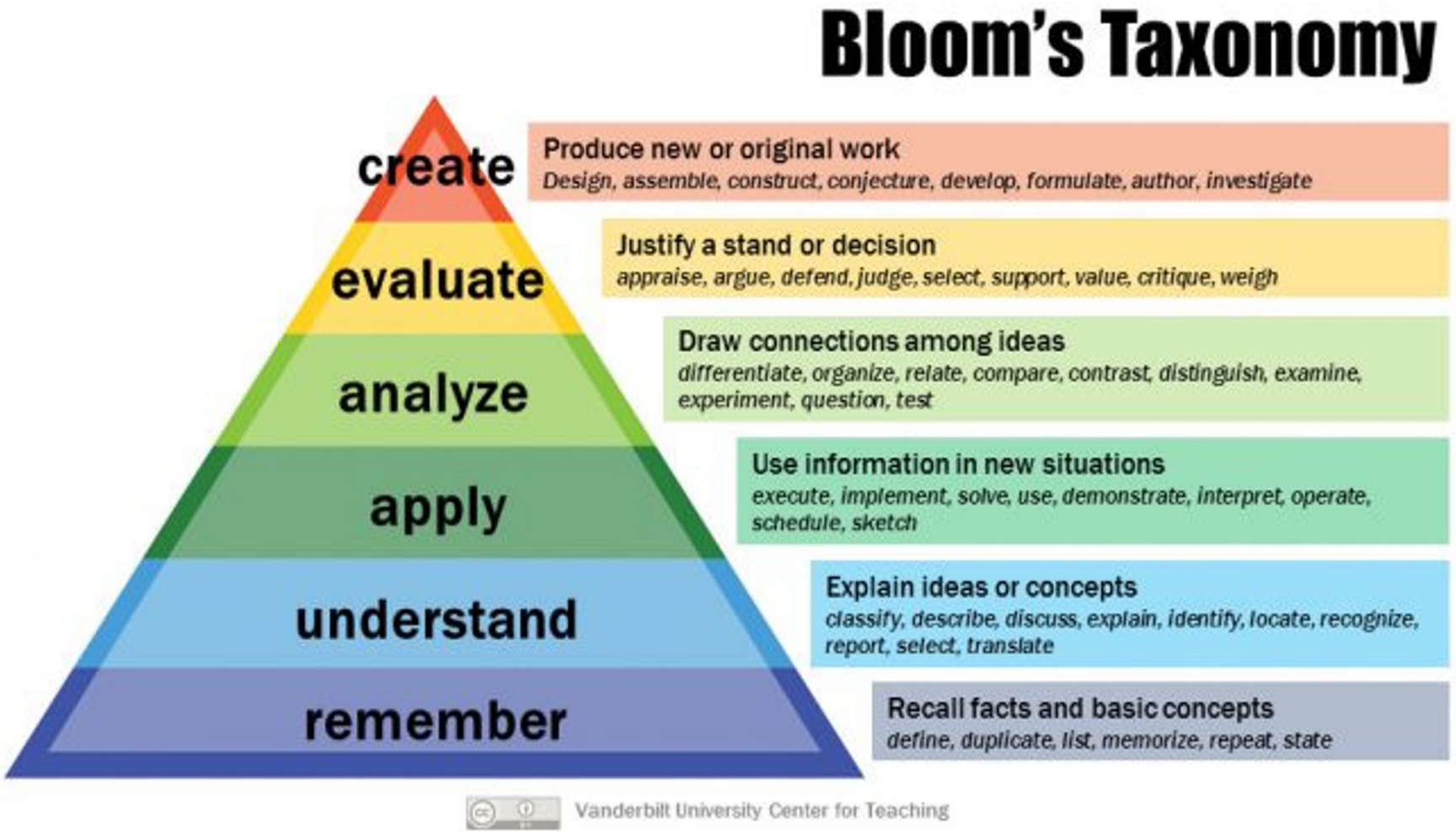
- Demands clear responsibilities
- What exactly is changing?
  - Deliverables
  - Tasks
  - Skills
- Risk: Are we doing this to please appsec?

Clarify

Who delivers what to whom?

How?

