three applications of model finding

Daniel Jackson · Tel Aviv, March 7, 2012
based on work with
Eunsuk Kang, Aleks Milicevic & Joe Near

CSAIL
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

model finding

finding a graph coloring

condition on adjacency and coloring:

```
all a, b | a->b in adj implies a.color != b.color
```

free variable

free variable

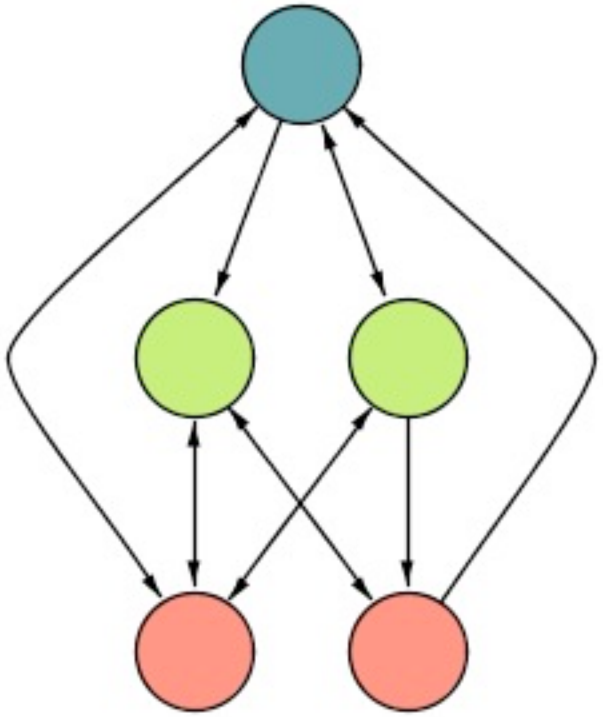
or, equivalently:

```
no adj.color & color
```

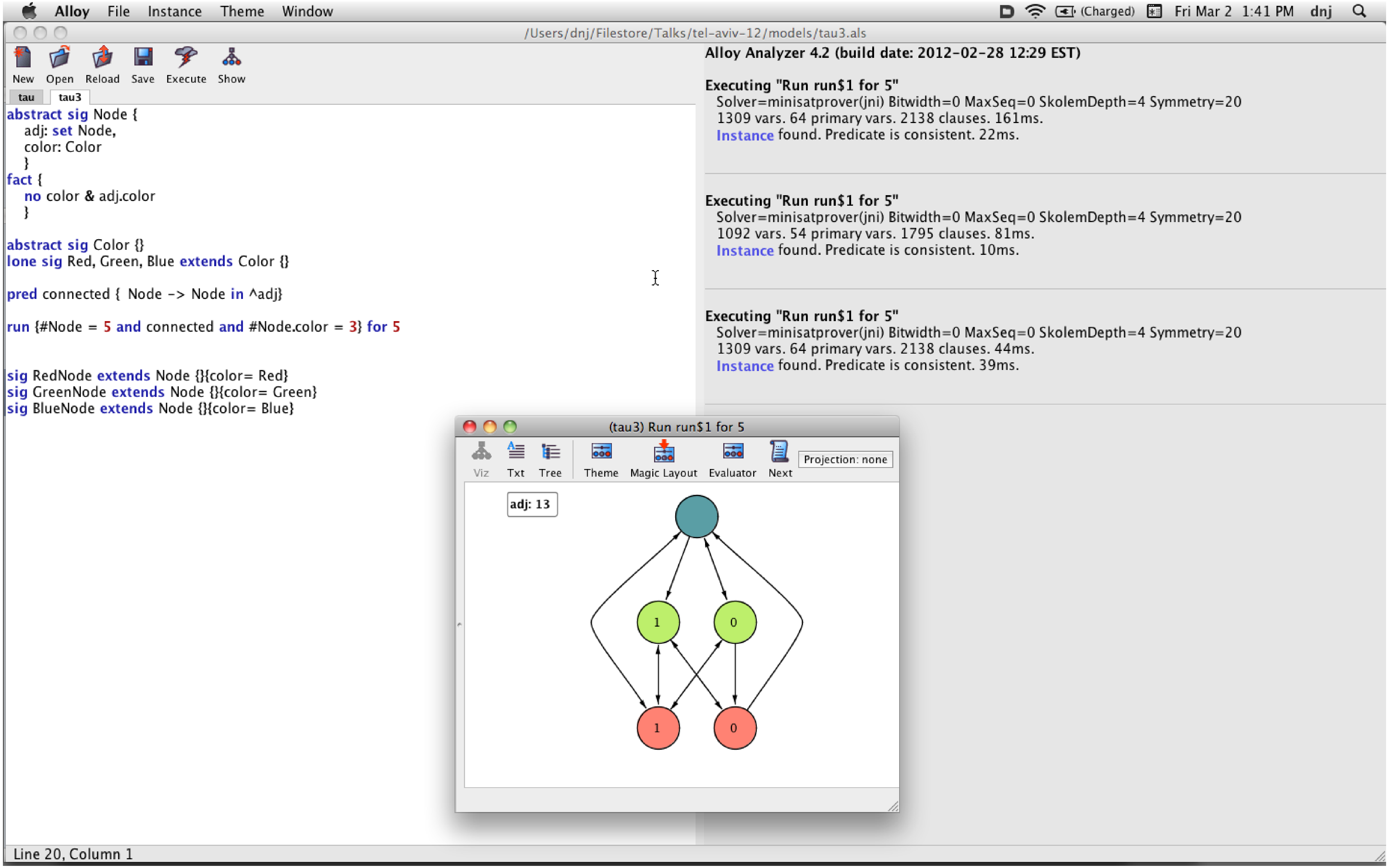
formalizing types:

```
some
  disj Node, Color: set univ,
  adj: Node -> Node,
  color: Node -> one Color |
    no adj.color & color
```

an instance:



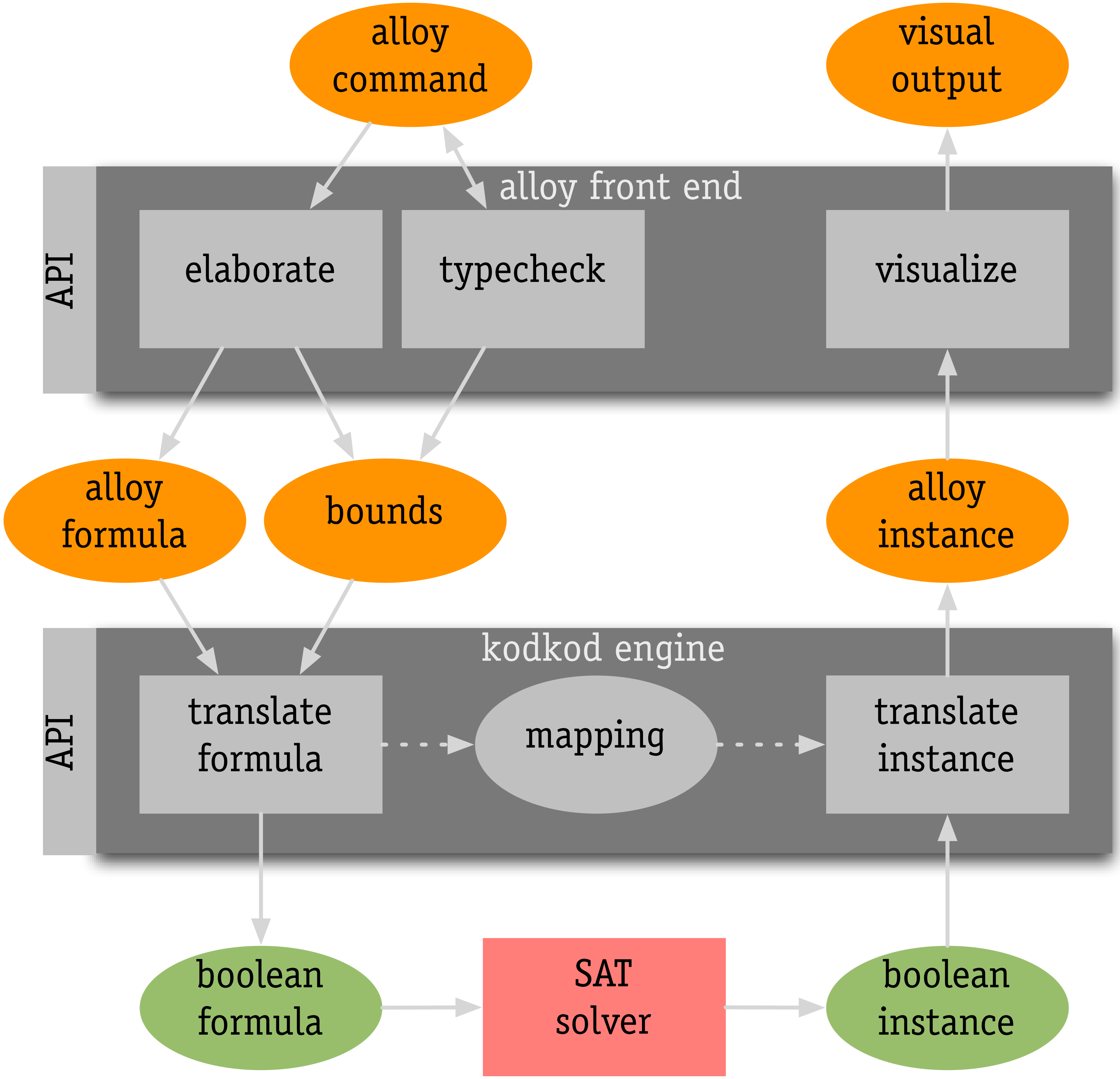
alloy analyzer


Alloy File Instance Theme Window
(Charged) Fri Mar 2 1:41 PM dnj
/Users/dnj/Filestore/Talks/tel-aviv-12/models/tau3.als
New Open Reload Save Execute Show
tau tau3
(tau3) Run run$1 for 5
Viz Txt Tree Theme Magic Layout Evaluator Next
Projection: none
adj: 13
1
0
1
0
Line 20, Column 1

```
abstract sig Node {
   adj: set Node,
   color: Color
   }
fact {
   no color & adj.color
   }

abstract sig Color {}
lone sig Red, Green, Blue extends Color {}

pred connected { Node -> Node in ^adj}

run {#Node = 5 and connected and #Node.color = 3} for 5


sig RedNode extends Node {}{color= Red}
sig GreenNode extends Node {}{color= Green}
sig BlueNode extends Node {}{color= Blue}
```

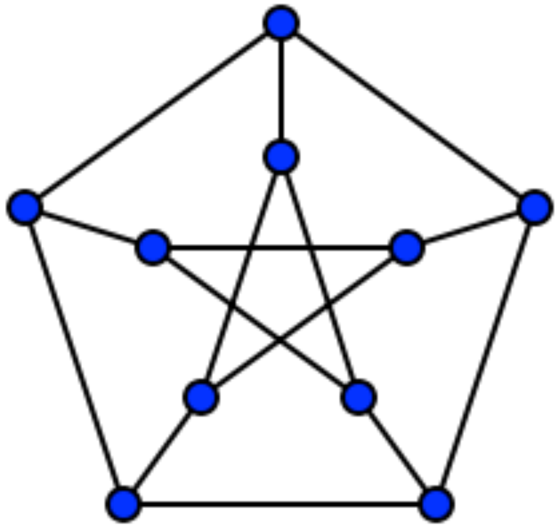
```
Alloy Analyzer 4.2 (build date: 2012-02-28 12:29 EST)

Executing "Run run$1 for 5"
   Solver=minisatprover(jni) Bitwidth=0 MaxSeq=0 SkolemDepth=4 Symmetry=20
   1309 vars. 64 primary vars. 2138 clauses. 161ms.
   Instance found. Predicate is consistent. 22ms.

Executing "Run run$1 for 5"
   Solver=minisatprover(jni) Bitwidth=0 MaxSeq=0 SkolemDepth=4 Symmetry=20
   1092 vars. 54 primary vars. 1795 clauses. 81ms.
   Instance found. Predicate is consistent. 10ms.

Executing "Run run$1 for 5"
   Solver=minisatprover(jni) Bitwidth=0 MaxSeq=0 SkolemDepth=4 Symmetry=20
   1309 vars. 64 primary vars. 2138 clauses. 44ms.
   Instance found. Predicate is consistent. 39ms.
```

how alloy works

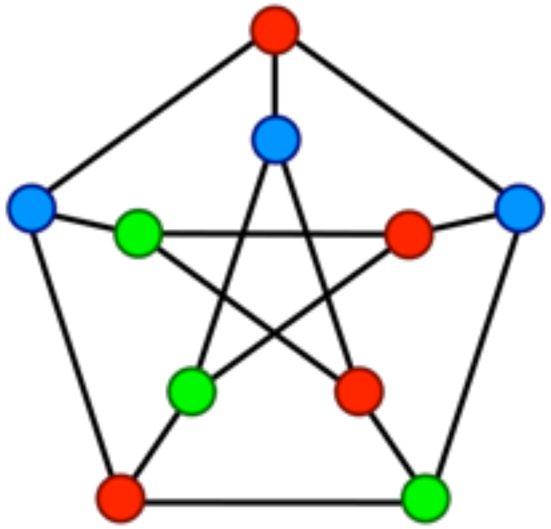


partial instances

Petersen graph



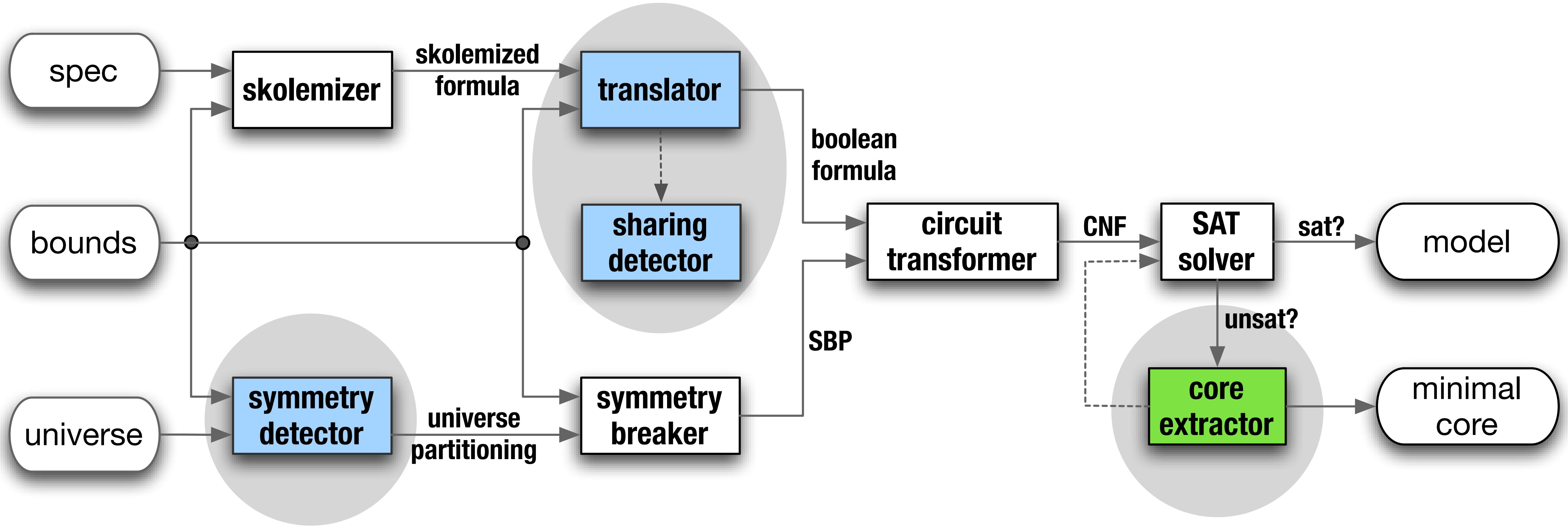
a coloring



	n0	n1	n2	n3	
n0	0	1	0	0	…
n1	0	0	1	0	
n2	0	0	0	1	
n3	1	0	0	0	

	c0	c1	c2	c3	
n0	c00	c01	c02	c03	…
n1	c10	c11	c12	c13	
n2	c20	c21	c22	c23	
n3	c30	c31	c32	c33	

kodkod architecture



🟦 [Torlak07]

