경량 퍼징을 위한 테스트케이스 선택 기법

박나은 [1], 김연진 [2], 이일구 [3]
[1] 성신여자대학교 미래융합기술공학과 박사과정
[2] 성신여자대학교 융합보안공학과 석사과정
[3] 성신여자대학교 융합보안공학과, 미래융합기술공학과 교수

nepark.cse@gmail.com, duswlsqhfk@gmail.com, iglee@sungshin.ac.kr

Testcase Selection Technique for Lightweight Fuzzing

Na-Eun Park[1], Yeon-Jin Kim[2], Il-Gu Lee[3]
[1] Dept. of Future Convergence Technology Engineering, Sungshin Women's University
[2] Dept. of Convergence Security Engineering, Sungshin Women's University
[3] Dept. of Convergence Security Engineering & Future Convergence Technology Engineering, Sungshin Women's University

요 약

최근 IoT (Internet of Things, IoT) 기기가 전 산업과 일상 생활에 활용되면서 취약점 탐지 기술이 중요해지고 있다. 그러나 리소스가 제약적인 IoT 기기에는 종래의 퍼징 기술을 적용하기 어렵다. 본 논문에서는 경량화 IoT 환경에서 퍼징 기술을 적용하기 위한 테스트케이스 선택 기법을 제안했다. 실험 결과에 따르면, 제안하는 방식은 무작위 입력을 생성하여 퍼징하는 종래 퍼저보다 평균 61.49% 빠르게 취약점을 탐지했다.

1. 서론

최근 사물인터넷(Internet of Things, IoT) 기술이 급격히 성장하고 있으며, 스마트 홈, 산업 인프라, 의료 환경 등 다양한 분야에서 IoT 기기가 이용되고 있다. 그러나 네트워크를 통해 연결되는 기기 수가 증가하면서, IoT 장치의 취약점과 위협도 증가하고 있다. 특히, IoT 장치의 펌웨어 취약점은 해킹에 악용되는 주요 공격 경로가 되고 있다[1]. 이러한 펌웨어 취약점을 식별하기 위해 다양한 기법들이 연구되었으며, 그 중 대표적인 취약점 분석 기법은 퍼징(fuzzing)이다. 퍼징은 충돌, 버그, 중단과 같은 예상치 못한 동작을 발견하기 위해 타겟 프로그램에 수많은 테스트 값을 무작위로 생성하여 입력하는 탐지 기술이다[2]. 최근 퍼징 기술은 가장 활발하게 사용되는 취약점 탐지 기술 중 하나로 주목받고 있으며 높은 효율성과 낮은 오탐률을 보인다는 장점이 있다[3].

그러나 종래의 퍼징 기술은 검사 대상 프로그램의 구조가 복잡할수록 입력 값 생성 시 많은 오버헤드가 발생한다는 한계점이 있다[2, 4]. 특히, IoT 기기와 같은 제한된 리소스를 가진 저전력 환경에서는 컴퓨팅 성능이 제한되어 있어서[5], 복잡한 퍼징 기술을 적용하기 어렵다.

본 논문에서는 저전력 IoT 환경에서 퍼징 기술을 적용할 때의 오버헤드를 줄이기 위한 테스트케이스 선택 기법을 제안한다. IoT 기기가 정상 동작을 수행하는 기간에는 퍼징하지 않고, 기기가 절전 상태가 되면 정상 동작 기간동안 수집된 로그 데이터를 분석하여 입력 값과 그에 대한 결과 정보를 수집한다. 이후 동일한 입력 값을 가진 테스트케이스는 삭제하고 남은 테스트케이스 중 탐지 필요성이 높은 테스트케이스에 더 높은 우선순위를 할당한다. 따라서 테스트케이스 입력을 최소화하면서 중요한 검사는 빠르게 수행할 수 있다.

본 논문의 주요 기여점은 아래와 같다.

- 테스트케이스를 최적화하는 경량 퍼징 방법을 제안한다.
- 퍼징 기법의 효율성을 비교 분석하는 프레임워크를 제안한다.

본 논문은 다음과 같이 구성된다. 2 장에서 경량 퍼징을 위한 종래 기술을 분석하고, 한계점을 도출한다. 3 장에서는 제안하는 테스트케이스 선택 기법에 대해 설명하고, 4 장에서는 제안하는 기법의 증명을 위한 시뮬레이션 설계 및 실험 결과를 분석한다. 5 장에서는 연구 결론과 한계점을 보완하기 위한 후속 연구 계획에 대해 설명하고 마무리한다.