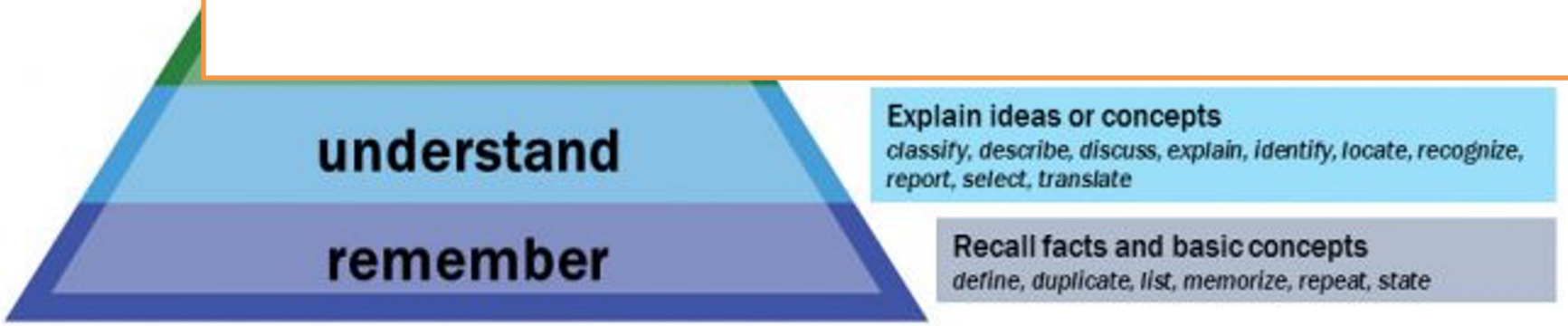
One tool - Bloom's Taxonomy



- Fundamental tool in learning
- Goals + evaluations

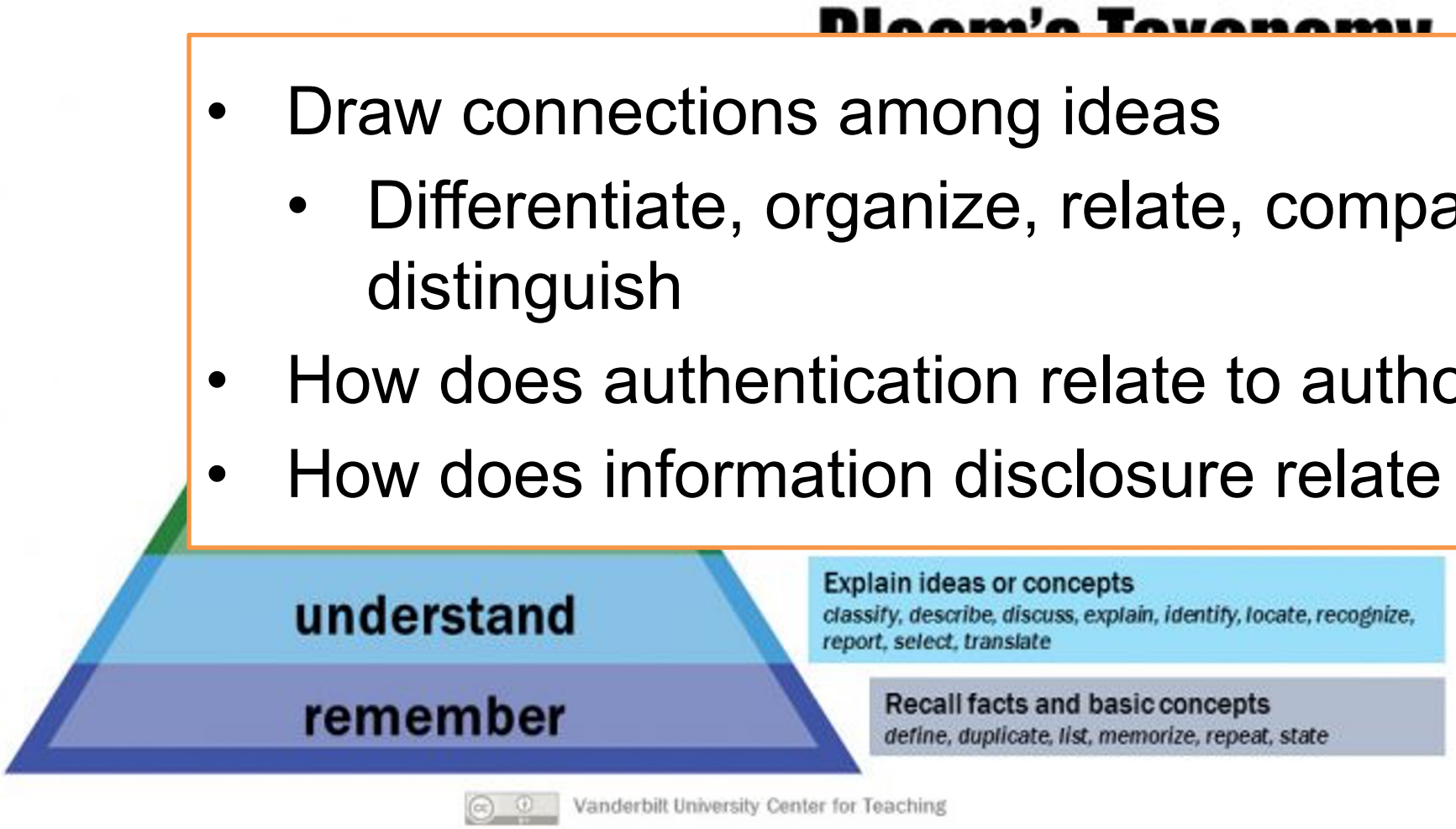
Bloom's Taxonomy: Remember

- Recall facts and basic concepts
  - Define, duplicate, list, repeat
- "Remember data sent over a network can be read by anyone"



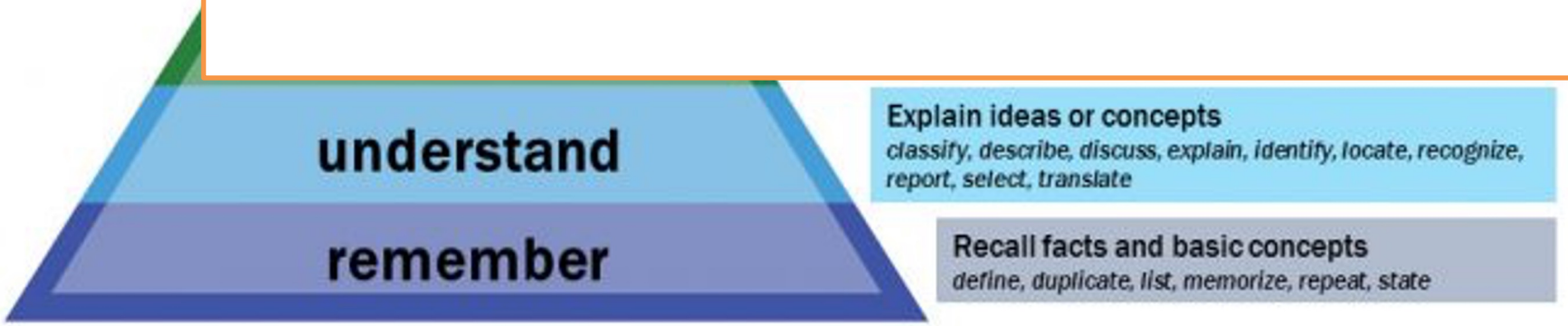
Bloom's Taxonomy: Analyze

- Draw connections among ideas
  - Differentiate, organize, relate, compare, contrast, distinguish
- How does authentication relate to authorization?
- How does information disclosure relate to tampering?



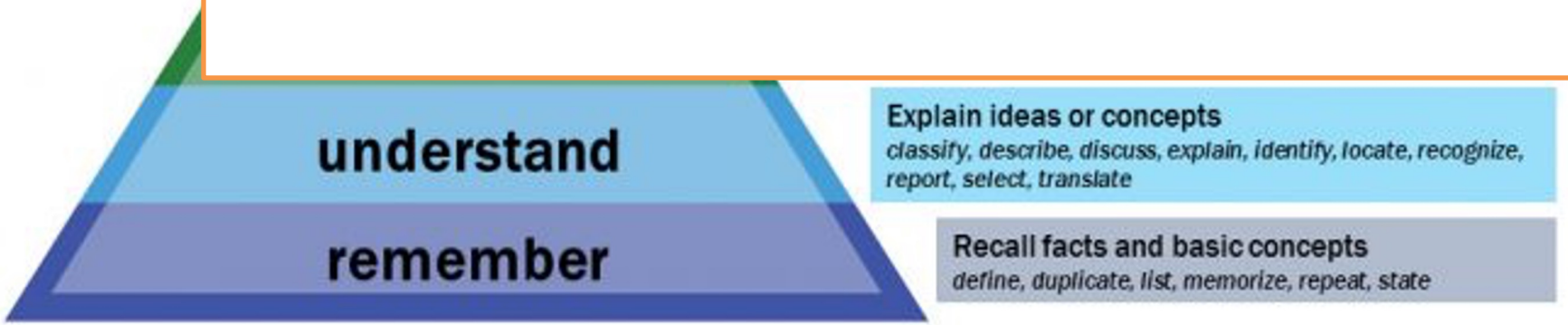
Bloom's Taxonomy: Evaluate

- Justify a stand or decision
  - Argue, defend, judge, select, critique, weigh
- Does encryption protect against that threat?