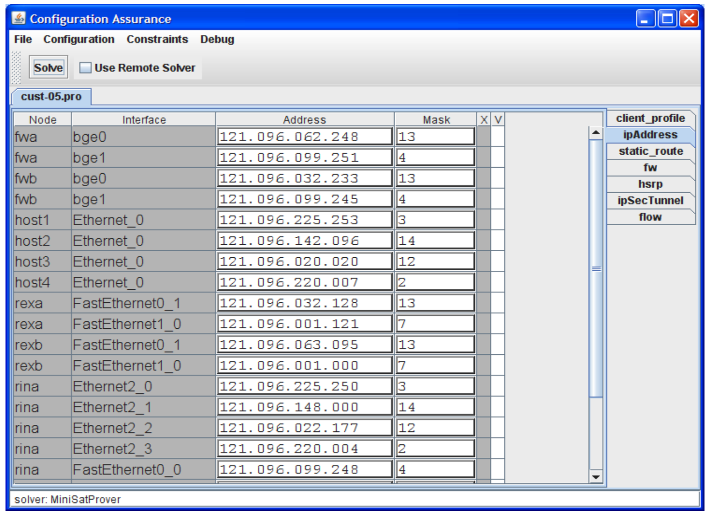
🟩 [Torlak08]

some applications of model finding

checking theorems
find a refutation
eg, Nitpick for Isabelle/HOL

software update
find packages to install
eg, Eclipse's Equinox P2

configuring networks
find router settings
eg, Telcordia's ConfigAssure



Node	Interface	Address	Mask	X	V
fwa	bge0	121.096.062.248	13		
fwa	bge1	121.096.099.251	4		
fwb	bge0	121.096.032.233	13		
fwb	bge1	121.096.099.245	4		
host1	Ethernet_0	121.096.225.253	3		
host2	Ethernet_0	121.096.142.096	14		
host3	Ethernet_0	121.096.020.020	12		
host4	Ethernet_0	121.096.220.007	2		
rexa	FastEthernet0_1	121.096.032.128	13		
rexa	FastEthernet1_0	121.096.001.121	7		
rexb	FastEthernet0_1	121.096.063.095	13		
rexb	FastEthernet1_0	121.096.001.000	7		
rina	Ethernet2_0	121.096.225.250	3		
rina	Ethernet2_1	121.096.148.000	14		
rina	Ethernet2_2	121.096.022.177	12		
rina	Ethernet2_3	121.096.220.004	2		
rina	FastEthernet0_0	121.096.099.248	4		

solver: MiniSatProver

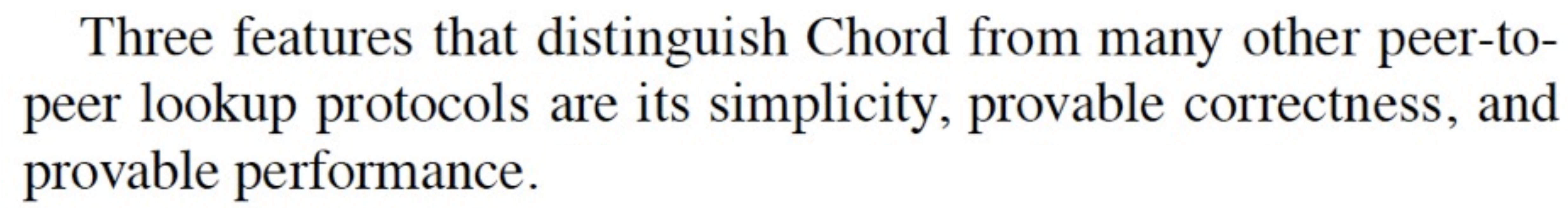
why alloy/kodkod?

	Kodkod	IDP1.3	Paradox2.3	DarwinFM	Mace4
language					
first order logic	◆	◆	◆	◆	◆
relational algebra	◆	◇	◇	◇	◇
partial models	◆	◆	◇	◇	◇
inductive definitions	⬙	◆	◇	◇	◇
types	◆	◆	◇	◇	◇
bitvector arithmetic	◆	⬙	◇	◇	◇
model finding					
partial models	◆	◆	⬙	⬙	⬙
inductive definitions	⬙	◆	◇	◇	◇
symmetry breaking	◆	◇	◆	◆	◆
high-arity relations	⬙	⬙	⬙	◆	⬙
nested quantifiers	⬙	⬙	◆	◆	◆
core extraction					
minimal core	◆	◇	◇	◇	◇

◆	**full support**
⬙	**partial support**
◇	**no support**

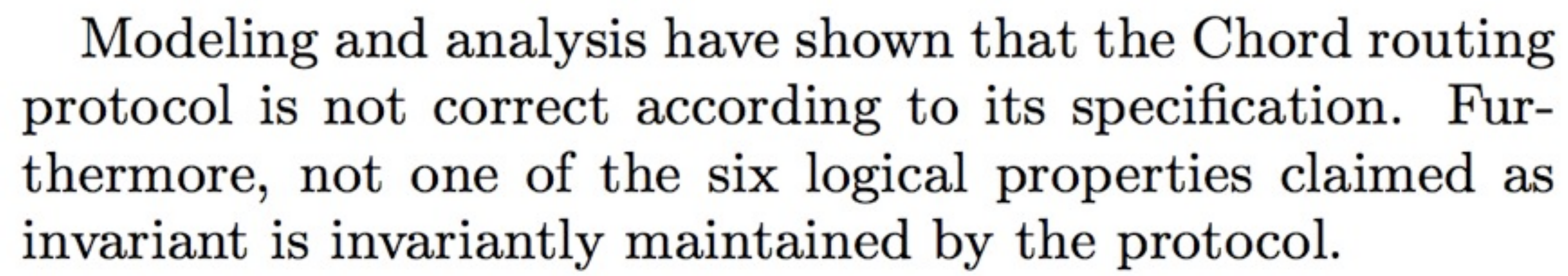
#0
design analysis

zave on chord



Three features that distinguish Chord from many other peer-to-peer lookup protocols are its simplicity, provable correctness, and provable performance.

Ion Stoica et al. Chord: A Scalable Peer to Peer Lookup Service for Internet Applications, SIGCOMM 2001 (also TON, 2003)



Modeling and analysis have shown that the Chord routing protocol is not correct according to its specification. Furthermore, not one of the six logical properties claimed as invariant is invariantly maintained by the protocol.

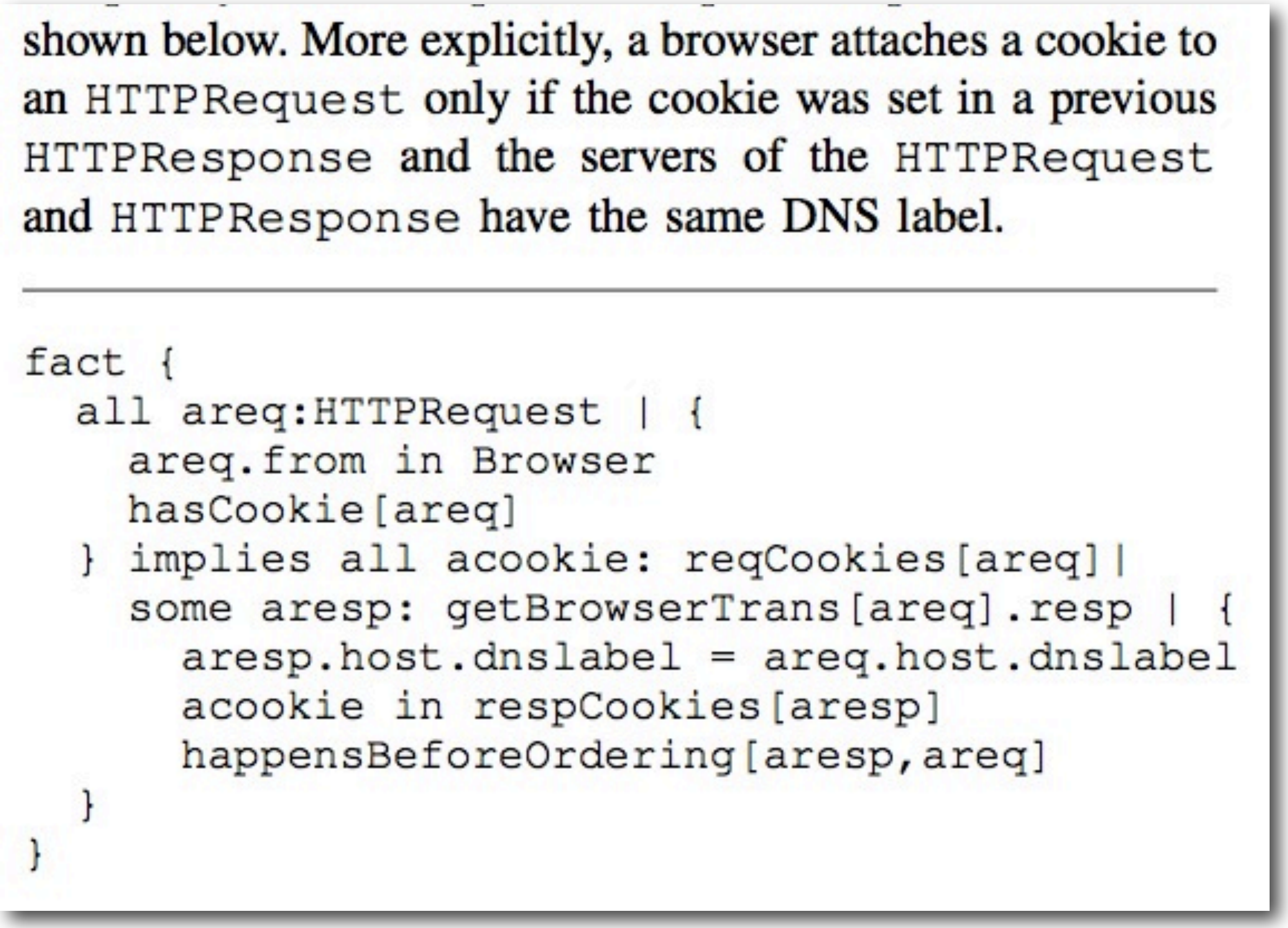
Pamela Zave. Invariant-Based Verification of Routing Protocols: The Case of Chord, 2009

akhawe+ on web security

generic model of web security

HTTP, certificates, cookies, script contexts

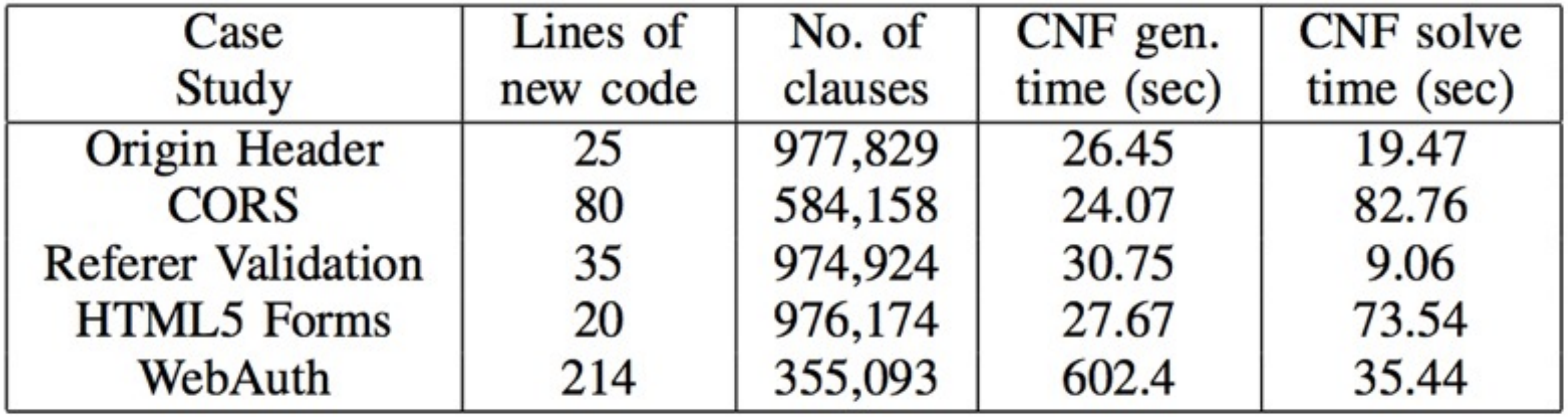
about 2,000 lines of Alloy



shown below. More explicitly, a browser attaches a cookie to an HTTPRequest only if the cookie was set in a previous HTTPResponse and the servers of the HTTPRequest and HTTPResponse have the same DNS label.

```
fact {
  all areq:HTTPRequest | {
    areq.from in Browser
    hasCookie[areq]
  } implies all acookie: reqCookies[areq]|
    some aresp: getBrowserTrans[areq].resp | {
      aresp.host.dnslabel = areq.host.dnslabel
      acookie in respCookies[aresp]
      happensBeforeOrdering[aresp,areq]
  }
}
```

results

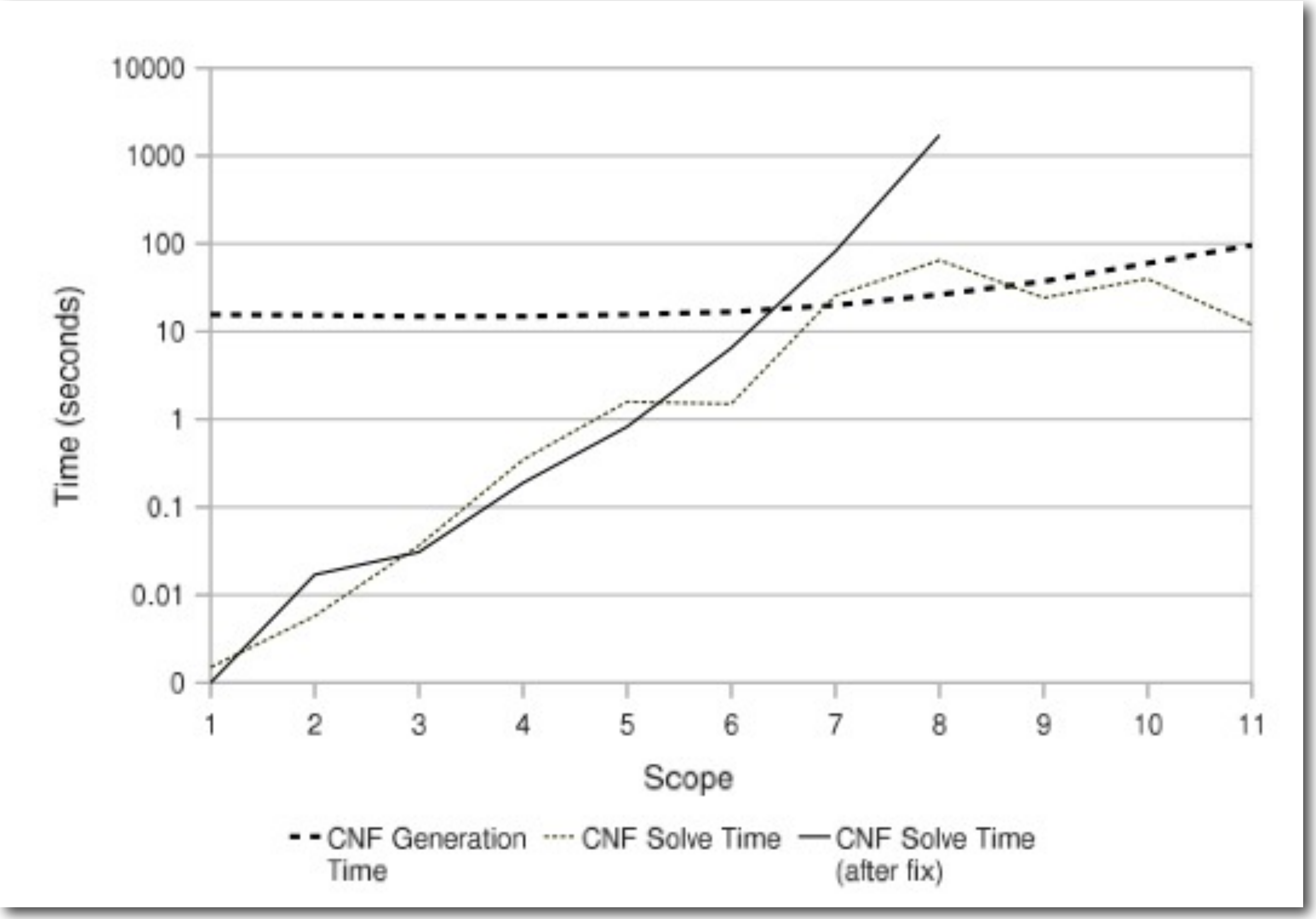


Case Study	Lines of new code	No. of clauses	CNF gen. time (sec)	CNF solve time (sec)
Origin Header	25	977,829	26.45	19.47
CORS	80	584,158	24.07	82.76
Referer Validation	35	974,924	30.75	9.06
HTML5 Forms	20	976,174	27.67	73.54
WebAuth	214	355,093	602.4	35.44

