

Software Assurance – Making the Software Ecosystem Rugged

ICSJWG

October 26th, 2011

me:

Richard Struse

Deputy Director

Software Assurance

National Cyber Security Div.

U.S. Department of Homeland Security

National Protection & Programs Directorate

richard.struse@dhs.gov



**Homeland
Security**



Software Assurance -

The level of confidence that software is free from vulnerabilities either intentionally designed into the software or accidentally inserted at anytime during its life cycle and that the software functions as intended. *Derived From: CNSSI-4009*

Making the Software Ecosystem

Software, firmware, protocols, interfaces, etc.

Rugged **rug·ged** (rŭg ' ĭd) *adj.*
Having a sturdy build or strong constitution

SOURCE: The Free Dictionary



The Rugged Software Manifesto

I am rugged... and more importantly, my code is rugged.

I recognize that software has become a foundation of our modern world.

I recognize the awesome responsibility that comes with this foundational role.

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.

I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic, and national security.

I recognize these things - and I choose to be rugged.

I am rugged because I refuse to be a source of vulnerability or weakness.

I am rugged because I assure my code will support its mission.

I am rugged because my code can face these challenges and persist in spite of them.

I am rugged, not because it is easy, but because it is necessary... and I am up for the challenge.

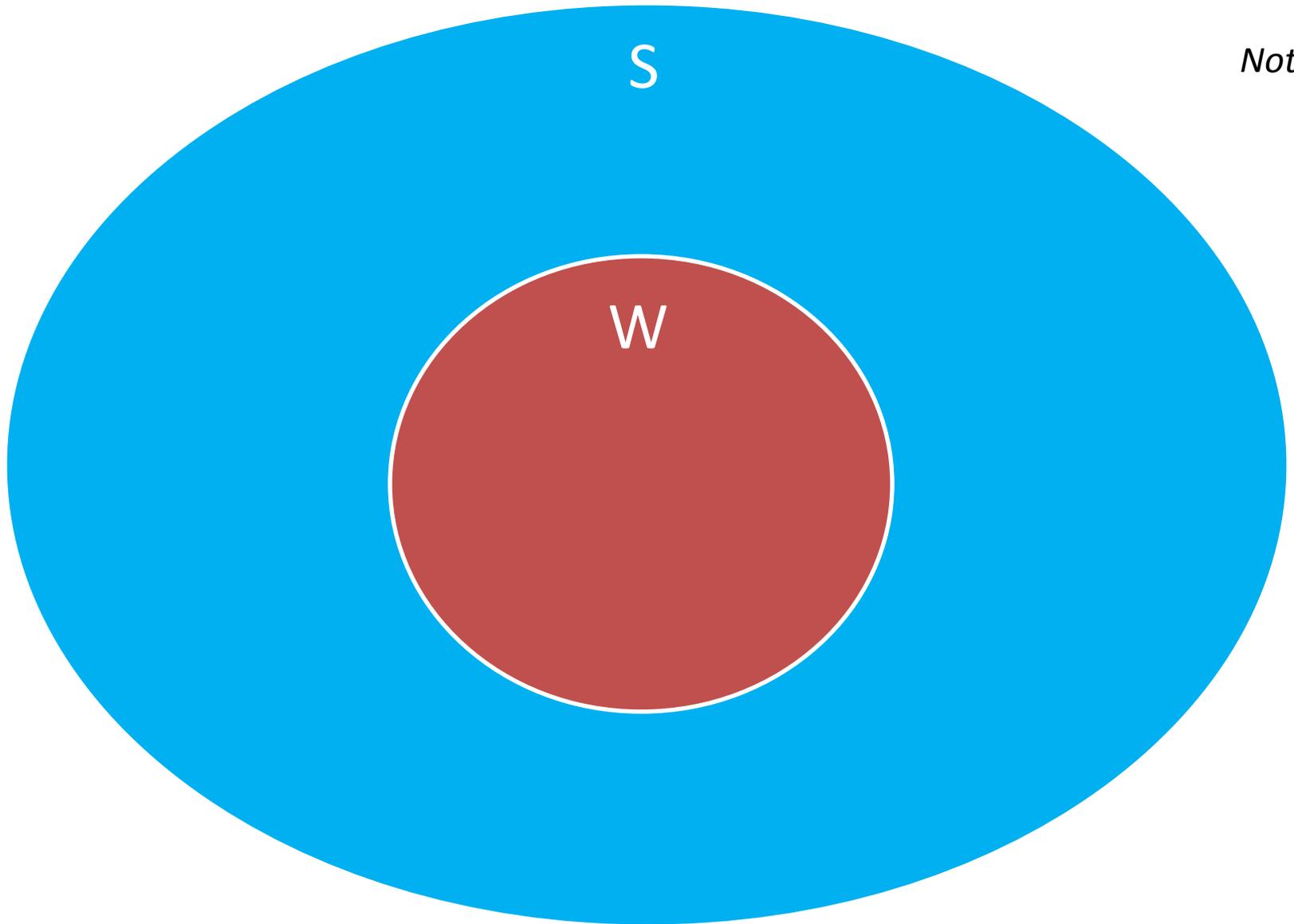
What is the context?

Where can automation help - *today*?

What problems are we trying to solve?

Where do we start?

S: The set of all software in existence at some point in time

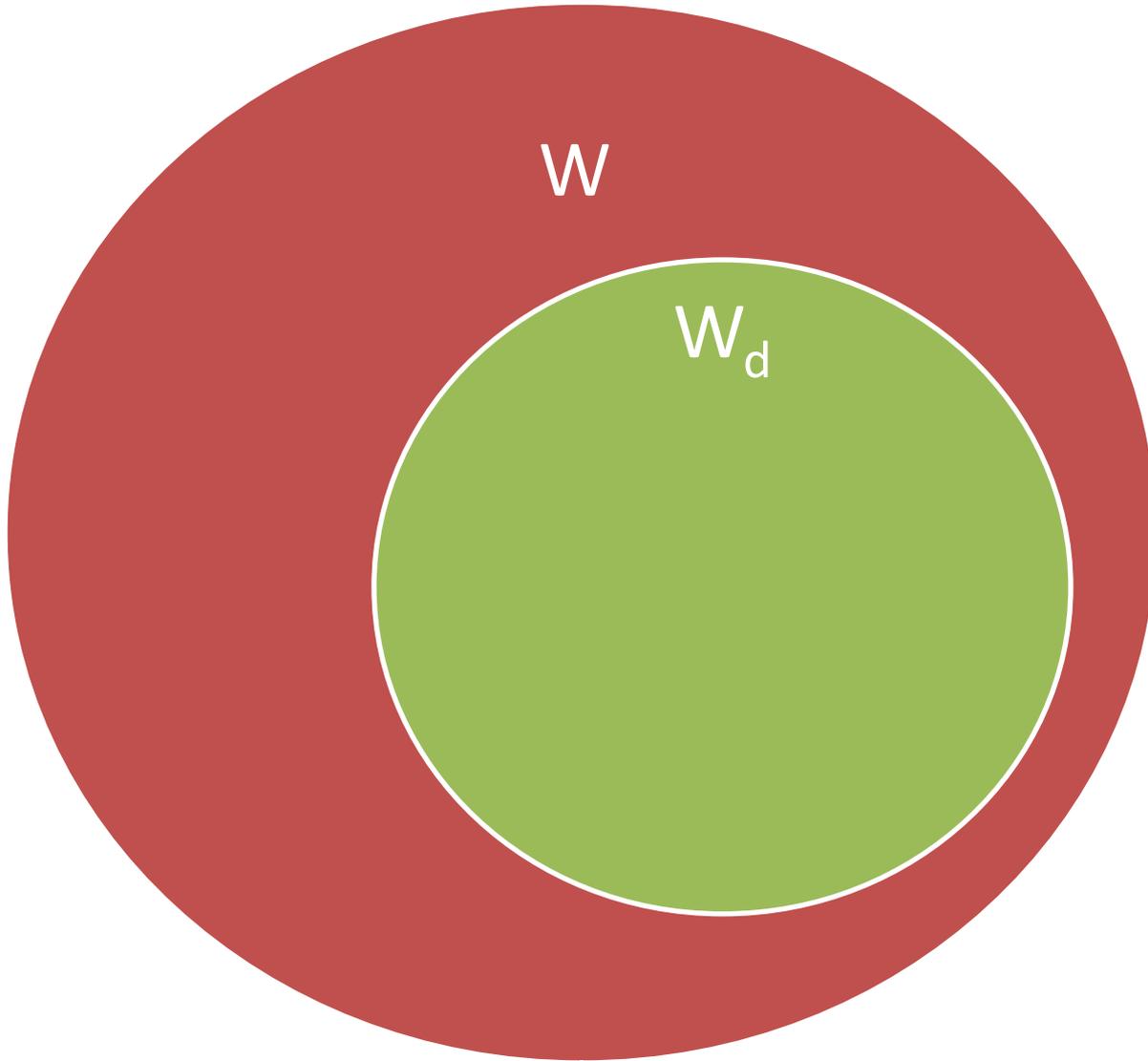


Notional

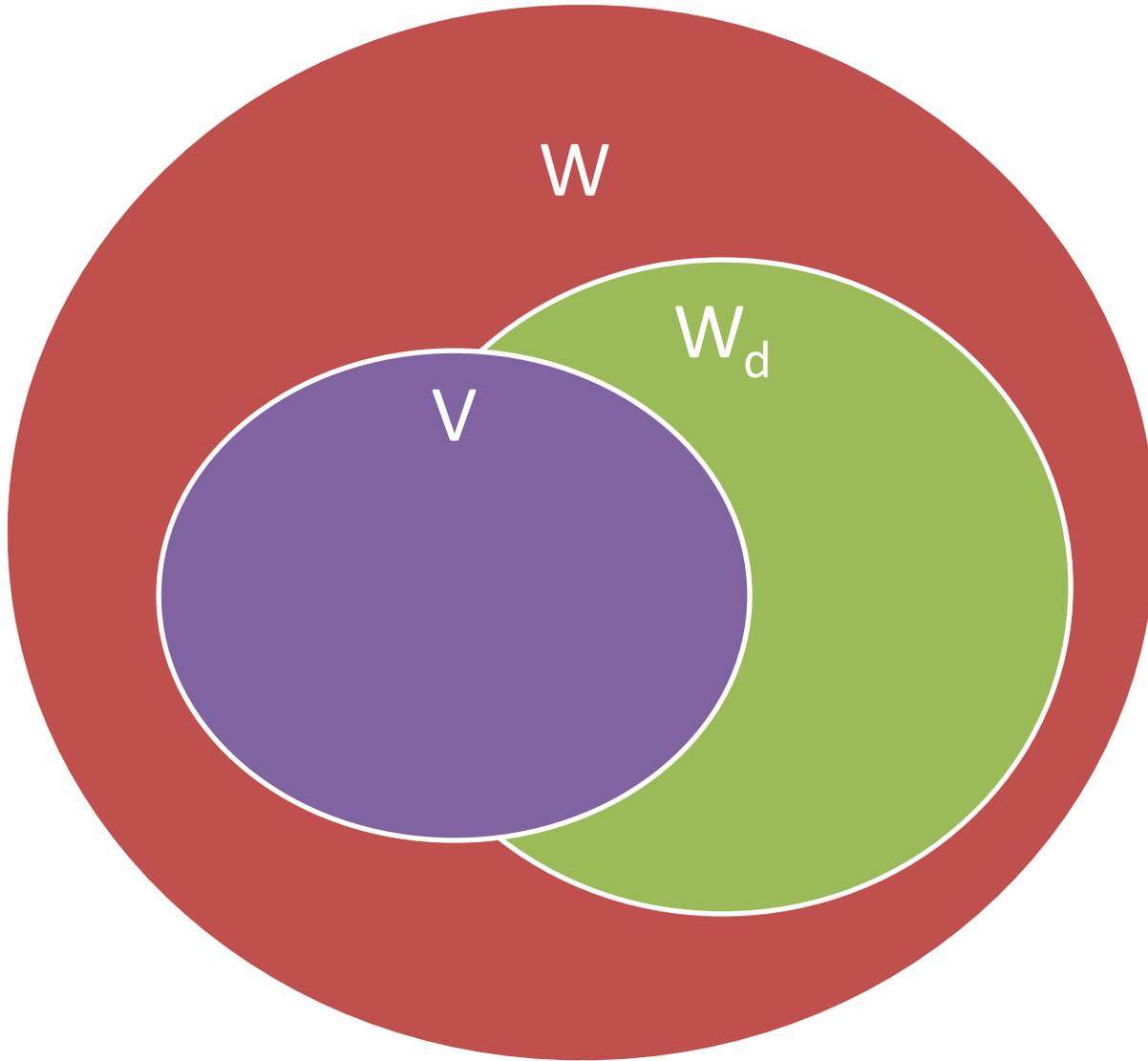
W: The set of all instances of software weaknesses in S