Bloom's Taxonomy: Create

- Produce new or original work
  - Design, assemble, investigate



Tools help us use Bloom to define skills + knowledge

Bloom Question Stems

Remembering

- Make a story map showing the main events.
- Make a time line of your typical day.
- Make a concept map of the topic.
- Write a list of keywords you know about….
- What characters were in the story?
- Make a chart showing…
- Make an acrostic poem about…
- Recite a poem you have learned.

Questions for Remembering

- What happened after...?
- How many...?
- What is...?
- Who was it that...?
- Name the ...?
- Find the definition of…
- Describe what happened after…
- Who spoke to...?
- Which is true or false...?

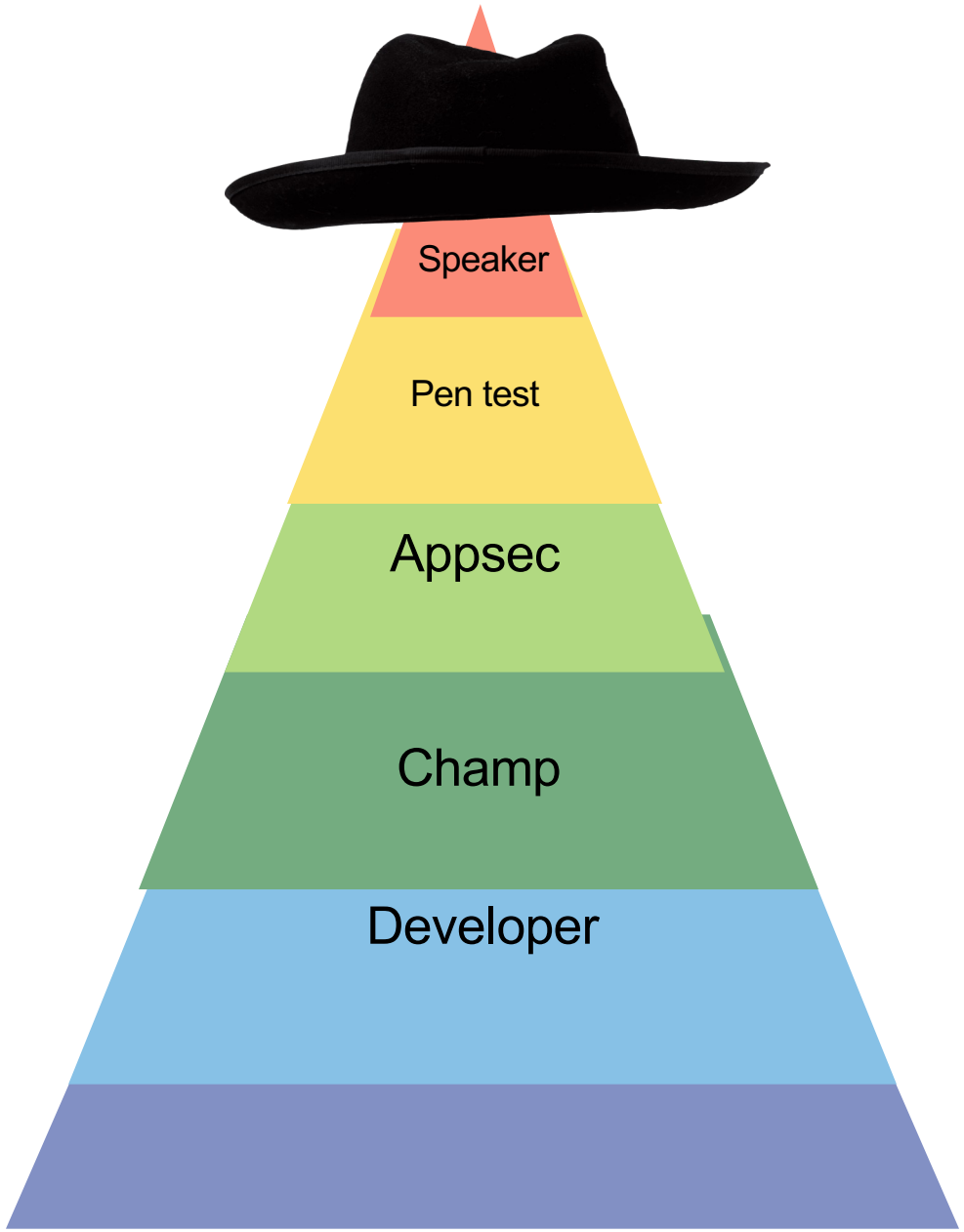
The Helpful Hundred – Planning for Instruction

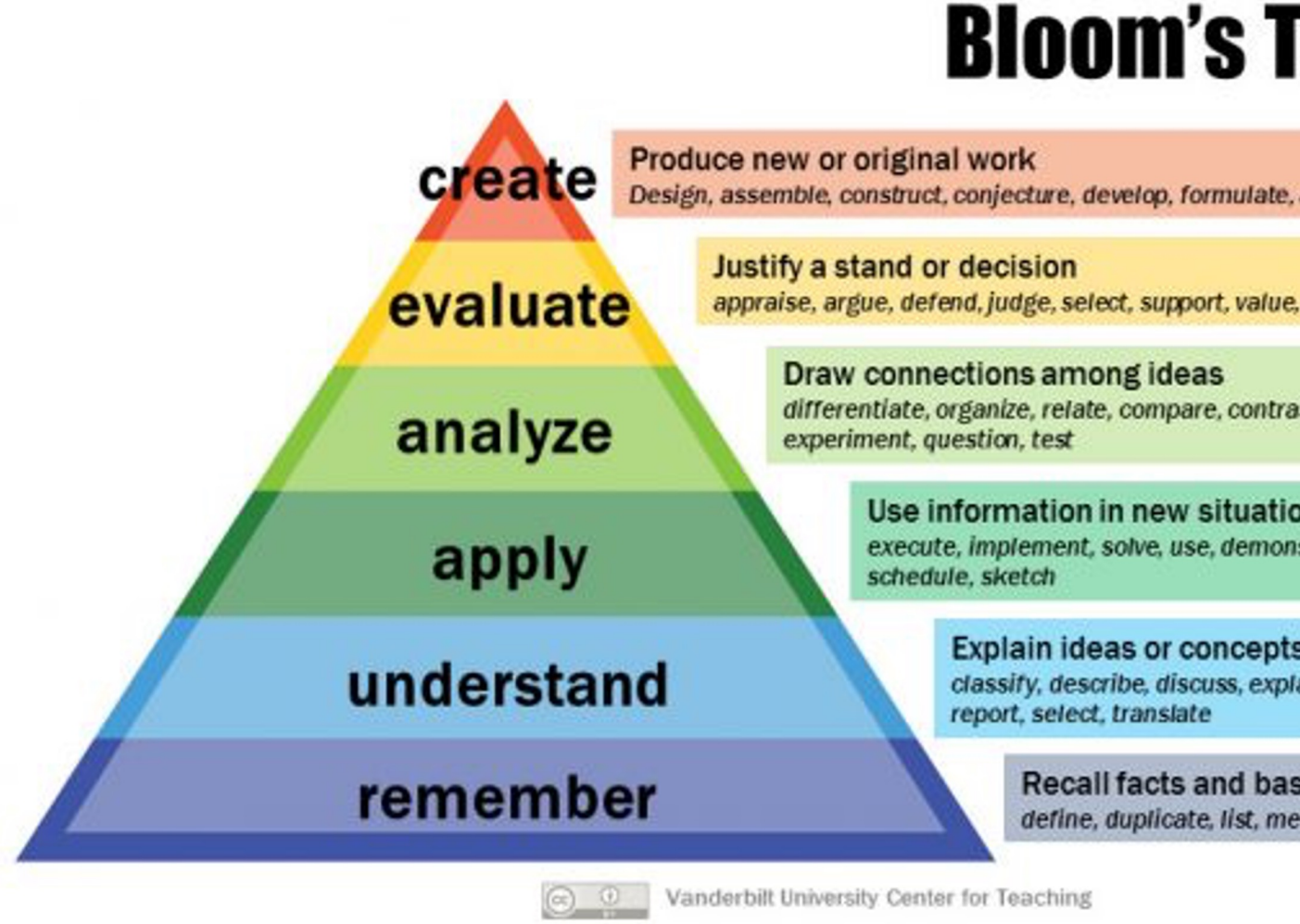
Smaldino, Lowther, and Russell (2008) suggest 100 verbs that highlight performance. Each of these verbs is observable and measurable, making them work quite well in writing objectives for learning. This is not to say that these 100 verbs are the only ones are can be used effectively; however, they provide a great reference.

