網安實務 - Project Midterm Presentation

組員資訊:

網工所 碩一 309552005 吳偉誠

預計的題目 :

重現「POSTER: AFL-based Fuzzing for Java with Kelinci」論文

Publication :

CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, October 2017

該論文Link : https://dl.acm.org/doi/abs/10.1145/3133956.3138820

題目說明 :

AFL(American Fuzzy Lop)是Fuzzing的一個很好用的工具，可以高效率的對binary程式進行Fuzzing，來找出可能存在的漏洞，如:Buffer-Overflow、Heap-Overflow、Double Free...等，AFL是Mutation-Based的Greybox Fuzzer，透過不斷對input進行Mutate，從而提高Coverage(利用代碼插樁，來計算「相對」的Coverage，但需要Target的Source Code)，一般來說越高的Coverage，代表找到Bug的機率就越高。另外，雖然AFL也有提供Blackbox的qemu-mode，是可以不需要Source Code的，但效率就很低，所以發現漏洞的可能性也就相對來說比較小。

而對於AFL來講，雖然有很多衍生出來的工具，可以支援如Python(python-afl), Go(AFLGO), Rust(afl.rt)...等，但並沒有一款能支援Java的工具，因此如果當碰到是由Java開發的程式就無法進行Fuzz了，雖然Java開發的程式可能不太會有特別嚴重的漏洞，如在C中產生ABR(Beyond Array Bounds Read)、ABW(Beyond Array Bounds Read)的問題，可能會有Arbitrary Memory R&W的漏洞。然而，在Java裡，如果有上述的問題產生，則頂多可能只有觸發IndexOutOfBoundsException的事件發生，其嚴重程度的差異就差非常非常多。但在Java要有危險的漏洞也不是不可能，只是通常機率就沒那麼高，但這並不代表Java就沒有Fuzz的價值，例如可能會有以下這些有價值的問題:Memory Leak(無用的變數一直佔用Memory)、RuntimeException(編譯器不會檢查到該異常)、Deserialization(反序列化) Vulnerability...等。