CHESS

Computers and Humans Exploring Software Security

Mr. Dustin Fraze

4/19/2018

CHESS

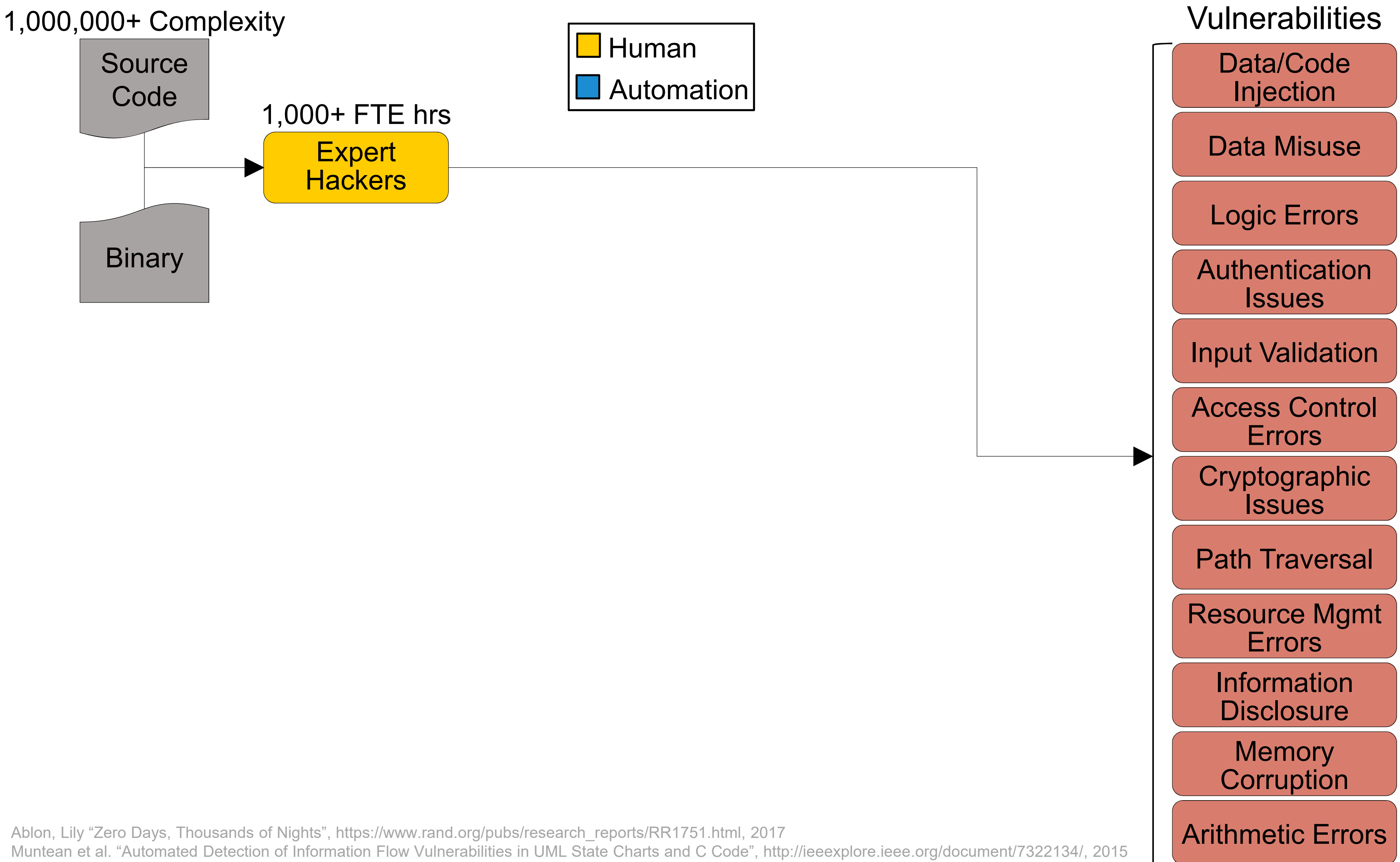
Develop computer-human systems to rapidly discover all classes of vulnerability in complex software

Limits of Current Approaches

Approach	Vulnerability Discovery Speed	Vulnerability Discovery Accuracy	Representative Software Complexity
Human	Low	Low	Web Browser
Computer	High	Low	Small Test Corpora
Computer-Human Experiments[1,2]	High	Moderate	Small Test Corpora
CHESS	High	High	Web Browser

