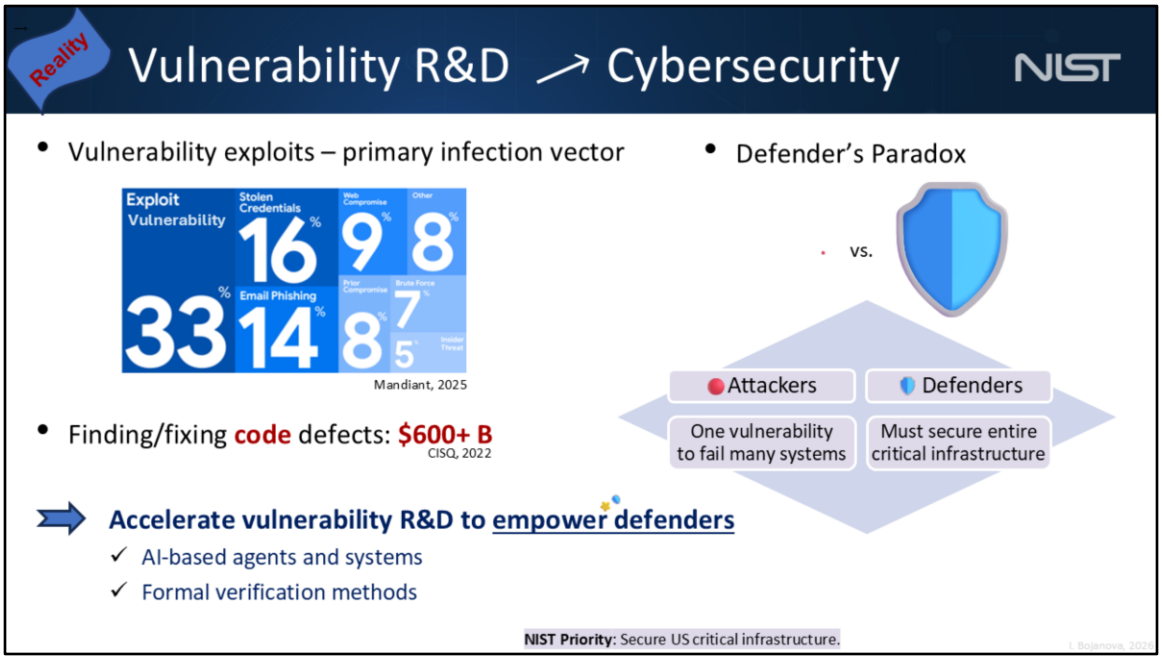
NVD–BF is envisioned as a formal vulnerability classifications platform that would advance hardware and software vulnerability analytics and accelerate AI and FM-based cybersecurity R&D.



The cybersecurity reality is:

- Hardware and software vulnerability exploits account for 33% of the initial vectors in cyberattacks.
- Finding and fixing code bugs/faults costs more than 600 billion dollars per year.
- Attackers need single exploitable vulnerability to compromise many systems, but defenders must secure the entire critical infrastructure.

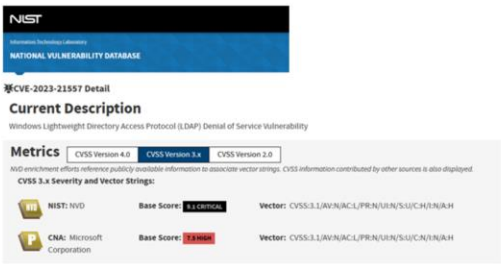
⟶ To empower defenders, we must accelerate cybersecurity vulnerability R&D.

Goal

Advance Vulnerability Analytics

NIST

- Vulnerability repositories
  - NVD, KEV
  - CVE (& CWE)
- Problem – limited analytics
  - Narrative descriptions
  - Static, subjective data
  - Lack of labeled datasets



Multi-dimensional vulnerability formalism?