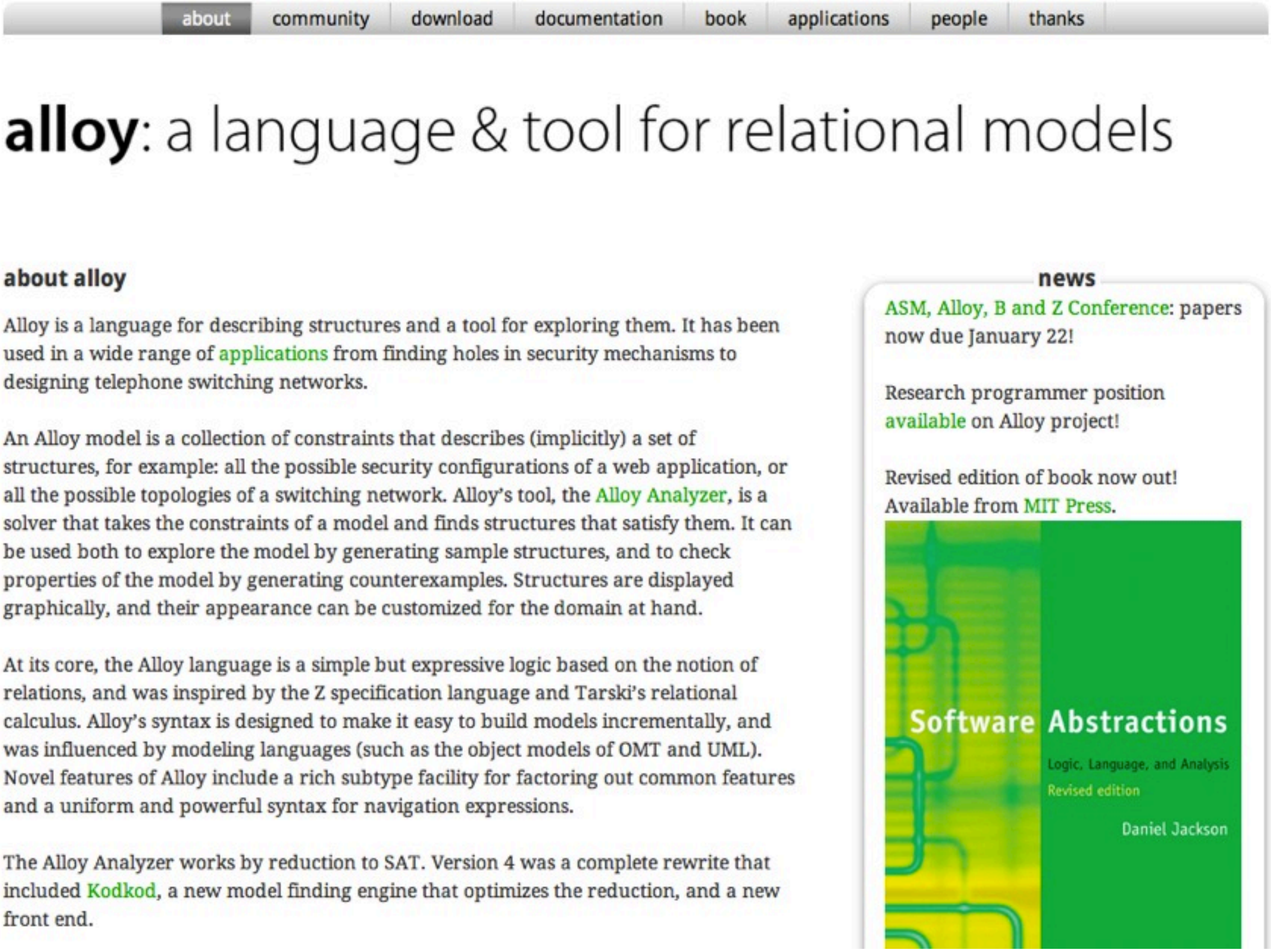
applied to 5 case studies
in each, found vulnerabilities
2 known, 3 unknown

sample vulnerability
referrer validation fails on redirects

falling over the cliff



more examples: alloy.mit.edu



alloy: a language & tool for relational models

about alloy

Alloy is a language for describing structures and a tool for exploring them. It has been used in a wide range of applications from finding holes in security mechanisms to designing telephone switching networks.

An Alloy model is a collection of constraints that describes (implicitly) a set of structures, for example: all the possible security configurations of a web application, or all the possible topologies of a switching network. Alloy's tool, the Alloy Analyzer, is a solver that takes the constraints of a model and finds structures that satisfy them. It can be used both to explore the model by generating sample structures, and to check properties of the model by generating counterexamples. Structures are displayed graphically, and their appearance can be customized for the domain at hand.

At its core, the Alloy language is a simple but expressive logic based on the notion of relations, and was inspired by the Z specification language and Tarski's relational calculus. Alloy's syntax is designed to make it easy to build models incrementally, and was influenced by modeling languages (such as the object models of OMT and UML). Novel features of Alloy include a rich subtype facility for factoring out common features and a uniform and powerful syntax for navigation expressions.

The Alloy Analyzer works by reduction to SAT. Version 4 was a complete rewrite that included Kodkod, a new model finding engine that optimizes the reduction, and a new front end.

news

ASM, Alloy, B and Z Conference: papers now due January 22!

Research programmer position available on Alloy project!

Revised edition of book now out! Available from MIT Press.

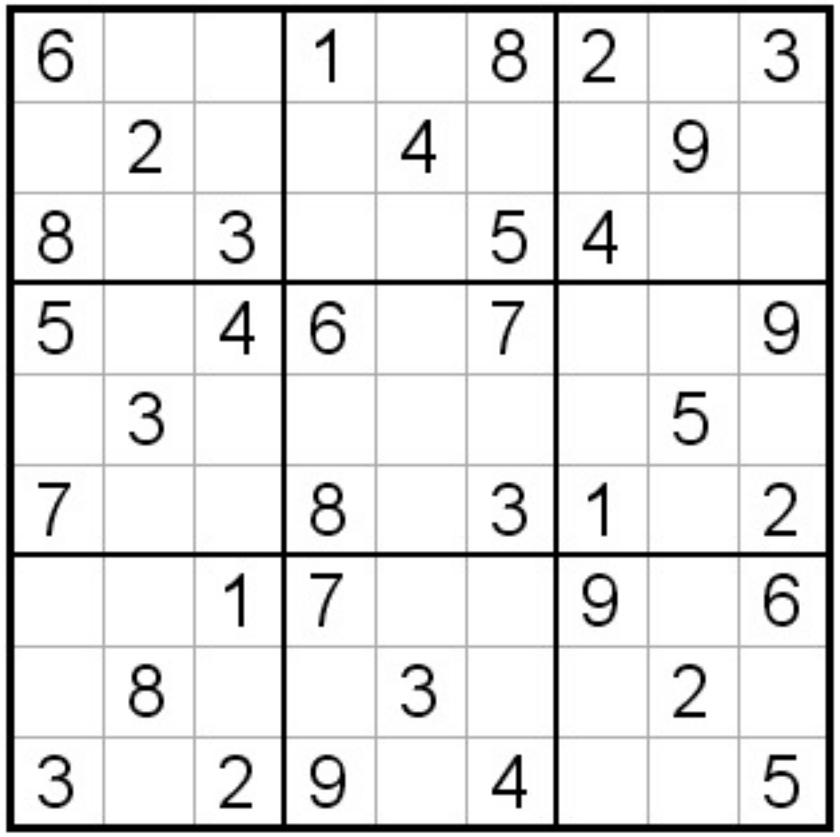
#1
declarative programming

work by Aleks Milicevic

sudoku

problem

fill in the empty cells so that
all rows, columns and squares contain 1..9



6			1		8	2		3
	2			4			9	
8		3			5	4		
5		4	6		7			9
	3						5	
7			8		3	1		2
		1	7			9		6
	8			3			2	
3		2	9		4			5

declaring the grid

```
public class Sudoku {
  private int [][] grid = new int [9][9];









public static void main(String[] args) {
    Sudoku s = new Sudoku();
    s.grid[0][3] = 1; ...; s.grid[8][8] = 5;


    }
```

6			1		8	2		3
	2			4			9	
8		3			5	4		
5		4	6		7			9
	3						5	
7			8		3	1		2
		1	7			9		6
	8			3			2	
3		2	9		4			5

specifying solve

```
public class Sudoku {
  private int [][] grid = new int [9][9];

@Ensures ({
      "all row in {0..8} | this.grid[row][int] = {1..9}",
      "all col in {0..8} | this.grid[int][col] = {1..9}",
      "all r , c in {0, 1, 2} |
        this.grid[{r∗3..r∗3+2}][{c∗3..c∗3+2] = {1..9}"
  })
@Modifies("this.grid[int].elems | _<2> = 0")
public void solve() {  ... }

public static void main(String[] args) {
   Sudoku s = new Sudoku();
   s.grid[0][3] = 1; ...; s.grid[8][8] = 5;
   s.solve( );

  }
```

6			1		8	2		3
	2			4			9	
8		3			5	4		
5		4	6		7			9
	3						5	
7			8		3	1		2
		1	7			9		6
	8			3			2	
3		2	9		4			5

implementing solve

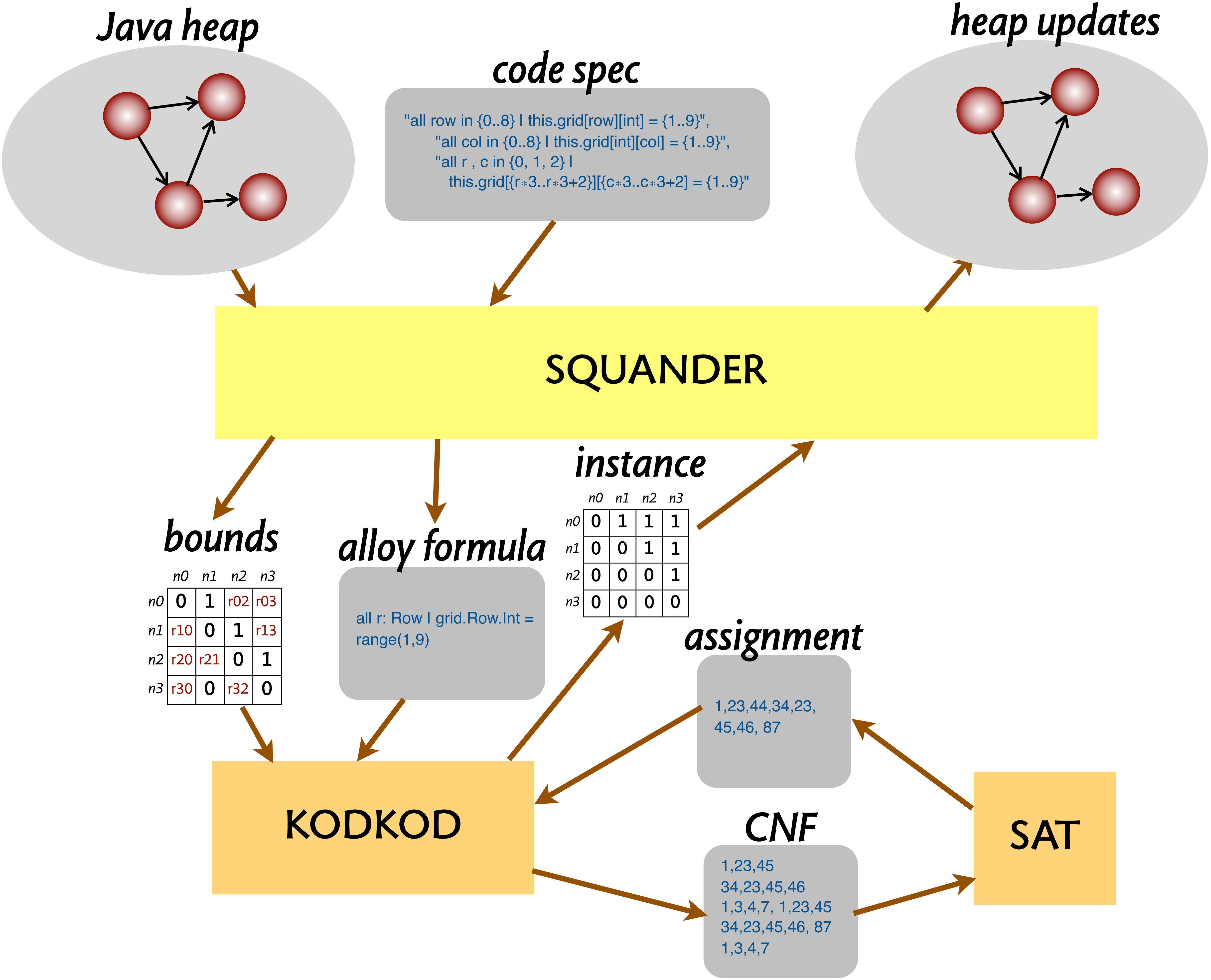
```
public class Sudoku {
  private int [][] grid = new int [9][9];

@Ensures ({
        "all row in {0..8} | this.grid[row][int] = {1..9}",
        "all col in {0..8} | this.grid[int][col] = {1..9}",
        "all r , c in {0, 1, 2} |
           this.grid[{r*3..r*3+2}][{c*3..c*3+2}] = {1..9}"
   })
@Modifies("this.grid[int].elems | _<2> = 0")
public void solve() { Squander.exe(this); }

public static void main(String[] args) {
   Sudoku s = new Sudoku();
   s.grid[0][3] = 1; ...; s.grid[8][8] = 5;
   s.solve( );

   }
```

6			1		8	2		3
	2			4			9	
8		3			5	4		
5		4	6		7			9
	3						5	
7			8		3	1		2
		1	7			9		6
	8			3			2	
3		2	9		4			5

printing the result

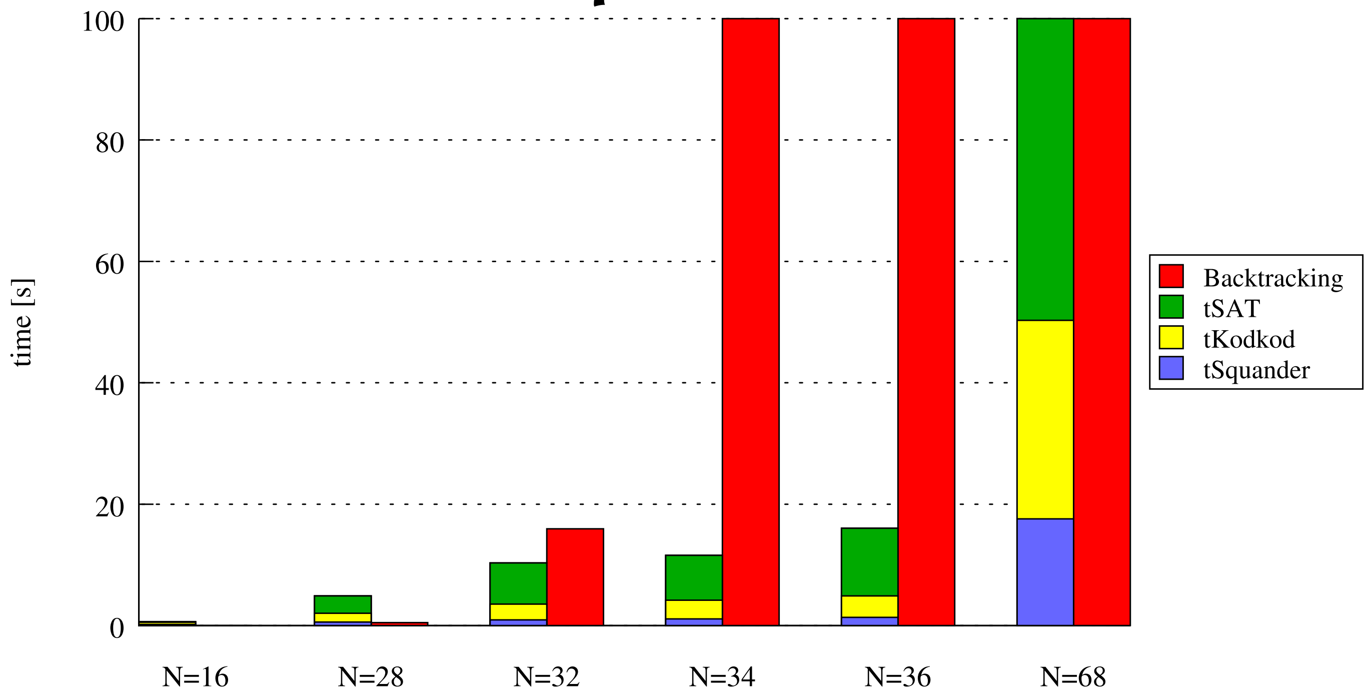
```
public class Sudoku {
  private int [][] grid = new int [9][9];

@Ensures ({
      "all row in {0..8} | this.grid[row][int] = {1..9}",
      "all col in {0..8} | this.grid[int][col] = {1..9}",
      "all r , c in {0, 1, 2} |
        this.grid[{r*3..r*3+2}][{c*3..c*3+2}] = {1..9}"
  })
@Modifies("this.grid[int].elems | _<2> = 0")
public void solve() { Squander.exe(this); }

public static void main(String[] args) {
   Sudoku s = new Sudoku();
   s.grid[0][3] = 1; ...; s.grid[8][8] = 5;
   s.solve( );
   System.out.println(s);
   }
```