There are many definitions of “weakness.”
What do we mean by weakness *in this context*?

A (software security) weakness is a property of software/systems that, under certain conditions, may permit unintended / unauthorized behavior.



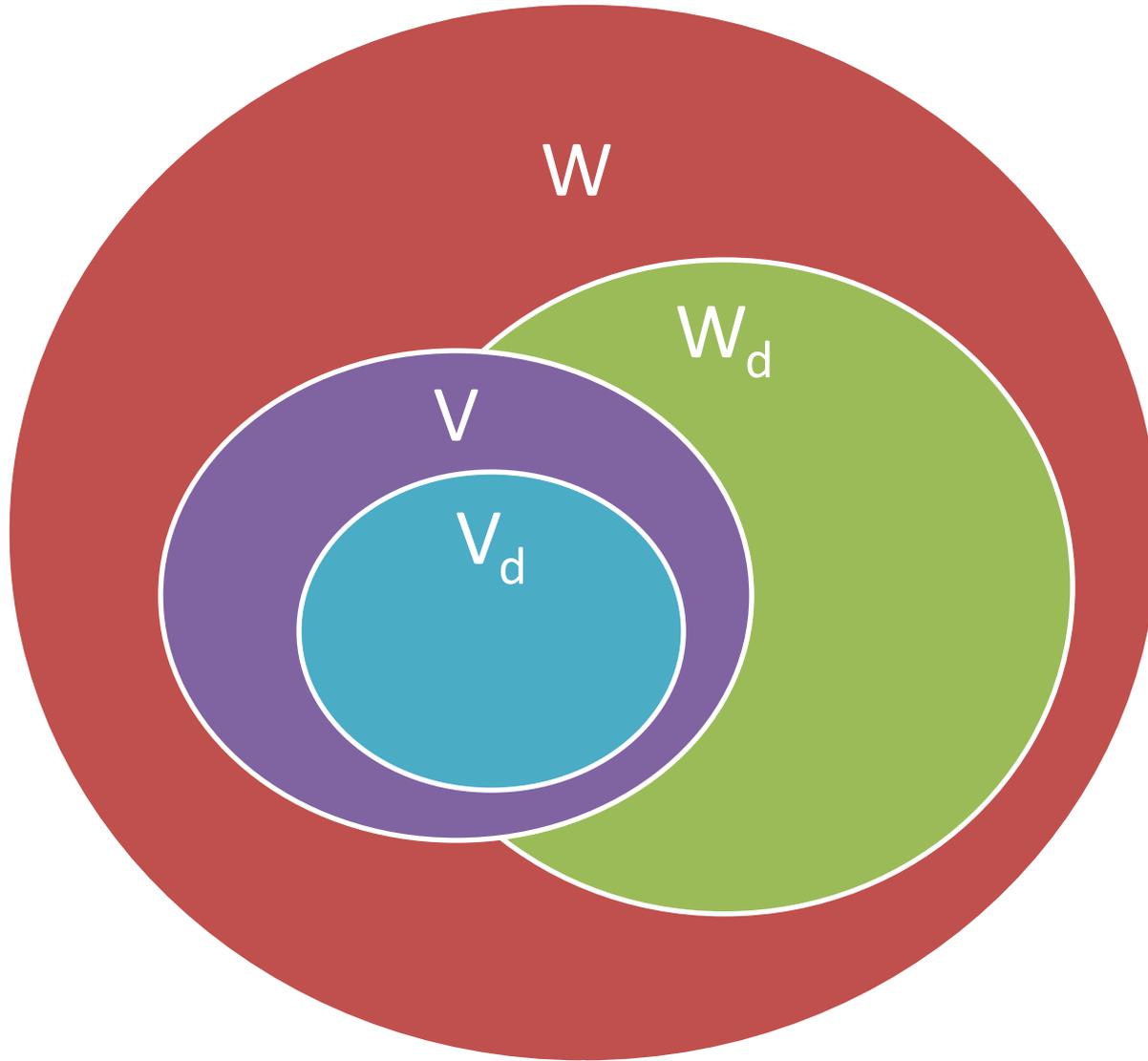
W_d : The set of all *discovered* software weaknesses in **W**



V: The set of all vulnerabilities in W

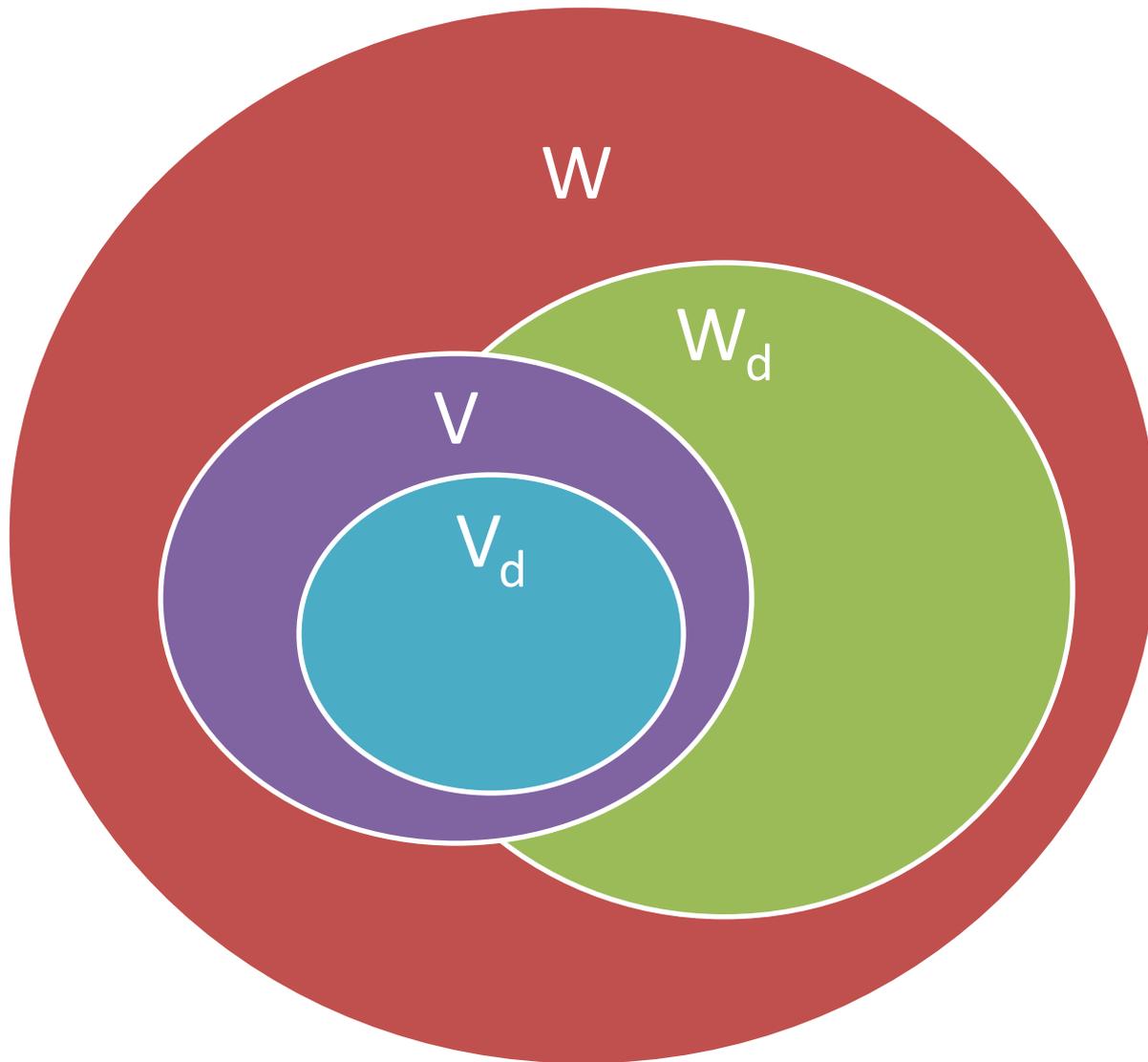
There are many definitions of “vulnerability.”
What do we mean by vulnerability *in this context*?