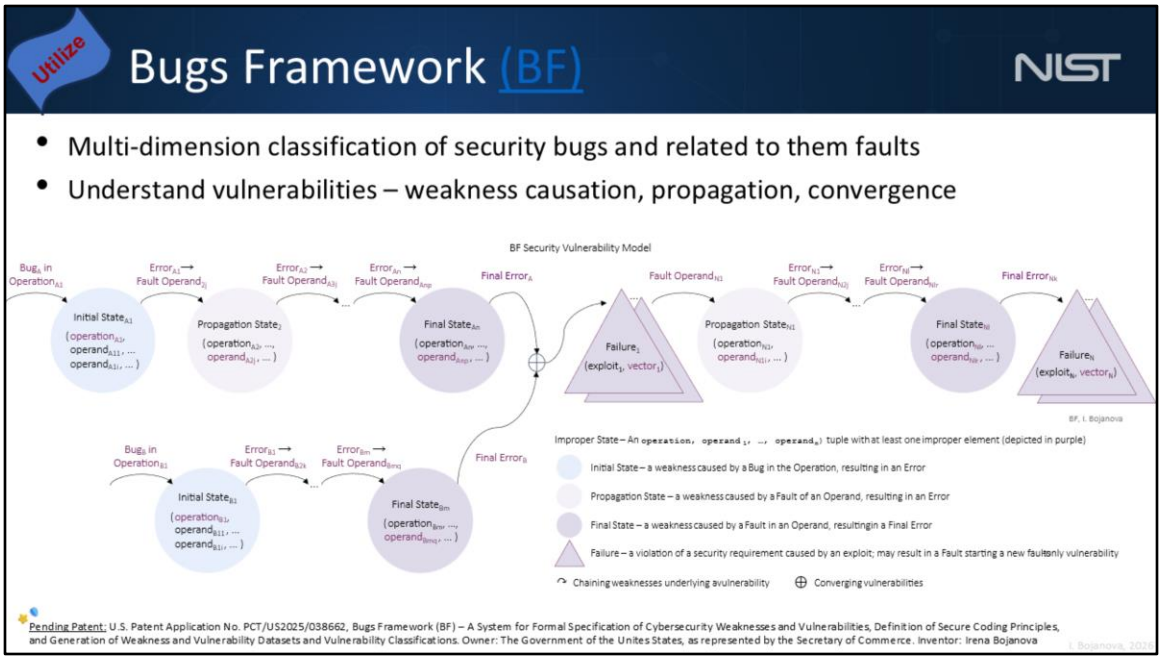
- ✓ Formal specification
- ✓ Secure coding principles
- ✓ Automatic generation
- ✓ Dynamic vulnerability classifications
- ✓ Contextual connections
- ✓ Analytics and scoring

NIST Priority: AI-based agents to secure US critical infrastructure.

I. Bojanova, 2025

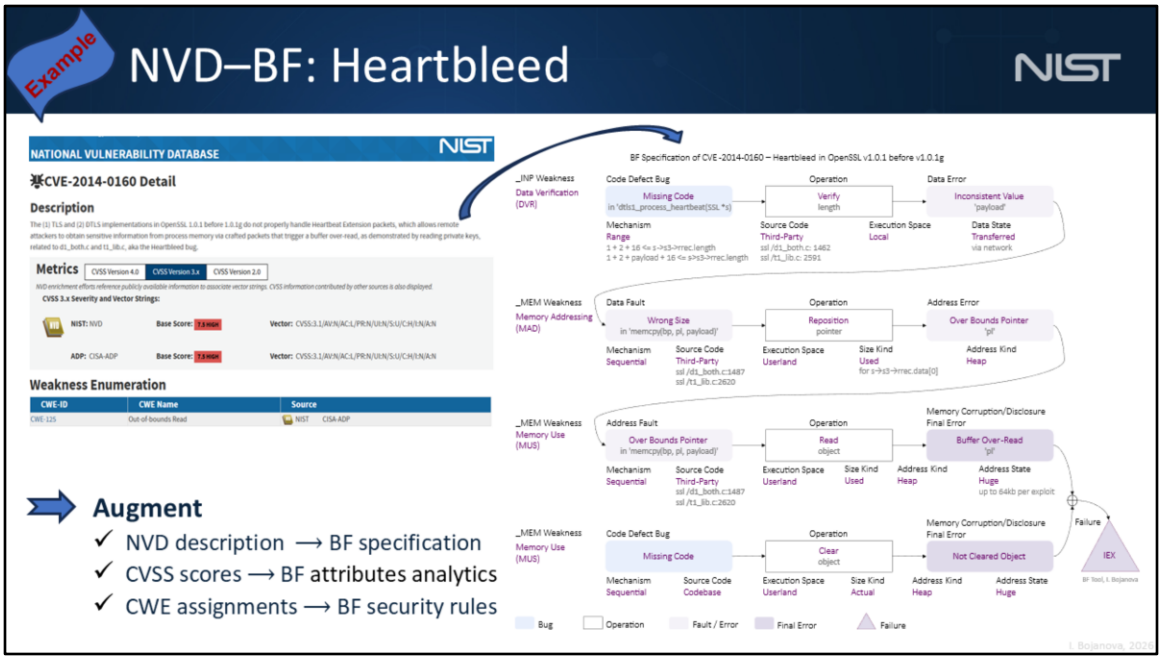
- Current vulnerability repositories are based on NL descriptions and static subjective data.
- Cybersecurity experts and automated systems struggle due to the lack of comprehensively labeled vulnerability datasets for AI and FM R&D.