add	compute	drill	label	predict	state
alphabetize	conduct	estimate	locate	prepare	subtract
analyze	construct	evaluate	make	present	suggest
apply	contrast	explain	manipulate	produce	swing
arrange	convert	extrapolate	match	pronounce	tabulate
assemble	correct	fit	measure	read	throw
attend	cut	generate	modify	reconstruct	time
bisect	deduce	graph	multiply	reduce	translate
build	defend	grasp	name	remove	type
cave	define	grind	operate	revise	underline
categorize	demonstrate	hit	order	select	verbalize
choose	derive	hold	organize	sketch	verify
classify	describe	identify	outline	ski	weave
color	design	illustrate	pack	solve	weigh
compare	designate	indicate	paint	sort	write
complete	diagram	install	plot	specify	
compose	distinguish	kick	position	square	

Source: Smaldino, S. E., Lowther, D. L., & Russell, J. D. (2008). Instructional Media and Technologies for Learning (9th ed). Upper Saddle River, NJ: Pearson.

What security work do we ask of different people?

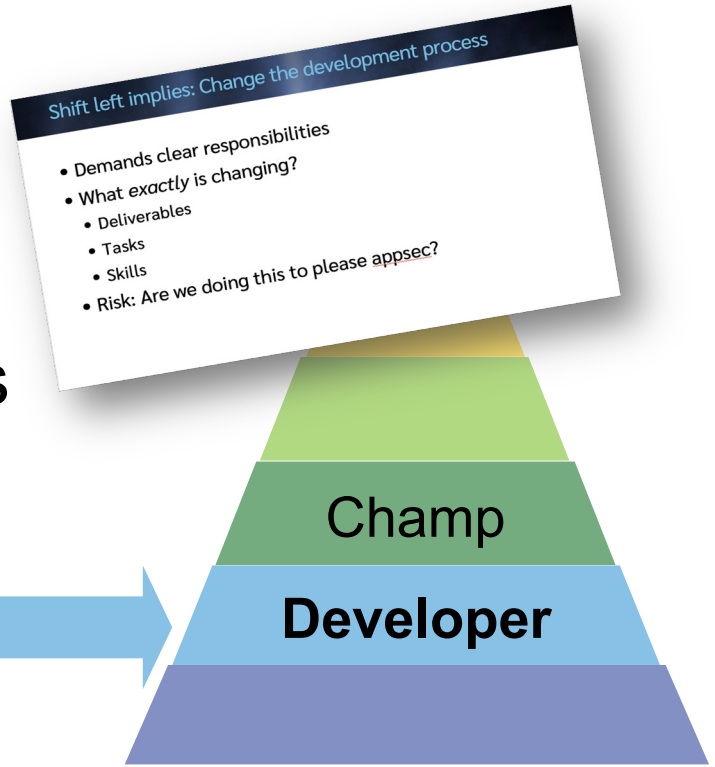




But instead... we teach like this?