6			1		8	2		3
	2			4			9	
8		3			5	4		
5		4	6		7			9
	3						5	
7			8		3	1		2
		1	7			9		6
	8			3			2	
3		2	9		4			5

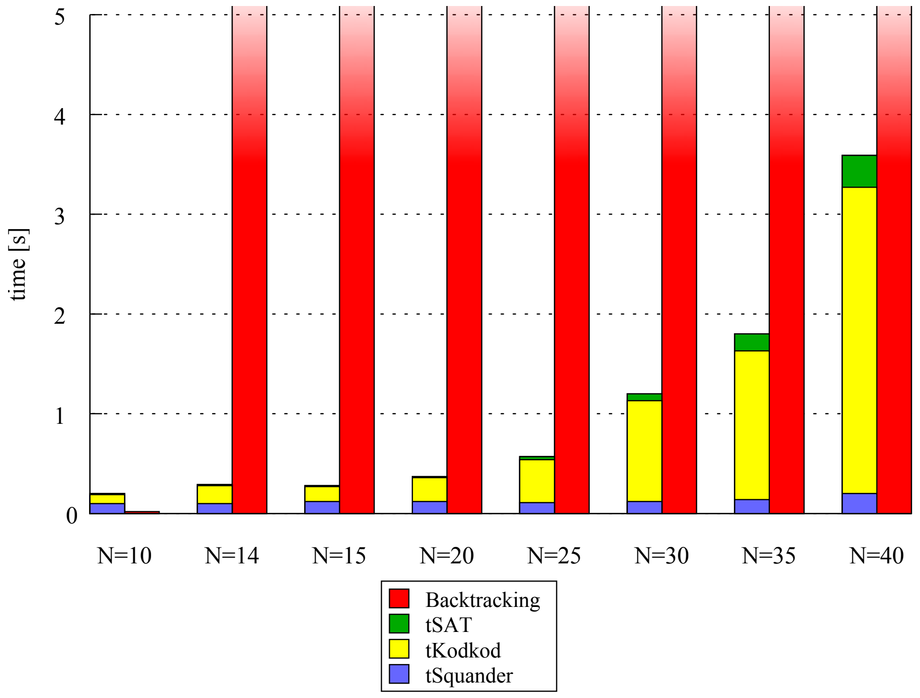


performance

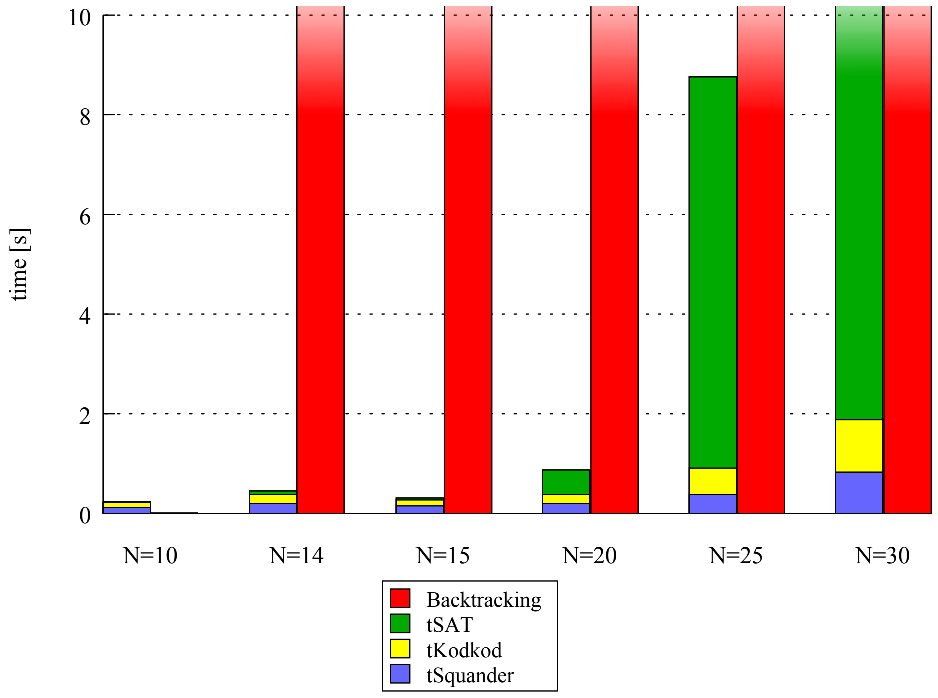
n-queens



hamiltonian path, none



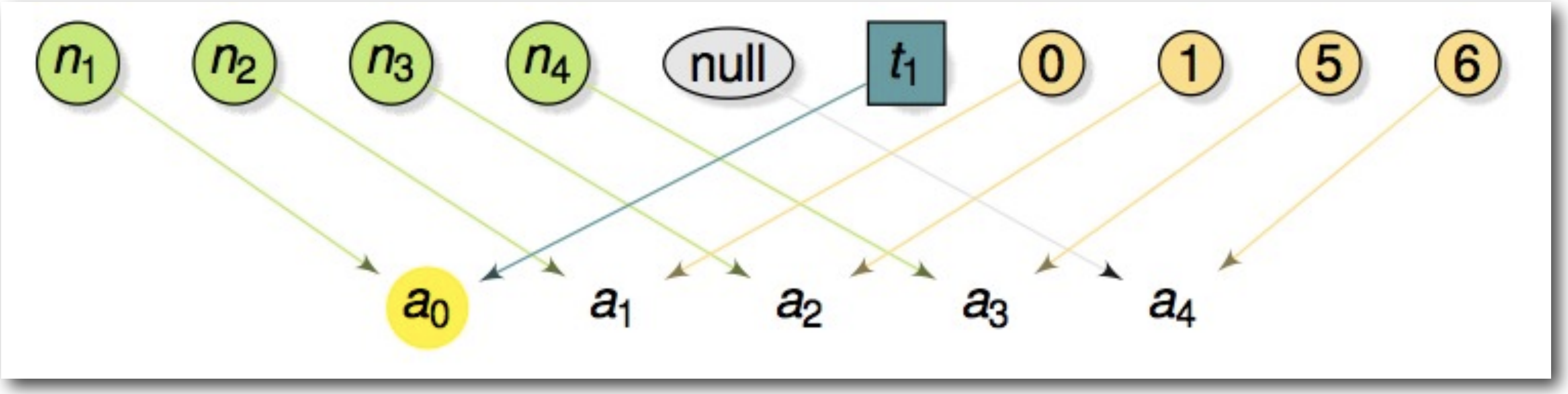
hamiltonian path, some



refinements

handling libraries
eg, Java collections
specs, spec fields, invariants

minimizing universe size
exploit type information in heap
map objects of different types to same atom



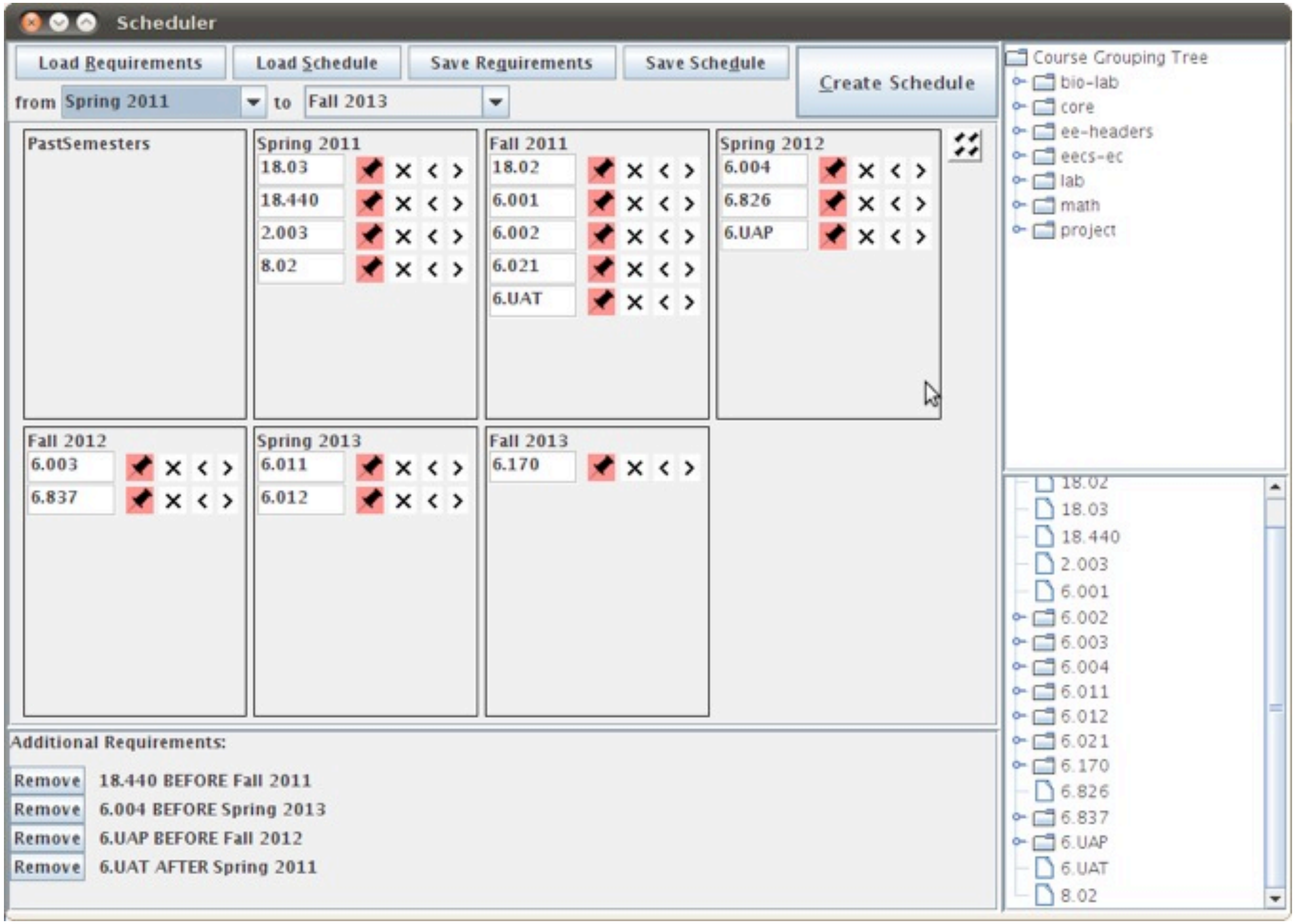
binary search tree

```
public class BalancedTree {
  private Node root;

  @SpecField("this.nodes: set Node | this.nodes = this.root.∗(left+right) – null")
  @Invariant({
    "all x: this.left.∗(left+right) – null | x.key< this.key",
    "all x: this . right .∗( left+right ) – null | x.key > this.key",
    "all n: this.nodes | (#n.left.ˆ(left+right) – #n.right.ˆ(left+right)) in {–1, 0, 1}"})

  public class public class Node {
    private Node left, right;
    private int key;
  }

  @Requires("z.key !in this.nodes.key")
  @Ensures ("this.nodes = @old(this.nodes) + z")
  @Modifies("this.root, this.nodes.left | <1>= null, this.nodes.right | <1>= null")
  public void insertNode(Node z) { Squander.exe(this, z); }
}
```

defines *nodes*

tree is balanced

course scheduler



18.440 BEFORE Fall 2011

6.004 BEFORE Spring 2013

6.UAP BEFORE Fall 2012

6.UAT AFTER Spring 2011

existing app
uses Alloy, but
embedded by hand

new version
Squander code

numbers
1500 lines of code
replaced by 30 of spec
2000 objects on heap
runs in 5s instead of 1s

related work

Kaplan [Koskal, Kuncak, Suter]
constraints integrated with Scala

Jeeves [Yang, Yessenov, Solar-Lezama]
declarative privacy policies enforced at runtime

PBnJ [Samimi, Aung, Millstein]
falling back to executable specs

data structure repair [Zaeem, Khurshid]
using contracts and Kodkod

#2
verification of web apps

work by Joseph Near

code checking by refutation

represent code & spec as formulas

Code(s,s’)

Spec(s,s’)

find instances of

Code(s,s’) **and not** Spec(s,s‘)

guarantees

every instance is a valid counterexample
but may miss bugs due to small scope

observations about web apps

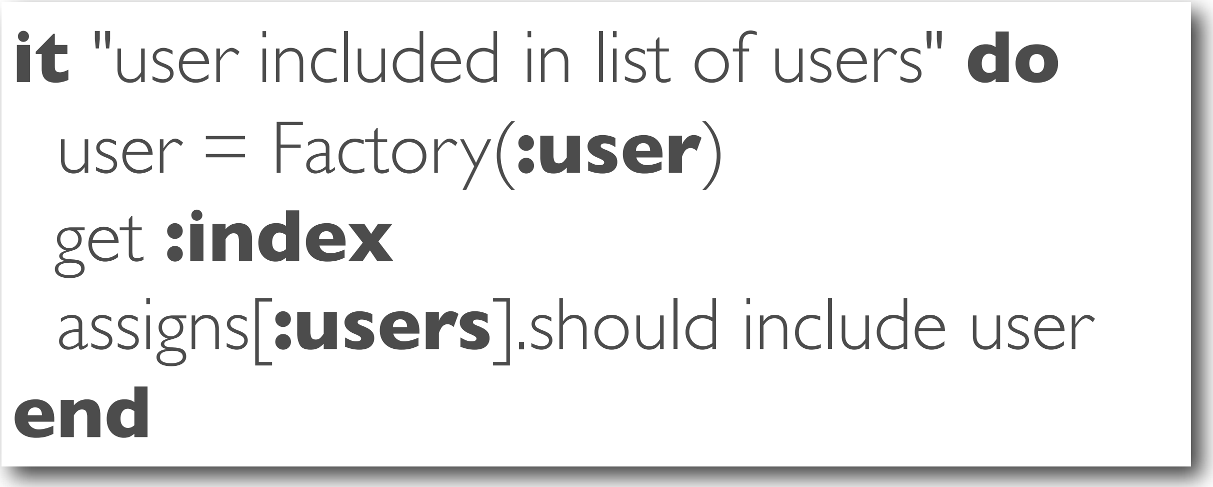
“CRUD”
little control structure
relational data

not just functionality
security critical
also relational, data-centric

unit tests
of controller actions
eg in RSpec

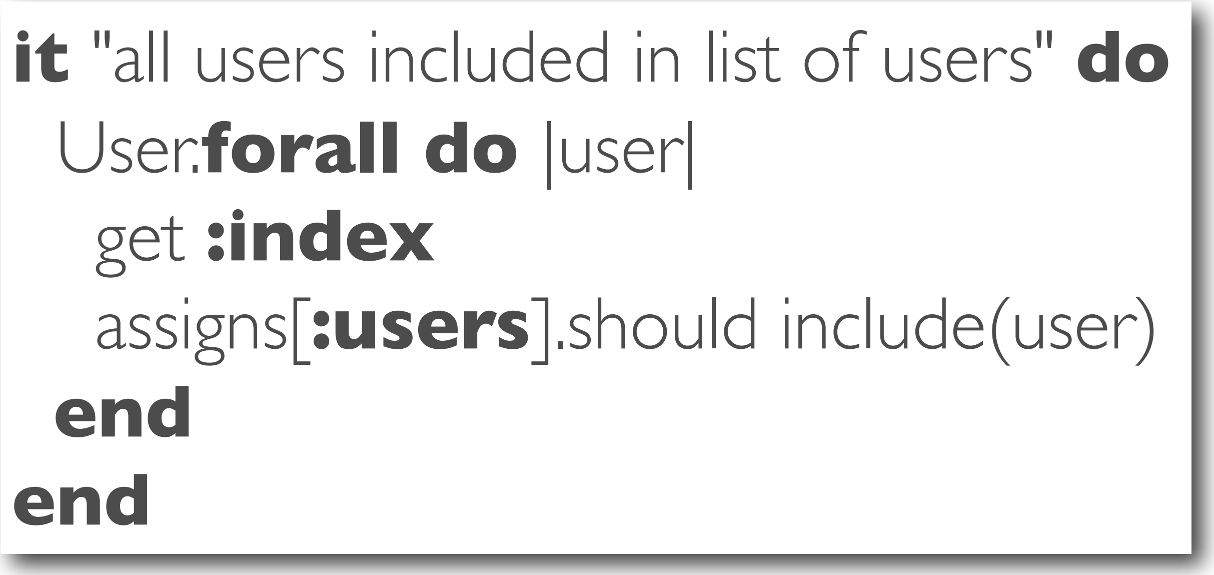
disciplined layering
data access factored out