⟶ To advance vulnerability analytics, we need dynamic, formal, multi-dimensional classifications.
⟶ Descriptions and classifications should be automatically generated utilizing a powerful vulnerability formalism.



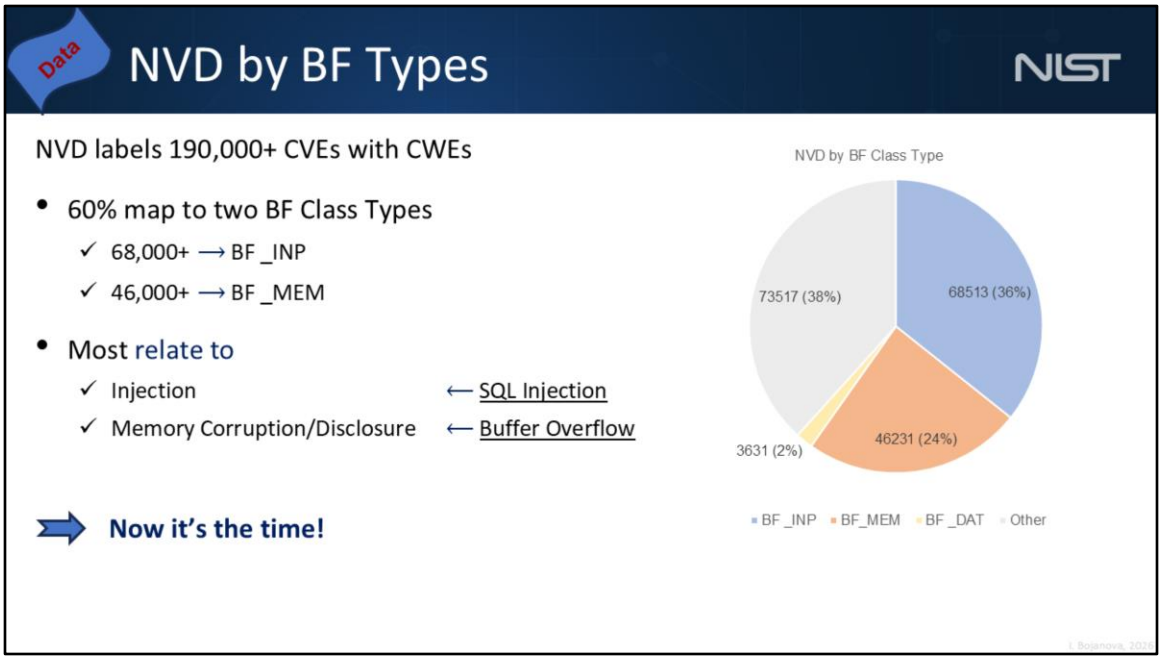
The Bugs Framework (BF) is a structured multi-dimensional classification of security bugs and related to them faults, supporting deeper understanding of vulnerabilities as chains of weaknesses, adhering to strict causation, propagation, and convergence rules.

BF's formal language is based on rigorous BF bugs models, causal taxonomies, and a vulnerability specification model.



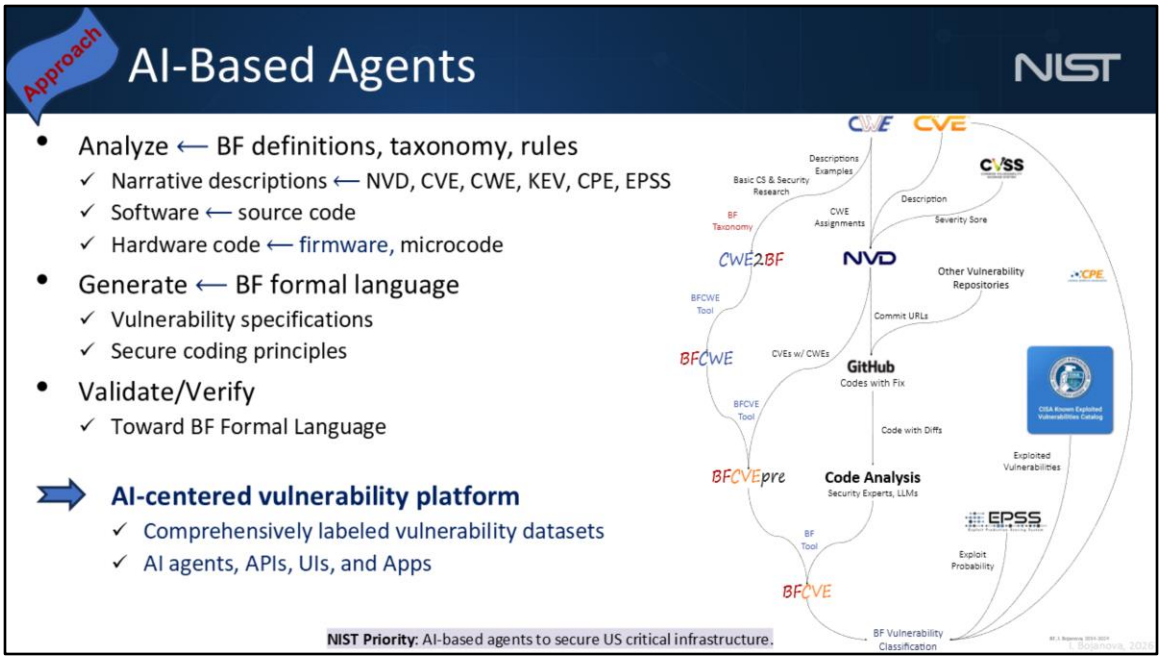
For example, the Heartbleed description in NVD will be augmented with its BF specification of two converging chains of weaknesses, as well as, provided in machine-readable formats.