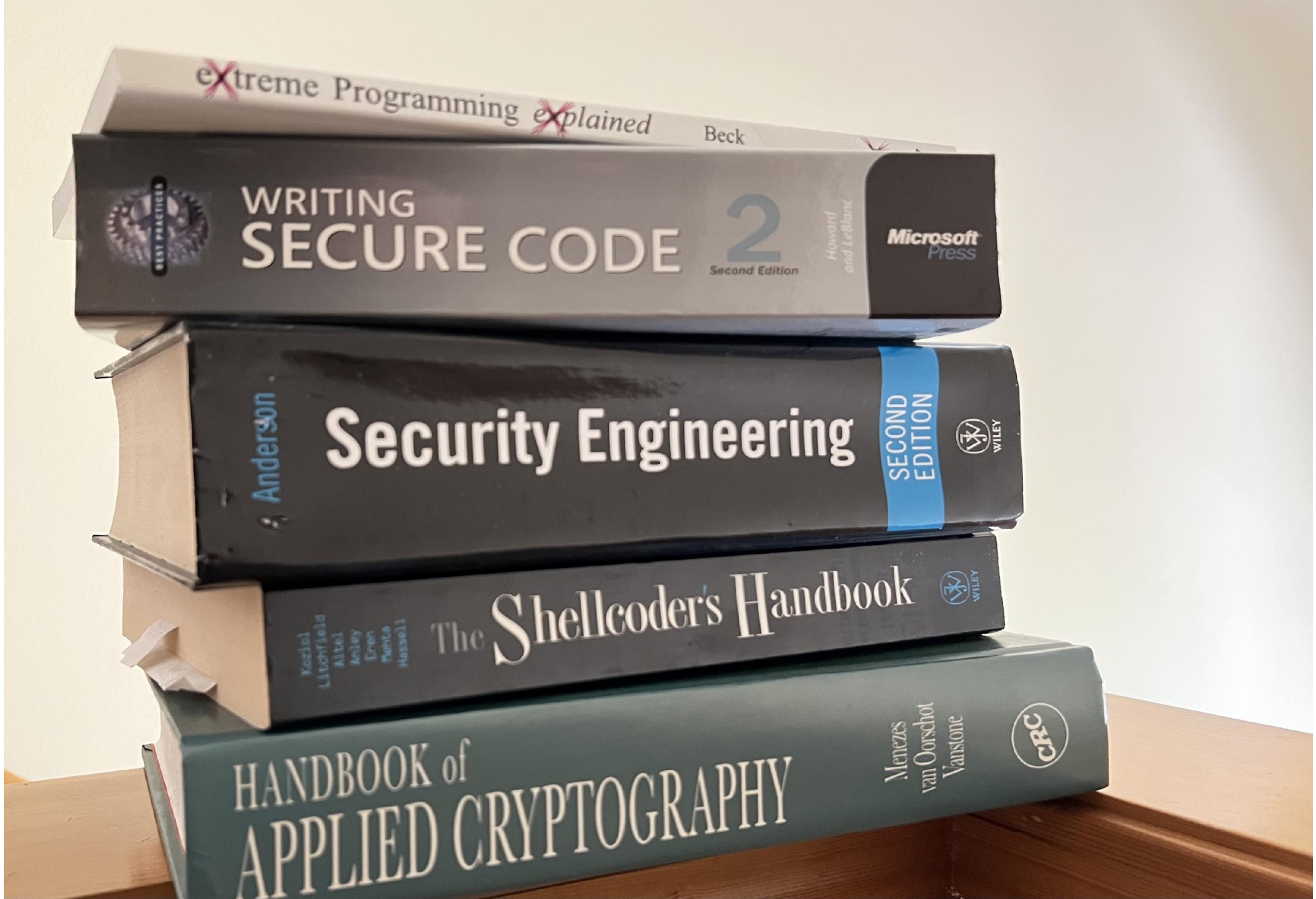
Criteria + constraints

- Align to job, aspirations
- Within reasonable training time
- Goals
  - Help people find, follow paved roads
  - Recognize danger signs



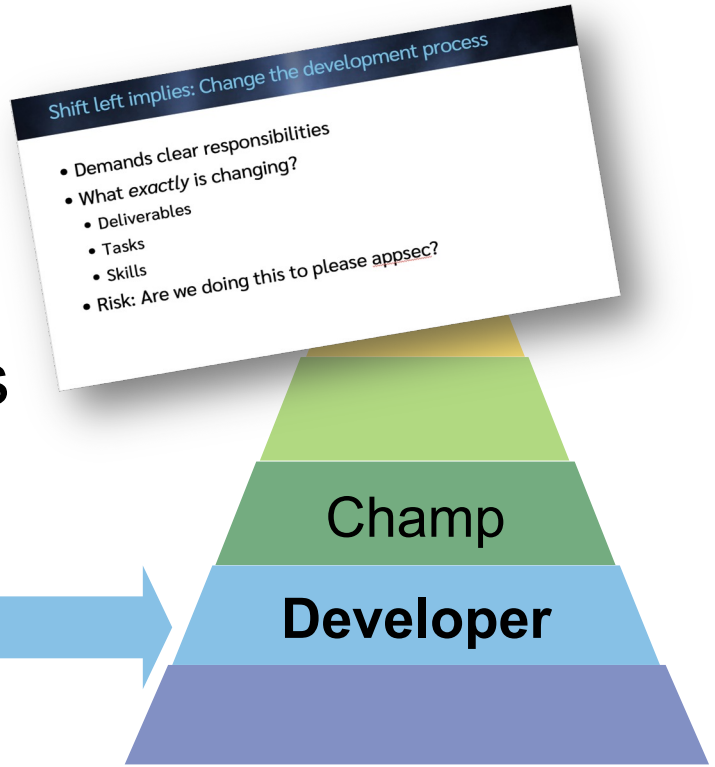
What fits here?

SHOSTACK + ASSOCIATES


eXtreme Programming eXplained
Beck
WRITING
SECURE CODE
2
Second Edition
Howard and LeBlanc
Microsoft Press
Anderson
Security Engineering
SECOND EDITION
WILEY
Kozial Litchfield Aitel Ankey Eren Mehta Hassell
The Shellcoder's Handbook
WILEY
HANDBOOK of
APPLIED CRYPTOGRAPHY
Menezes van Oorschot Vanstone
CRC

Criteria + constraints

- Align to job, aspirations
- Within reasonable training time
- Goals
  - Help people find, follow paved roads
  - Recognize danger signs



What fits here?

SHOSTACK + ASSOCIATES

Chunking is crucial

- Our brains are really, really good at pattern recognition
  - Dealing with information in “chunks”
- Short term memory is 7 +/- 2 chunks
- 1,1,2,3,5,8,13,34,55…
- If we don’t define the chunks, our students will
  - (They may anyway!)

Categories and frames

- Exploit techniques?
- Threat actors?
- Compliance?
- Cyberwar?
- Top ten?
- Threats?