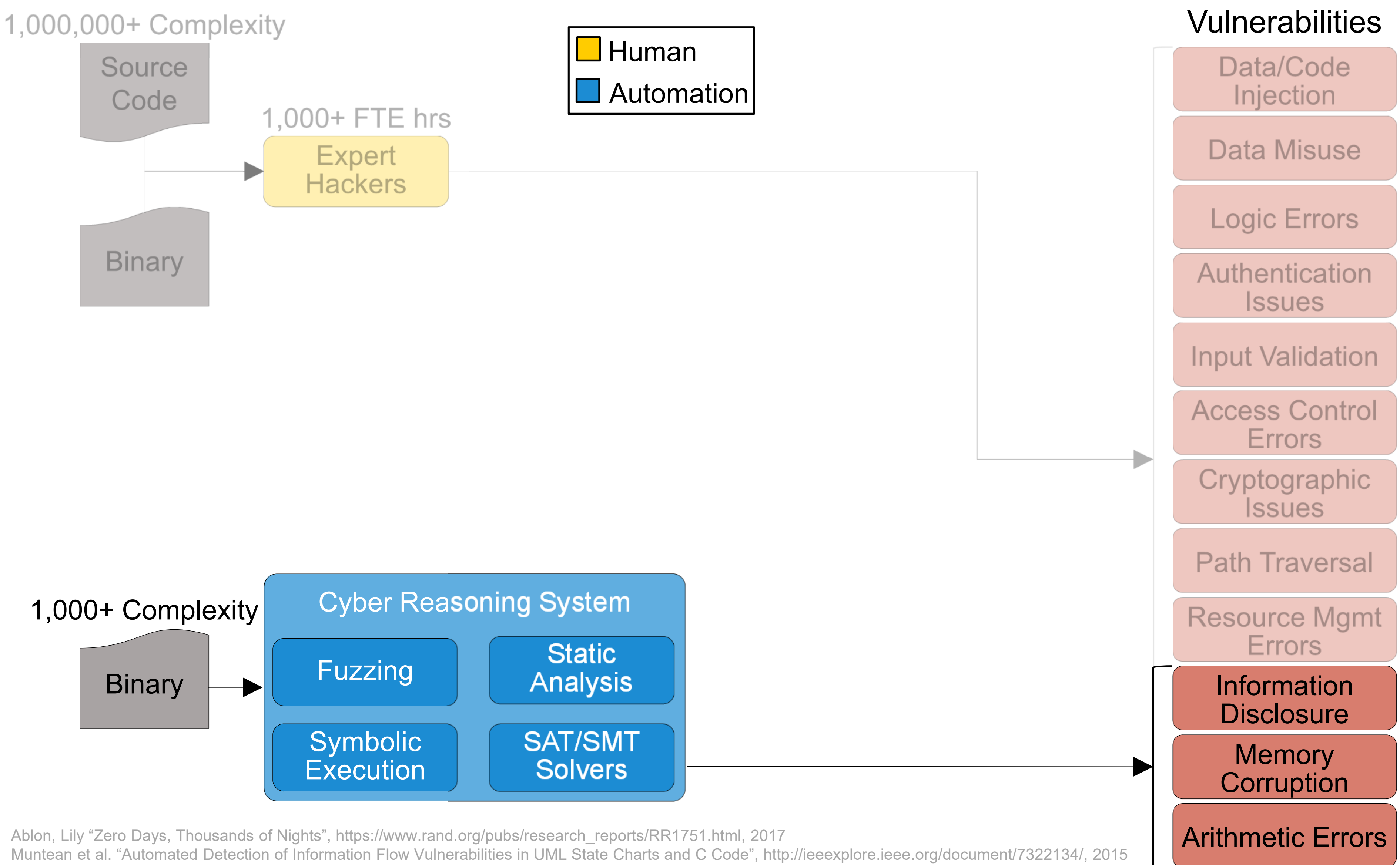
[1] Muntean et al. "Automated Detection of Information Flow Vulnerabilities in UML State Charts and C Code", http://ieeexplore.ieee.org/document/7322134/, 2015
[2] Shoshitaishvili et al. "Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance", https://arxiv.org/abs/1708.02749, 2017