A *(software security) vulnerability* is a collection of one or more weaknesses that contain the necessary conditions to permit unauthorized parties to cause the software to perform unintended behavior (a.k.a. “is exploitable”)



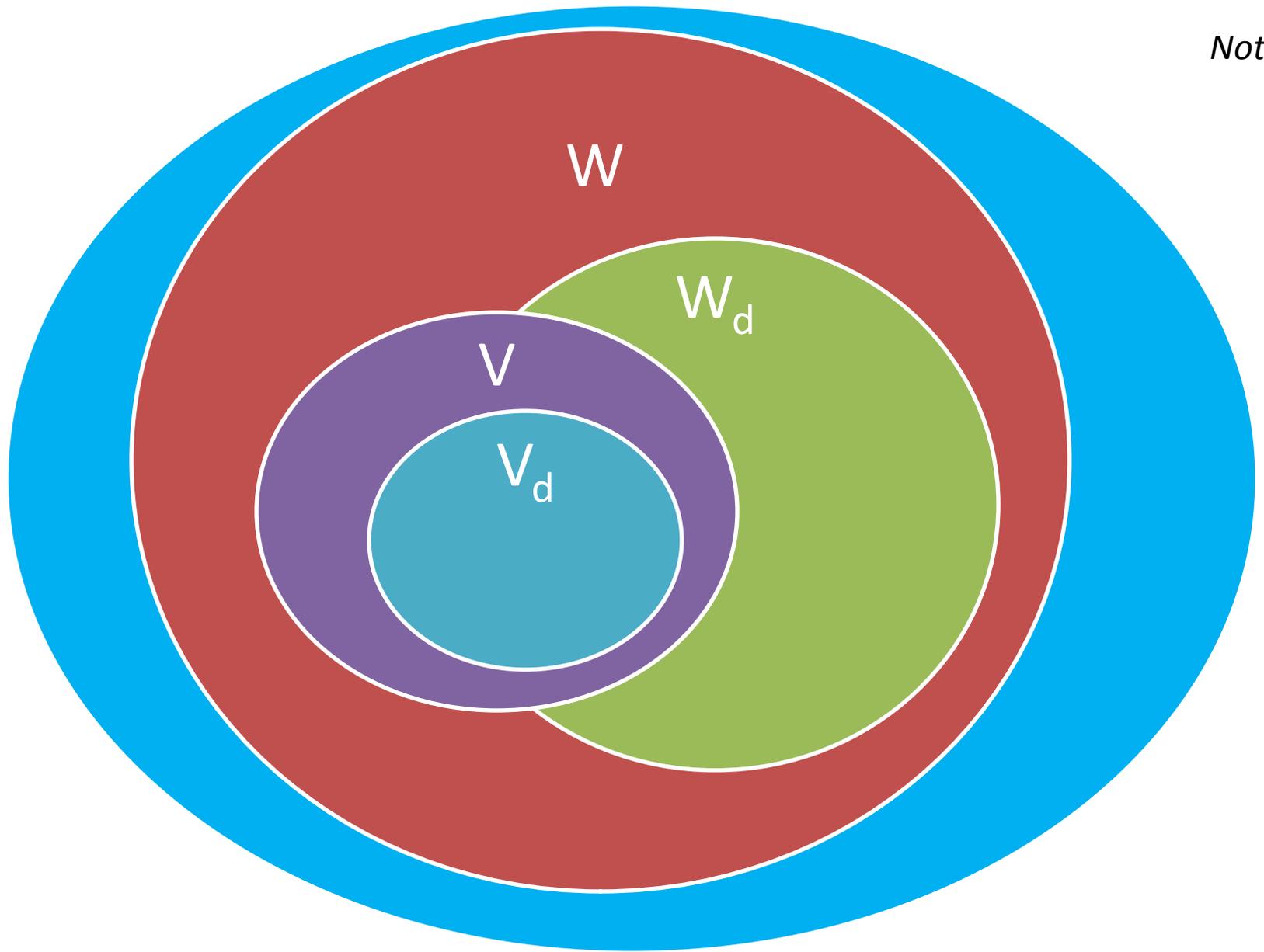
V_d : The set of all *discovered* vulnerabilities in V

Notional



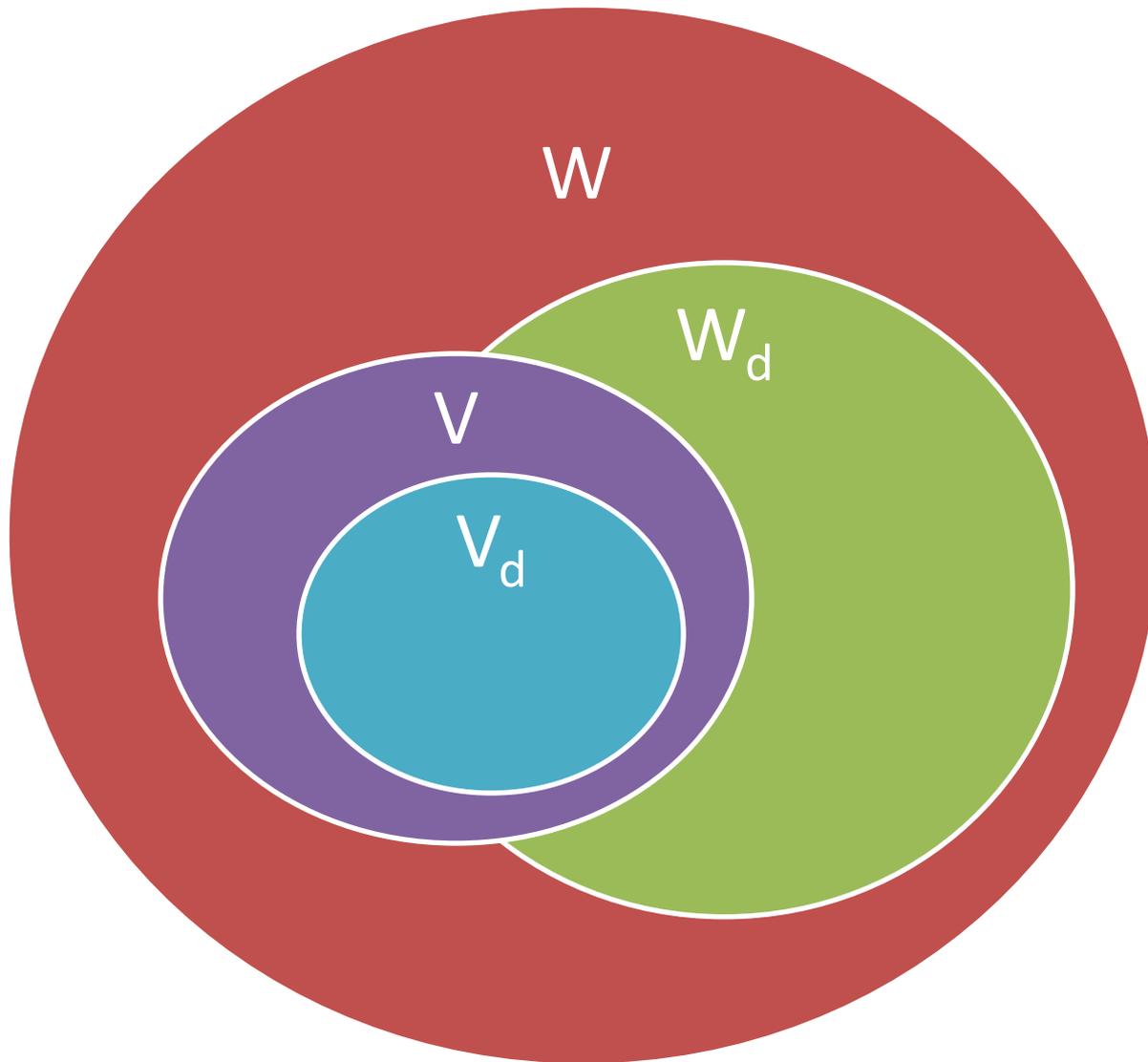
What does the future hold?

Notional



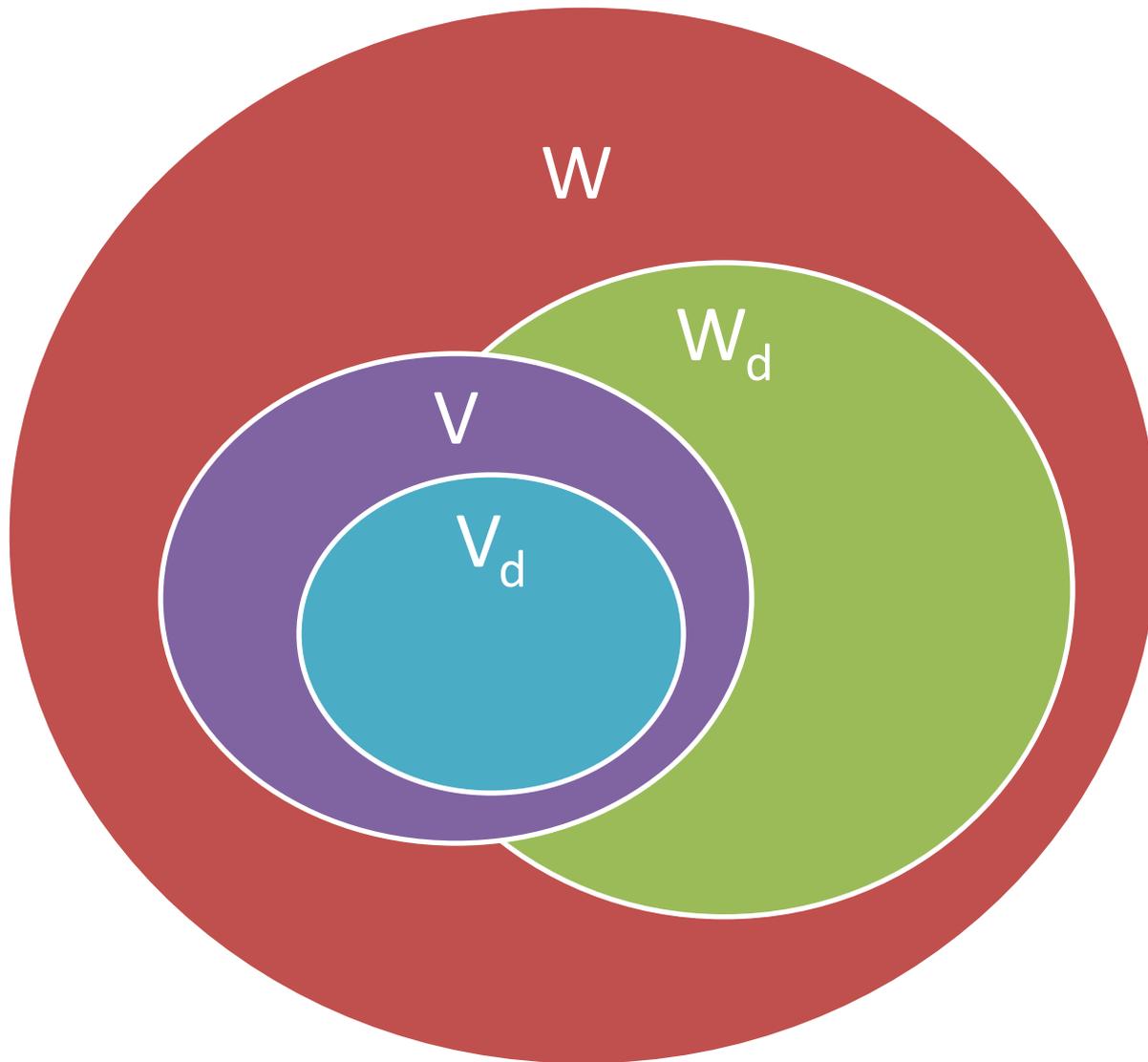
We know it's *not* this, at least not in the near-term