“What can go wrong”
focuses our attention on threats

“What can go wrong?” is a powerful framing question

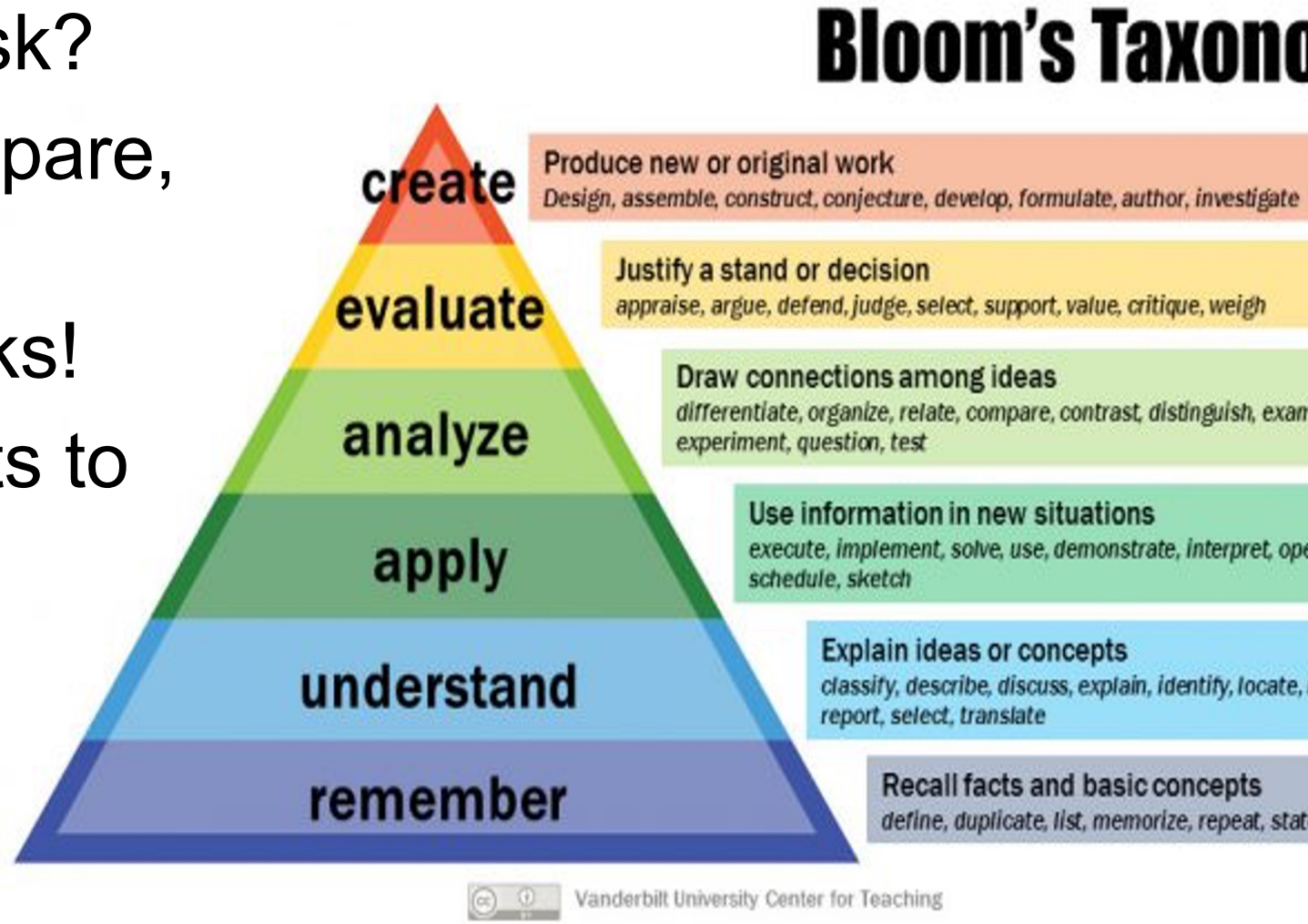
- Everyone has an answer — if you ask and encourage
  - Across industries, technical skill, execs
- Variants
  - “What keeps you up at night?”
  - “How would you attack this”
  - Fortunately/unfortunately game

“What can go wrong?” is an umbrella

- Open ended is easier to answer, but answers vary a lot
- Structures
  - Finance execs … ORX
  - Security … OWASP top ten
  - FDA …. inability to update
  - Compliance… see my Threat Modeling Compliance (BHAsia ’21)
- Flaws, not bugs

What's the single best toolset?

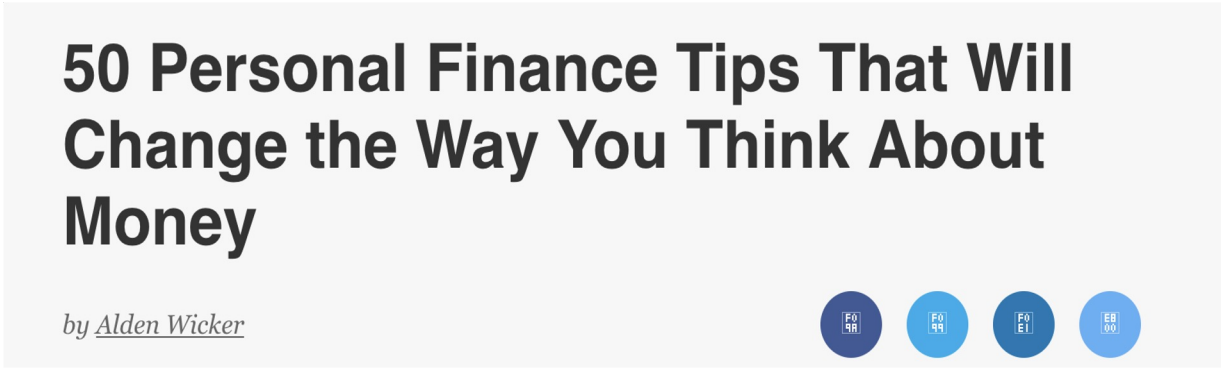
- 4 ways to do a side task?
- I have to analyze, compare, evaluate?
  - Those are expert tasks!
- So people need experts to offer specific advice



Single best tool need: Personal finance example

- Max out your tax advantaged, matched accounts…

50?!?!!



- Target date funds

SHOSTACK + ASSOCIATES

What does every engineer need to know?

- The question is catalyzed by a few projects
  - Fast, Cheap + Good: An Unusual Tradeoff (whitepaper)
  - *Threats: What Every Engineer Should Learn from Star Wars*
  - Nothing Is Good Enough
- All of which started with a simple question…

Whitepaper available now Shostack.org/whitepapers/ |Others forthcoming: Wiley, Feb, 2022 — IEEE S+P Magazine, Jan/Feb 2023

SHOSTACK + ASSOCIATES

A simple question

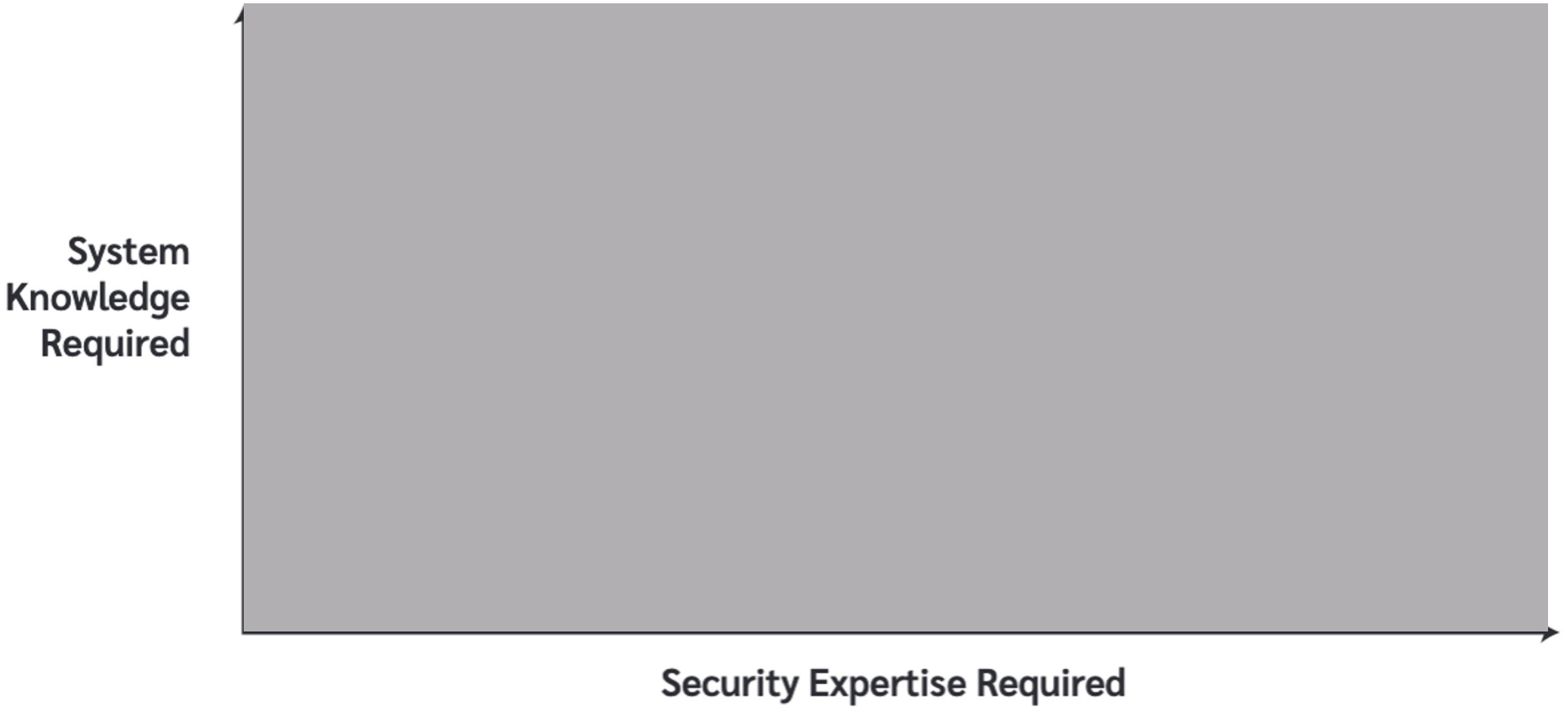
Is every flaw unique?