Today's Approach to Vulnerability Discovery



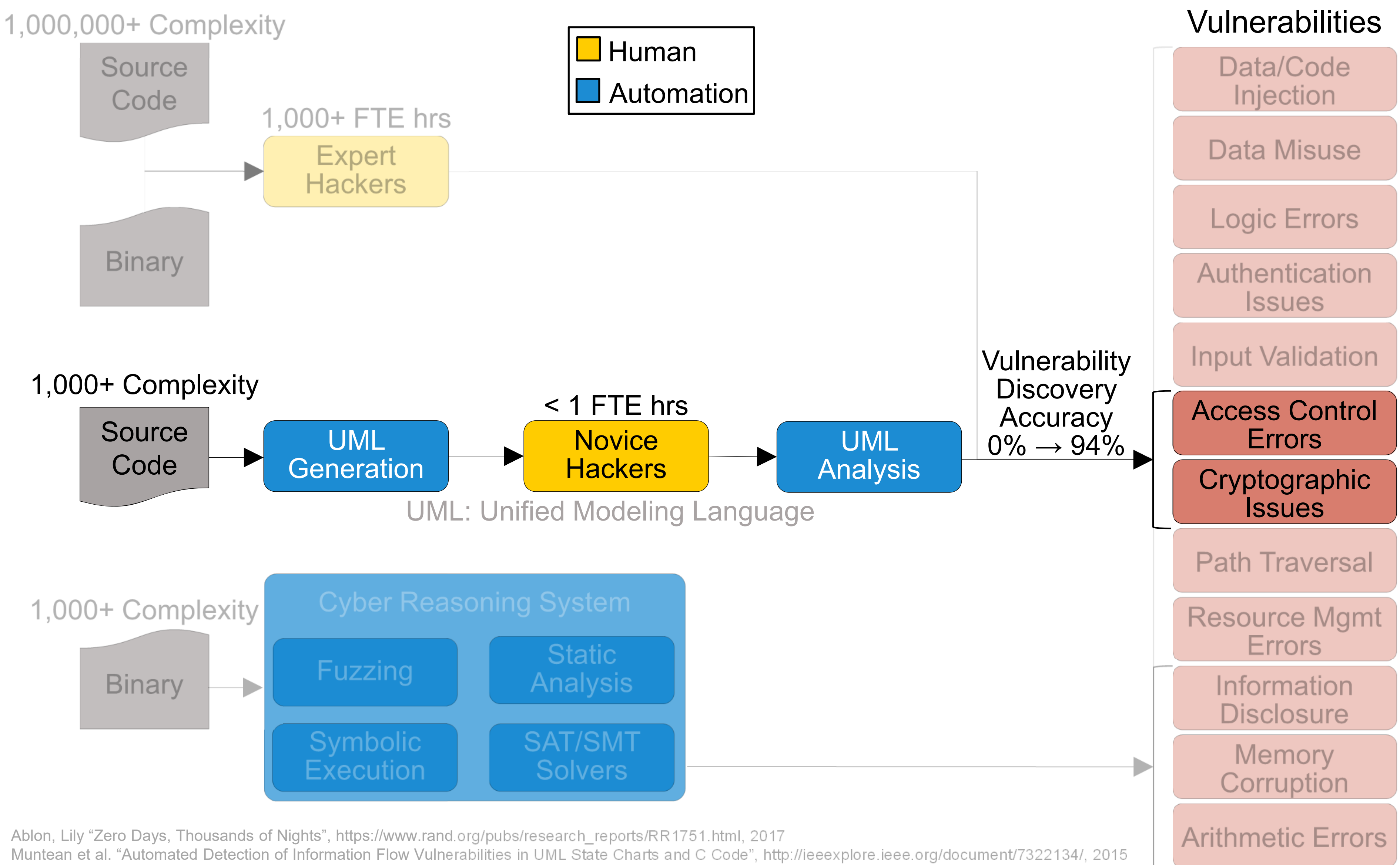
Ablon, Lily "Zero Days, Thousands of Nights", https://www.rand.org/pubs/research_reports/RR1751.html, 2017
Muntean et al. "Automated Detection of Information Flow Vulnerabilities in UML State Charts and C Code", http://ieeexplore.ieee.org/document/7322134/, 2015
Shoshitaishvili et al. "Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance", https://arxiv.org/abs/1708.02749, 2017