Notional



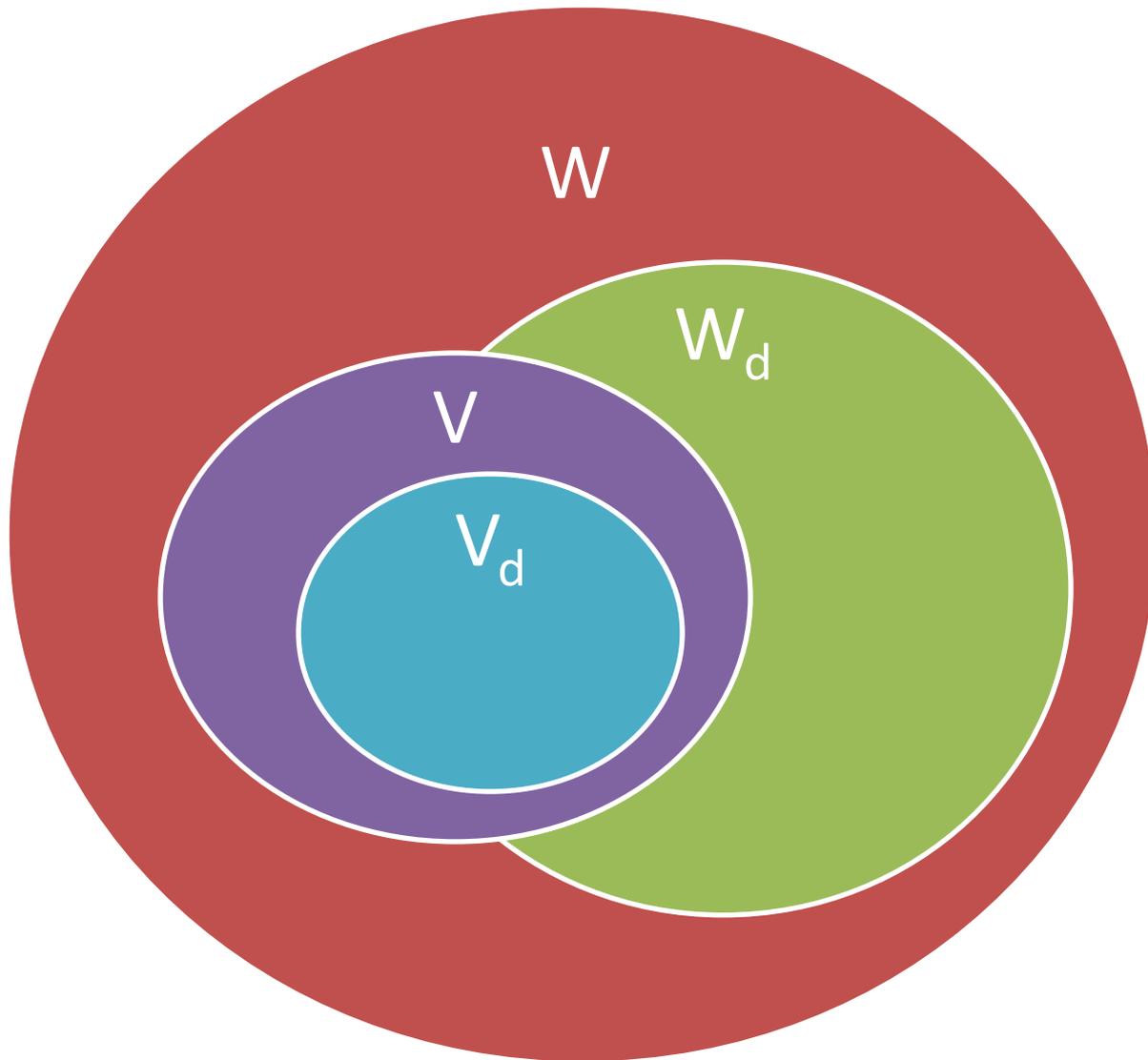
Maybe the problem grows unbounded?

Notional

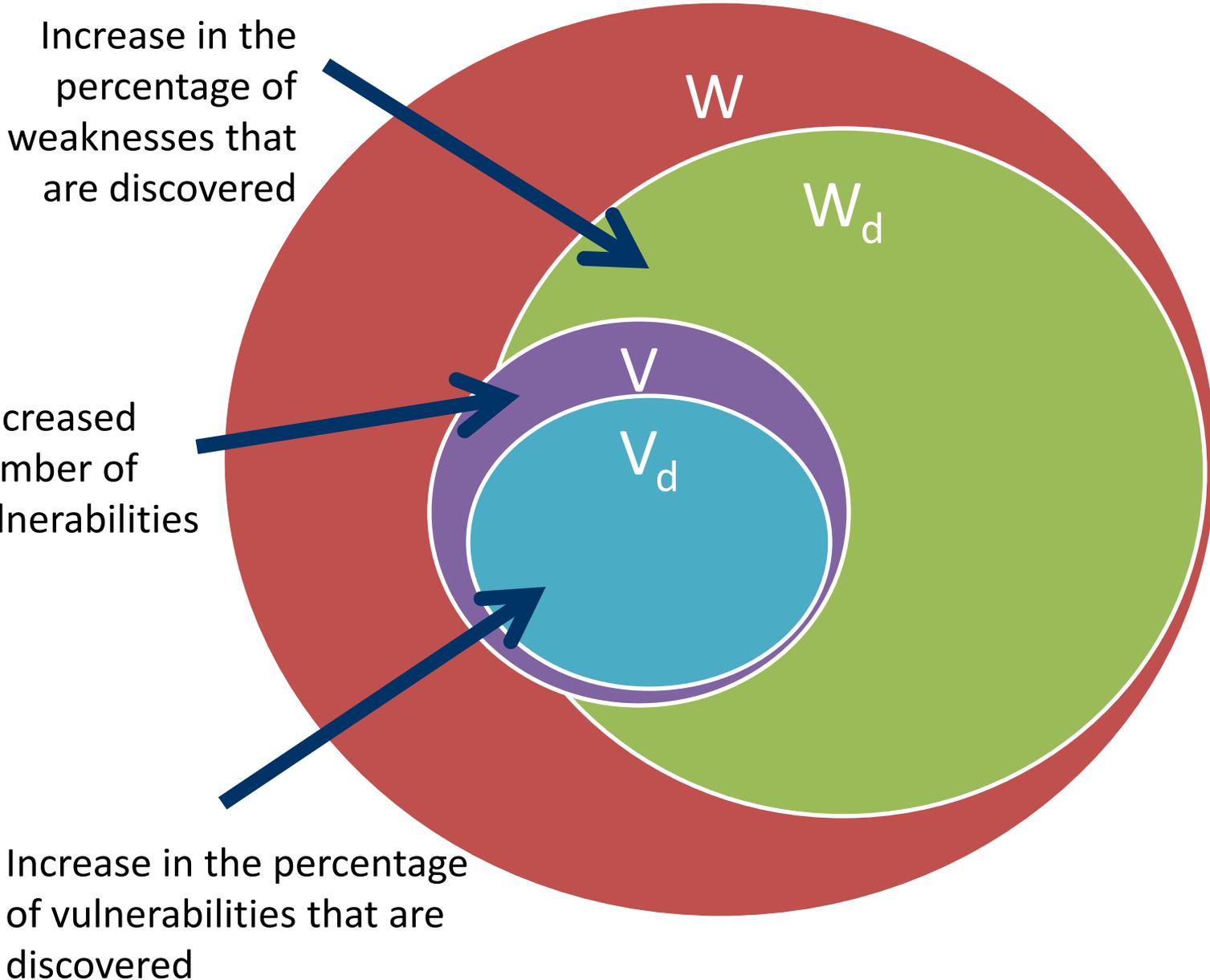


Maybe just some things get worse?

Notional

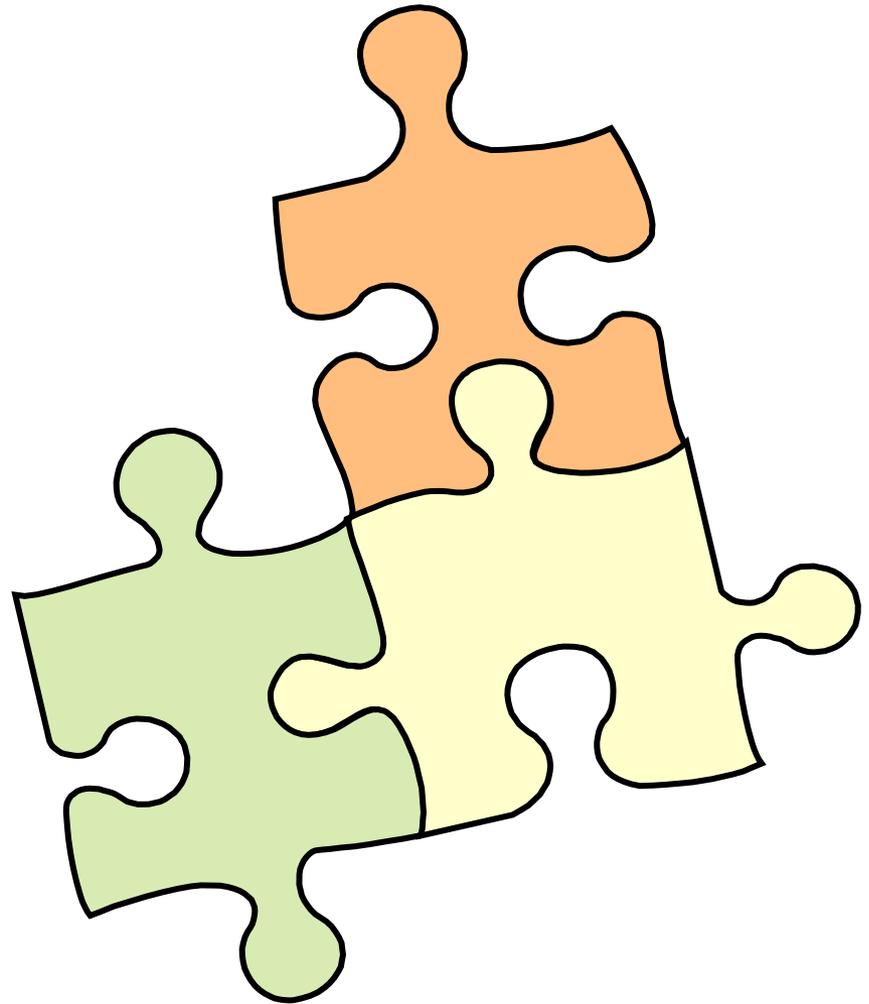
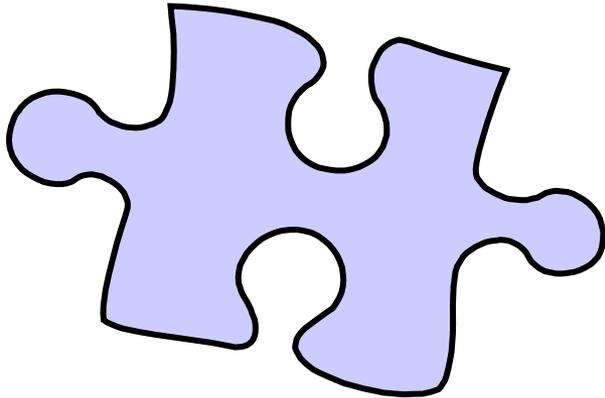


One reasonable near-term goal



Is this really the case? Yes?

Automation is *one piece*



of the SwA puzzle.

automation can help...



