A Potentially Specious Cyber Security Offering for 5G/B5G/6G:

Software Supply Chain Vulnerabilities within Certain Fuzzing Modules

Steve Chan
Decision Engineering Analysis Lab
schan@dengineering.org

Researcher Bio

Dr. Steve Chan is an International Academy, Research and Industry Association (IARIA) Fellow. He is an inventor with both international and U.S. patents and serves as a reviewer for 21 peer-reviewed journals/conference proceedings, such as IEEE Access in the area of cyber security for real-time control and monitoring for smart grids.

He has been active in the areas of Cyber as well as Power & Energy/Power Electronics Societies of various IEEE chapters, and he has been an invited speaker, such as at the IEEE Smart Grid Utility Cybersecurity Workshop.

He has authored/co-authored papers that were presented at the IEEE International Conference on Distributed Computing Systems (ICDCS) Workshop, IEEE International Conference on Condition Monitoring and Diagnosis (CMD), IEEE Sensors Applications Symposium (SAS), IEEE Computing and Communication Workshop and Conference (CCWC), IEEE Information Technology, Electronics & Mobile Communication Conference (IEMCON), IEEE Technically Sponsored Future of Information and Communication (FICC) Conference, IEEE International Conference on Information and Communications Technology (ICOIACT), IEEE Future Technologies Conference (FTC), IEEE International Conference on Digital Ecosystems and Technologies (DEST), and the IEEE International Conference on Collaborative Computing (CollaborateCom).

Research Domains

The Decision Engineering Analysis Laboratory has engaged in a variety of cyber-related research, such as in the areas of:

- Leveraging Sidecars for a More Probabilistic Cyber Convergence
- Systems Resilience: Reliable Cyber-protection (cyber defense, guaranteed reliability, cyber awareness, cyber-space, on-line cyber protection, traffic, etc.)
- Log Analysis
- Challenges to Cyber Services
- The Nexus of Cognitive Computing, Artificial Intelligence and Cyber Security – Anomaly Detection at Scale
- Leveraging Artificial Intelligence/Cognitive Computing to Meet the Increasing Cycles of Adaptation within the Cyber Domain
- Advances to Protect Critical Assets
- Cyber Attack Surfaces and the Interoperability of Architectural Application Domain Resiliency
- Advanced Approaches to Enhance Cyber Applications
- Cyber-Centered Major Challenges, Monitoring and Evaluating the Cyber-health of Industrial Systems
- Enhancing Cyber Infrastructural Resilience for Cyber Cities

The Challenge

The issue of software supply chain vulnerabilities has become prevalent. Various governmental directives, such as the “Improving the Nation’s Cybersecurity” (Executive Order 14028, which was issued on 12 May 2021 and proceeded to direct the National Institute of Standards and Technology or NIST to enhance software supply chain security guidelines) underscore the significance.

Given the rise in software supply chain vulnerabilities, fuzz testing (a.k.a., fuzzing) has been invaluable for uncovering a plethora of security vulnerabilities (e.g., improper handling of procedures, invalid integrity protection, and security procedure bypasses) within software.

Hence, it seems ironic that severe software supply chain vulnerabilities have been uncovered within certain mission-critical software fuzzing paradigms — the very mechanism that is supposed to discern cyber vulnerabilities and enhance the cyber posture.

The Posited Approach

Traditionally, white-box fuzzers produce higher quality inputs, but the computational overhead is much higher, while black-box fuzzers that focus upon random mutation have computational overhead that is much lower, but produce lower quality inputs.

To address these challenges, we present a bespoke grey-box concolic fuzzing module, which is comprised of four differing bespoke Grey-Box Concolic Fuzzers (GBCFs).

The GBCFs are able to achieve higher coverage (on average) and able to more robustly discern which parts of a software program they visit and how consistent they are in doing so.

In turn, this GBCF set is fuzzed by tertiary and quaternary GBCFs, so as to mitigate against inadvertently not discerning vulnerabilities within the primary and secondary GBCFs themselves. The utilization of distinct and disparate tertiary and quaternary fuzzers (which utilize different classes for mutating a seed as well as seeding schedules) increases the likelihood of increased coverage (on average).