Vulnerability Discovery with CGC



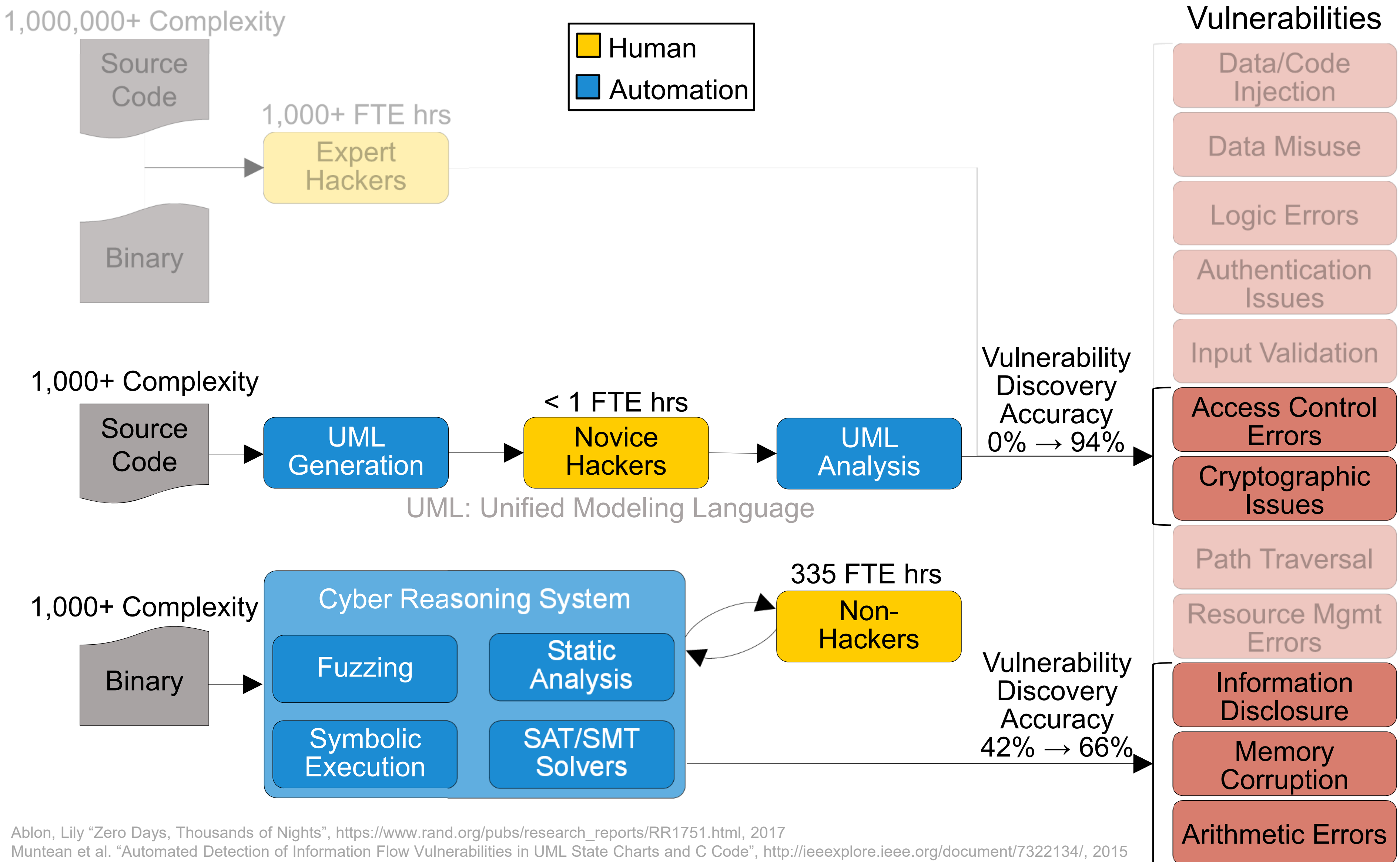
Ablon, Lily "Zero Days, Thousands of Nights", https://www.rand.org/pubs/research_reports/RR1751.html, 2017
Muntean et al. "Automated Detection of Information Flow Vulnerabilities in UML State Charts and C Code", http://ieeexplore.ieee.org/document/7322134/, 2015
Shoshitaishvili et al. "Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance", https://arxiv.org/abs/1708.02749, 2017

Experimental Vulnerability Discovery with Novice Hackers



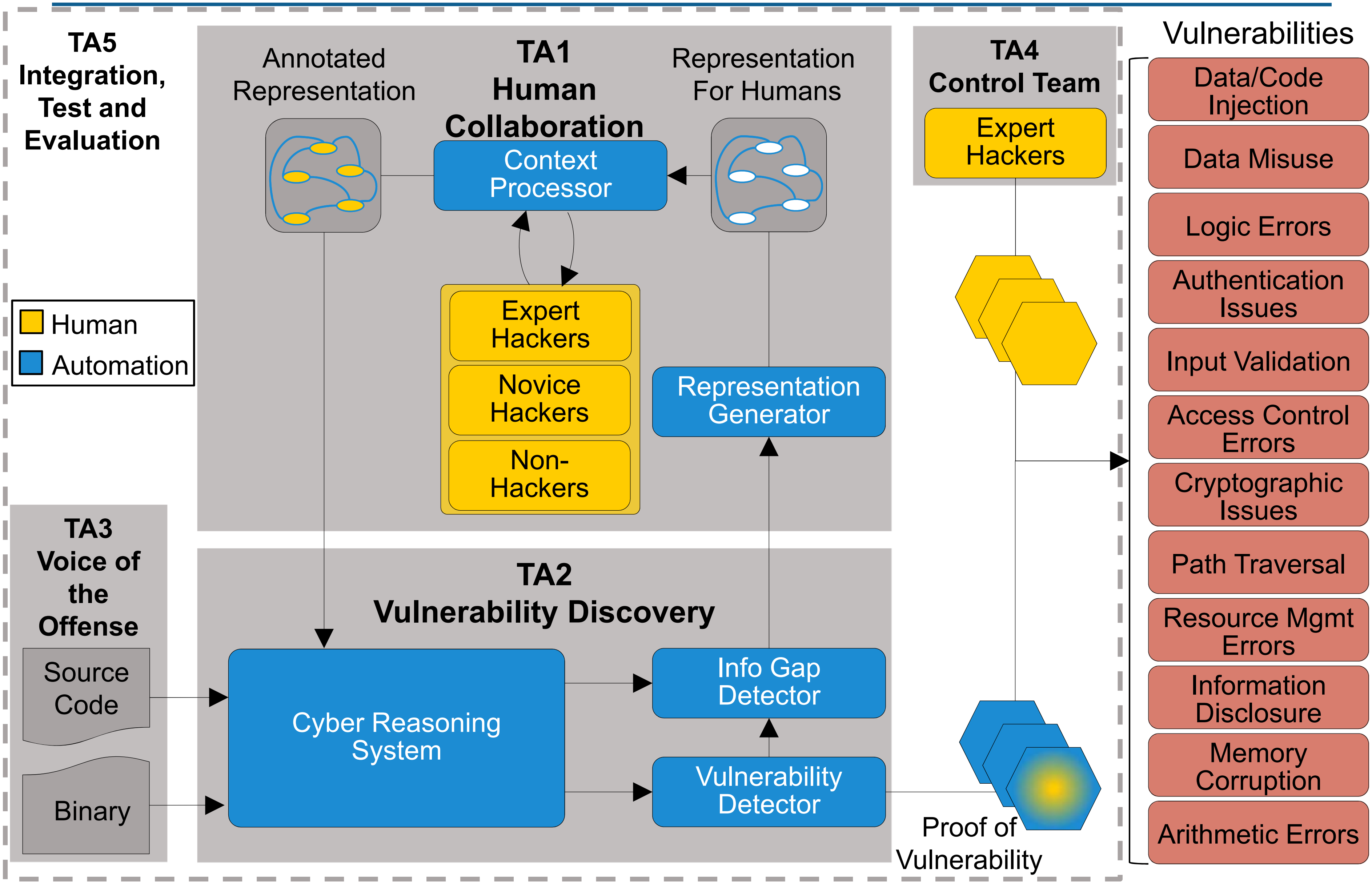
Ablon, Lily “Zero Days, Thousands of Nights”, https://www.rand.org/pubs/research_reports/RR1751.html, 2017
Muntean et al. “Automated Detection of Information Flow Vulnerabilities in UML State Charts and C Code”, http://ieeexplore.ieee.org/document/7322134/, 2015
Shoshitaishvili et al. “Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance”, https://arxiv.org/abs/1708.02749, 2017