Rubicon specs



```
it "user included in list of users" do
  user = Factory(:user)
  get :index
  assigns[:users].should include user
end
```

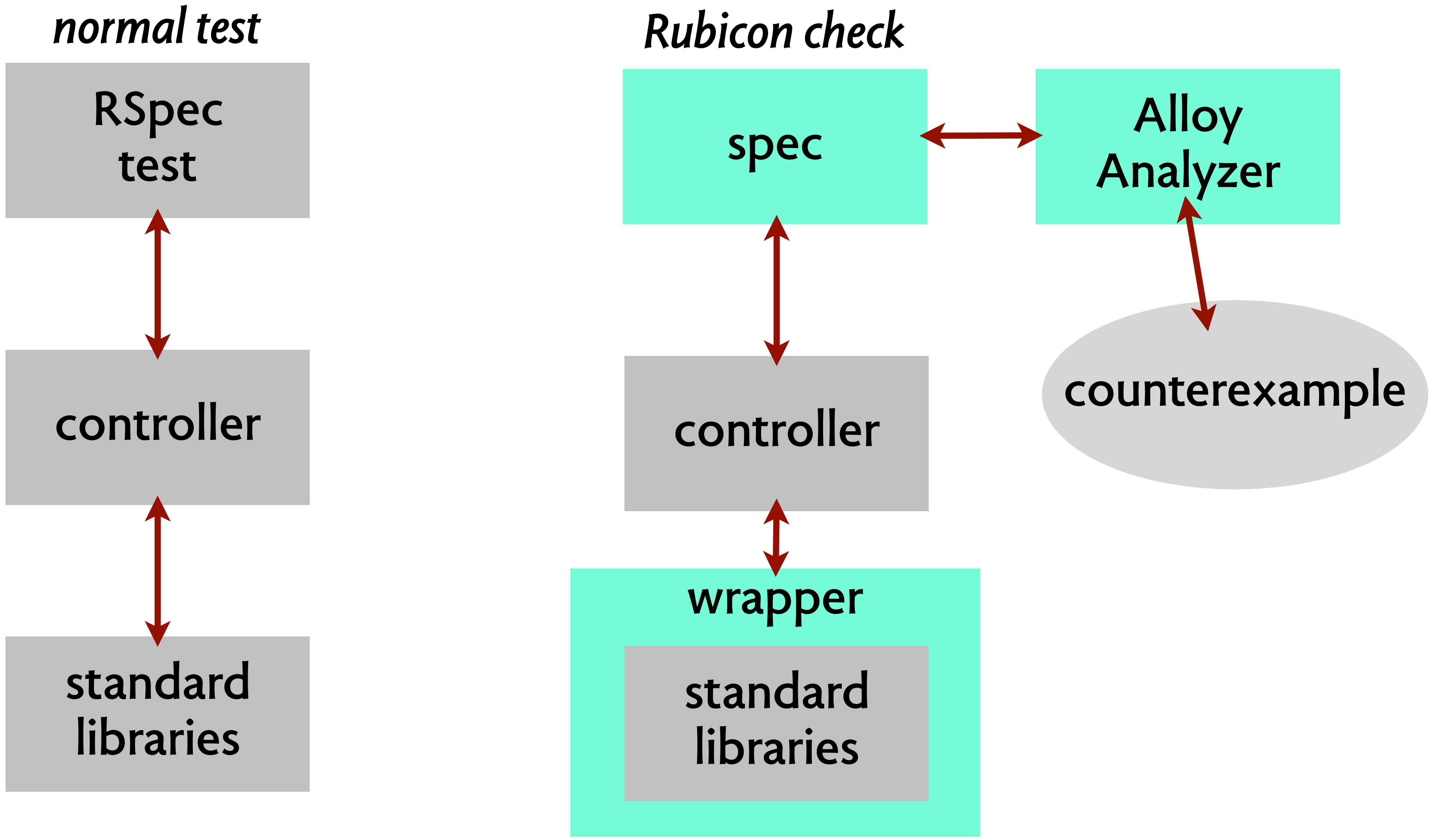
RSpec test



```
it "all users included in list of users" do
  User.forall do |user|
    get :index
    assigns[:users].should include(user)
  end
end
```

Rubicon spec

how Rubicon works



stubbing active record

```
klasses = ActiveRecord::Base.descendants klasses.each do |klass|
  metaklass = class << klass; self; end
  metaklass.send(:define_method, :all,
  lambda {|*args|
    if $symbolic_execution then ExprApp.new(:all, [self])
    else super end})
end
```

User.all **evaluates to**

in Rails: list of database records

in Rails+Rubicon: ExprApp(User)

stubbing subclass methods

```
klass.column_names.each do |name|
  klass.send(:define_method, name.to_sym,
    lambda {|*args|
      if $symbolic_execution then
        ExprApp.new(:field_get, [self, name.to_sym])
      else super end})
end
```

some_user.id **evaluates to**

in Rails: 1, eg

in Rails+Rubicon: ExprApp(**:field_get**, some_user, **:id**)

sample spec & action

```
class UsersController < ApplicationController
def profile
 @current_user = User.find_by_id(session[:user_id])
 all_posts = Micropost.where(:user => @current_user.friends)
 @posts = all_posts.select do |post|
    (post.privacy == 'friends') |
    (post.privacy == 'public')
    end
  end
end
```

```
it "all users see only their friends' posts" do
    User.forall do |user|
        session[:user_id] = user.id
        get :profile
        Micropost.forall do |post|
            ((post.privacy == 'friends') & (!user.friends.include? post.user)).
            implies do
            assigns[:posts].should_not include post
                end
            end
    end
end
```

sample verification condition

```
Implies(
 And(symbolic_post.privacy = 'friends',
       Not(include(symbolic_user.friends, symbolic_post.user))),
 Not(include(Query(Micropost,
                   And(include(:user, symbolic_user.friends)),
                       Or(=(:privacy, 'friends')), =(:privacy, 'public'))))),
     symbolic_post))
```

converted to:

```
all u: User, p: Micropost |
   p.privacy = friends and not p.user in u.friends
   implies p not in
      { p': Micropost |
            p.user in u.friends
            and (p'.privacy = friends or p'.privacy = public)}
```

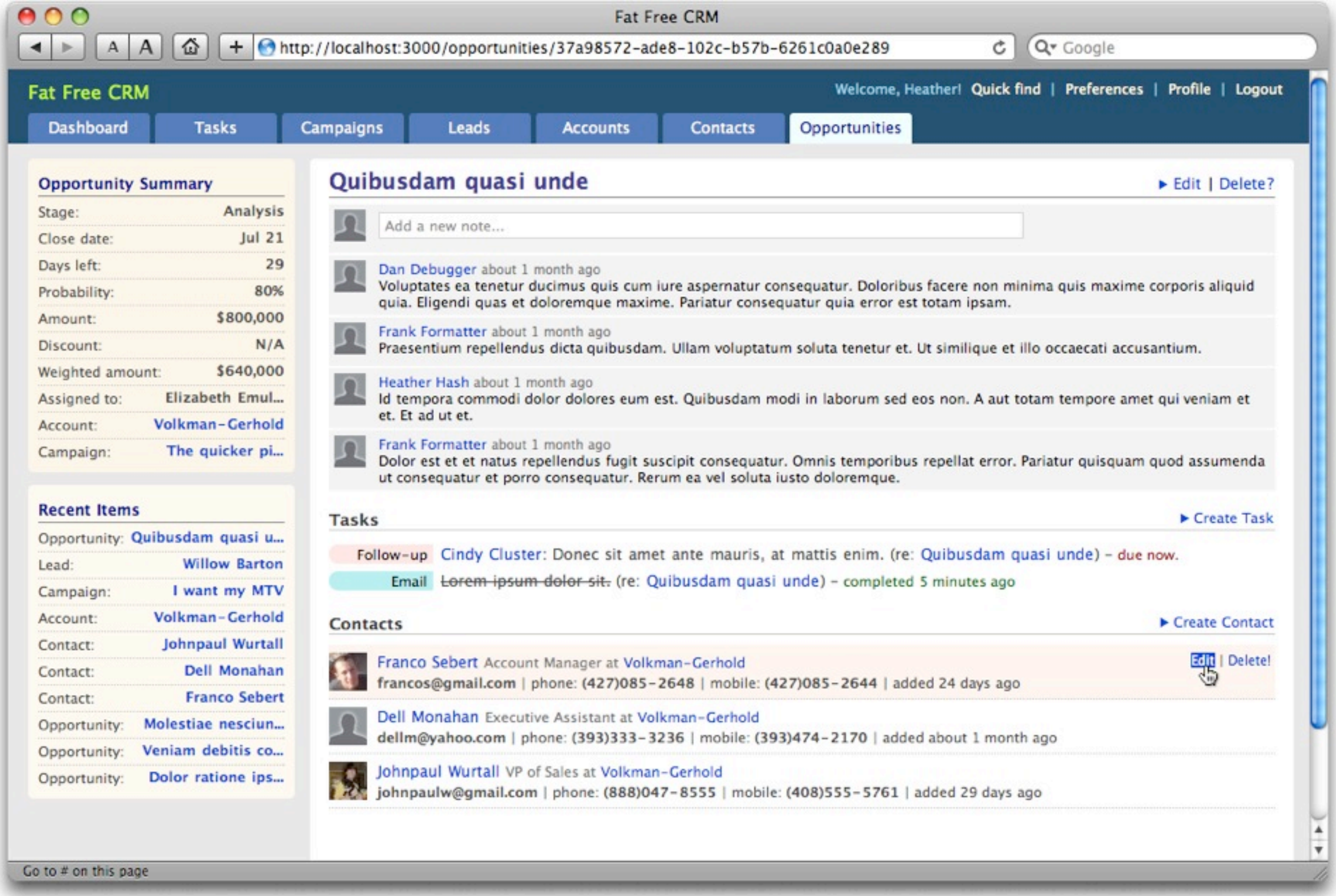
results to date

wrote specs
5 open-source apps
c.150 specs, 1kloc

ran analyses
average about 3s in scope of 5

founds bugs
2 bugs found in Fat Free CRM
one spurious
one serious security bug

Fat Free CRM


Fat Free CRM
http://localhost:3000/opportunities/37a98572-ade8-102c-b57b-6261c0a0e289
Google
Fat Free CRM
Welcome, Heather! Quick find | Preferences | Profile | Logout
Dashboard
Tasks
Campaigns
Leads
Accounts
Contacts
Opportunities
Go to # on this page

Opportunity Summary

Stage:	Analysis
Close date:	Jul 21
Days left:	29
Probability:	80%
Amount:	$800,000
Discount:	N/A
Weighted amount:	$640,000
Assigned to:	Elizabeth Emul...
Account:	Volkman-Gerhold
Campaign:	The quicker pi...

Recent Items

Opportunity:	Quibusdam quasi u...
Lead:	Willow Barton
Campaign:	I want my MTV
Account:	Volkman-Gerhold
Contact:	Johnpaul Wurtall
Contact:	Dell Monahan
Contact:	Franco Sebert
Opportunity:	Molestiae nesciun...
Opportunity:	Veniam debitis co...
Opportunity:	Dolor ratione ips...

Quibusdam quasi unde ▸ Edit | Delete?

Add a new note...

Dan Debugger about 1 month ago
Voluptates ea tenetur ducimus quis cum iure aspernatur consequatur. Doloribus facere non minima quis maxime corporis aliquid quia. Eligendi quas et doloremque maxime. Pariatur consequatur quia error est totam ipsam.

Frank Formatter about 1 month ago
Praesentium repellendus dicta quibusdam. Ullam voluptatum soluta tenetur et. Ut similique et illo occaecati accusantium.

Heather Hash about 1 month ago
Id tempora commodi dolor dolores eum est. Quibusdam modi in laborum sed eos non. A aut totam tempore amet qui veniam et et. Et ad ut et.

Frank Formatter about 1 month ago
Dolor est et et natus repellendus fugit suscipit consequatur. Omnis temporibus repellat error. Pariatur quisquam quod assumenda ut consequatur et porro consequatur. Rerum ea vel soluta iusto doloremque.

Tasks ▸ Create Task

Follow-up Cindy Cluster: Donec sit amet ante mauris, at mattis enim. (re: Quibusdam quasi unde) – due now.

Email ~~Lorem ipsum dolor sit.~~ (re: Quibusdam quasi unde) – completed 5 minutes ago

Contacts ▸ Create Contact

Franco Sebert Account Manager at Volkman-Gerhold Edit | Delete!
francos@gmail.com | phone: (427)085-2648 | mobile: (427)085-2644 | added 24 days ago

Dell Monahan Executive Assistant at Volkman-Gerhold
dellm@yahoo.com | phone: (393)333-3236 | mobile: (393)474-2170 | added about 1 month ago

Johnpaul Wurtall VP of Sales at Volkman-Gerhold
johnpaulw@gmail.com | phone: (888)047-8555 | mobile: (408)555-5761 | added 29 days ago

related work

model finding for checking Java
[Vaziri], [Taghdiri], [Dennis] & co
example: KOA vote tallying program

model checking for web apps
eg, [DeAlfaro], [Castelluccia]
focused on navigation

symbolic security analysis
[Chaudhuri & Foster]

checking Rails data model in Alloy
[Nijjar & Bultan]

#3
security configuration

work by Eunsuk Kang

problem

most security attacks not subtle
badly configured firewall
failure to sanitize queries
missing access controls

but hard to fix
complex configuration settings
interactions between components
changing defaults & behaviors

standard approach

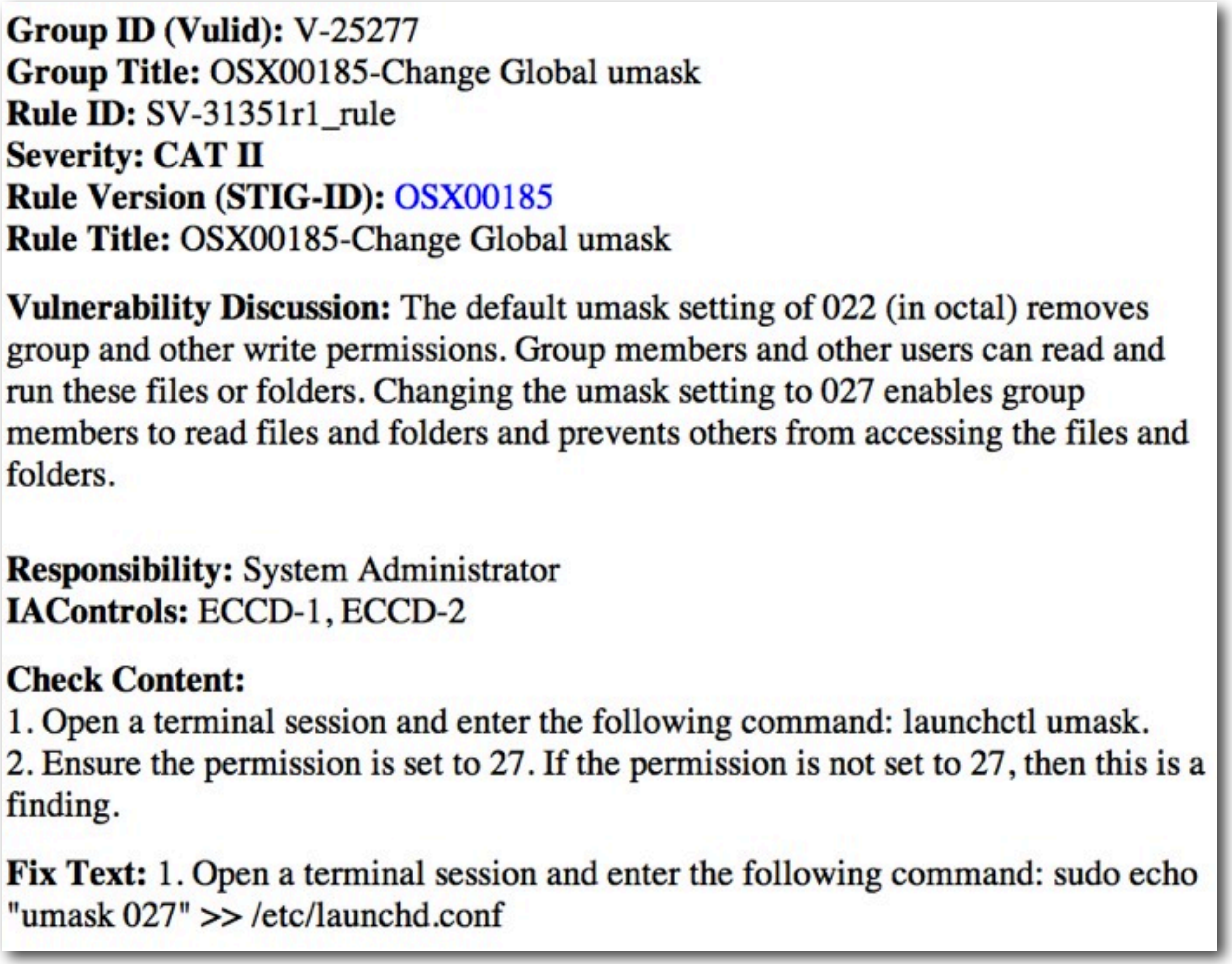
designer of application
relies on experts for component properties

administrator picks conservative settings
eg DISAs ‘Security Technical Implementation Guides’

no explicit argument
connecting the components

application-independent
too stringent?
not stringent enough

a sample STIG entry



Group ID (Vulid): V-25277
Group Title: OSX00185-Change Global umask
Rule ID: SV-31351r1_rule
Severity: CAT II
Rule Version (STIG-ID): OSX00185
Rule Title: OSX00185-Change Global umask