The CVSS severity scores – via BF attributes analytics and the CWE weakness assignments – via specific BF security rules.



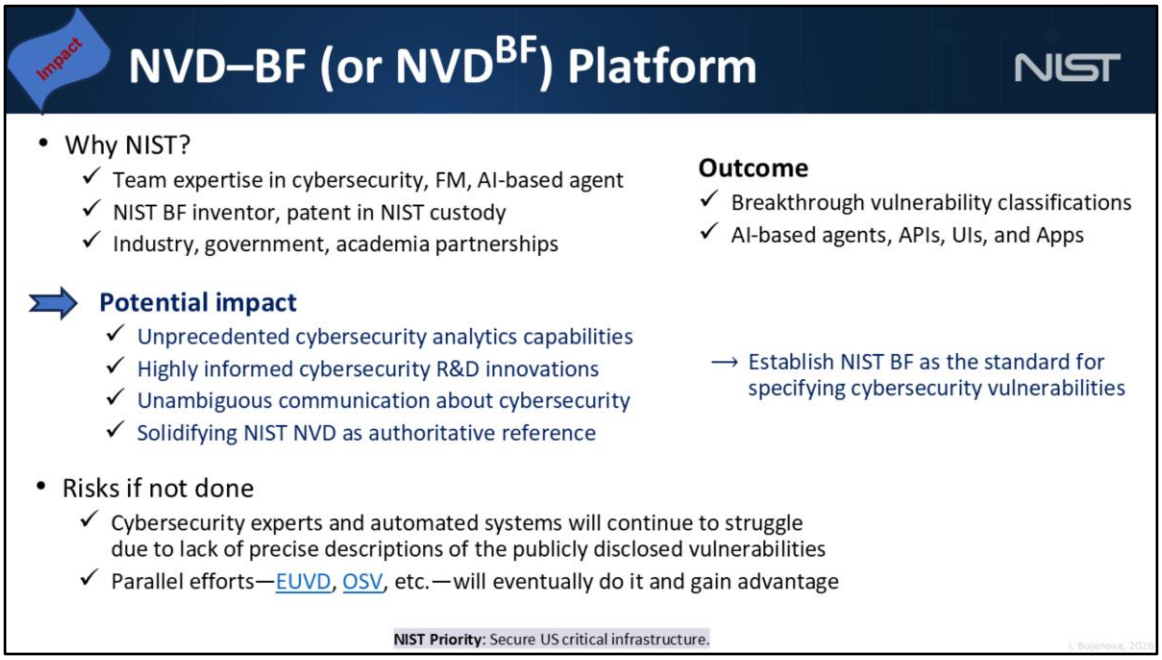
- NVD labels close to 200,000 CVEs with CWEs, which allows mapping them to BF classes.
- 60% of these CVEs map to BF Input Check or Memory Bugs classes.
- Many lead to the most dangerous exploitable errors: Injection (e.g., SQL injection) and Memory Corruption (e.g., buffer overflow).

⟶The thousands of known vulnerabilities form the basis for a first comprehensively labeled vulnerability dataset and multi-dimensional classifications.



We will create BF-based AI autonomous agents:

- agents to autonomously process NL descriptions from NVD and other vulnerability repositories
- agents to analyze firmware and software code with bugs and fixes
- agents to generate formal vulnerability specifications and secure coding principles
- agents to validate and verify generated specifications toward the BF formal language.



Why NIST?

