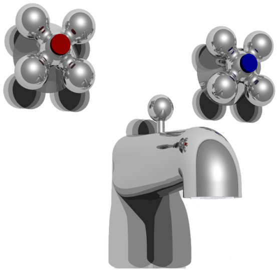
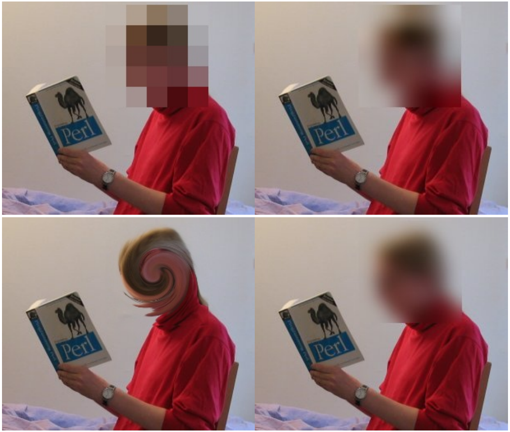


Techniques for Quantitative Information-flow Measurement

Stephen McCamant, James Newsome, Michael D. Ernst, and Dawn Song

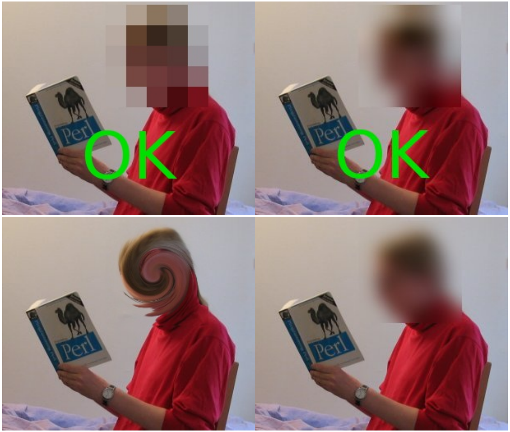
UC Berkeley, CMU, and MIT

Example: image transformation



Policy: reveal at most 3k (1%) of information about my face.

Example: image transformation



Policy: reveal at most 3k (1%) of information about my face. Our tool measures: top two satisfy policy

Goal: information security

- Confidentiality or integrity policy: keep secret data in or malicious data out
- Information flow: account for all influences through a program, not just direct copying
- How much information flows?
- Number of bits gives a mathematical limit on inferences or attacker influence

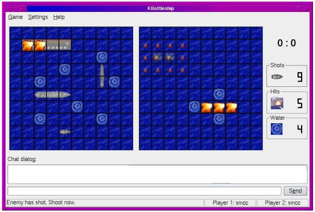
Example 2: attacking a network server

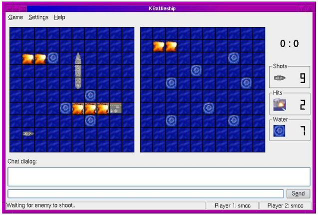
- A server is influenced by clients:
  A. Good clients request one of several legal operations
  B. Bad clients might force the server to jump to an attacker-chosen address
- Goal: reliably distinguish A from B (avoid false negatives and false positives)

Example 2: attacking a network server

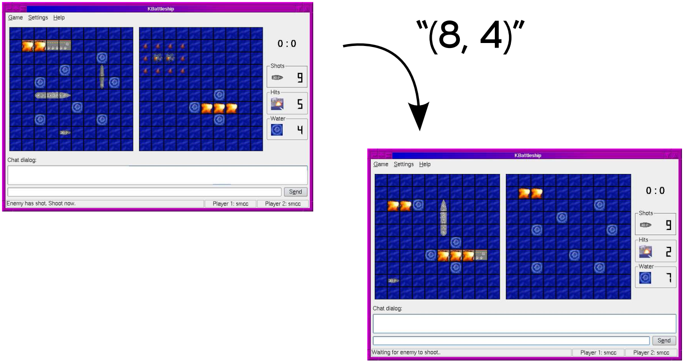
- A server is influenced by clients:
  A. Good clients request one of several legal operations
  B. Bad clients might force the server to jump to an attacker-chosen address
- Goal: reliably distinguish A from B (avoid false negatives and false positives)
- Influence is 3.3 bits in A (benign), 32 bits in B (exploitable)