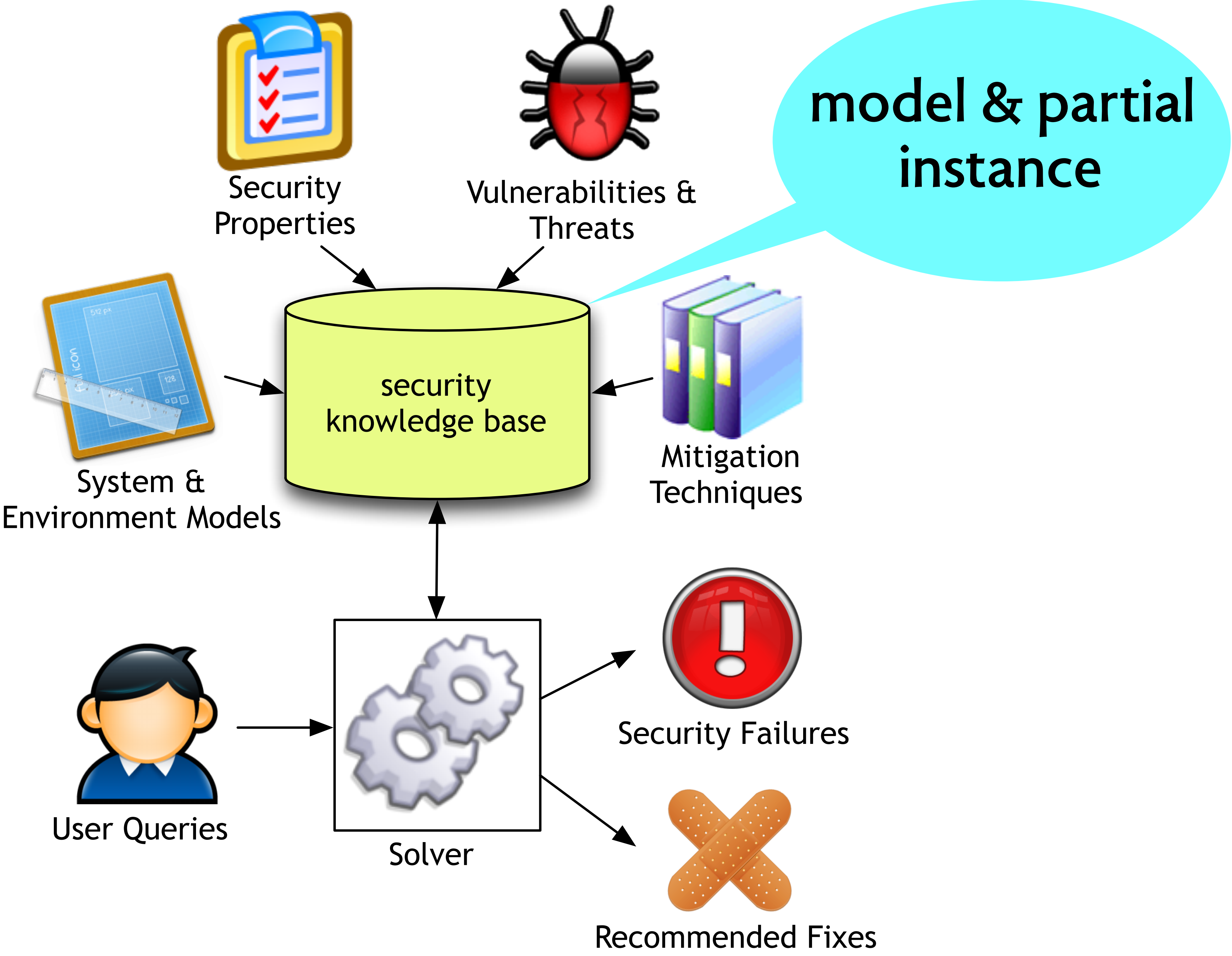
Vulnerability Discussion: The default umask setting of 022 (in octal) removes group and other write permissions. Group members and other users can read and run these files or folders. Changing the umask setting to 027 enables group members to read files and folders and prevents others from accessing the files and folders.

Responsibility: System Administrator
IAControls: ECCD-1, ECCD-2

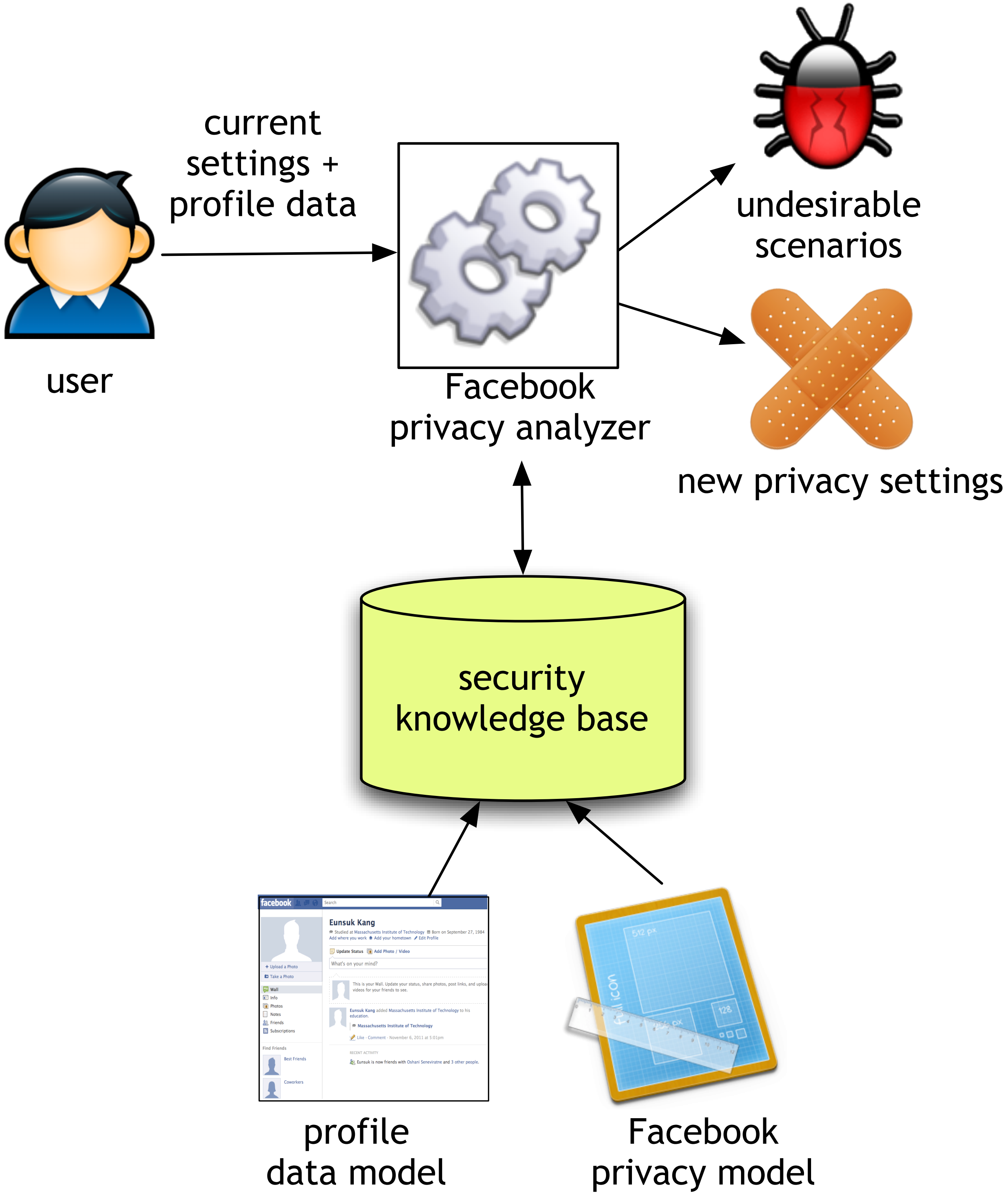
Check Content:
1. Open a terminal session and enter the following command: launchctl umask.
2. Ensure the permission is set to 27. If the permission is not set to 27, then this is a finding.

Fix Text: 1. Open a terminal session and enter the following command: sudo echo "umask 027" >> /etc/launchd.conf

architecture



example: Facebook privacy



user

current settings + profile data

Facebook privacy analyzer

undesirable scenarios

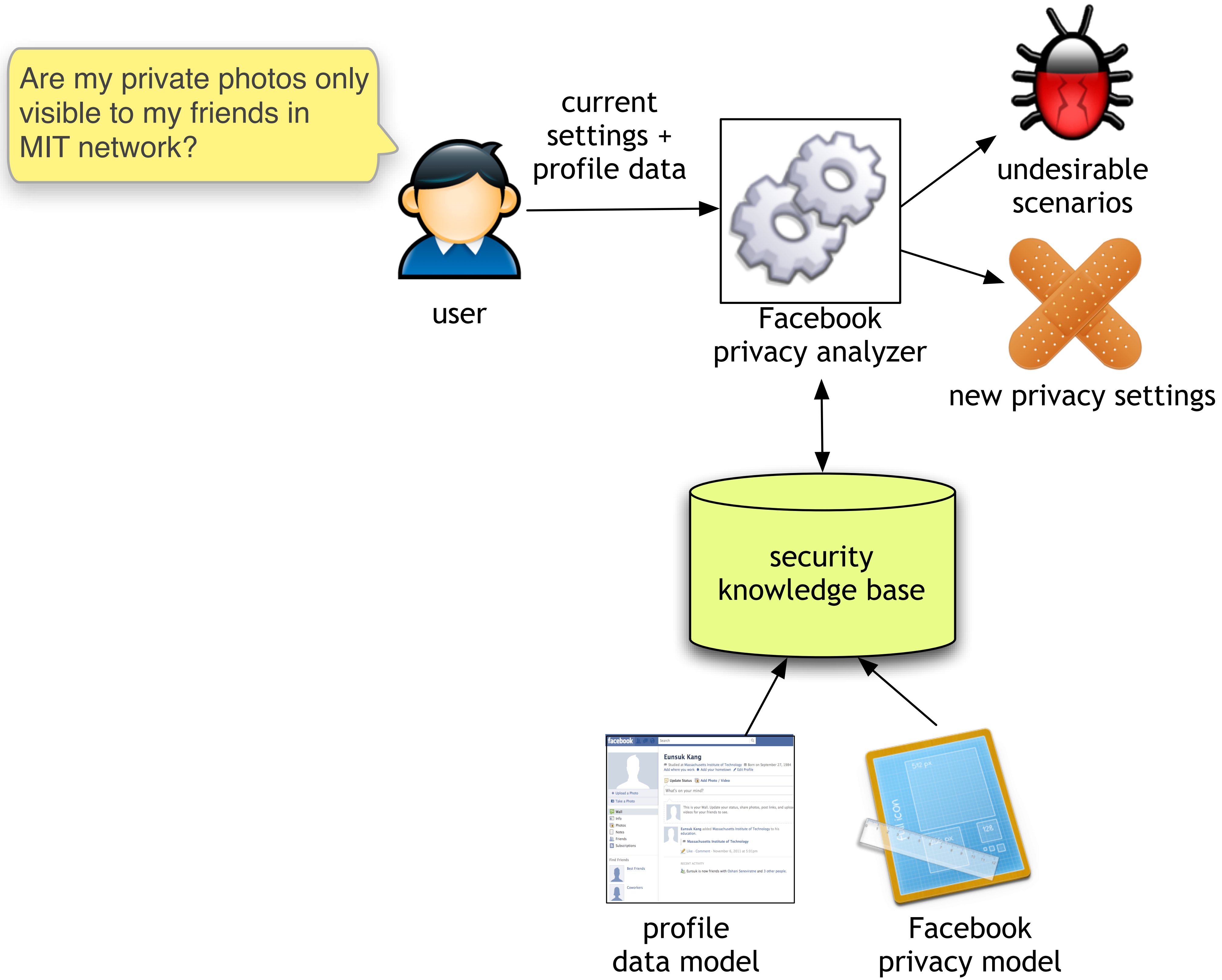
new privacy settings

security knowledge base

profile data model

Facebook privacy model

example: Facebook privacy


Are my private photos only visible to my friends in MIT network?
user
current settings + profile data
Facebook privacy analyzer
undesirable scenarios
new privacy settings
security knowledge base
profile data model
Facebook privacy model

facebook
Search

Eunsuk Kang

Studied at Massachusetts Institute of Technology Born on September 27, 1984
Add where you work Add your hometown Edit Profile

Update Status Add Photo / Video

What's on your mind?

+ Upload a Photo
Take a Photo

Wall
Info
Photos
Notes
Friends
Subscriptions

This is your Wall. Update your status, share photos, post links, and upload videos for your friends to see.

Eunsuk Kang added Massachusetts Institute of Technology to his education.

Massachusetts Institute of Technology

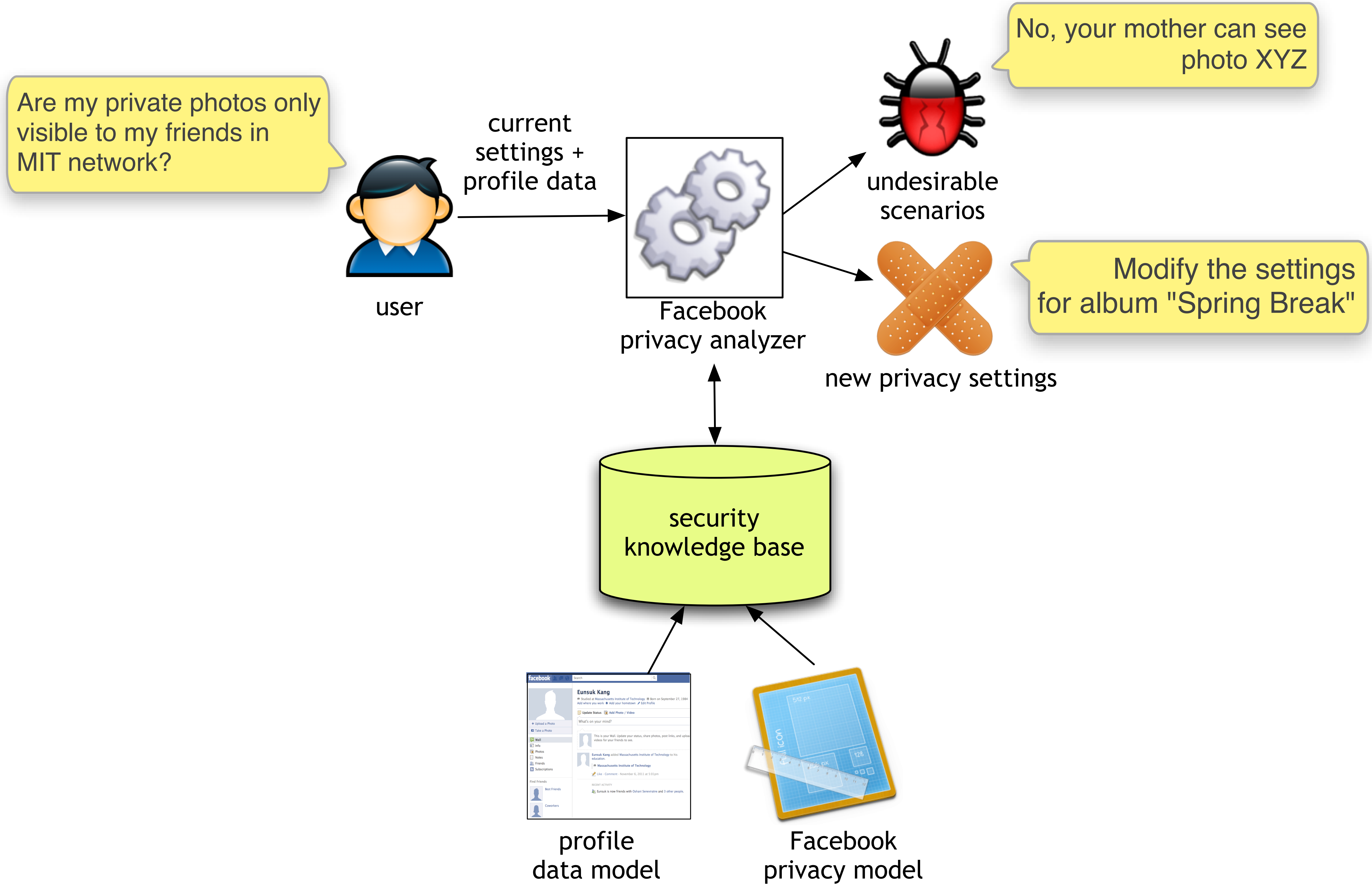
Like · Comment · November 6, 2011 at 5:01pm

RECENT ACTIVITY

Eunsuk is now friends with Oshani Seneviratne and 3 other people.

