Decker: Attack Surface Reduction via On-Demand Code Mapping

Chris Porter, Sharjeel Khan, Santosh Pande

ASPLOS 2023
Reduction code surface as a defense

- Modern applications are bloated with unused functionality
- Novel code reuse attacks leverage this bloated code
- Recent code reduction techniques are typically
  - Sound but conservative (too much attack surface)
  - Aggressive but unsound (can lead to crashes)
- In this work, we build a prototype called Decker that takes a constructive approach to attack surface reduction
Example

A deck

Image source: Wikipedia (deck)
Example

```
main()
foo()
bar()
```
Example

```
main()
...
if(x)
   foo()
   bar()
foo()
...
bar()
...
```
Example

Program counter

```plaintext
main()
...
if(x)
  foo()
  bar()

foo()
...

bar()
...
```
Example

main()

... 

if (x)

foo()

bar()

foo()

...

bar()

...

CREATING THE NEXT®
Example

```
main()
...
if(x)
  foo()
  bar()
foo()
...
bar()
...
```
Example

```
main()
...
if(x)
  foo()
  bar()
foo()
...
bar()
...
```
Example

```
main()
...
if (x)
  foo()
  bar()
foo()
...  
bar()
...  
```
Example

Code reuse attack components: 🐜🐜🐞 ✓

main()

foo()

bar()
Example

Code reuse attack components:
Thank you!

- Please come to our talk for more details!
- Static compiler and dynamic runtime techniques
- Technique is **sound**
- nginx case study:
  - Windows and Linux support
  - Breaks real-world attacks
  - Large attack surface reduction (~80%)
  - Performant (~2% overhead)