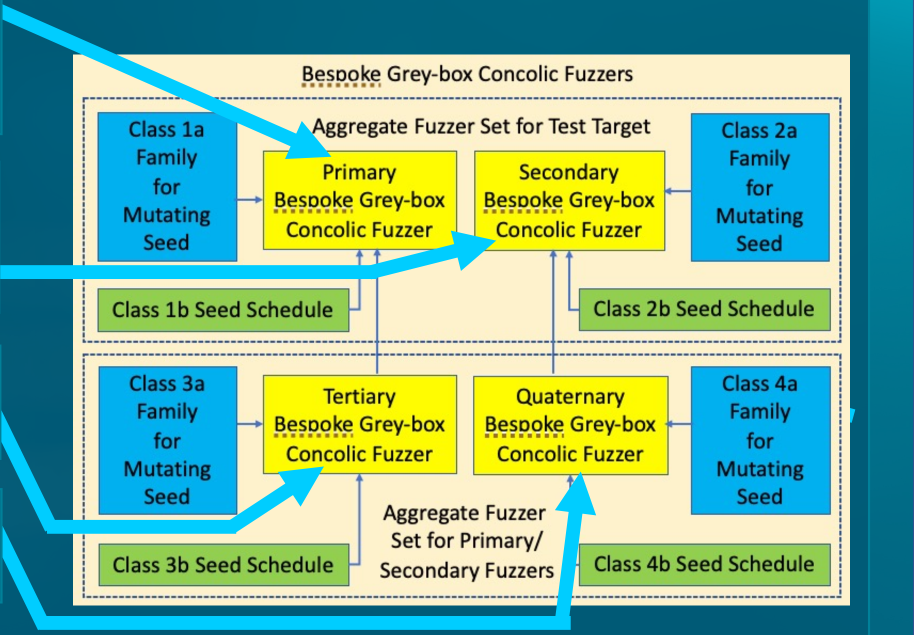
Figure 1

A primary GBCF is able to achieve higher coverage (on average) and able to more robustly discern which parts of a software program it visits and how consistent it is in doing so.

A secondary GBCF, which utilizes different classes (from that of the primary fuzzer) for mutating a seed, contributes toward higher coverage.

A tertiary GBCF fuzzes the primary grey-box concolic fuzzer so as to discern potential vulnerabilities within the fuzzer itself.

A quaternary GBCF fuzzes the secondary grey-box concolic fuzzer so as to discern potential vulnerabilities within the fuzzer itself.