2. 선행 연구 분석

선행연구 [6]은 IoT 웹 퍼징 기술의 최적화를 위해 정적 분석 기술을 활용하여 정보를 수집한 후 동적 분석을 수행하는 2 단계 퍼징 시스템인 IoTParser 를 제안했다. IoT 펌웨어에 대한 정적 분석을 통해 웹 페이지 경로와 인터페이스 정보 등을 수집하고 중요도에 따라 시드의 우선순위를 설정하거나 동적 분석에 활용했다. 실험을 통해 IoTParser 의 취약점 탐지율과 효율성을 평가하였으며 5 개 유형의 IoT 장치를 선택하고 W13scan 과 Firmhunter 기술과 비교했다. 실험 결과에 따르면 IoTParser 는 취약점 탐지율을 44% 개선할 수 있었고, 효율성은 48.2% 개선했다. 그러나 이 연구는 압축된 펌웨어를 대상으로 퍼징을 수행할 수 없었다. 펌웨어를 암호화하여 패키지화 한 최근 IoT 펌웨어 유형에는 적용할 수 없어 제한된 환경에서의 적용 및 성능 향상만 가능하다는 한계점이 있다.

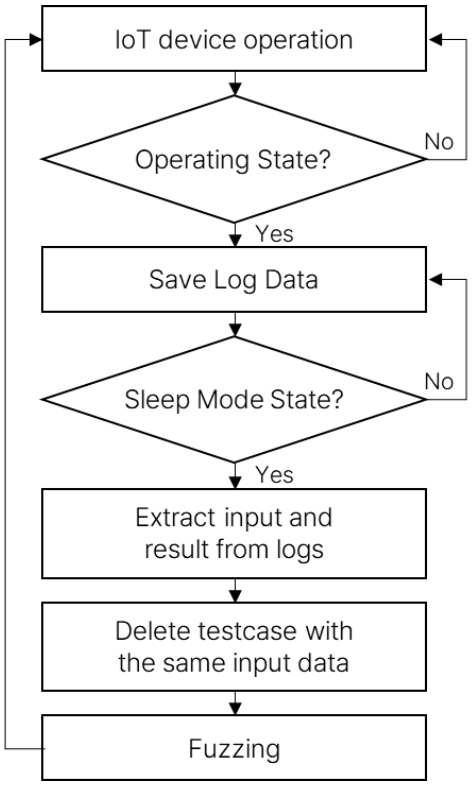
선행연구 [7]은 종래 퍼징 기술들이 테스트 리소스 증가를 무시하고 효율성 향상에 초점을 두는 문제에 주목하여, 종래 퍼징의 자원 낭비 문제를 해결하기 위한 분산 환경 기반 퍼저인 UltraFuzz 를 제안했다. 퍼징과 스케줄링을 분리하여 적절한 리소스를 할당함으로써 전력 소모를 최적화하고, 중앙 집중식 동적 스케줄링을 통해 시드 값을 수집하고 평가하여 중복 테스트케이스를 삭제했다. 이 연구에서는 AFL, AFL-P, PAFL, EnFuzz 와 같은 종래 도구들과 비교하여 취약점 발견 및 오버헤드 측면의 성능을 검증했다. 실험 결과에 따르면 UltraFuzz 는 AFL 보다 10 개 더 많은 취약점을 탐지했고, PAFL, EnFuzz, AFL-P 보다 16 개 이상 더 많은 취약점을 탐지했다. AFL-P 는 오버헤드를 1.42% 개선했고, UltraFuzz 는 1.26% 개선했다. 그러나 중앙 집중식 스케줄러에 의존하기 때문에 분산 IoT 환경에서는 별도의 중앙 집중 환경이 구현되어야 한다. 또한, UltraFuzz 의 핵심 메커니즘 중 하나인 Super-Linear Acceleration 기법은 active phase 에서는 효과가 있지만, smooth phase 에서는 테스트가 포화되어 장기적으로 사용하기 어려운 문제가 있다.