Example 3: Battleship game

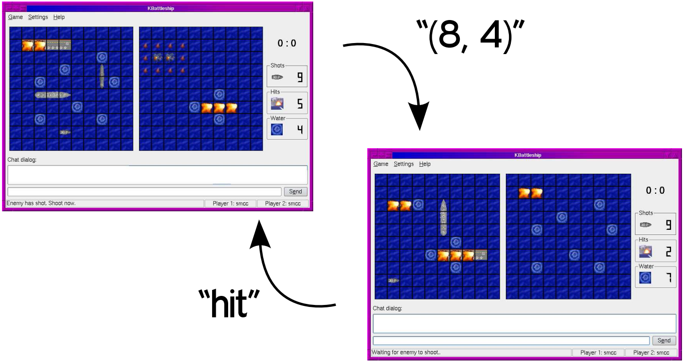




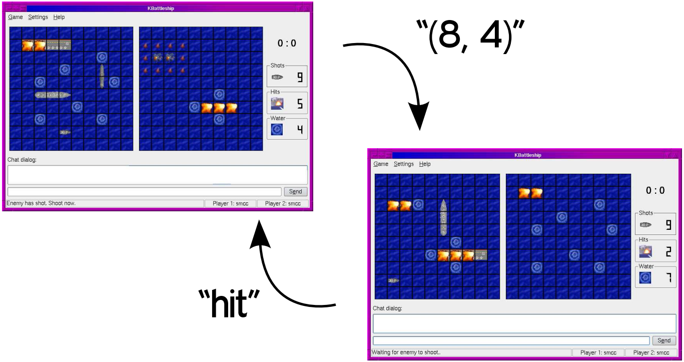
Example 3: Battleship game



Example 3: Battleship game



Example 3: Battleship game



Want to minimize ship location information revealed

Outline

Introduction to information flow

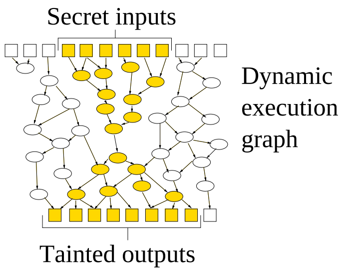
Upper bounds via maximum flow

Lower bounds via a decision procedure

Case studies

Conclusion

Start with: tainting



- Track which values might be transitively influenced by secret inputs
- In other words, graph reachability

Challenge 1: implicit flows

```
if (age > 50)
    salary = salary + 10000;
```

- Indirect influence via control flow, array indexes, and pointers
- Solution: annotations that bound the side-effects of secret-dependent code
  - Added by hand or via automatic analysis

Challenge 2: tainting imprecision

- Many pieces of tainted data may carry the same information
  - Copying multiplies taint but not information
- Solution: model information as a finite substance, compute maximum flow
- Graph algorithms with program-sensitive compression for efficiency

Implementation: Flowcheck

- Based on the Valgrind dynamic analysis framework
- For x86/Linux binaries (scales to: X server, KDE apps)
- GPLed and available for download: `http://people.csail.mit.edu/smcc/projects/secret-flow/`

Outline

Introduction to information flow

Upper bounds via maximum flow

Lower bounds via a decision procedure

Case studies

Conclusion

A complementary approach

- Flowcheck scales well, but gives no guarantee about precision: upper bound might be too conservative
- Alternative approach: verify specific possible outputs
- Can give lower bounds and approximations with statistically bounded error

Decision procedure approach

- Convert program or trace into logical formula giving output in terms of inputs
- Give formula to decision procedure to determine which outputs can be produced
- Can find smallest or largest possible output, enumerate examples, or check random sample outputs

Decision procedure implementation

- Built using BitBlaze infrastructure: TEMU whole-system dynamic tracing, Vine instruction-level static analysis
- Used STP bitvector decision procedure
- Works with COTS binary applications and severs, on both Windows and Linux

Outline

Introduction to information flow

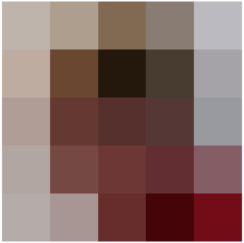
Upper bounds via maximum flow

Lower bounds via a decision procedure

Case studies

Conclusion

Image transformation #1



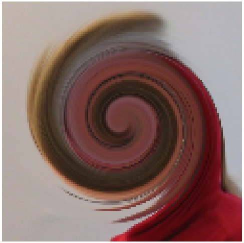
`me.ppm`: 375120 bits

$(125 \cdot 125 \cdot 24 + 120)$

```
% convert me.ppm \
    -resize 5x5 \
    -sample 125x125
1464 bits leaked
```