Experimental Vulnerability Discovery with Non-Experts



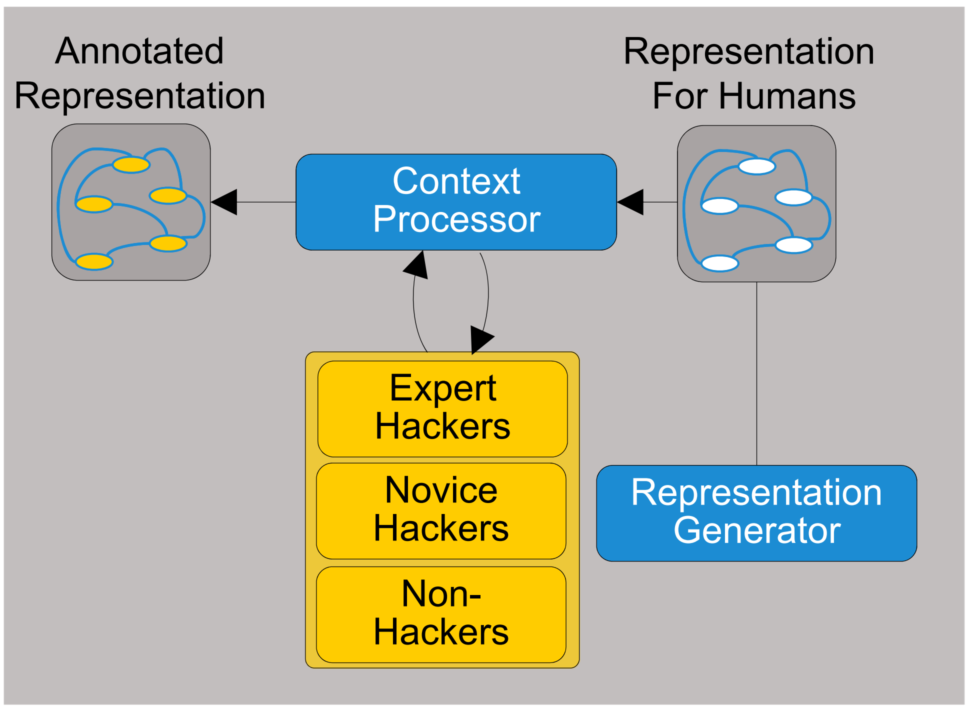
Ablon, Lily "Zero Days, Thousands of Nights", https://www.rand.org/pubs/research_reports/RR1751.html, 2017
Muntean et al. "Automated Detection of Information Flow Vulnerabilities in UML State Charts and C Code", http://ieeexplore.ieee.org/document/7322134/, 2015
Shoshitaishvili et al. "Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance", https://arxiv.org/abs/1708.02749, 2017