Construction

- Common Weakness Enumeration (**CWE**)
- Common Attack Pattern Enumeration and Classification (**CAPEC**)
- CWE Coverage Claims Representation (**CCR**)



Verification

- Common Weakness Enumeration (**CWE**)
- Common Weakness Risk Analysis Framework (**CWRAF**)
- Common Weakness Scoring System (**CWSS**)
- Common Attack Pattern Enumeration and Classification (**CAPEC**)
- CWE Coverage Claims Representation (**CCR**)



Deployment

- Security Content Automation Protocol (**SCAP**) Components, including:
 - Common Vulnerabilities and Exposures (**CVE**)
 - Common Configuration Enumeration (**CCE**)
 - Open Vulnerability Assessment Language (**OVAL**)

“Making Security Measureable”:
measurablesecurity.mitre.org

Sponsored by the USG

Resources provided for
voluntary adoption

Open, community efforts that
are *free* to use

XML-based

Some important things to note

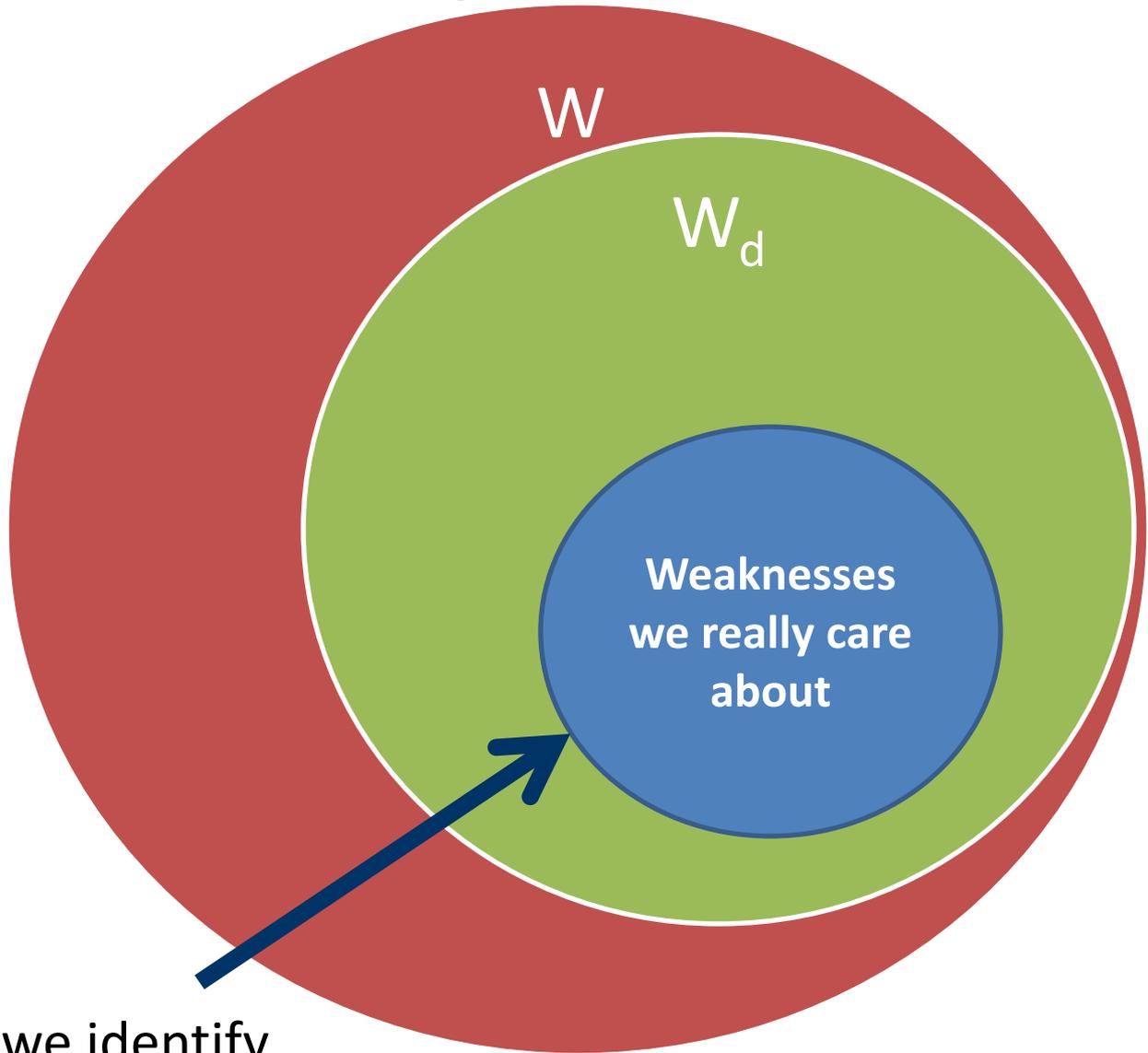
Differing levels of maturity...

Effort	Maturity
CVE	Very Mature
OVAL	Very Mature
XCCDF	Very Mature
CCE	Mature
CPE	Mature
CWE	Mature
CAPEC	Somewhat Mature
CWE CCR	Brand-new
CWSS	Brand-new
CWRAF	Brand-new

We encourage you to get involved in these communities

For the software we're responsible for

Notional

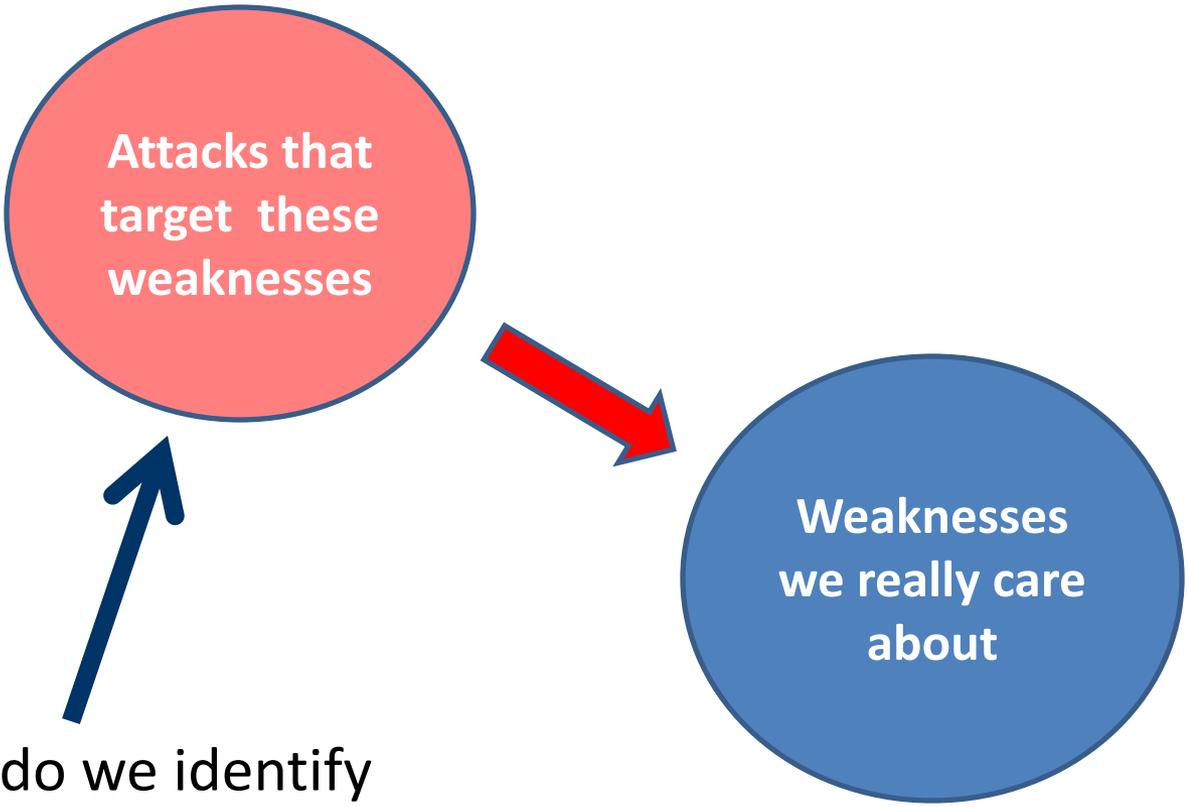


How do we identify these?

which weaknesses are most important?

For the software we're responsible for

Notional



How do we identify these?

how can those weaknesses be attacked?