- We will form a team with the exact expertise in cybersecurity, FM, AI models, and AI-based agents.
- I am the inventor of the BF, as the theoretical basis for this project. The patent is owned by the Government of the Unites States, as represented by the Secretary of Commerce; the patent is in the custody of NIST.
- Strong industry, government, academia partnerships

The resulting outcome will be a breakthrough platform with formal multi-dimensional vulnerability classifications and AI-based agents, APIs, UIs, and Apps for cybersecurity R&D.

The potential impact would be

- Unprecedented multi-dimensional cybersecurity analytics capabilities
- Highly informed innovations in bug/fault detection, vulnerability resolution or mitigation, and development of effective countermeasures against potential security threats and specific exploits
- Unambiguous communication about cybersecurity across industry, government, and academia

- Solidifying NIST NVD as authoritative reference

⟶ Finally, the NVD-BF would establish NIST BF as the standard for specifying cybersecurity vulnerabilities

It this is not done, cybersecurity experts and automated systems will continue to struggle due to lack of precise descriptions of the publicly disclosed vulnerabilities.

Parallel efforts—European Union Vulnerability Database (EUVD), Open Source Vulnerability (OSV) Database, etc.—will eventually go in this direction and gain advantage.

Potential Collaborators

NIST

- **NIST Researchers**

Irena Bojanova	—cybersecurity, formal methods—	ITL/775.01
Tim Blattner	—AI models, AI-based agents—	ITL/775.02
Jason Collard	—AI, NLP, formal methods—	ITL/775.04
Jeff Voas	—cybersecurity, digital twin—	ITL/773.03
Rick Kuhn	—combinatorial testing—	ITL/773.02

- **Non-NIST PhD Candidates**

Purdue University
Colorado State University
Idaho National Laboratory
JHU, Applied Physics Laboratory
University of West Attica
University of Quebec

References

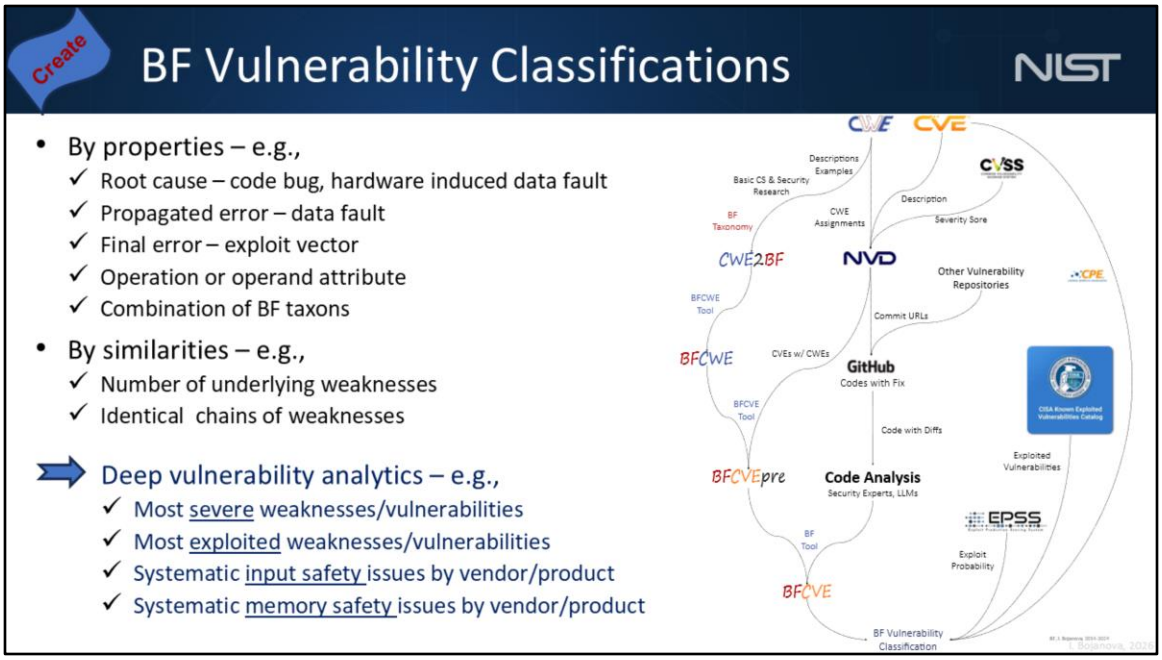
[1] Bojanova I (2024) Bugs Framework (BF): Formalizing Cybersecurity Weaknesses and Vulnerabilities. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP), NIST SP 800-231. https://doi.org/10.6028/NIST.SP.800-231

PATENT PENDING

[2] U.S. Patent Application No. PCT/US2025/038662, Bugs Framework (BF) – A System for Formal Specification of Cybersecurity Weaknesses and Vulnerabilities, Definition of Secure Coding Principles, and Generation of Weakness and Vulnerability Datasets and Vulnerability Classifications. Inventor: Irena Bojanova, NIST.