Another simple question

Do flaws cluster?

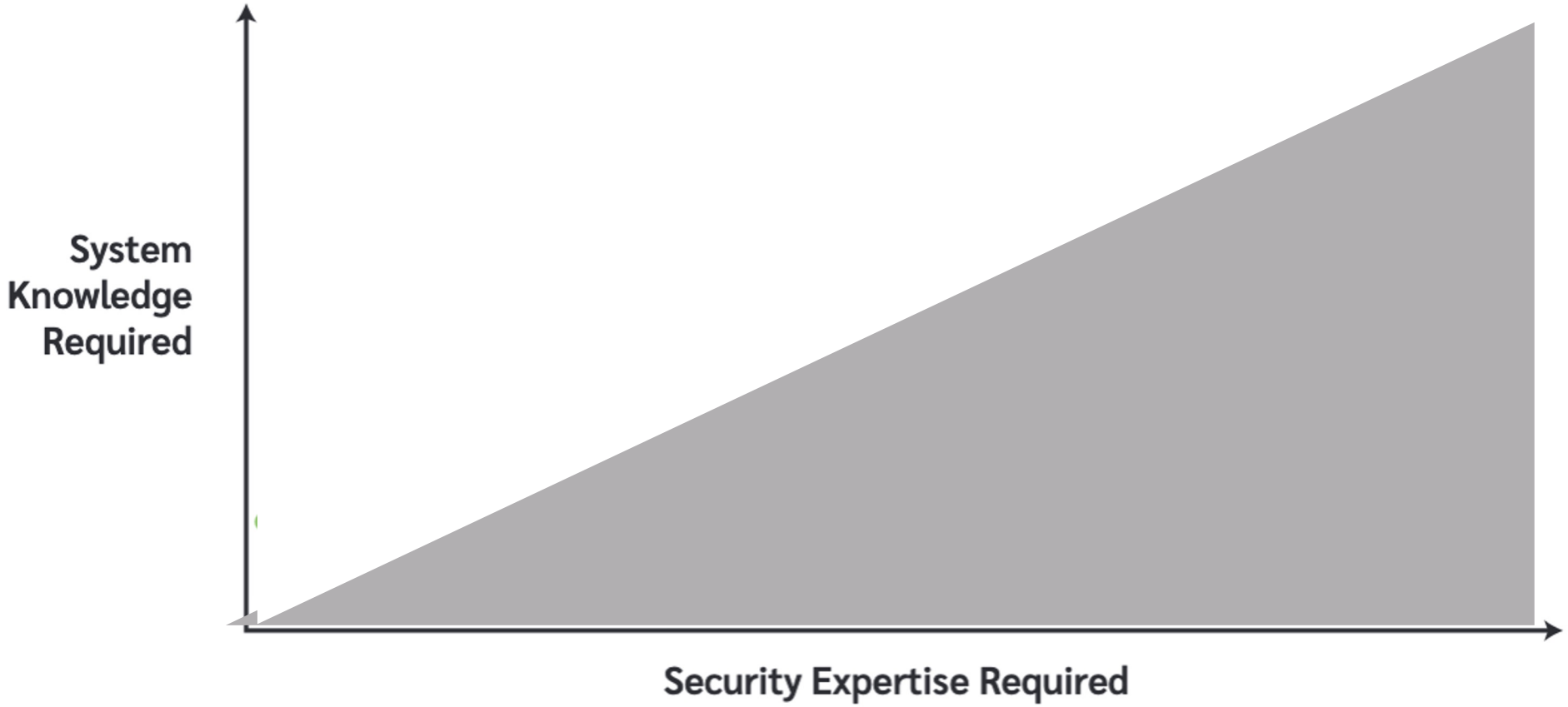
What do we need to know to find them?

Where are the flaws? (1)