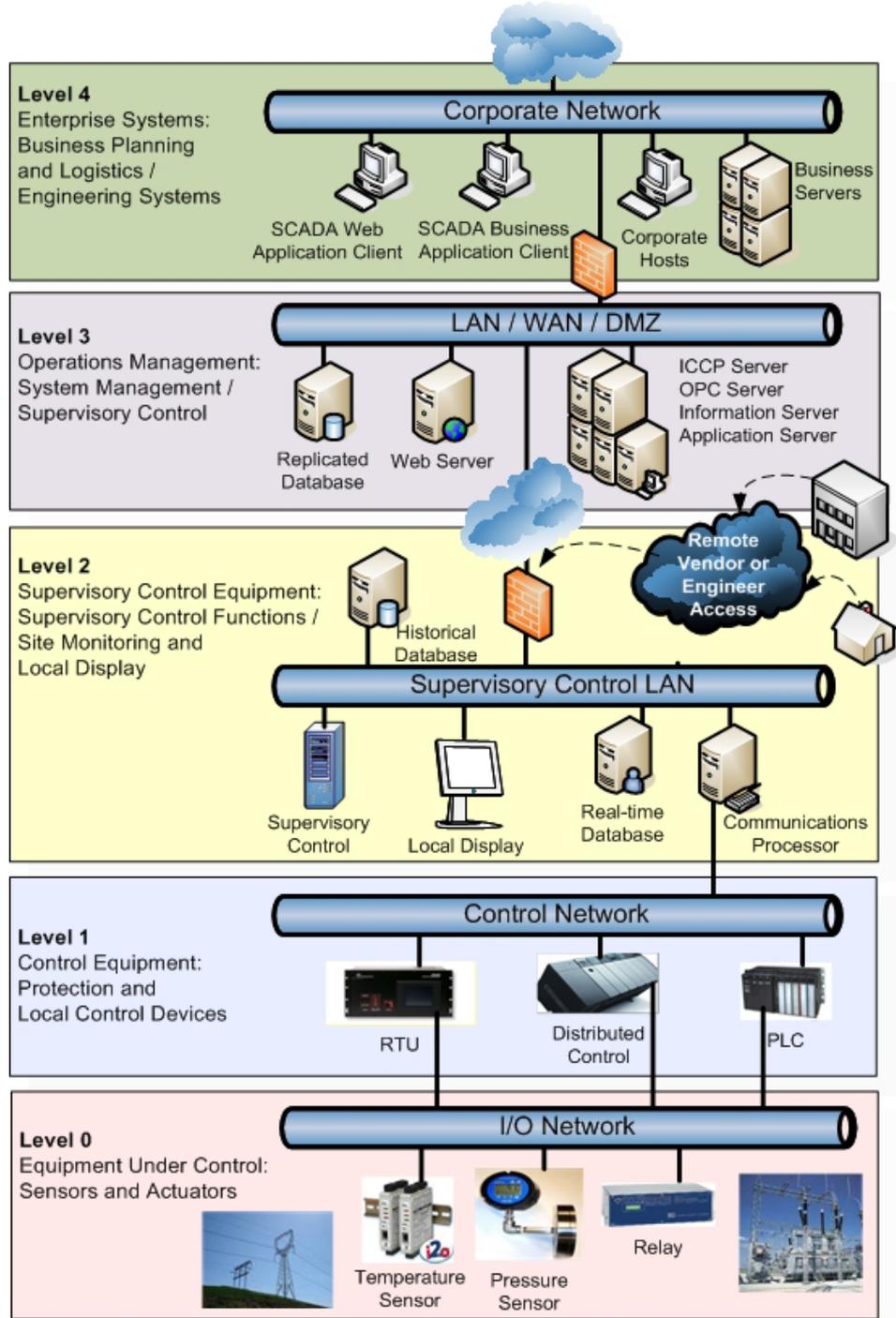
Weaknesses and attacks relevant here →

and here →

and here →

and here →

May be different than those relevant here →



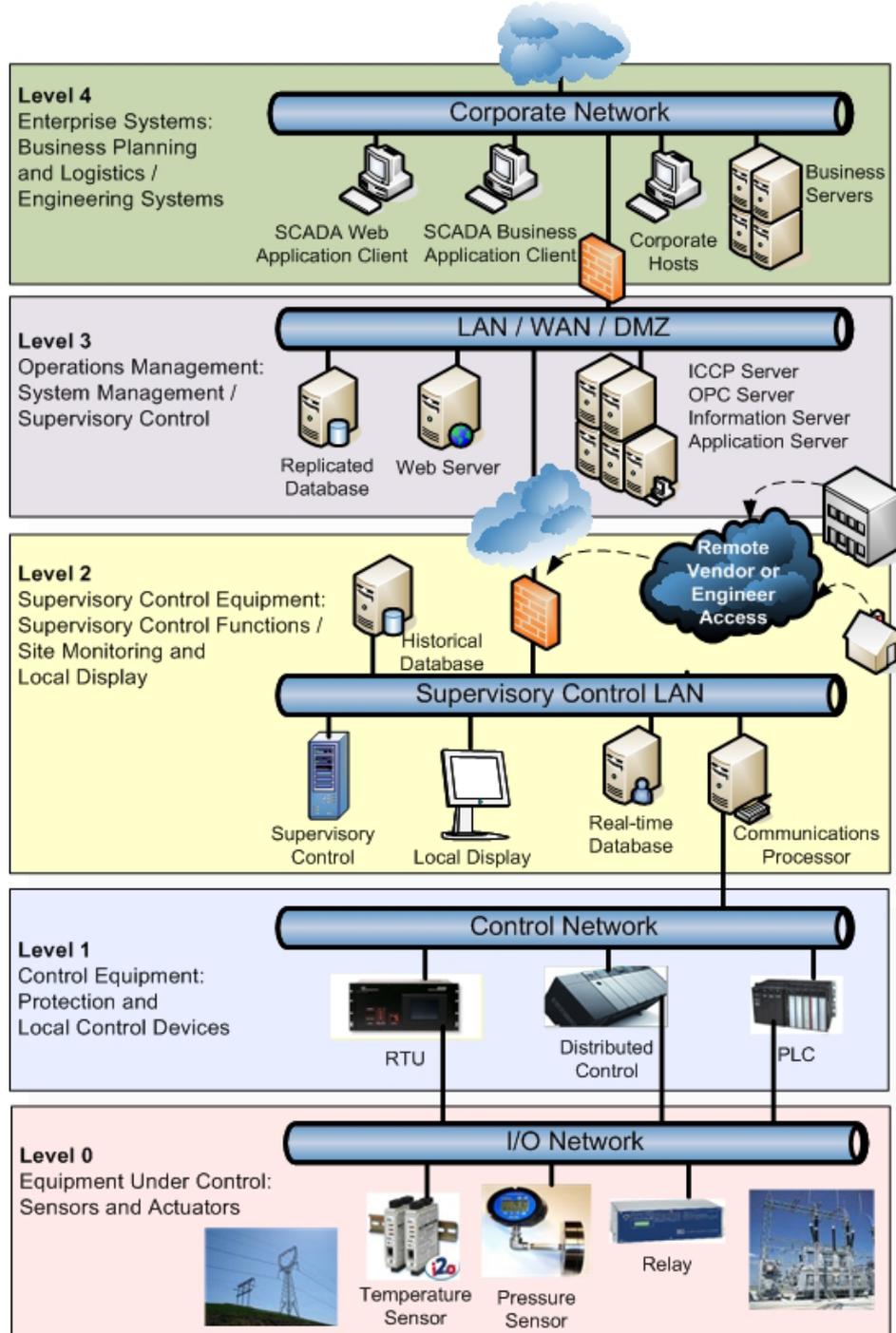
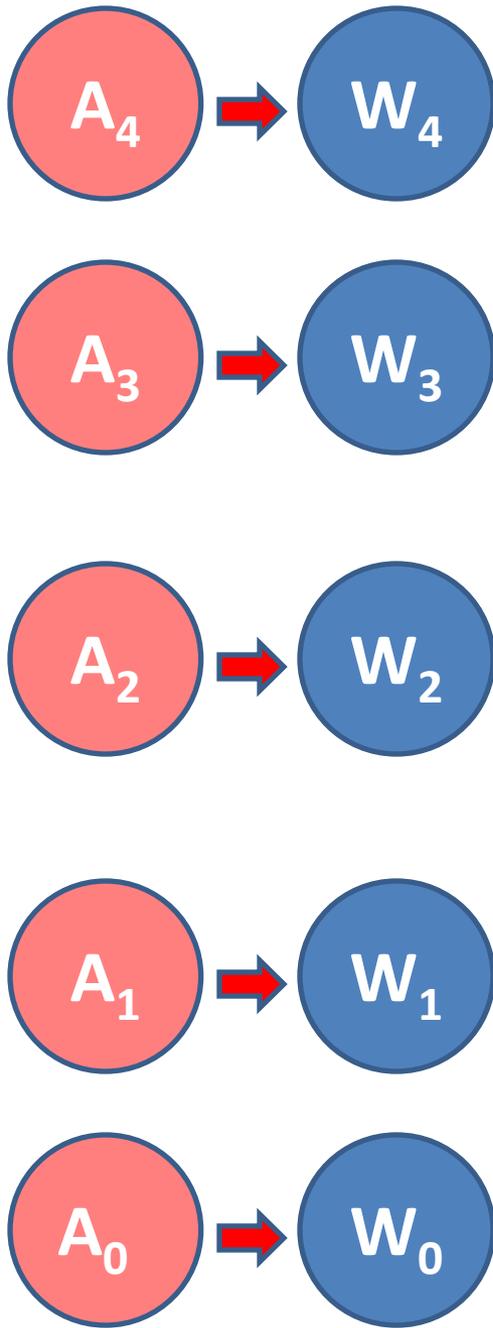
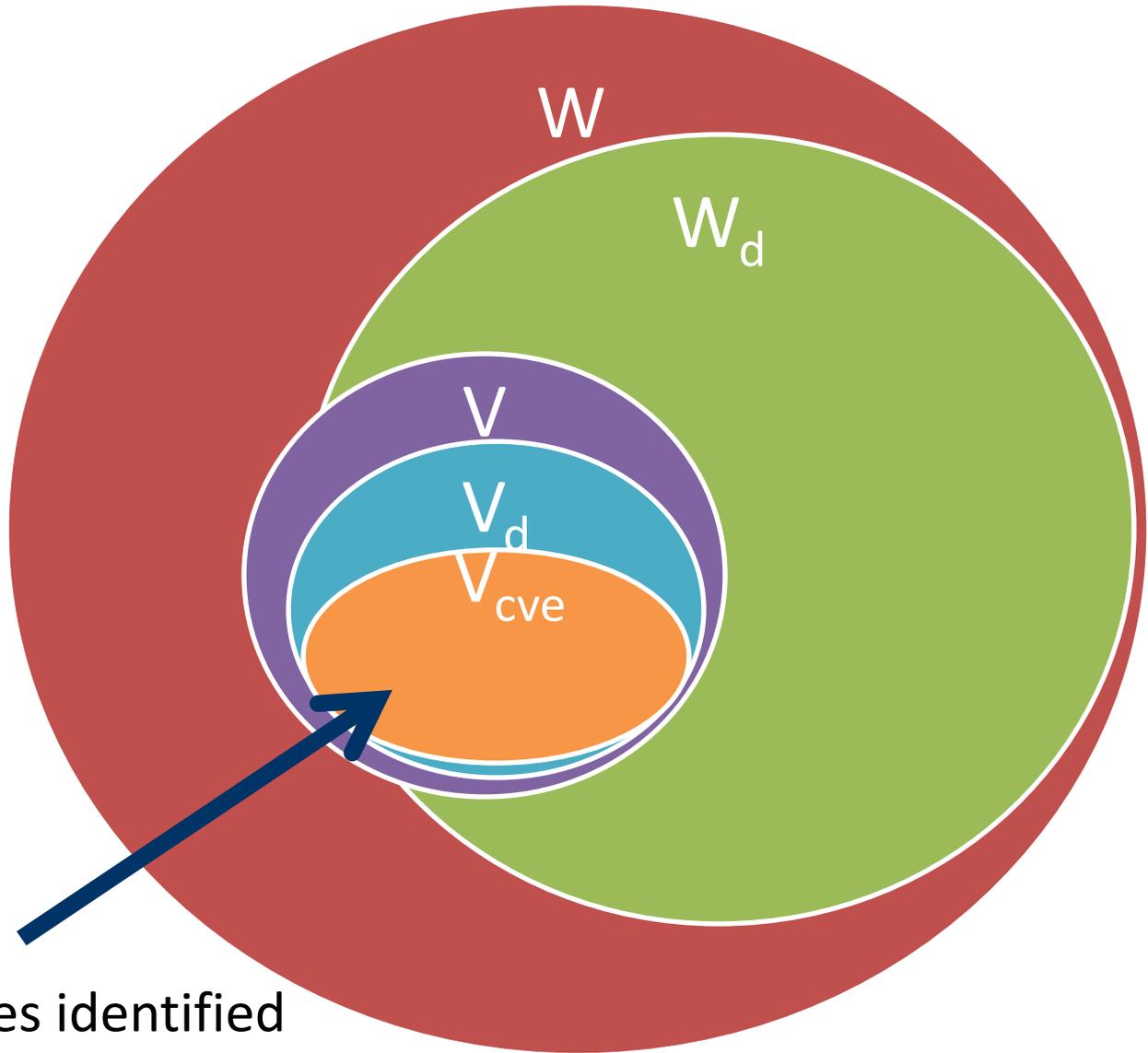


Diagram from: *Vulnerability Analysis of Energy Delivery Control Systems*, INL, September 2011

For the software we're responsible for

Notional



Vulnerabilities identified with a CVE are a good starting point

where should we start?