National Institute of Standards and Technology
U.S. Department of Commerce

Extras



Dataset of diverse multi-dimensional vulnerability classifications based on common properties and similarities. Security vulnerabilities could be classified by common root causes (i.e., software or firmware bugs or hardware defect-induced bugs or faults), such as declaring a variable of a wrong data type.

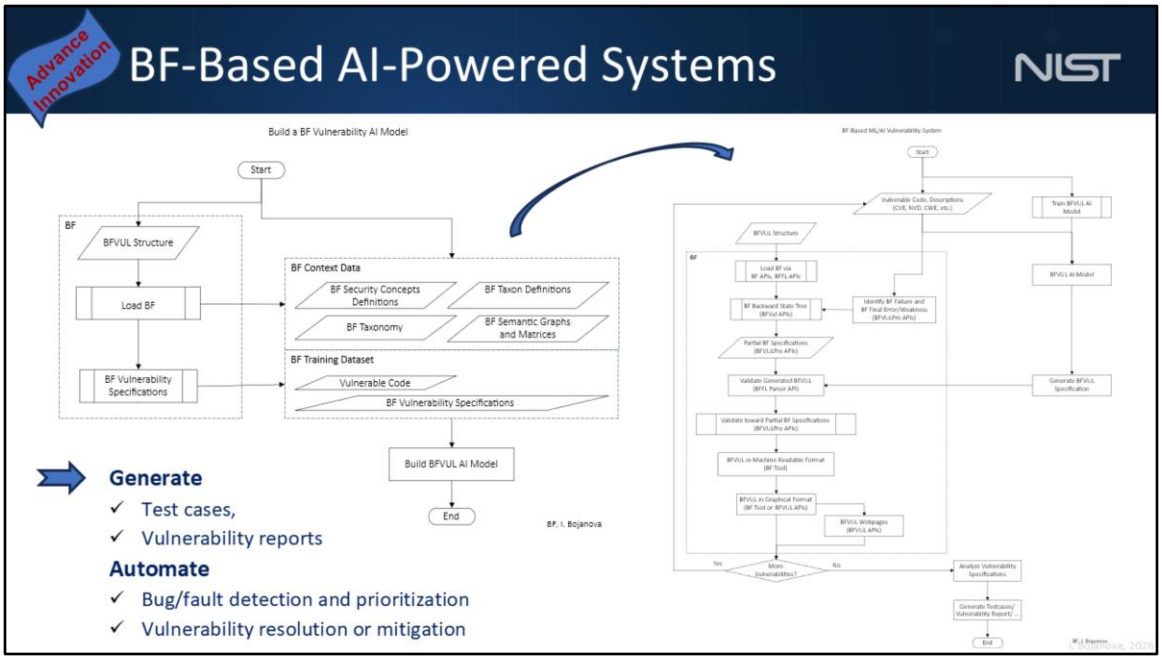
They could also be classified by any other BF taxons, such as propagating faults, common final errors, operation and operand attributes, identical BF specifications (i.e., chains of weaknesses), and even the number of underlying weaknesses.
The BF operation and operand attributes provide insight into the severity of the weaknesses and how they relate to commonly used scores, such as CVSS and EPSS.

Their analysis would allow for deeper research on the most significant and most exploited weaknesses and vulnerabilities.

Intriguing classifications by BF classes and CPE data may reveal systematic
input/output check safety, memory safety, data type safety, and other secure coding problems by particular vendors and products.