Collaborative Vulnerability Discovery with CHESS



TA1 Human Collaboration

Challenges	Possible Approaches
Identify and generate representations that communicate information gaps to humans	• UML Diagrams (Class, Activity, etc.) • Control Flow Graphs • Hilbert Curves for Cyclic Activity
Capture and process the insights humans generate by reasoning over the representations	• Annotation/Label Sets • Instrumented Program Interaction • Human Mental Model Analysis



1. Process identified information gaps into human-understandable representations
2. Summarize and minimize software artifact data
3. Interact with human teammates using generated representations
4. Capture contextual insights from human
5. Process human feedback into machine-ingestible formats

TA1 Human Collaboration

Strong Proposals will:

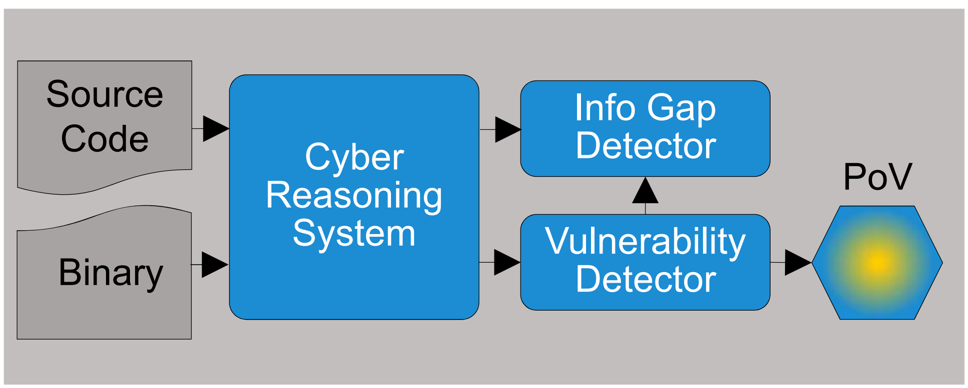
- Reduce the cognitive load and effort required by human collaborators
- Explore new representations and methods of human-computer interaction for capturing human insights
- Empower non-expert collaborators (novice hackers, non-hackers)
- Scale from single computer-human collaboration to N:N team collaboration
- Address any relevant HSR issues (data collection, data anonymization, test subject recruitment, etc.)

Strong Proposals will NOT:

- Involve invasive medical technology
- Only improve performance of expert hackers

TA2 Vulnerability Discovery

Challenges	Possible Approaches
Identify information required to discover classes of vulnerabilities not addressed by automation	• Type Usage • Semantic Metadata • Complexity Inference
Extend CRS technology to scale up and reason over new and existing representations	• Compilation Instrumentation • Type Chain Analysis
Develop new vulnerability detection techniques to leverage human-provided insights	• Object/Data Type Classification • Function Call Context • Semantic Concreteness/Clustering



1. Analyze source code and related software artifacts for potential vulnerabilities
2. Identify regions of uncertainty and other obstacles to automated analysis in source code and related software artifacts
3. Identify vulnerabilities in target categories
4. Generate Proofs of Vulnerability (PoV) and patches

TA2 Vulnerability Discovery

Strong Proposals will:

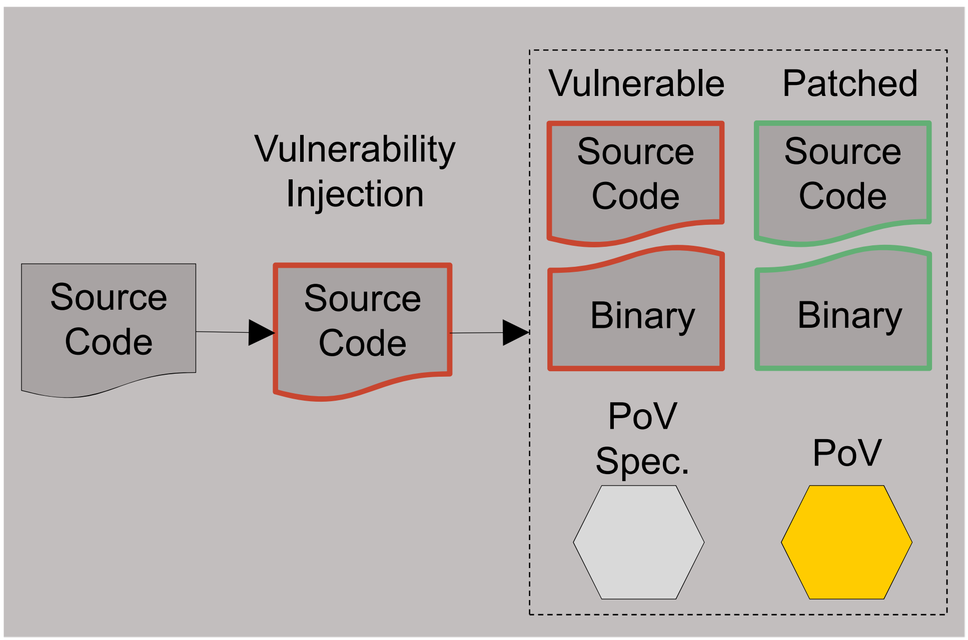
- Identify vulnerability discovery techniques that may benefit from human collaborator insights
- Address vulnerability classes in a thorough and scalable manner
- Generate patches that address underlying vulnerabilities completely and specifically
- Scale from single computer-human collaboration to N:N team collaboration