Dictionary of publicly-disclosed vulnerabilities with unique identifiers

- CVE ID
- Status
- Description
- References

Note: Each CVE entry is the result of expert analysis to verify, de-conflict and de-duplicate public vulnerability disclosures

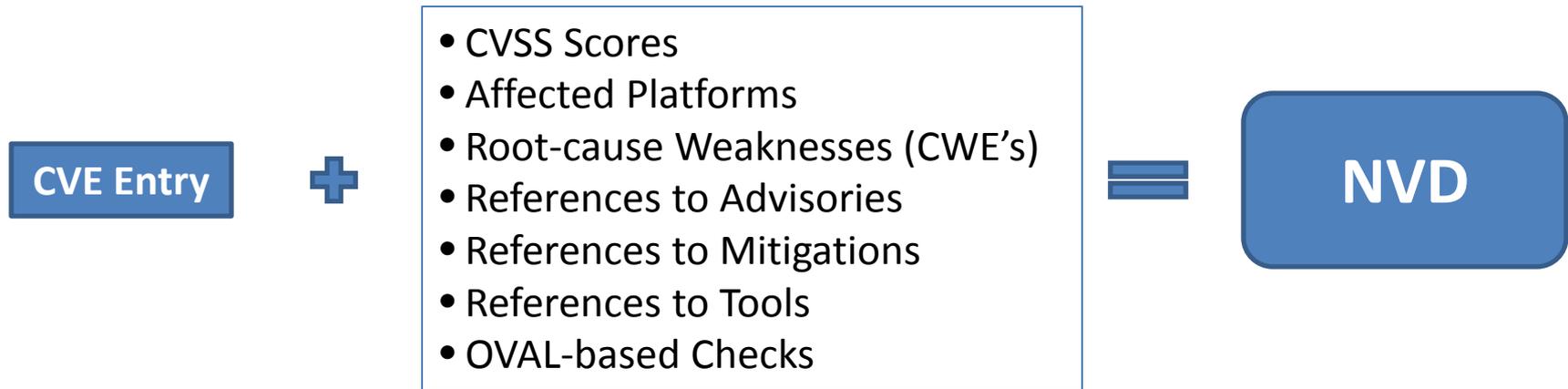
CVE entries feed into *NVD*

```
assert(CVE != Bug_Database);
```

47,258 entries (as of last week)

Common Vulnerabilities and Exposures (CVE)

National Vulnerability Database (NVD)



U.S. government repository of
standards-based vulnerability
management data

website: nvd.nist.gov

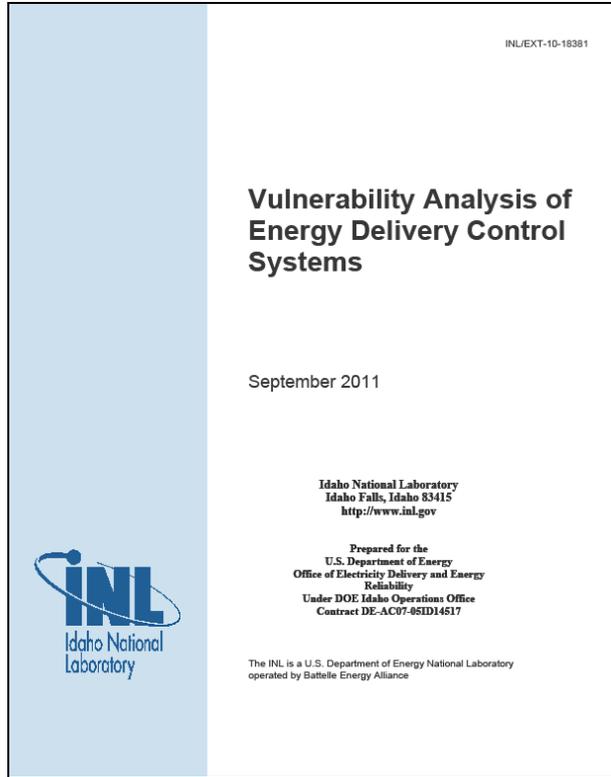
Dictionary of software weakness *types*

- CWE ID
 - Name
 - Description
 - Alternate Names
 - Applicable Platforms
 - Applicable Languages
 - **Technical Impacts**
 - Potential Mitigations
 - **Observed Instances (CVE's)**
 - **Related Attack Patterns (CAPEC's)**
 - Examples
- Plus much, much more*

860+ entries in a tree-structure

Common Weakness Enumeration (CWE)

Prioritizing weaknesses to be mitigated



Lists are a good start but they
are designed to be broadly
applicable



OWASP
The Open Web Application Security Project
<http://www.owasp.org>

OWASP Top 10



CWE/SANS Top 25

**We would like a way to specify priorities
based on business/mission risk**

Common Weakness Risk Analysis Framework (CWRAF)

*How do I **identify** which of the 800+ CWE's are most important for my specific business domain, technologies and environment?*

Common Weakness Scoring System (CWSS)

*How do I **rank** the CWE's I care about according to my specific business domain, technologies and environment?*

How do I identify and score weaknesses important to my organization?

Common Weakness Risk Analysis Framework (CWRAF)

Technical Impacts

1. Modify data
2. Read data
3. DoS: unreliable execution
4. DoS: resource consumption
5. Execute unauthorized code or commands
6. Gain privileges / assume identity
7. Bypass protection mechanism
8. Hide activities

Weightings

- W1=0
W2=0
W3=10
W4=4
W5=10
W6=0
W7=0
W8=0



Layers

1. System
2. Application
3. Network
4. Enterprise



Technical Impact
Scorecard

Multiple pieces – we'll focus on "Vignettes"

CWRAF: Technical Impact Scorecard

and each technical impact

	MD	RD	UE	RC	EA	GP	BP	HA
Application								
System		8						
Network								
Enterprise								3

For each layer

assign a weighting from 0 to 10

CWRAF: Technical Impact Scorecard

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	2
System	8	8	4	2	10	9	5	1
Network	9	5	6	2	10	5	7	1
Enterprise	4	7	6	2	10	6	4	3

These weightings can now be used to evaluate individual CWE's based on each CWE's Technical Impacts

Note: Values for illustrative purposes only

Note: Values for illustrative purposes only

Notional

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	2
System	8	8	4	2	10	9	5	1
Network	9	5	6	2	10	5	7	1
Enterprise	4	7	6	2	10	6	4	3

CWE-78
Technical
Impacts

CWSS
Formula

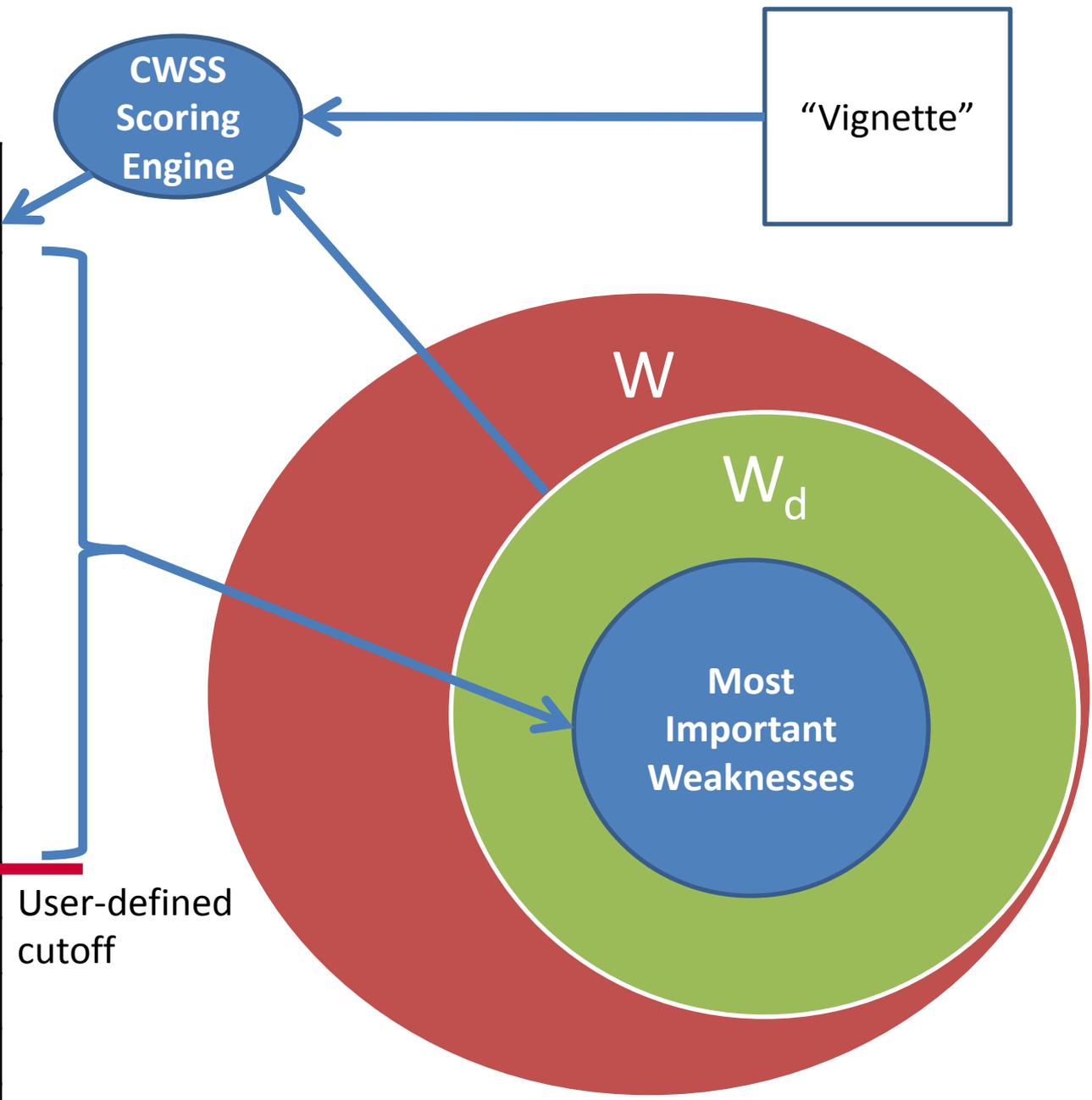
=

95

CWSS Score for CWE-78
for this vignette

Common Weakness Scoring System (CWSS)

CWSS Score	CWE
97	CWE-79
95	CWE-78
94	CWE-22
94	CWE-434
94	CWE-798
93	CWE-120
93	CWE-250
92	CWE-770
91	CWE-829
91	CWE-190
91	CWE-494
90	CWE-134
90	CWE-772
90	CWE-476
90	CWE-131
...	