선행연구 [8]은 종래 퍼징 방식이 사용하는 시드 스케줄링 방식이 제어 흐름 내 탐색되지 않은 영역을 고려하지 않는다는 한계점을 해결하기 위해, 엣지 커버리지 계측 기술을 이용한 그레이박스 퍼저 UntouchFuzz 를 제안했다. 종래의 AFL 퍼저를 개선한 것으로, 검사하지 않은 엣지 커버리지가 결정되면 시드 스케줄링을 이용하여 검사하지 않은 엣지에게 우선순위를 부여하여 최적의 시드 세트를 출력함으로써 오버헤드를 개선하고자 했다. 이 연구에서는 AFL, AFLFast, EcoFuzz 와 탐지 커버리지를 비교하였으며 종래 연구 대비 약 5.87% 향상된 커버리지 개선을 보여 제안 방식의 성능을 입증했다. 그러나 이 연구는 제안한 퍼징 기술이 오버헤드를 개선할 수 있는지 증명하지 못했다.

[6-8] 연구와 같이, 종래 경량 IoT 환경에서 퍼징 기술을 적용하기 위해 다양한 최적화 기법들이 연구되고 있다. 그러나 종래 연구들은 특정 환경에 제한된 연구 기술을 제안하여 범용적인 환경에서 적용이 어렵거나, 장치가 지속적으로 동작할 때 비효율적이다.

3. 테스트 케이스 최적화 기법

본 장에서는 제안하는 테스트케이스 최적화 기법에 대해 설명한다. 제안하는 방식의 전체적인 흐름은 그림 1 과 같다.



(그림 1) Flowchart of the Proposed Technique

IoT 기기가 일반적인 통신을 수행할 때, 퍼저는 별도의 분석을 수행하지 않고 대기한다. 그러나 기기가 절전 모드가 되면, 퍼저는 IoT 기기가 통신을 수행하면서 수집한 로그 데이터를 분석한다. 이후, 퍼저는 로그 데이터를 이용하여 입력 값에 대한 동작 결과가 있었는지 추출하고, 중복된 입력 값을 가진 테스트케이스를 삭제한다. 삭제 후에는 남은 테스트케이스 중 중요도가 높은 테스트케이스의 우선순위를 높여서 절전 모드 시간동안 분석한다. 우선순위가 높은 테스트케이스에 대한 검사가 완료되면, 남은 테스트케이스를 절전 모드 기간동안 무작위 검사하여 퍼징을 수행

한다. 따라서 제안 방식을 적용할 경우 IoT 기기는 일반적인 통신 기간동안 별도의 전력을 소모하지 않고 동작하며, 절전 모드 기간동안 동작함으로써 IoT 기기의 통신 성능에 영향을 미치지 않고 퍼징을 수행할 수 있다.

4. 성능 평가 및 분석

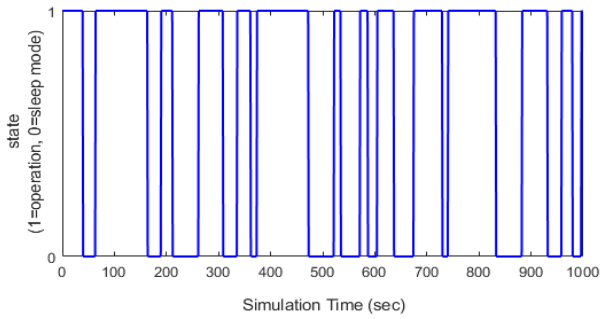
본 논문에서 제안하는 방식의 개념적 검증을 위해 시뮬레이터를 설계하고 평가했다. 시뮬레이터는 파이썬으로 설계되었으며, 동작 방식과 환경 조건은 방식의 단순성을 위해 시스템 파라미터와 환경 조건을 가정하여 시뮬레이션했다. 시뮬레이션에서 사용한 변수는 표 1 과 같다.

(표 1) Simulation Parameters

Parameter	Value
Simulation time	1,000 sec
Operation time	1~10% of the simulation time
Power saving time	1~5% of the simulation time
Number of total testcase	1,000
Number of important testcase	100 (10% of the total testcases)
Number of logs generated during the operation time	Poisson random value (λ=1,5,10)