Strong Proposals will NOT:

- Identify vulnerabilities inserted in challenge sets via diffing
- Focus only on memory corruption and arithmetic errors
- Rely primarily on fuzzing for vulnerability discovery

TA3 Voice of the Offense

Challenges	Possible Approaches
Develop challenge problems scaling to 1M+ complexity	• Large-scale Automated Vulnerability Addition (LAVA)
Ensure challenge problems are representative of required vulnerability classes	• Vulnerability test corpora (Juliet, CGC, OSS-FUZZ, etc.) • Public n-day databases



1. Develop challenge problems with vulnerabilities across all required classes and scaling from 10K to 1M+ complexity
2. Develop a source code patch for each challenge problem vulnerability
3. Develop a binary patch for each challenge problem vulnerability
4. Create a proof of vulnerability (PoV) specification for each vulnerability class
5. Develop a PoV for each challenge problem vulnerability

TA3 Voice of the Offense

Strong Proposals will:

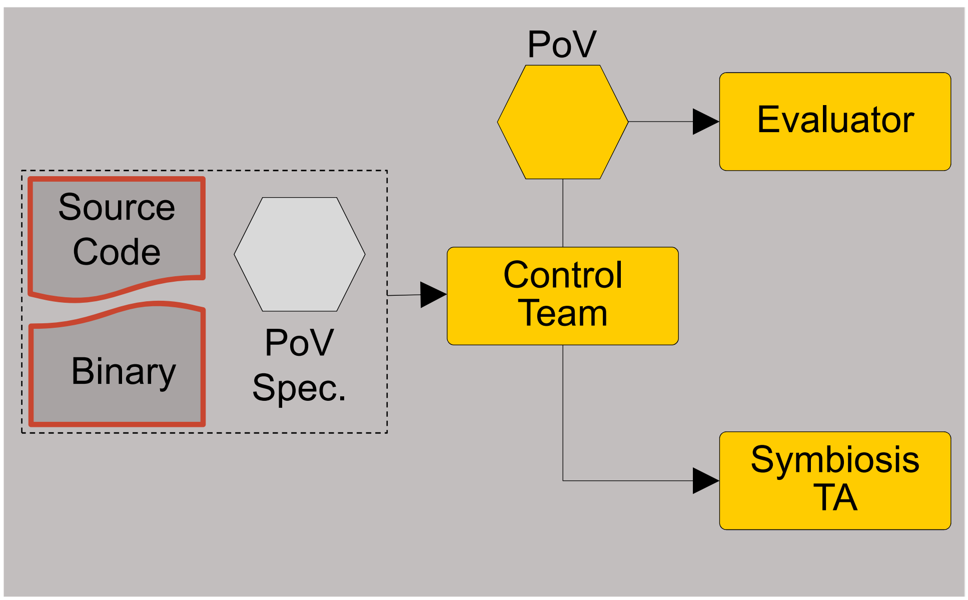
- Ensure challenge set coverage of all vulnerability classes
- Scale challenge sets to be representative of large, complex codebases

Strong Proposals will NOT:

- Allow challenge set vulnerabilities to impact production software
- Search for 0-day vulnerabilities in production software

TA4 Control Team

Tasks
Create an expert hacker performance baseline against TA3 challenge problems
Ensure CHESS R&D teams are aware of edge of the art techniques in software reverse engineering and exploitation



1. Leverage state of the art tools to find vulnerabilities in source code and binary challenge problems developed by TA 3
2. Develop a PoV for each vulnerability discovered in the challenge problems according to the provided PoV specification
3. Collect feedback during evaluations for post-evaluation review by the Symbiosis TA
4. Identify divergent and/or conflicting evaluation performance between the Control Team and CHESS system

TA4 Control Team

Strong Proposals will:

- Demonstrate expertise in the state of the art in vulnerability discovery
- Address both source-assisted and binary vulnerability discovery

Strong Proposals will NOT:

- Identify vulnerabilities inserted in challenge sets via diffing

TA5 Integration, Test and Evaluation

Tasks
Integrate technology and techniques from TA1 and TA2 into a single platform for evaluation and transition
Design and execute tests to measure CHESS system performance against TA3 challenge problems

1. Integrate components from TA1 and TA2 into a single working platform
2. Promote collaboration between performers
3. Evaluate integrated CHESS system performance against TA3 challenge problems
4. Recruit human collaborators for evaluations
5. Demonstrate and transition CHESS technology to identified industry and government partners

TA5 Integration, Test and Evaluation

Strong Proposals will:

- Integrate CHESS system components in a continuous and collaborative manner
- Develop instrumented testbed environments for evaluations
- Promote collaboration between all CHESS performers
- Address any relevant HSR issues (data collection, data anonymization, test subject recruitment, etc.)