CWRAF/CWSS in a Nutshell

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	6
System	8	3	4	2	10	9	5	1
Network	9	5	3	2	7	5	7	1
Enterprise	4	7	6	2	10	6	4	3

	MD	RD	UE	RC	EA	GP	BP	HA
Application	4	7	2	4	6	8	8	2
System	6	8	4	2	5	6	5	1
Network	5	5	6	3	5	5	7	4
Enterprise	4	7	6	2	4	6	4	3

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	8	6	10	1
System	9	8	4	2	9	6	5	2
Network	9	5	6	2	9	5	7	4
Enterprise	7	7	6	2	7	6	4	4

	MD	RD	UE	RC	EA	GP	BP	HA
Application	7	6	4	2	4	8	7	2
System	7	8	4	2	5	9	5	1
Network	7	5	6	2	5	5	7	1
Enterprise	3	5	6	2	5	6	4	3

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	2
System	8	8	4	2	10	9	5	1
Network	9	5	6	2	10	5	7	1
Enterprise	4	7	6	2	10	6	4	3

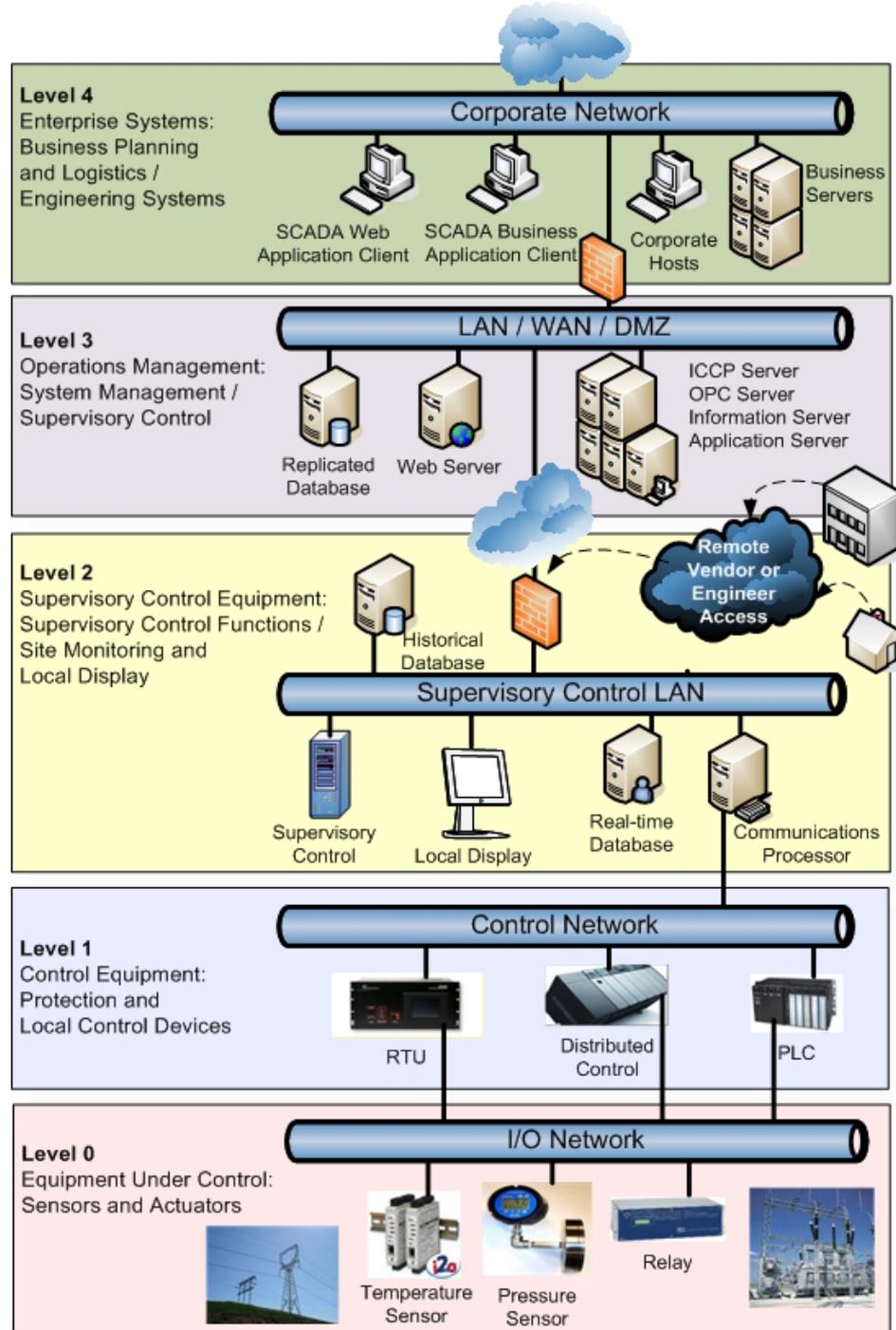


Diagram from: *Vulnerability Analysis of Energy Delivery Control Systems*, INL, September 2011

Common Attack Pattern Enumeration and Classification (CAPEC)

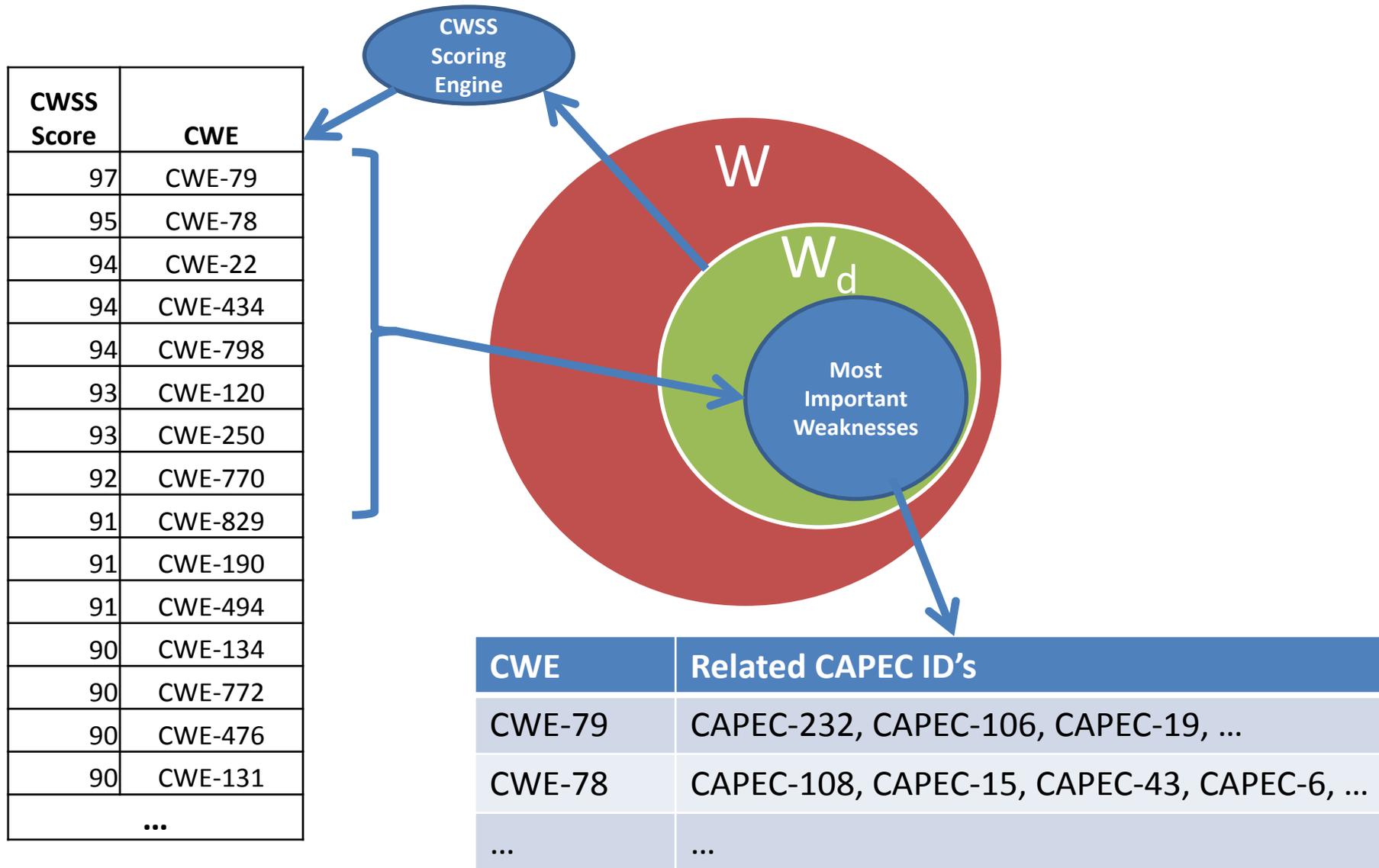
Dictionary of attack types (mostly software)

- CAPEC ID
- Name
- Description
- Attack Prerequisites
- Indicators of Attack
- Examples
- **Related Weaknesses (CWE's)**
- Mitigations

Plus much, much more

386 patterns, organized
by categories, with views

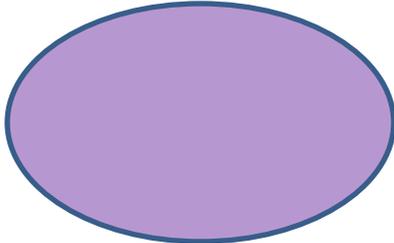
What types of attacks should I test my system against?



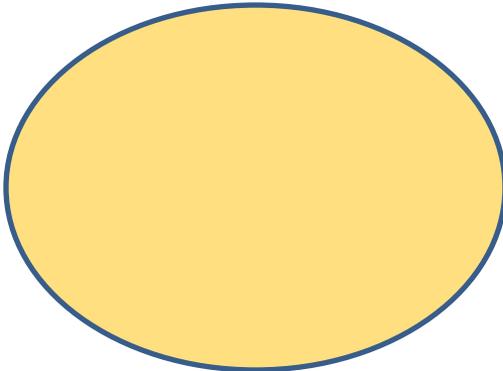
CWE Coverage Claims Representation

Set of CWE's tool *claims* to cover

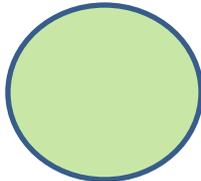
Tool A



Tool B



Tool C



Which static analysis tools find the CWE's I care about?

Which published vulnerabilities apply to *my* systems?

- CVE/
NVD
- OVAL
- CPE

- CVE/
NVD
- CVSS

Which published vulnerabilities should I fix *first*?

Are my systems configured properly?

- OVAL
- XCCDF
- CCE
- NVD

For asset owner/operators...

For technology developers/integrators...

What types of attacks
should I design my
system to withstand?

CAPEC

CWE

CWRAF

CWSS

What architecture, design
and coding weaknesses are
most critical to avoid?

For technology developers/integrators...

What analysis tools might help me find the weaknesses I care about?

CCR

CWE

CAPEC

What types of attacks do I need to test my code against?

CWE

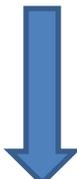
CWRAF

CWSS

What weaknesses are highest priority to eliminate from my code?

For asset owners *and* technology vendors...

What weaknesses are most important?

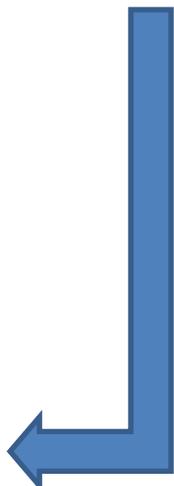


Does the delivered system contain any of those weaknesses?

Does my testing cover all of those weaknesses?



What types of attacks exploit those weaknesses?



SwA Working Groups – Next meeting: Week of Nov 28 @ MITRE in McLean, VA

All SwA Program events are free and open to the public

SwA Forum – Next Forum: Week of March 26, 2012 @ MITRE in McLean, VA

SwA Websites: www.us-cert.gov/swa

Making Security Measureable:
measurablesecurity.mitre.org

Email: software.assurance@dhs.gov

Software Assurance Resources

Copyrighted Material
New Studies in Archaeology

The Collapse of Complex Societies

JOSEPH A. TANTER



Copyrighted Material

A Closing Thought

“investment in sociopolitical complexity as a problem-solving response often reaches a point of declining marginal returns”

Tainter, *The Collapse of Complex Societies*, p. 194

While we must continue to incrementally secure existing systems...

We should also be pursuing ***fundamental technological improvements*** that will reduce overall system complexity

Questions?

Richard Struse

Deputy Director

Software Assurance

National Cyber Security Div.

U.S. Department of Homeland Security

National Protection & Programs Directorate

richard.struse@dhs.gov



**Homeland
Security**

thank you.