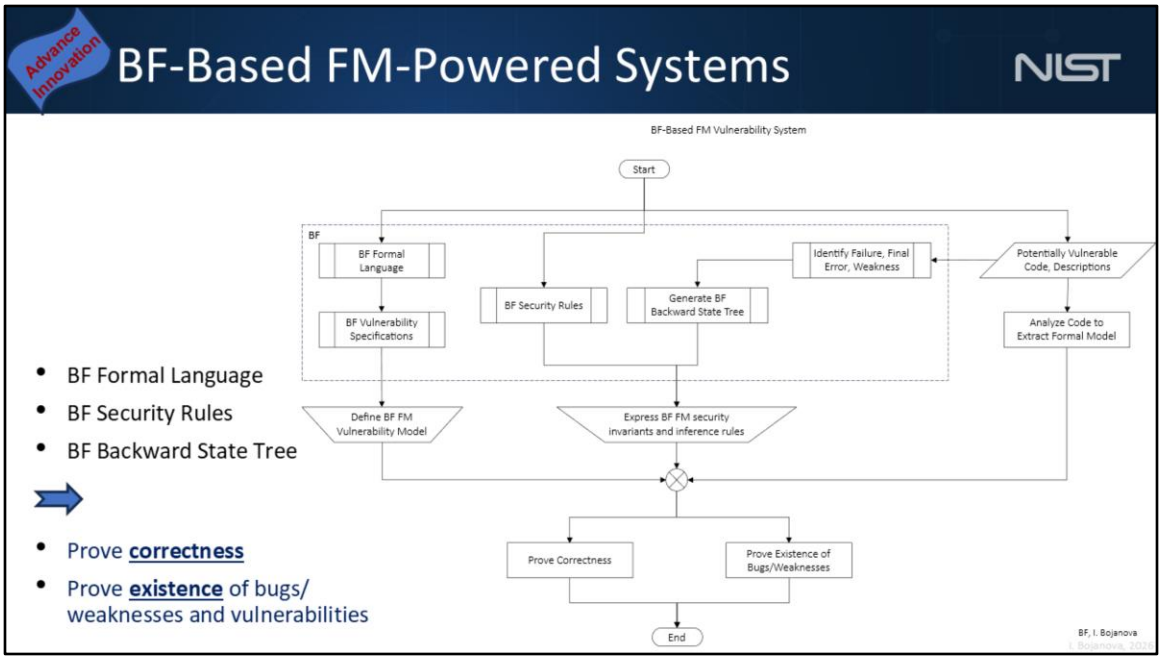
The multi-dimensional BF vulnerability classifications will contribute to deeper analysis and refined understanding of security weaknesses, vulnerabilities, exploits, and failures.

They will enable more focused cybersecurity research and highly informed development of effective countermeasures against potential security threats and specific exploits.



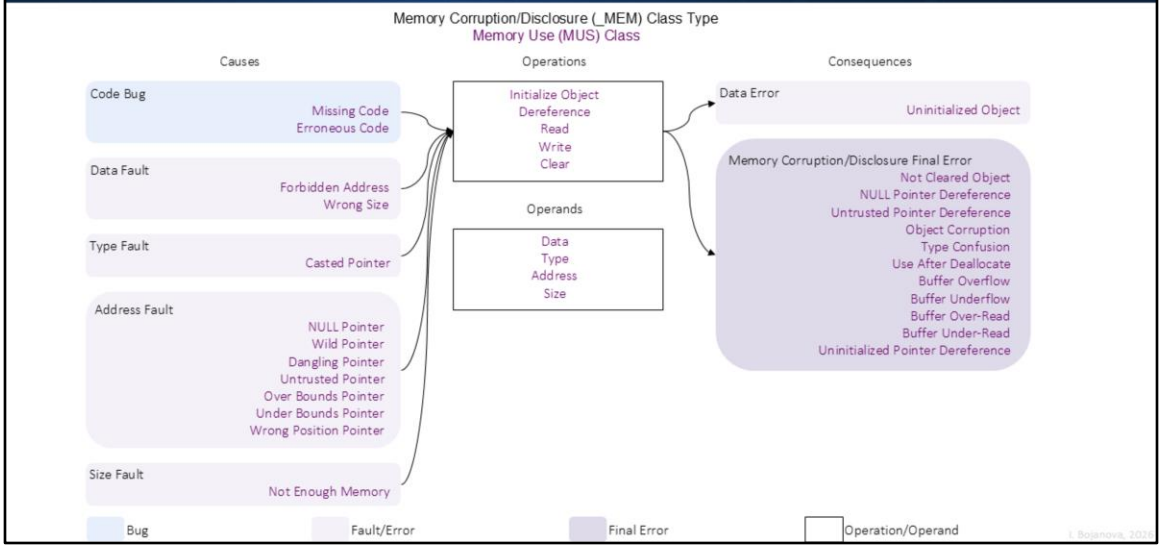
Broader applications support would be for:

- Automation of bug/fault detection and vulnerability resolution or mitigation
- Generation of testcases and vulnerability reports