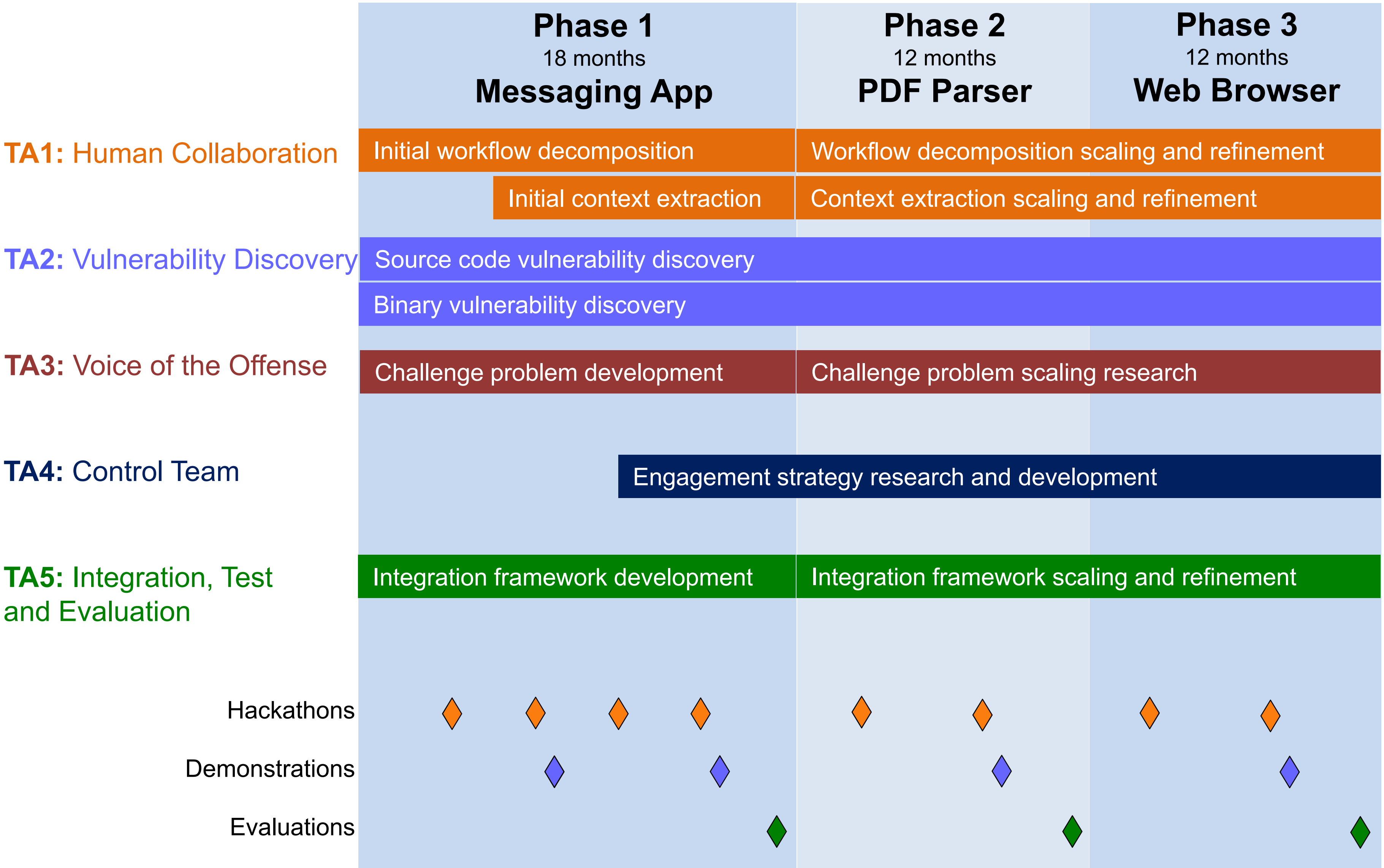
Strong Proposals will NOT:

- Allow challenge set vulnerabilities to impact production software

CHESS Metrics

Phase Duration	Phase 1 18 months	Phase 2 12 months	Phase 3 12 months
Vulnerability Discovery Speed	As fast as control	10x faster than control	100x faster than control
Vulnerability Discovery Accuracy with Source Code	70%	85%	99%
Vulnerability Discovery Accuracy without Source Code	50%	75%	99%
Software Complexity	Messaging App (10K)	PDF Parser (150K)	Web Browser (1M)

CHESS Schedule



www.darpa.mil