

The background features several large, overlapping, semi-transparent swirls in shades of purple, green, and blue. Scattered throughout are numerous small, yellow, triangular shapes, some pointing towards the center and others pointing outwards, creating a dynamic and energetic feel.

Chaos for a Fast, Secure, and Predictable Future

John Criswell and Vikram Adve
University of Illinois at Urbana-Champaign



Software is in Danger!

- Attackers exploit code for fun/profit!
- Buffer overflows often violate:
 - Call Graph
 - Points-to Graph
 - Reaching Definitions Analysis
- Other attacks violate:
 - Type Safety
 - Information Flow Policies



The Secret Life of Run-time Checks

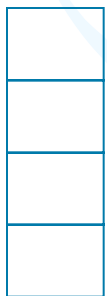
- Attacks thwarted by run-time checks
- Challenges
 - Performance
 - Physical memory usage
 - Virtual address space usage
 - Concurrency Overhead



Predictably Unpredictable!

- Encrypt data between registers and memory
 - Stores encrypt
 - Loads decrypt
- Using a single key is boring and useless!
- Assign keys to stores/loads based on semantic property
- Violations: Predictably unpredictable!

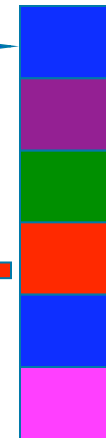
Registers:



Encrypt/Decrypt:



Memory Locations

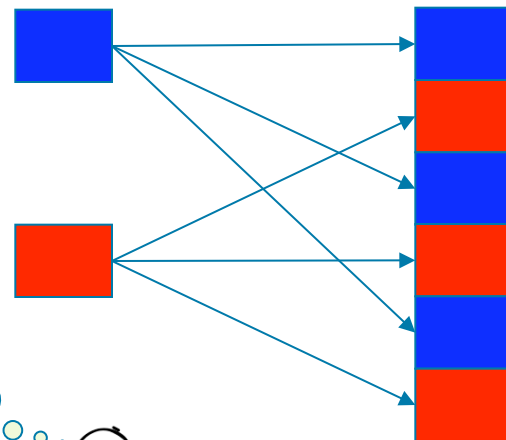


Points-To Analysis^{1,2}

- Input: points-to graph
- Assign distinct key for each points-to set
- Use fast XOR keys
- Stores encrypt
- Loads decrypt

Pointers:

Memory Objects:



Violation of points-to analysis yields unpredictable result!

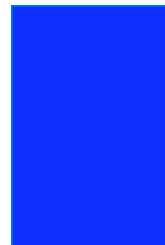
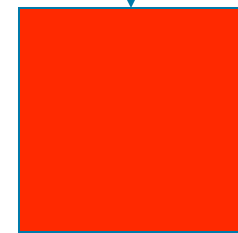
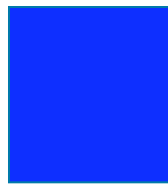
1. Bhatkar and Sekar, DIMVA 2008
2. Cadar et. al., TechReport 2008

Secure Information Flow

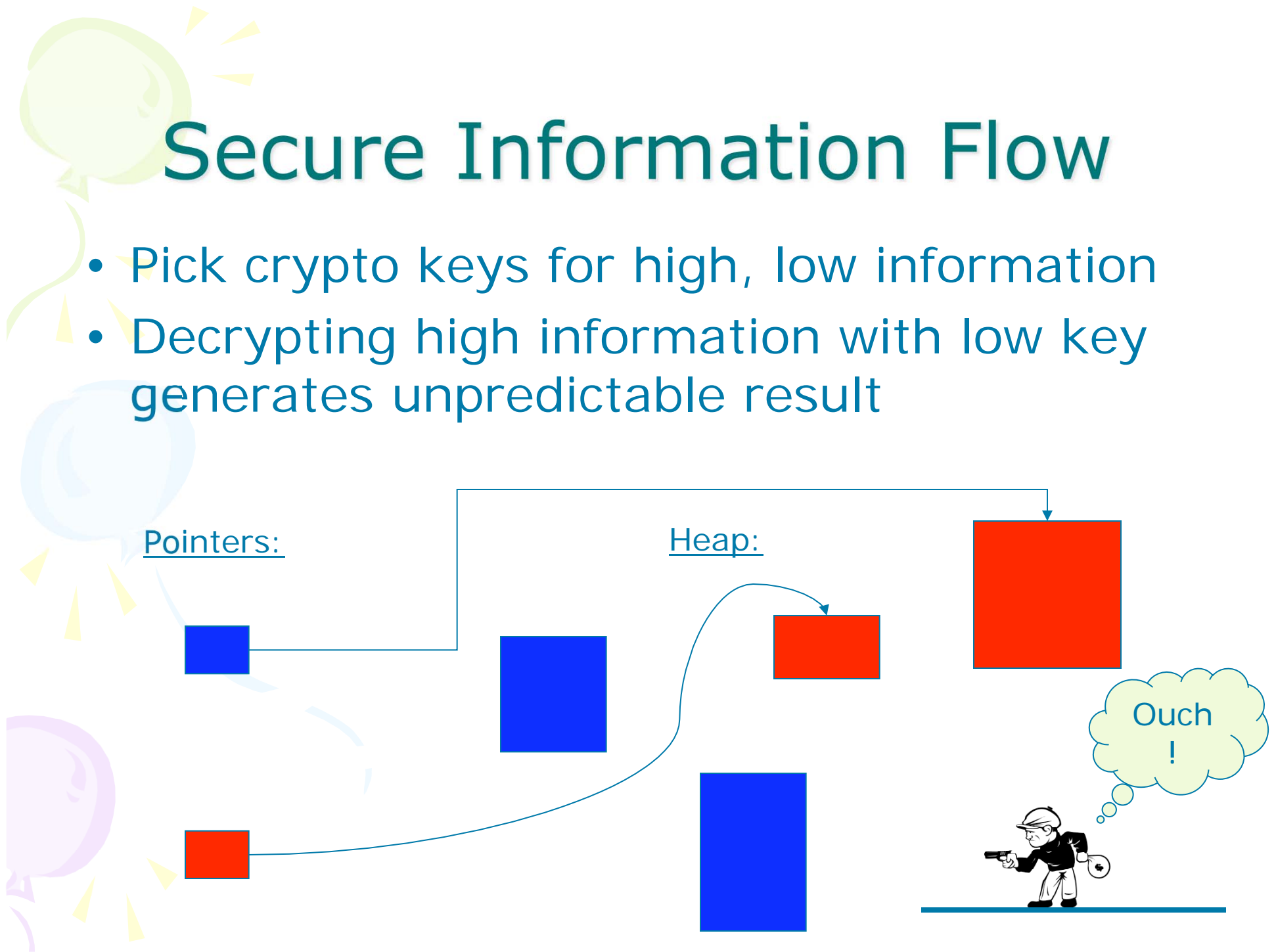
- Pick crypto keys for high, low information
- Decrypting high information with low key generates unpredictable result