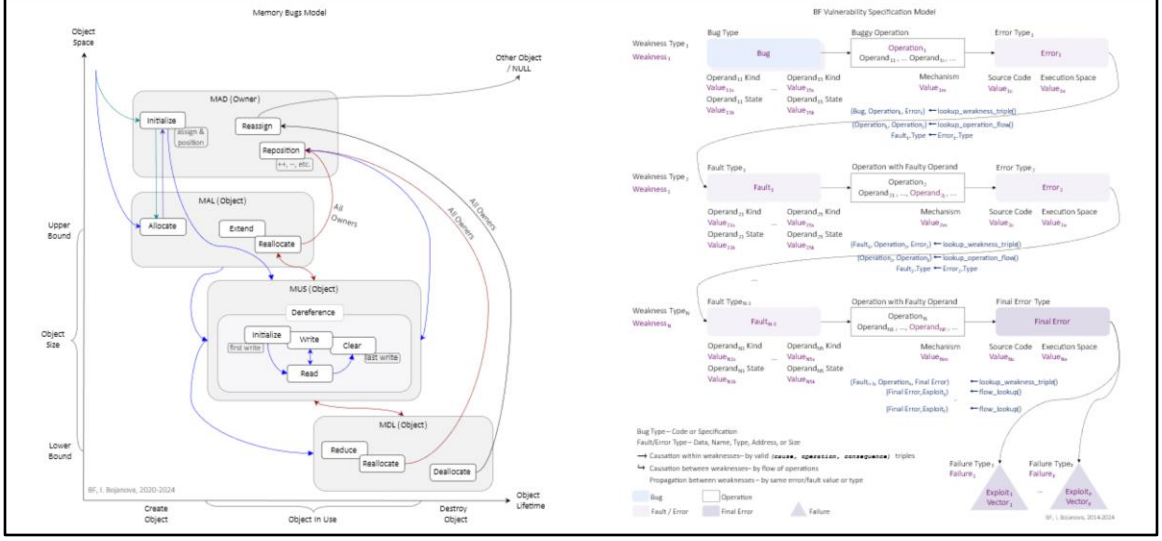
Given the formal specification of code and the BF security concepts definitions, FM could be applied to prove correctness or prove the existence of bugs/faults and related vulnerabilities.

BF Causal Taxonomies

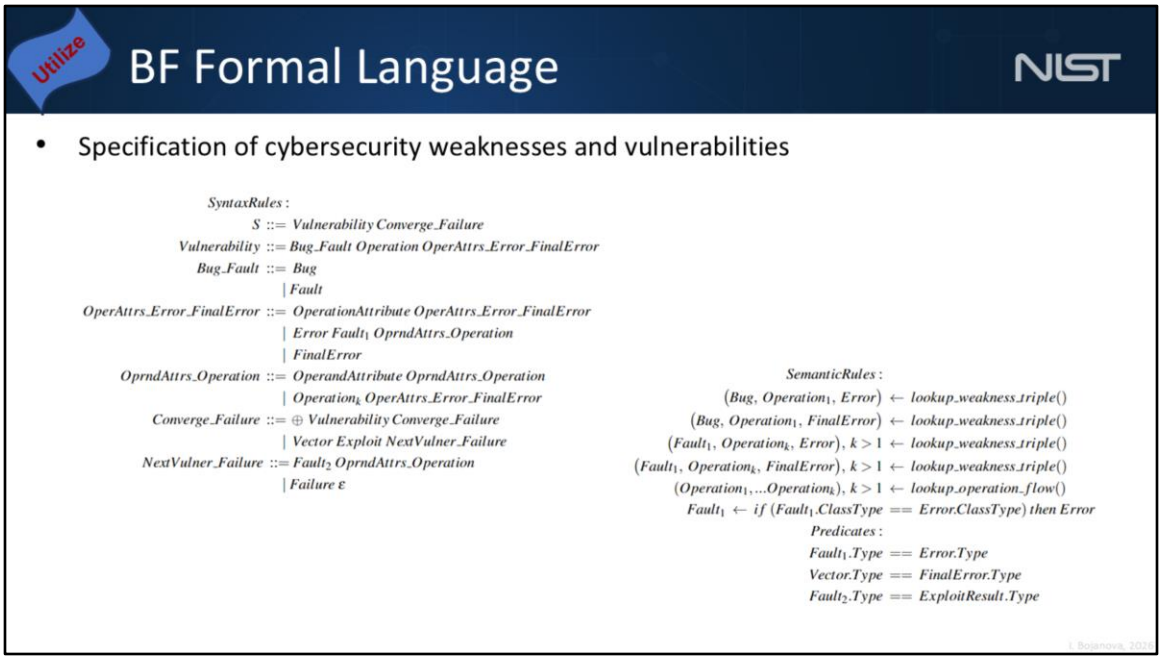


BF organizes bugs by the operations of distinct execution phases, faults by their input operands, and errors by their output results. Bugs and faults are the possible causes for security weaknesses; errors and final errors -- their possible consequences.

BF Bugs and Specification Models



BF's weakness taxonomies and bugs and specification models form the basis for defining secure coding principles, such as memory safety, input/output safety, and data type safety.



BF's formal language is based on rigorous BF bugs models, causal taxonomies, and vulnerability specification models.