Bespoke Grey-box Concolic Fuzzers

Aggregate Fuzzer Set for Test Target

Class 1a Family for Mutating Seed

Primary Bespoke Grey-box Concolic Fuzzer

Secondary Bespoke Grey-box Concolic Fuzzer

Class 2a Family for Mutating Seed

Class 1b Seed Schedule

Class 2b Seed Schedule

Class 3a Family for Mutating Seed

Tertiary Bespoke Grey-box Concolic Fuzzer

Quaternary Bespoke Grey-box Concolic Fuzzer

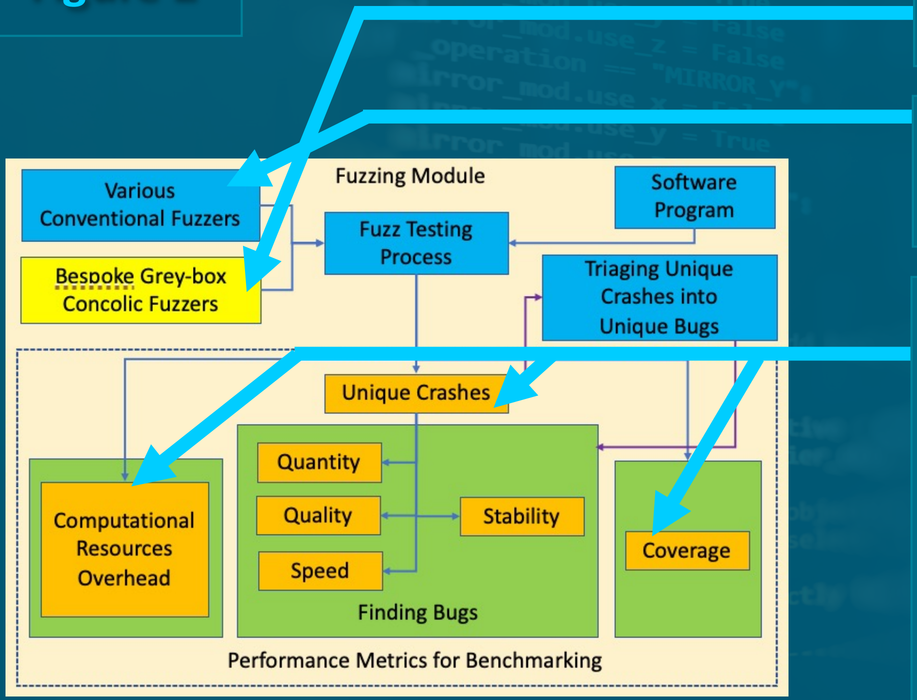
Class 4a Family for Mutating Seed

Aggregate Fuzzer Set for Primary/ Secondary Fuzzers

Class 3b Seed Schedule

Class 4b Seed Schedule

Figure 2



Fuzzing Module

Various Conventional Fuzzers

Bespoke Grey-box Concolic Fuzzers

Fuzz Testing Process

Software Program

Triaging Unique Crashes into Unique Bugs

Unique Crashes

Quantity

Quality

Speed

Stability

Finding Bugs

Computational Resources Overhead

Coverage

Performance Metrics for Benchmarking

The entirety of the prior Figure 1 is situated within the yellow box of Figure 2.

Various conventional fuzzers are still quite useful, but in many cases, they are sub-optimal at discerning ‘hard-to-trigger’ bugs. For these cases, the GBCF are invaluable.

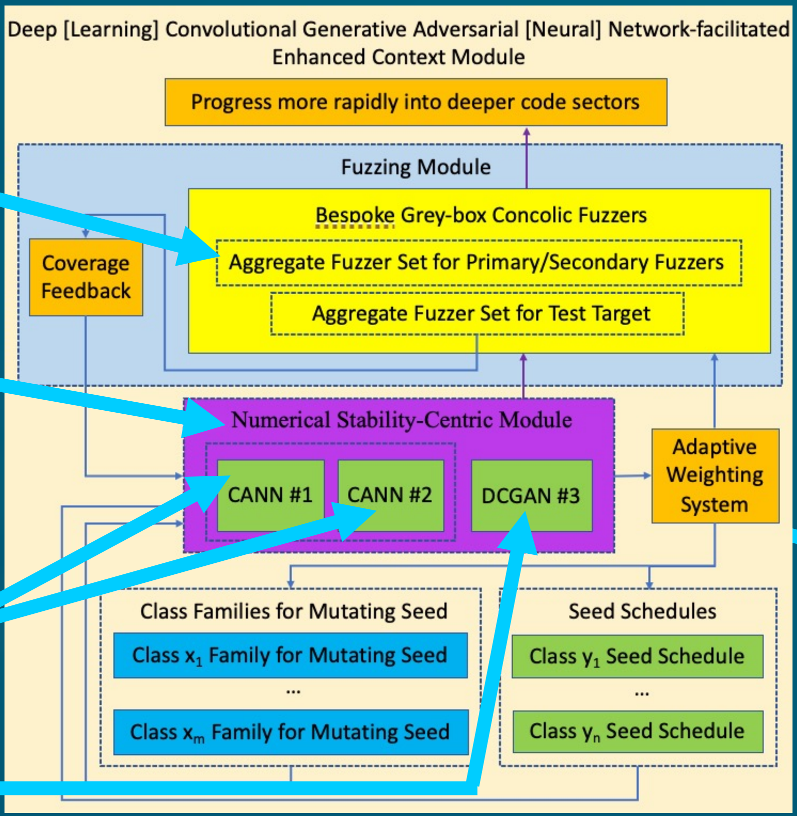
Performance metrics for assessing the GBCF include, but are not limited to: (1) Unique Crashes, (2) Computational Resources Overhead, and (3) Coverage. For (1), a lower Relative Standard Deviation (RSD) implies higher performance stability (as the discovery rate of bugs is above the acceptable threshold). For (2), if a particular paradigm is effective at finding more bugs, but the resources consumed are disproportionate, then that must be taken into consideration. Finally, (3) signifies the intrinsic ability of the fuzzer for exploring new pathways; this is of import for pursuing relevant pathways, which lead to vulnerable code.