Pointers:

Heap:

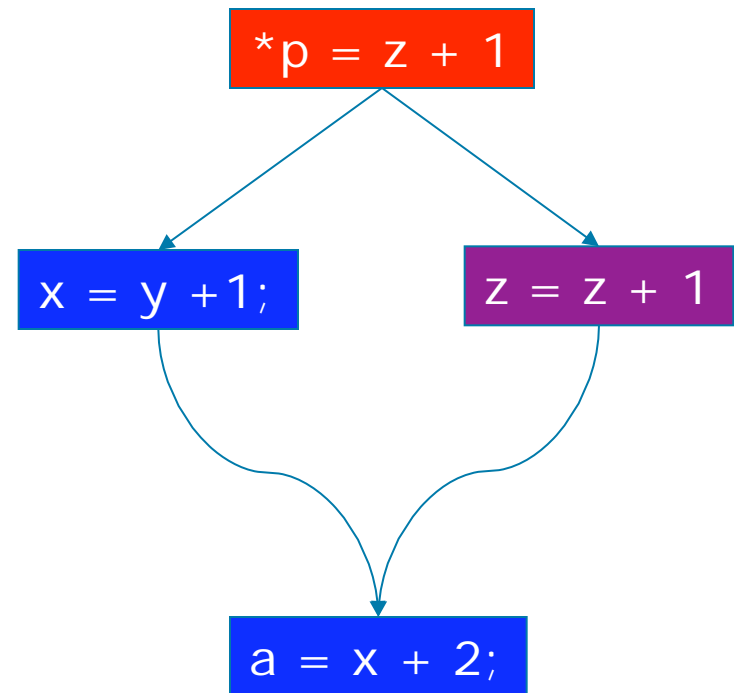


Ouch !



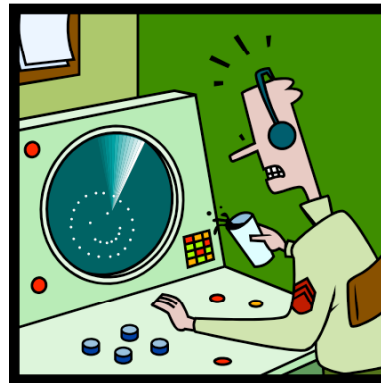
Reaching Definitions

- Merge sets of overlapping reaching definitions
- Each set gets a unique crypto key
- Violation causes unpredictable behavior



Challenges

- Buffer overreads¹ can leak key data
- Small values can be brute-forced^{1,2}
- XOR encryption not sufficient



1. [Strackx et. al., EuroSec 2009]
2. [Shacham et. al., CCS 2004]



Parting Thoughts

- Are there semantic safety properties that could be protected using chaos?

- 
- Control flow integrity
 - Type-safety
 - Thread-privacy
 - Data hiding
 - Variable scopes
 - Dangling stack pointers
- 

Extras!



Type-Safety

- Create a crypto key for each field
- Create a crypto key for stack
- Type-unsafe accesses generate unpredictable behavior

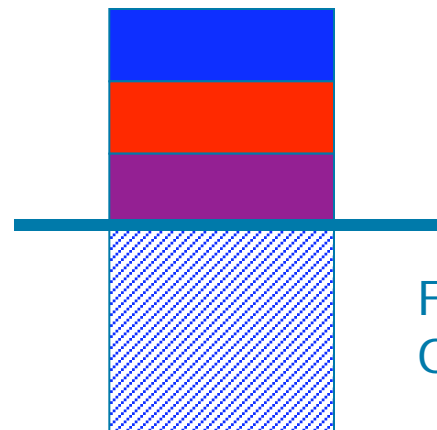
Subclass with Fields:



Pointers:



Super Class with Fields:



Fields from
Other object