NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

ITL Science Day
03/26/2026

NVD–BF (or Alf. NVD^{BF}) Formal Vulnerability Classifications Platform

Irena Bojanova, NIST

Bugs Framework (BF)
https://usnistgov.github.io/BF

Motivation

Mandiant, 2025

- Finding/fixing **code** defects: **$600+ B**;
- Need for automated bug identification, vulnerability detection, and analysis

Objective

✓ AI-based agents and systems
✓ Formal verification methods

⇨ Accelerate AI and Formal Methods (FM) Cybersecurity R&D to **secure** critical infrastructure

Methodology

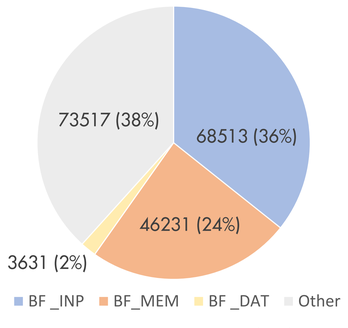
✓ Formal language; Vulnerability models
✓ Secure coding principles

- Utilize Bugs Framework (BF) – multi-dimensional vulnerability formalism
- Develop BF-based AI agents and AI-powered systems for generation of:
  - ✓ Formal vulnerability specifications
  - ✓ Dynamic **formal** multi-dimensional vulnerability classifications
- Develop BF-based FM-powered systems to prove code correctness or existence of hardware or software bugs/weaknesses and vulnerabilities

NVD by BF Type

NVD labels **200,000+** CVEs with CWEs

- 60% map to two BF Class Types
  - ✓ 68,000+ ⟶ BF Input Check (_INP)
  - ✓ 46,000+ ⟶ BF Memory (_MEM)
- Most relate to
  - ✓ Injection ⟵ **SQL Injection**
  - ✓ Memory Corruption/ Disclosure ⟵ **Buffer Overflow**



BF, I. Bojanova

BF–Based AI Agents

- Analyze ⟵ BF definitions, taxonomies
  - ✓ Narrative descriptions ⟵ NVD, CVE, CWE, KEV, CPE, EPSS
  - ✓ Hardware bug/weakness ⟵ firmware (incl. microcode), circuit logic
  - ✓ Software bug/ weaknesses ⟵ source code (e.g., on GitHub)
- Generate
  - ✓ BFVUL Specifications
  - ✓ BF Secure Coding Principles
- Validate/Verify
  - ✓ Toward BF Formal Language

BF–Based FM Proves

- Security Rules
- Code correctness
- Existence of bugs/weaknesses

NVD^{BF} Heartbleed

Example

NATIONAL VULNERABILITY DATABASE NIST

CVE-2014-0160 Detail

Description

The (1) TLS and (2) DTLS implementations in OpenSSL 1.0.1 before 1.0.1g do not properly handle Heartbeat Extension packets, which allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read, as demonstrated by reading private keys, related to d1_both.c and t1_lib.c, aka the Heartbleed bug.

Metrics CVSS Version 4.0 | CVSS Version 3.x | CVSS Version 2.0

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displa

CVSS 3.x Severity and Vector Strings:

NVD **NIST:** NVD Base Score: 7.5 HIGH Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N

ADP: CISA-ADP Base Score: 7.5 HIGH Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-125	Out-of-bounds Read	NIST CISA-ADP

BF Specification of CVE -2014-0160 – Heartbleed in OpenSSL v1.0.1 before v1.0.1g

_INP Weakness — Data Verification (DVR)
Code Defect Bug: Missing Code in 'dtls1_process_heartbeat(SSL *s)'
Operation: Verify length
Data Error: Inconsistent Value 'payload'
Mechanism: Range — 1 + 2 + 16 <= s->s3->rrec.length; 1 + 2 + payload + 16 <= s>s3->rrec.length
Source Code: Third-Party — ssl /d1_both.c: 1462; ssl /t1_lib.c: 2591
Execution Space: Local
Data State: Transferred via network