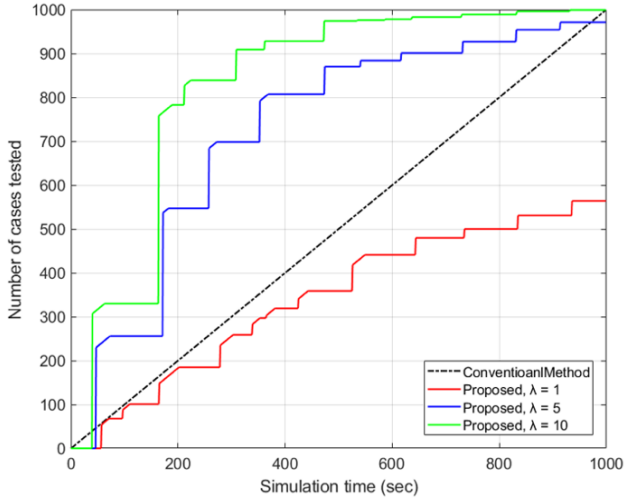
시뮬레이션 중에 동작 시간과 절전 모드 시간은 랜덤하게 발생한다. 할당된 시간이 종료되면 다른 동작으로 전환된다. 동작 시간 동안 발생하는 로그 데이터 수는 random.poisson() 함수를 이용하여 포아송 랜덤 분포로 발생하도록 설계했다. 로그 데이터 발생 빈도는 λ 값을 1, 5, 10 으로 설정하여 로그 데이터가 적게 생성될 때(λ=1)와 많이 생성될 때(λ=10)로 환경을 나눠서 평가했다. 동작 시간 동안 발생하는 로그 데이터는 중복된 값이 생성될 수 있으며, 중복된 로그 데이터는 탐지에 반영하지 않았다. 퍼저의 동작은 종래 방식과 제안한 방식 모두 1 초에 1 개의 테스트 케이스를 검사한다.

취약점이 발생하는 테스트케이스를 사전에 중요한 테스트케이스로 선별하여 평가했다. 중요한 테스트케이스를 검사한 경우 취약점을 탐지하고, 종래의 무작위 입력 값을 생성하여 퍼징을 수행했을 때와 제안 방법을 적용했을 때의 퍼징 수행 횟수와 취약점 탐지율을 평가했다. 시뮬레이션 시간동안 전환된 동작 상태와 절전 모드 상태의 변화는 그림 2 와 같다.



(그림 2) State changes during simulation time

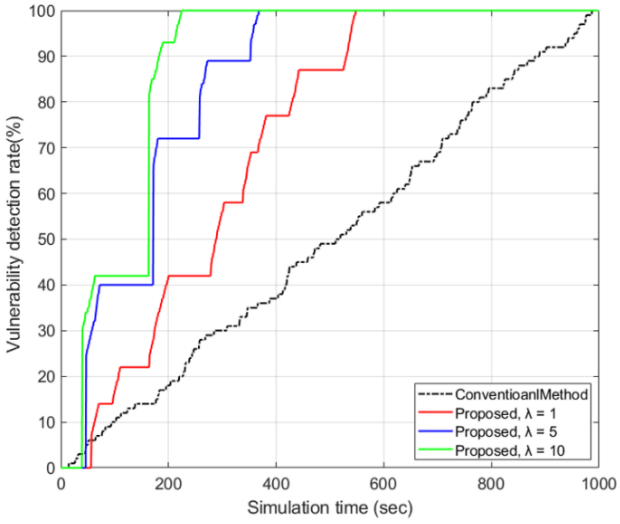
시뮬레이션 시간 동안 시간 흐름에 따른 퍼징 수행 횟수에 대한 결과는 그림 3 과 같다.



(그림 3) Number of fuzzing during simulation time

무작위 입력 생성 및 검사를 수행하는 종래 방식은 매초마다 1 개의 테스트를 수행하므로 시간과 횟수가 선형적으로 증가한다. 그러나 제안하는 방식은 동작 상태에서는 퍼징을 수행하지 않고, 절전 모드 상태일 때만 퍼징을 수행한다. 또한, 제안한 방식은 동작 상태일 때 수집된 로그 데이터로 삭제한 테스트케이스를 절전 모드 상태에서 테스트하여 취약점 검출율이 계단 형태였다. 또, 제안 방식은 동작 과정에서 발생하는 데이터 발생 빈도에 따라 성능에 차이가 있다. 가장 적은 데이터가 발생하는 환경(λ=1)에서는 입력 값을 추출할 로그 데이터가 적으므로, 종래 방식보다 퍼징 횟수가 적다. 그러나 λ 값이 증가함에 따라 같은 시간 대비 λ=5 는 약 73% 였고, λ=10 은 약 114% 더 많은 퍼징 횟수를 보였다.