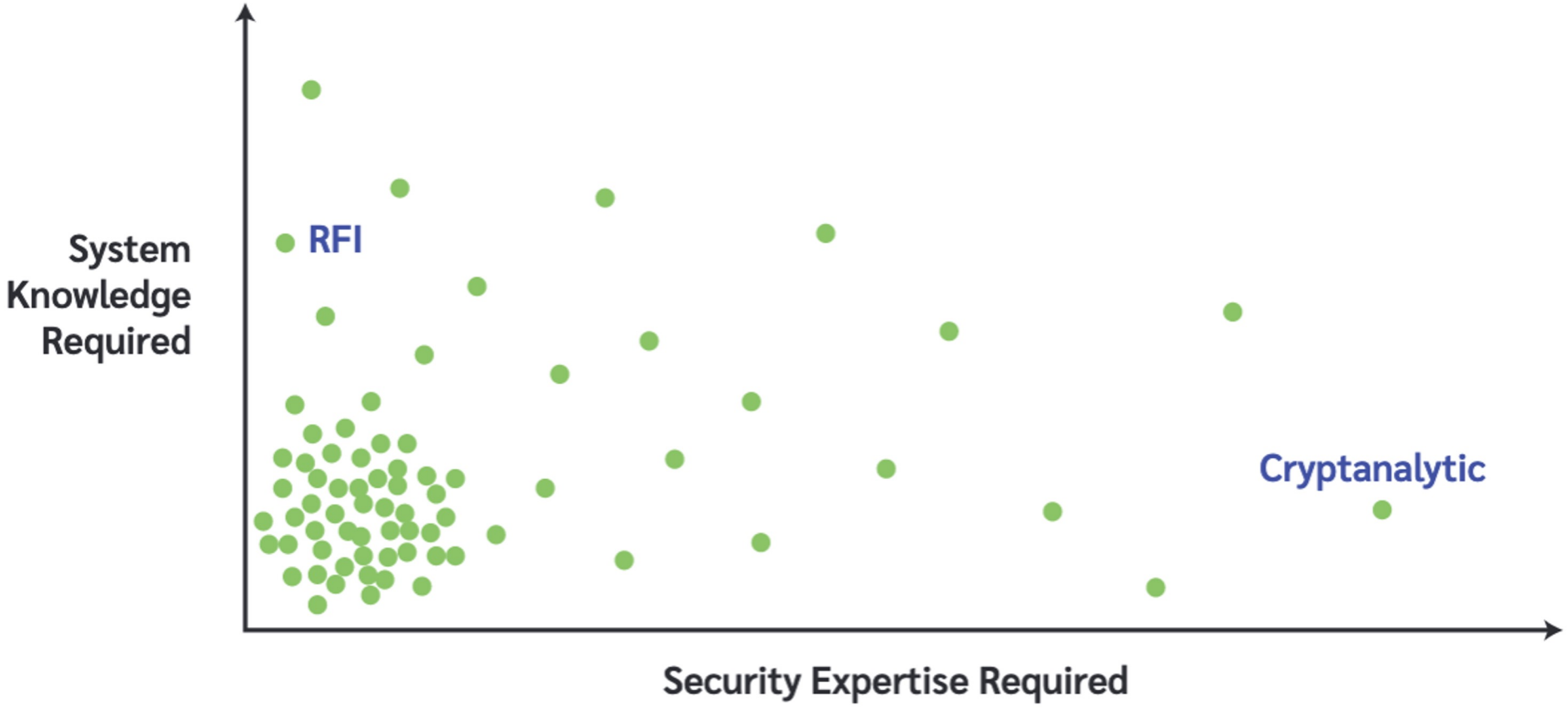
SHOSTACK
+ ASSOCIATES

Where are the flaws? (2)



SHOSTACK
+ ASSOCIATES

Maybe they’re easy to find?



SHOSTACK + ASSOCIATES

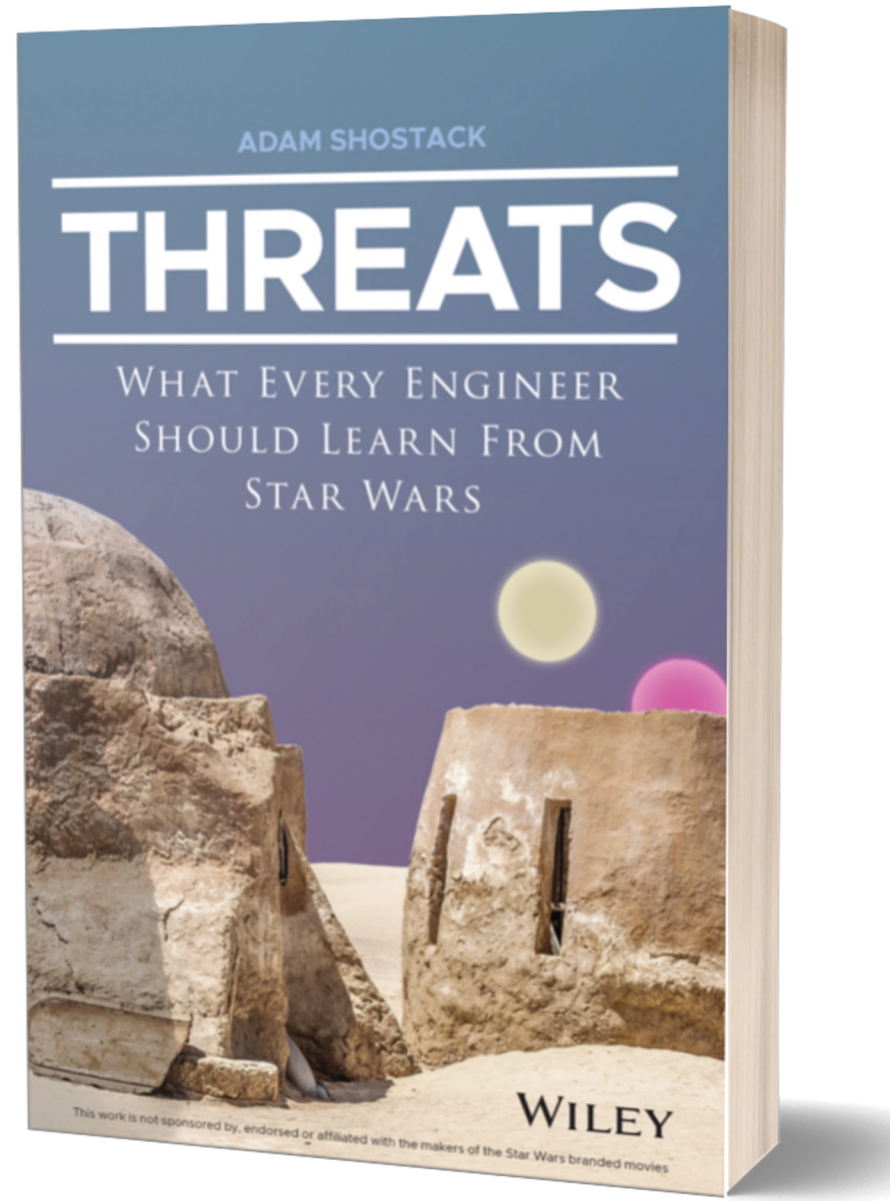
If lightweight flaws are common,
we should transform how
we work with engineers

Why do lightweight flaws exist?

- Developers stink?
- Bad tools? (Languages, scanners)
- Lack of knowledge?

What developers need to know: Framework

- My proposal:
  - STRIDE threats
    - (Spoofing, Tampering, Repudiation, Info disclose, DoS, Expansion of Authority)
  - Parsing + predictability generate danger
  - Kill chains bring these together



SHOSTACK + ASSOCIATES

Recap

- Code issues underly many (most?) security issues
- Shifting left is an admirable goal
- Only works when we're clear about change

A possible future

- Normal levels of security are defined
- Developers able to build more secure systems
- Less rework, fewer escalations, more predictable delivery

Thank you!

SHOSTACK
+ ASSOCIATES

Questions?

adam@shostack.org

SHOSTACK + ASSOCIATES