All You Ever Wanted to Know About Dynamic Taint Analysis and Forward Symbolic Execution

موارد کاربر تحلیل آلایش پویا و اجرای نمادین پیش‌رو:

- تشخیص آسیب‌پذیری
- تحلیل دژافزار
- تولید موردآزمون

در این مقاله دو مفهوم تحلیل آلایش پویا و اجرای نمادین به صورت صوری تعریف شده‌اند. برای بیان صوری ابتدا یک زبان ساده (Simple Intermediate Language) همراه با syntax و sematic ارائه شده است. سپس معنای زبان را برای تحلیل آلایش یا اجرای نمادین تغییر می‌دهد.

برای تحلیل آلایش باید خط مشی مربوطه تعریف شود. یک خط مشی آلایش از سه پارامتر اصلی تشکیل می شود: ۱- چطور آلایش ایجاد می شود؟ ۲- چطور آلایش پخش می شود؟ ۳- چطور آلایش در طول اجرا بررسی می شود؟

با توجه به کاربرد هر کدام از سه سوال بالا باید تعریف شوند.

CRAXDroid

- برای برخورد با UI از فازر استفاده می‌کند. در این صورت نمی تواند ورودی‌های خوبی تولید کند.
- برای پوشش مسیرهای مختلف از روش پویا-نمادین استفاده می کند.
- ابزار روی برنامه های اندروید X86 کار می کند. (یعنی محیط واقعی نیست.) دلیل این کار این بوده است که نویسندگان مقاله می خواستند که از ابزار S2E استفاده کنند که ابزار کشف آسیب‌پذیری باینری در فضای پلتفرم PC هست و روی معماری x86 کار می کند.
- CRAXDroid در دو حالت جعبه سفید و سیاه کار می کند. در حالت جعبه سیاه فرض شده که فایل باینری x86 در اختیار هست. در حالت جعبه سفید هم با استفاده از JNI (Java Native Interface) کدهای سطح بالای جاوا به فایل های باینری سطح پایین نبدیل می شوند بعد عملیات آزمون انجام می شود.
- چون CRAXDroid روی S2E نوشته شده فقط می تواند آسیب‌پذیری‌های سرریز بافر را تشخیص دهد که در مورد برنامه های اندروید بسیار نادر است چون این اینگونه برنامه ها معمولا به زبان جاوا نوشته می شوند و امکان وقوع این نوع آسیب پذیری در آن وجود ندارد مگر این که برنامه به زبان C نوشته شود که فراوانی آنها به نسبت بسیار کمتر است.

- در قسمت آزمون مقاله در روش جعبه سفید و تشخیص آسیب‌پذیری نویسنده خود یک برنامه دارای خطا را آزموده است. در مورد آزمون نشت اطلاعات هم ادعا کرده که روی برنامه های بازار گوگل آزمون را انجام داده است ولی نه تعداد کل برنامه ها را عنوان کرده و نه تعداد برنامه های خطا دار یا سالم!!

2016, Automatically Discovering, Reporting and Reproducing Android Application Crashes

هدف از این مقاله کشف Crashهای موجود در برنامه های اندروید هست. ابتدا این مقاله ابزارهای این حوزه را در چهار دسته کلی قرار داده است:

- تولید ورودی به صورت دلخواه: ابزارهای فازر مثل Monkey
- تولید ورودی به صورت سیستماتیک: ACTEve ابزارهایی که از اجرای نمادین استفاده می کنند.
- تولید ورودی بر اساس مدل UI استخراج شده از برنامه
- روش های ترکیبی بالا

ابزارهای مختلف با هم مقایسه شده اند و گفته شده که هیچ کدام از ابزارها گزارش به زبان طبیعی تولید نمی کنند همچنین گزارشی تولید نمی کنند که به وسیله آن بتوان محل وجود خطا را به توسعه دهنده نشان داد یا کدی که به وسیله آن بتوان دوباره خطا را تولید کرد. همچنین گفته شده که ابزارهای حاضر در عمل خوب کار نمی کنند.

CrashScope سعی دارد روشی ارائه دهد که به وسیله آن بتوان به صورت خودکار یک گزارش خوب برای مهندسین نرم افزار تولید کند.

این کار چون از روش اجرای نمادین استفاده نمی کند از عنوان پروژه ارشد دور هست. ولی ایده های پیاده سازی آن و محک‌های آزمون استفاده شده آن مفید خواهد بود.