그림 4 는 종래 방식과 제안 방식의 시간에 따른 취약점 탐지율을 비교한 것이다. 무작위 입력 값으로 테스트하는 종래 방식과 중요한 테스트케이스가 미리 선별된 제안 방식의 성능을 비교했다.



(그림 4) Vulnerability detection rate during simulation time

종래의 방식은 전체 입력 값 중 무작위로 선정하고 생성하여 테스트하므로 시뮬레이션 시간에 따라 선형적인 검출률이 측정된다. 그러나 제안한 방식은 절전 모드 상태가 되면 중요한 테스트케이스로 선별된 입력 값을 우선으로 선정하여 테스트하기 때문에 취약점을 더 빠르게 탐지할 수 있다. 특히, 퍼징 수행 횟수 측면에서 종래 방식보다 낮은 성능을 보였던 λ=1 환경은 종래 방식이 중요 취약점 중 53%를 탐지한 시점에 중요 취약점 전체를 탐지하여 속도 향상을 보였다. λ=5 에서는 종래 방식이 36%를 탐지한 시점에 중요 취약점 전체를 탐지할 수 있었고, λ=10 에서는 종래 방식이 20%를 탐지한 시점에 제안 방식이 중요 취약점 전체를 탐지했다. 즉, 종래 방식보다 평균 61.49% 빠르게 중요 취약점을 탐지할 수 있음을 입증했다.

5. 결론

사물인터넷 환경이 확대되면서 취약점 탐지를 위한 퍼징 기술이 활발하게 연구되고 있다. 그러나 종래의 퍼징 기술은 경량 IoT 기기의 리소스 제한 환경을 고려하지 않았다. 본 논문에서는 경량화 환경에서 효율적인 퍼징 기술을 위한 테스트케이스 선택 기법을 제안했다. 제안하는 방식을 종래의 방식과 비교 평가한 결과에 따르면 무작위 입력을 사용하는 종래 퍼징 방식보다 61.49% 더 빠르게 중요 취약점을 탐지할 수 있었다. 후속 연구로는 제안 방식을 실제 환경에 적용하기 위해 취약점을 내포한 테스트케이스를 선별하고 우선순위를 지정하는 방법을 연구할 계획이며, 여러 환경에서 제안 방식을 구현, 평가하고자 한다.

ACKNOWLEDGEMENT

본 논문은 2024 년도 산업통상자원부 및 한국산업기술진흥원의 산업혁신인재성장지원사업 (RS-2024-00415520)과 과학기술정보통신부 및 정보통신기획평가원의 ICT 혁신인재 4.0 사업의 연구결과로 수행되었음 (No. IITP-2022-RS-2022-00156310)

참고문헌

[1] L. Situ et al., “Physical Devices-Agnostic Hybrid Fuzzing of IoT Firmware,” in IEEE Internet of Things Journal, vol. 10, no. 23, pp. 20718-20734, 1 Dec.1, 2023, doi: 10.1109/JIOT.2023.3303780

[2] Zhao, X. et al., “A systematic review of fuzzing,” Soft Comput, 28, 5493–5522, 2024.

[3] S. Qin et al., “NSFuzz: Towards Efficient and State-Aware Network Service Fuzzing,” ACM Trans. Softw. Eng. Methodol. 32, 6, Article 160, 26 pages, 2023.

[4] Y. Zhao et al., “Grammar-aware test case trimming for efficient hybrid fuzzing,” Journal of King Saud University - Computer and Information Sciences, Volume 36, Issue 1, 2024.

[5] S. -E. Jeon et al., “Two-Step Feature Selection Technique for Secure and Lightweight Internet of Things,” 2023 32nd International Conference on Computer Communications and Networks (ICCCN), pp. 1-6, 2023.

[6] Gao, Y. et al., “Optimizing IoT Web Fuzzing by Firmware Infomation Mining,” Applied Sciences 12, no. 13, 6429, 2022.

[7] X. Zhou et al., “UltraFuzz: Towards Resource-Saving in Distributed Fuzzing,” in IEEE Transactions on Software Engineering, vol. 49, no. 4, pp. 2394-2412, 2023.

[8] Xie, C. et al., “Not All Seeds Are Important: Fuzzing Guided by Untouched Edges,” Applied Sciences 13, no. 24, 2023.