Figure 3

The coverage feedback derived by both primary and secondary GBCFs (Aggregate Fuzzer Set for Primary/Secondary Fuzzers) helps to operationalize an underpinning numerical stability-centric Deep Learning Convolutional Generative Adversarial Neural Network-facilitated Enhanced Context Module, which is underpinned by a Numerical Stability-Centric Module (NSCM).

The NSCM contains two Convolutional Adversarial Neural Networks (CANNs), each with a different implementation and version of PyTorch; PyTorch v0.4.1 (more numerically stable) is used in CANN #1, and PyTorch v1.7.0 (less numerically stable) is used CANN #2.

The Tensorflow-based DCGAN #3 helps to stabilize the entire NSCM paradigm.



Deep [Learning] Convolutional Generative Adversarial [Neural] Network-facilitated Enhanced Context Module

Progress more rapidly into deeper code sectors

Fuzzing Module

Bespoke Grey-box Concolic Fuzzers

Aggregate Fuzzer Set for Primary/Secondary Fuzzers

Aggregate Fuzzer Set for Test Target

Coverage Feedback

Numerical Stability-Centric Module

CANN #1

CANN #2

DCGAN #3

Adaptive Weighting System

Class Families for Mutating Seed

Class x_1 Family for Mutating Seed

...

Class x_m Family for Mutating Seed

Seed Schedules

Class y_1 Seed Schedule

...

Class y_n Seed Schedule

Experimentation Findings

The Deep Learning Convolutional Generative Adversarial Neural Network-facilitated Enhanced Context Module (ECM) is the mainstay of the Fuzzing Module; its purpose is to serve as a macro feedback loop. In essence, the ECM selects a seed, mutates it, and serves it as input to the test target. If the input causes a crash, it will be added to the ECM's crash set. Alternatively, if the input segues to new coverage, it will be added to the search seed pool.

In turn, the Fuzzing Module derives Coverage Feedback from the primary and secondary GBCFs (Aggregate Fuzzer Set for Primary/Secondary Fuzzers) for the Test Target. This then serves as input to the NSCM, which processes the information and informs an Adaptive Weighting System, which dynamically weights the Class Families for the Mutating Seed and Seed Schedules. This should segue to a more optimal Seed Schedule for decreasing Time-to-Exposure (TTE) - the speed at which bugs are found - as well as RSD. The resulting lower RSD/higher performance stability can be attributed to the NSCM and Adaptive Weighting System.

Overall, the bespoke grey-box concolic fuzzing module, which is comprised of four differing GBCFs, shows promise.

Conclusion

In a not insignificant portion of the 5G/B5G/6G ecosystem cyber cases, the more serious security problems are implementation imperfections (e.g., network protocols); these constitute attack surface areas, which are often exploited. In the case for which 5G/B5G/6G protocols are still evolving and being defined, these implementation imperfections can be amplified. Conventional software cyber security frameworks, which involve code review, risk analysis, penetration testing, and prototypical fuzzing, do not currently suffice for robustly addressing a domain space, such as the 5G/B5G/6G ecosystem, wherein the protocols are evolving at a rapid pace.

Prototypical fuzzers are challenged by the coverage issue, and conventional Coverage-based Grey-box Fuzzers (CGFs) are as well. In an endeavor to provide a mitigation pathway, we presented an architectural stack comprised of a sequence of bespoke GBCFs; as the primary GBCF (used against the testing target) is designed to work in conjunction with a secondary GBCF so as to better mitigate against coverage issues (e.g., increasing the probability of visiting certain blocks/lines of code of the software program), and both are fuzzed by tertiary and quaternary GBCFs (which utilize different classes for mutating a seed as well as seeding schedules), so as to mitigate against inadvertently not discerning vulnerabilities within the primary and secondary GBCFs themselves, the likelihood of increased coverage (on average) is enhanced. The feedback for coverage and adaptive weighting as well as seed scheduling schemas contribute to the efficacy.