_MEM Weakness — Memory Addressing (MAD)
Data Fault: Wrong Size in 'memcpy(bp, pl, payload)'
Operation: Reposition pointer
Address Error: Over Bound Pointer 'pl'
Mechanism: Sequential
Source Code: Third-Party — ssl /d1_both.c:1487; ssl /t1_lib.c:2620
Execution Space: Userland
Address State: Heap
Size Kind: Used — for s->s3->rrec.data[0]

_MEM Weakness — Memory Use (MUS)
Address Fault: Over Bound Pointer in 'memcpy(bp, pl, payload)'
Operation: Read object
Memory Corruption/Disclosure Final Error: Buffer Over-Read 'pl'
Mechanism: Sequential
Source Code: Third-Party — ssl /d1_both.c:1487; ssl /t1_lib.c:2620
Execution Space: Userland
Address Kind: Huge — up to 64kb per exploit
Address State: Heap
Size Kind: Used

_MEM Weakness — Memory Use (MUS)
Code Defect Bug: Missing Code
Operation: Clear object
Memory Corruption/Disclosure Final Error: Not Cleared Object
Failure: IEX
Mechanism: Sequential
Source Code: Codebase
Execution Space: Userland
Address Kind: Huge — up to 64kb per exploit
Address State: Heap
Size Kind: Actual

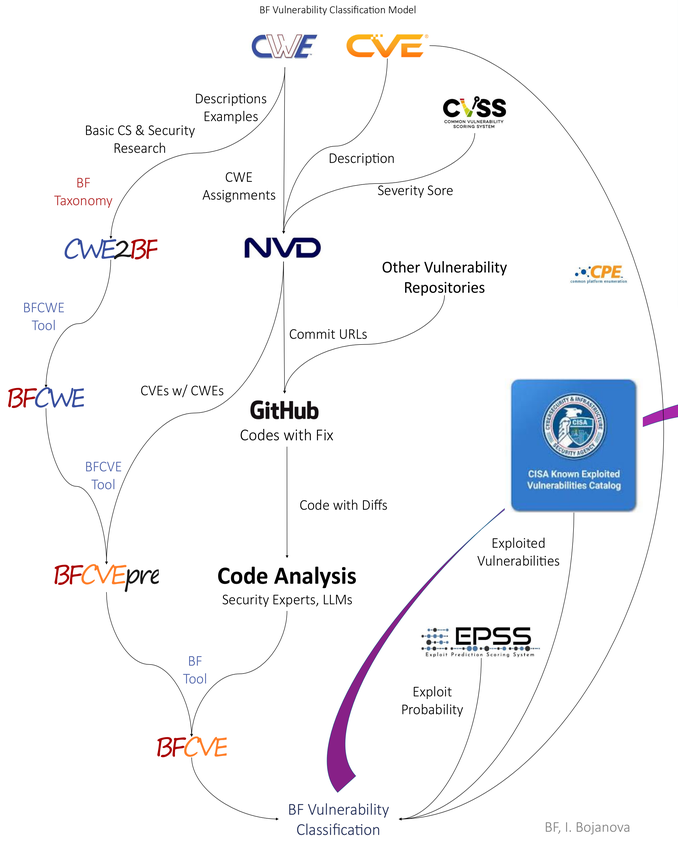
Bug | Operation | Fault / Error | Final Error | Failure

BF Tool, I. Bojanova

Augment NVD

- NVD description ⟵ BF specification
- CVSS scores ⟵ BF attributes analytics
- CWE assignments ⟵ BF security rules

Vulnerability Classifications



BF, I. Bojanova

Multi-dimensional vulnerability classifications

- By properties – e.g.,
  - ✓ **Root cause** – BF code bug or hardware induced BF data fault
  - ✓ Final error – BF exploit vector
- By similarities – e.g.,
  - ✓ Number of underlying weaknesses
  - ✓ Identical chains of weaknesses
  - ✓ Severity – BF attributes

Potential Impact

- Unprecedented cybersecurity analytics capabilities
- Highly informed cybersecurity R&D innovations
- Solidifying NIST NVD as authoritative reference
- Establish NIST BF as the standard for specifying cybersecurity vulnerabilities

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
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.

02/08/2026