Find Friends
Best Friends
Coworkers

example: Facebook privacy



Eunsuk Kang

Studied at Massachusetts Institute of Technology Born on September 27, 1984

Add where you work Add your hometown Edit Profile

What's on your mind?

This is your Wall. Update your status, share photos, post links, and uploa videos for your friends to see.

Eunsuk Kang added Massachusetts Institute of Technology to his education.

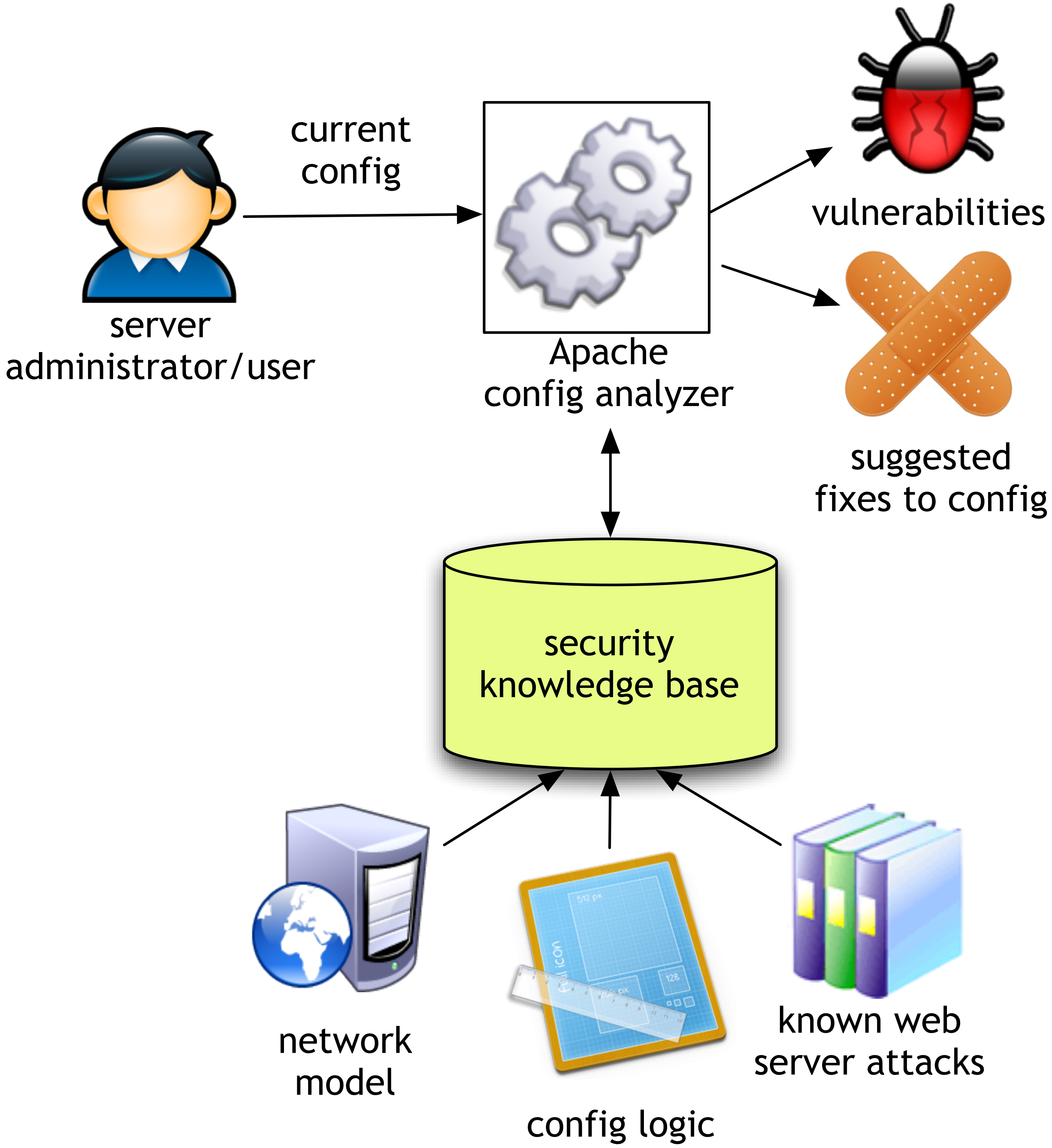
Massachusetts Institute of Technology

Like · Comment · November 6, 2011 at 5:01pm

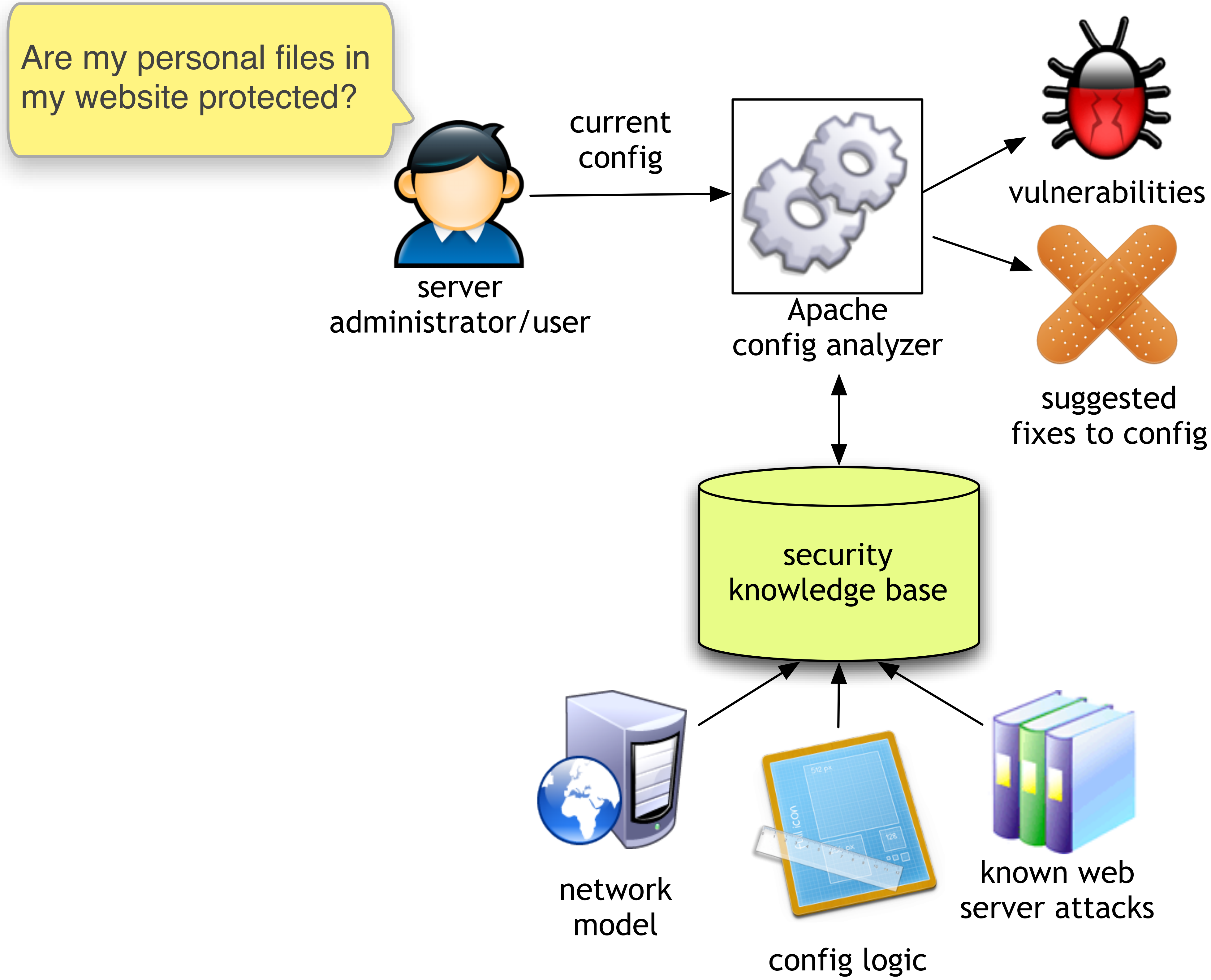
RECENT ACTIVITY

Eunsuk is now friends with Oshani Seneviratne and 3 other people.

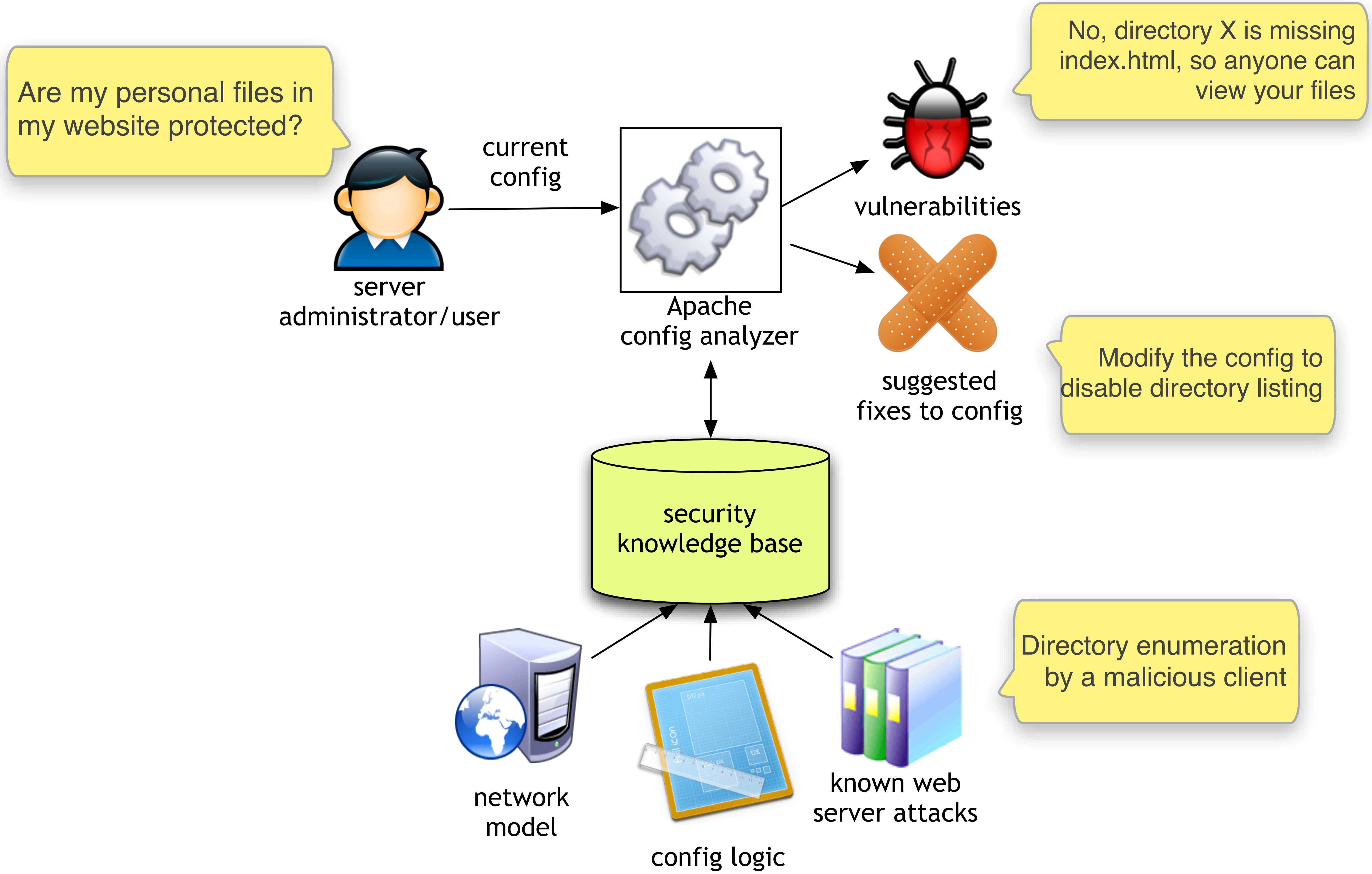
example: Apache security



example: Apache security



example: Apache security



Are my personal files in my website protected?