$(5 \cdot 5 \cdot 48 + 264)$

Image transformation #2



`me.ppm`: 375120 **bits**

$(125 \cdot 125 \cdot 24 + 120)$

```
% convert me.ppm \
   -swirl 720
```

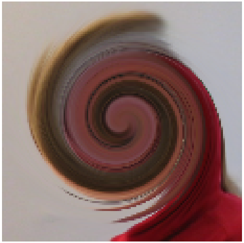
`375120 bits leaked`

(= file size)

Image transformation #2

me.ppm: 375120 bits

$(125 \cdot 125 \cdot 24 + 120)$



```
% convert me.ppm \
    -swirl 720 \
    swirl.ppm
375120 bits leaked
```

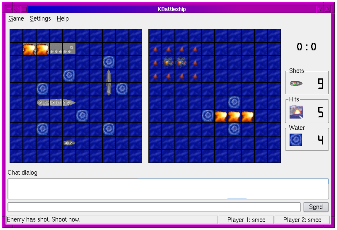
(= file size)

```
% convert swirl.ppm \
  -swirl -720
```

Attacks on network servers

- Samba file server uses network input to choose a function pointer
  - Leads to false positives in previous tainting systems
  - Our tool measures the exact influence: $\log_2 10 = 3.3$ bits
- Another jump pointer in Microsoft DCOM server can be influenced by network input
  - Our tool measures influence of $[27.5, 32.0]$ bits
  - True positive: vulnerability exploited by the Blaster worm

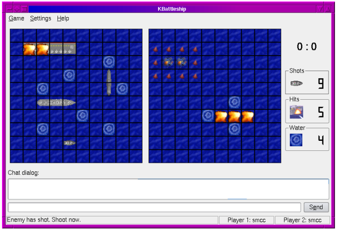
Running KBattleship




```
% kbattleship
0 bits leaked
...
8 bits leaked
...
16 bits leaked
...
24 bits leaked
```

Eight bits per round seems like too much...

Running KBattleship




```
% kbattleship
0 bits leaked
...
8 bits leaked
...
16 bits leaked
...
24 bits leaked
```

Eight bits per round seems like too much. . .

Previously unknown bug: protocol includes type of ship on non-fatal hit.

Outline

- Introduction to information flow
- Upper bounds via maximum flow
- Lower bounds via a decision procedure
- Case studies
- **Conclusion**

Summary

- Quantitative information-flow policies allow for precise distinctions
- Instruction-level analysis can give accurate measurements for real software
- New techniques:
  - Upper bounds using maximum network flow
  - Lower bounds using a decision procedure

Thank you

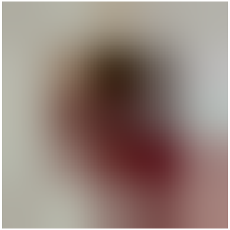
Enclosure annotations

- Enclosure leverages static information to bound behavior of alternate executions
- Similar: RIFLE [VBC+'04], [MPL'04], Trishul [NSCT'07], [CF'08], etc.
- Sufficient for soundness: one big region around whole program
- For precision: infer via static analysis, or annotate by hand

Enclosure region details

- Annotations written in source, appear at machine level
- Cause flow edges from branch conditions to region outputs
- Most annotations can be found with a simple analysis
- Uncommon, easy to add by hand (average 10/program)

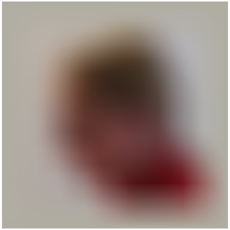
Blur details



ImageMagick -resize
5x5 Interpolation

(Hand) lower bound:
600 bits
Upper bound:
1720 bits

ImageMagick -blur
Gaussian kernel
convolution
(Hand) lower bound:
3456 bits
Upper bound:
375120 bits



Custom blur
Square kernel
convolution
(Hand) lower bound:
375120 bits
Upper bound:
375120 bits

KBattleship bug

```
void KBattleshipWindow::
     slotSendEnemyFieldState(int fieldx,
                             int fieldy)
{
    /* ... */
    data = m_ownshiplist->
             shipTypeAt(fieldx, fieldy);
    /* ... */
    msg->addField(QString("fieldstate"),
                  QString::number(data));
    /* ... */
}
```