server administrator/user

current config

Apache config analyzer

vulnerabilities

No, directory X is missing index.html, so anyone can view your files

suggested fixes to config

Modify the config to disable directory listing

security knowledge base

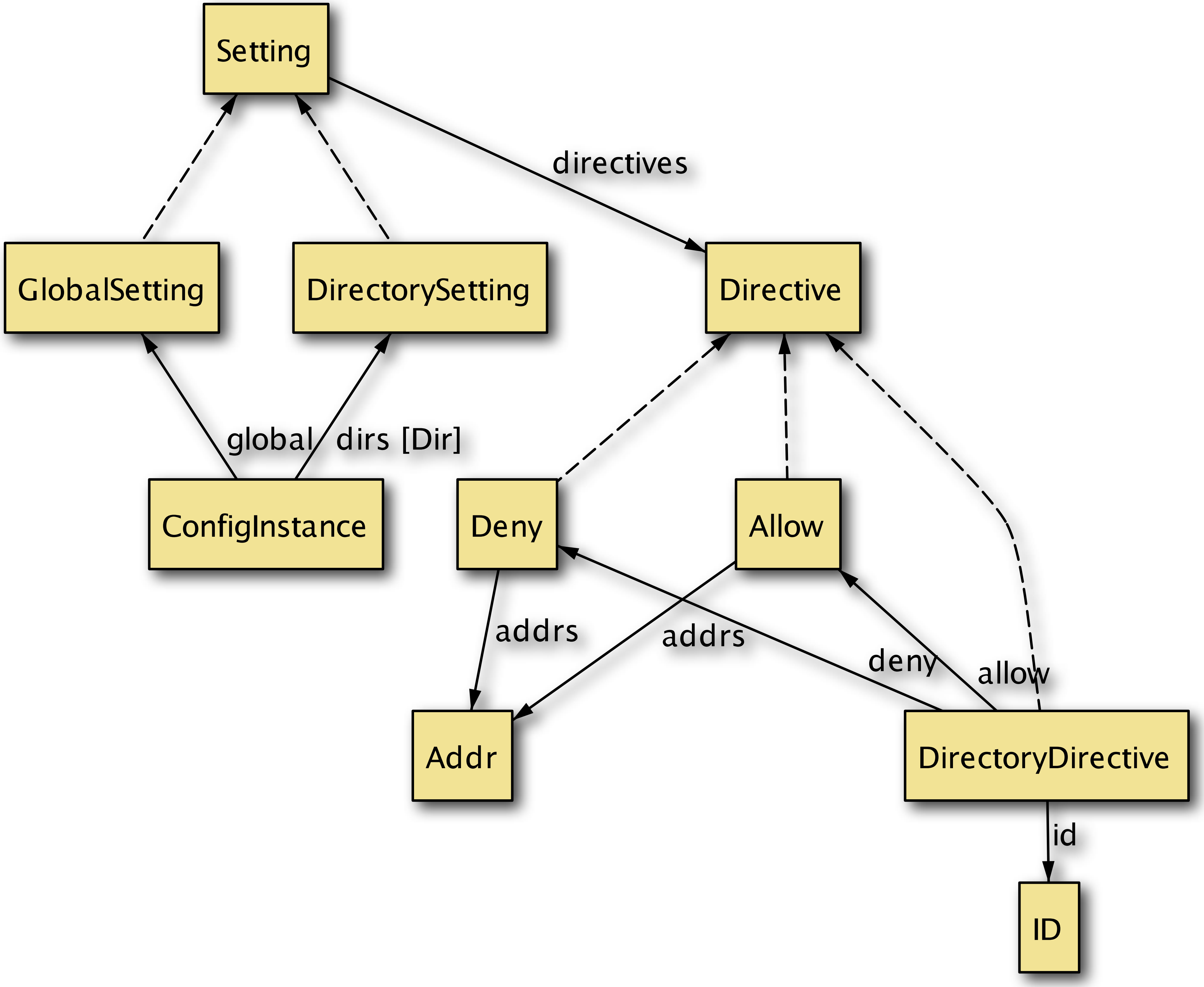
network model

config logic

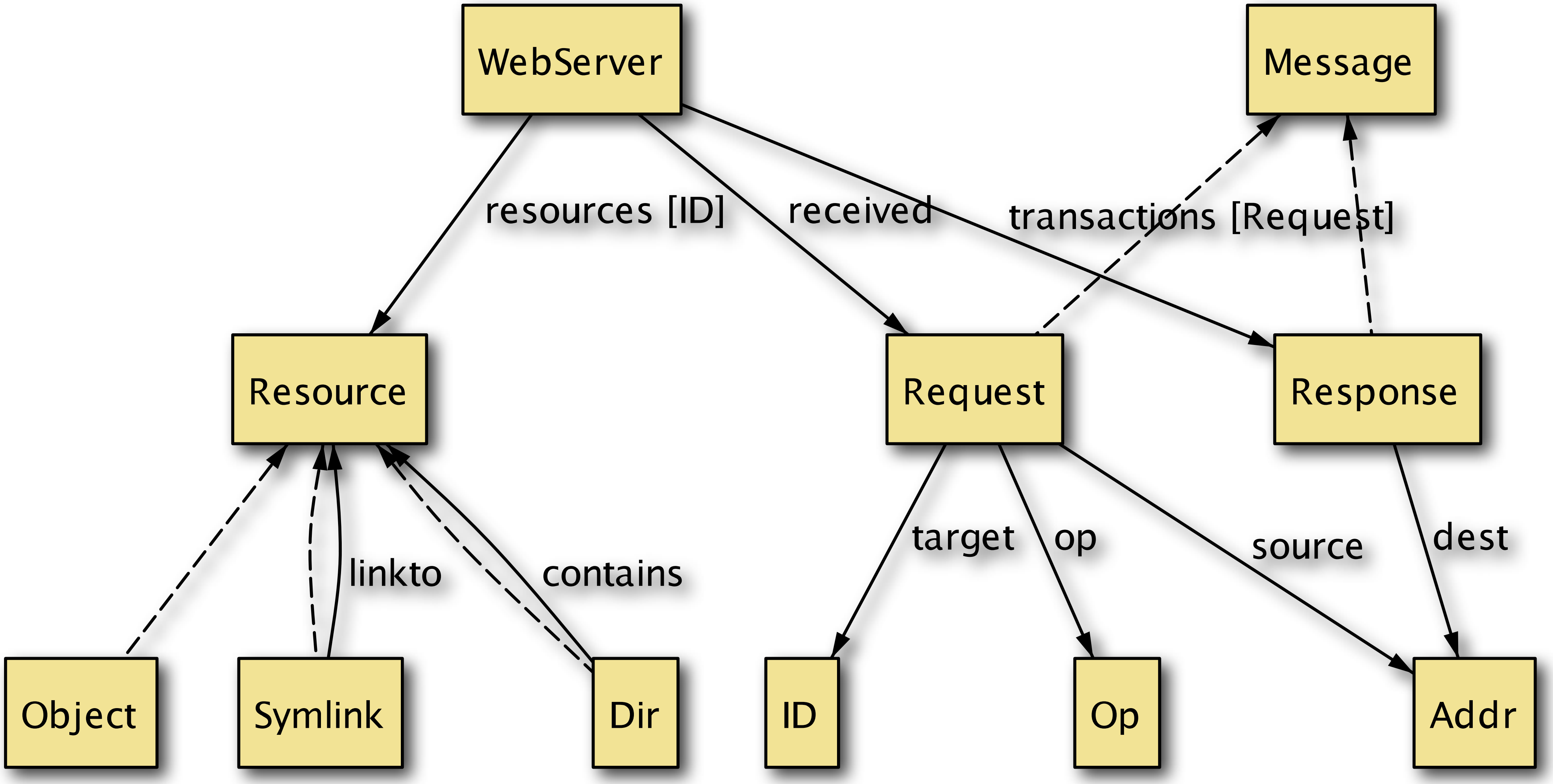
known web server attacks

Directory enumeration by a malicious client

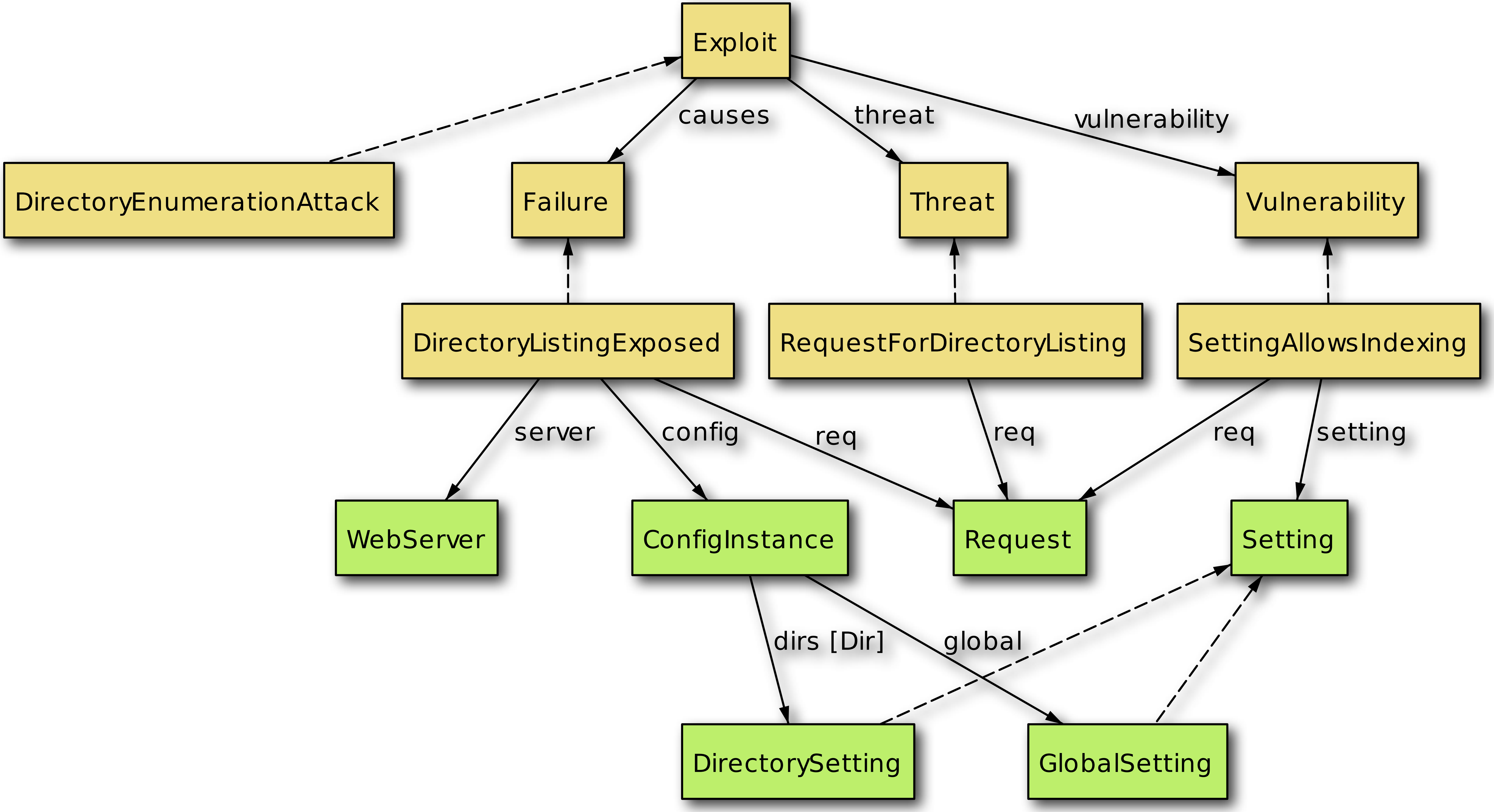
apache configuration model



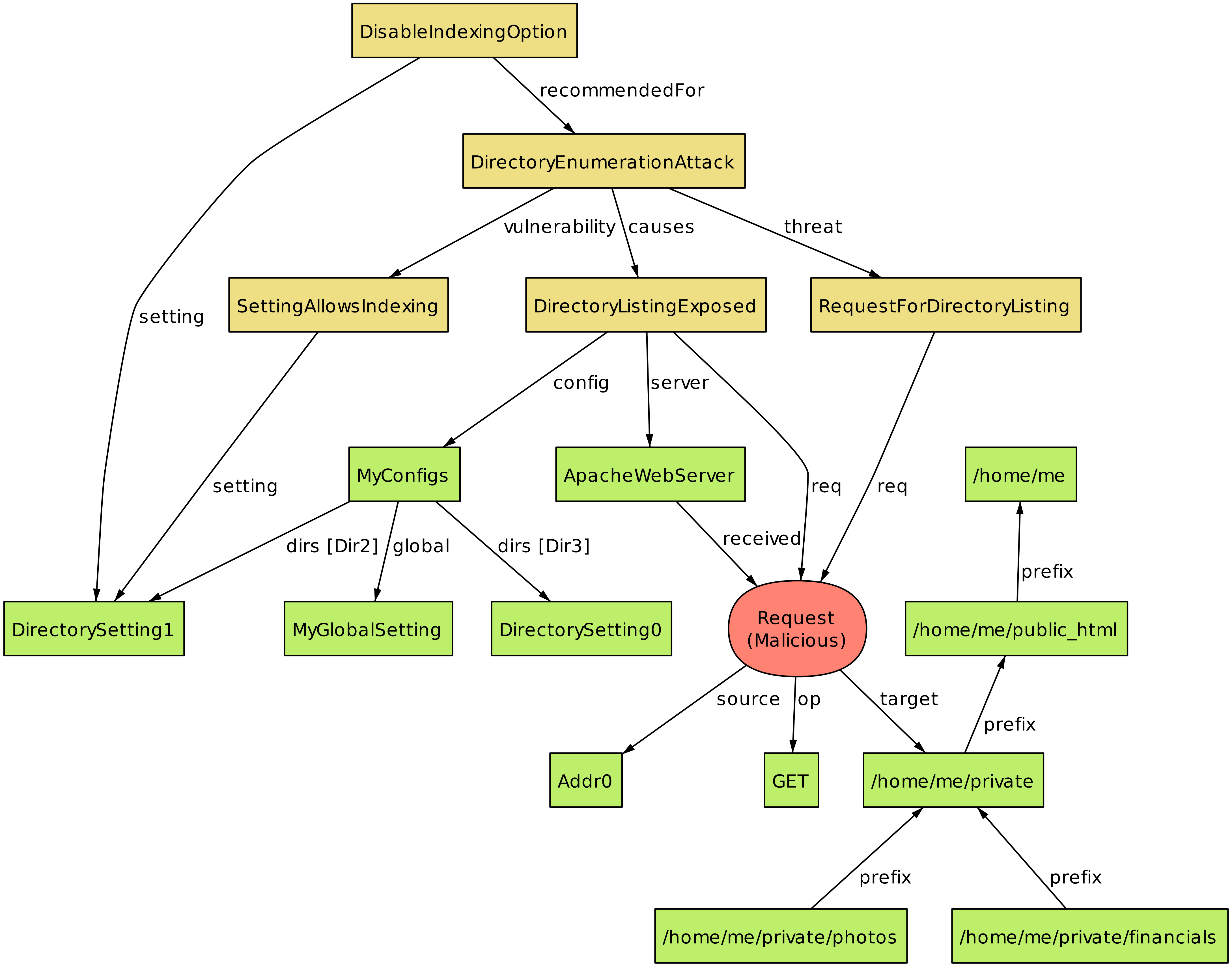
apache behavior model



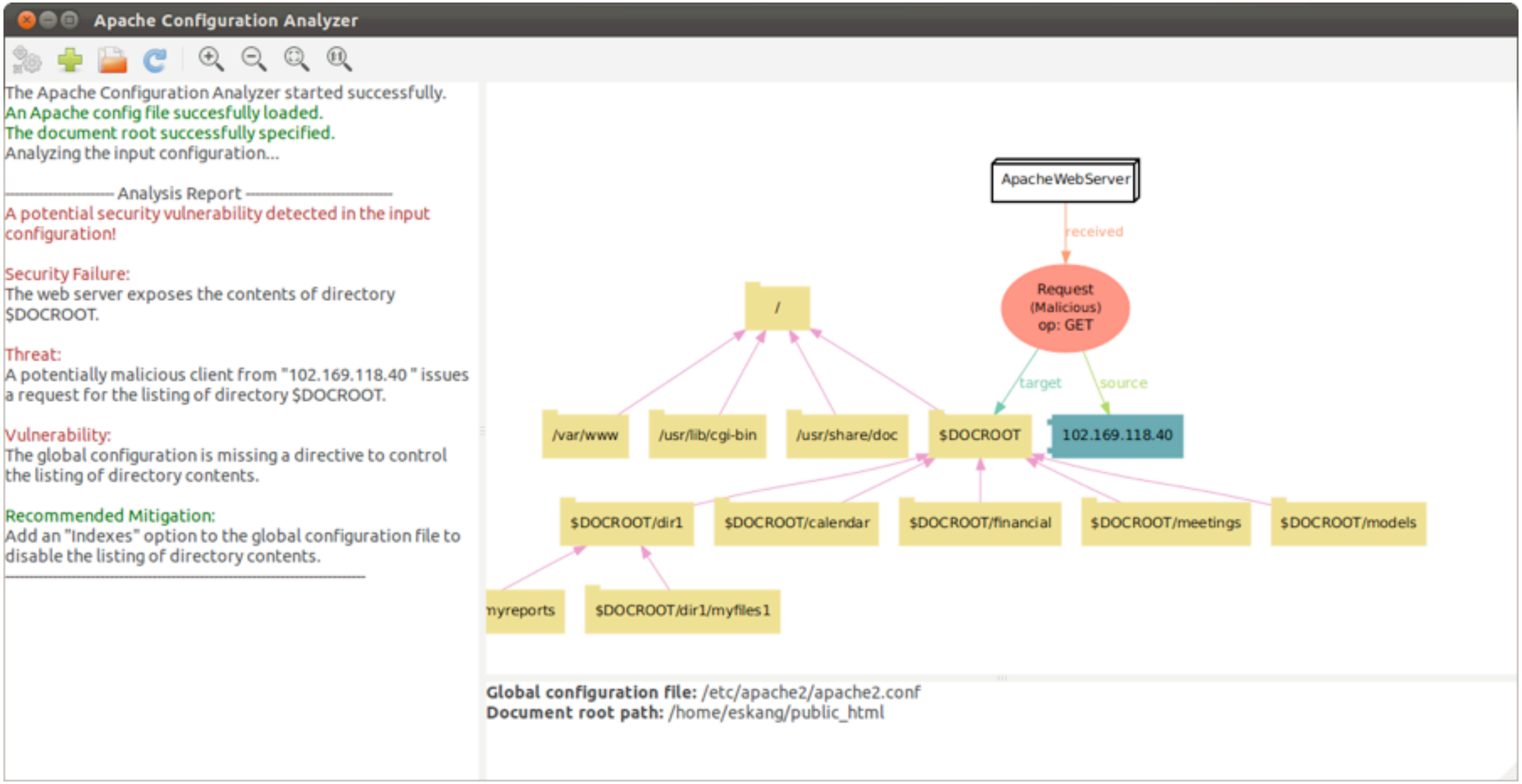
apache threat model



sample attack



prototype Apache analyzer



The Apache Configuration Analyzer started successfully.
An Apache config file succesfully loaded.
The document root successfully specified.
Analyzing the input configuration...

------------------- Analysis Report -------------------------

A potential security vulnerability detected in the input configuration!

Security Failure:
The web server exposes the contents of directory $DOCROOT.

Threat:
A potentially malicious client from "102.169.118.40 " issues a request for the listing of directory $DOCROOT.

Vulnerability:
The global configuration is missing a directive to control the listing of directory contents.

Recommended Mitigation:
Add an "Indexes" option to the global configuration file to disable the listing of directory contents.

Global configuration file: /etc/apache2/apache2.conf
Document root path: /home/eskang/public_html

related work

SAT-based configuration

firewalls [Margrave (Nelson et al, 10)]

packages [eg, Opium, Mancoosi, Zypp]

rule-based configuration

networks [eg, MulVal (Ou et al., 05)]

model-based diagnosis

[eg, Reiter, Kleer, Williams]

explain symptoms at run-time

summary: 3 provocations

three provocations

relational logic + SAT
cf. “the expressiveness/tractability balance”

focus on failures vs proofs
counterexamples, explanations, fixes

high-level reasoning vs state machine
may scale better & provide better feedback?