Automating End User Security Tasks

Eunsuk Kang and Daniel Jackson
MIT Security Seminar
Nov 26, 2012
Problem
I recently set up my own home server, running Ubuntu Linux. The intent I'm having for this is to have a private wiki for my family, so we can have a place to collect information like phone numbers, and other relevant information. Currently, access is limited to my LAN, essentially prohibiting outside access, but I'm considering allowing external access, if I can figure out how to meet a level of security. Specifically, I'm looking for a solution which:

1. Prevents read access of any web content without some kind of authentication.
2. The authentication should be passed in such a way to prevent packet sniffing to allow permanent access to the network.
3. Ideally, I don't want this to cost anything, or if it costs something, to be a small one time cost.
4. As one of the most common locations I will be accessing this will be work, I would prefer a solution which does not require any additional software on the client end.
5. Access to within the LAN must be unrestricted, ie, no password required.
6. I would like to be able to access the site occasionally from my Android phone, so any solution should not prevent that from working.
7. I would like to expand this someday to allow more than just wiki software, but if there's a clever way to do this with the wiki software, I'll consider it.

My current thought is leaning towards producing a self-signed SSL certificate, and using Apache authentication, either basic or digest, but requiring it to be passed over HTTPS, but I'm opened to alternative solutions.

My full config:

- Ubuntu 11.10
- Apache 2.2
- Media Wiki 1.18
I recently set up my own home server...a private wiki for my family...considering allowing external access, if I can figure out how to meet a level of security.
I have a website with a webhost company. I wrote a little php/mysql application - no file uploads, just administering data from the DB and creating pdf with TCPDF. There are folders like css and js but nothing else.

As I have limited control on the shared host, what would be the best practice to keep the files safe but executable.

So let’s say:

index.php
connect_db.php
page1.php
functions.php
IMAGES folder
CSS folder
TCPDF folder
.htaccess
php.ini

Update:

the server is Linux server with PHP5.2 (optional 5.3) and I have unlimited webservice... The main reason that I am asking because I had an attack and someone managed to put some folder (netbank) and some email form to phish personal details. Now I am building the whole site from scratch and would like to cover every aspect.

I also broke this question to more questions so people might use as well as a good practice resources.
How to protect off-the-shelf web applications?

As an exercise, please consider the following:

If you are faced with having to deploy off-the-shelf web applications that you are unsure about the vendors security coding practices, which techniques or technologies would be worthwhile applying or deploying for some protection?

Assuming the following:

- You don't have access to the source code or can't easily influence changes.
- You don't have the resources, funds or skills, for penetration testing or vulnerability assessment of individual applications but funds maybe available for protection technologies that could be used across a range of web applications.
- You have no ability to veto the deployment of the applications.

Update: Think SysAdmin rather than InfoSec professional, that is you have some idea about security but it isn't your main thing.

What are the approaches, techniques or technologies that should be considered at the SysAdmin level without a Security Team to call on that would help them sleep better at night?

web-application  attack-prevention

share  improve this question  edited Jun 22 '11 at 2:54  asked May 19 '11 at 13:03

web-application  webserver

share  improve this question  edited Sep 14 '11 at 11:54  asked Sep 14 '11 at 9:25

share  improve this question  asked Feb 21 at 13:33
How to protect off-the-shelf web applications?

How do I run a security check on my WordPress server?

I have a cloud-based server running with CentOS 6.0 and CSF installed. Today I got a message from my host that one of my WordPress installations is hacked and used for phishing.

But I don’t know how this has happened. I do not know where it is happening. What the best practice here? How can I check what has happened and stop it?

EDIT: My host said now that he wasn’t sure it was wordpress, but was suspecting it. Do I really need to delete everything and reinstall. This is over 10 sites, and it will be a lot of work.

web-application  incident-response  forensics  wordpress

Not strictly related to Wordpress but you could install rkhunter which and etckeeper. rkhunter will perform daily scans for root kits and you can use etckeeper (or git) to track changes to important directories like /etc – javano Aug 10 at 10:46

Or tripwire.org – Tie-fighter Aug 10 at 13:57
Security implications of having files owned by the apache user?

Currently running a LAMP instance that developers are using for a variety of webapps. I have the following scenario:

- Multiple developers need access to create and modify files under /var/www/html
- The developers need to be able to access each other’s code in case they are out on vacation, sick, etc.

My thoughts were the following:

1. For each web application (directory) under /var/www/html, recursively change the owner to ‘apache’ and the group to a ‘developers’ group. Set the permissions at 570.
2. For each file under /var/www/html, recursively change the owner to ‘apache’ and the group to a ‘developers’ group. Set the permissions at 470.

Keeping the files and directories as read-only to the Apache user is a good security practice, but is having such files owned by the Apache user a bad one? My concern was around what would happen if Apache got hit by a 0-day vulnerability or something of the sort.

If anyone has a more elegant way to do this, I’m interested in hearing it as well.
End User Products

Web Servers

Routers & Firewalls

Social Networks

Web Frameworks
What Users Do Now

User
Stating a Policy

“I want to share my files only with my family members”
Collecting Domain Knowledge

Policy

User

Available through: Articles, forums, books, papers, etc.

System model
Threats & Vulnerabilities
Mitigations
Performing Analysis

Check system against policy

User

Policy

Analysis

System model

Threats & Vulnerabilities

Mitigations
Challenges

Policy

Hard to formulate

Analysis

User

System model  Threats & Vulnerabilities  Mitigations
Challenges

Policy

Analysis

User

How much is enough?

System model
Threats & Vulnerabilities
Mitigations
Challenges

- Policy
- Analysis

User

Tedious & error prone

System model
Threats & Vulnerabilities
Mitigations
How Things Should Be

- System model
- Threats & Vulnerabilities
- Mitigations
Encoding Domain Knowledge

Database of knowledge

Expert

System model

Threats & Vulnerabilities

Mitigations
User’s Role

User provides desired policy

Policy

Knowledge Base
Automated Analysis

Policy → Analyzer → Knowledge Base

Violation

Leverage knowledge for checking policies
Providing Feedback to User

Policy → Diagnostic Feedback → Analyzer

Policy → Knowledge Base

Policy → Violation
Research Questions

Can we construct

1. A knowledge representation useful for describing a class of security tasks?

2. An analysis that efficiently detects security issues in the user’s system?

3. A form of feedback that the user can understand & act on?
Example: Web Server Configuration
Web Server Configuration

Apache HTTP Server
Serving 55% of all active web sites

Configuration
Global (httpd.conf) and local (.htaccess)
Powerful, flexible options (directives)
Some directives have (in)direct implications on security
Server-Client Interaction

req(url) → fetch

resp(doc) ←
Server-Client Interaction

Check directives & client info
If req is OK, fetch resource
Send back response
Map URI to filename
Check directives & client info
Execute file system command(s)
Not Whole Picture?

Map URI to filename
Check directives & client info
Execute file system command(s)

Check file permissions
If perms OK, run cmds
Return data to server
HTTP Server

Map URI to filename
Check directives & client info
Execute file system command(s)
Send back output as response

HTTP Server

File System

Check file permissions
If perms OK, run cmds
Return data to server

req(url)

resp(doc)

r/w/x(fname)

tFetch

Not Whole Picture?
A Simple Exploit

Only clients with password can access my private docs.

Use **AuthBasic** directive
Protect folder using **htpasswd**
A Simple Exploit

Bob

Mallory

HTTP Server

File System

Threat:
Assume shared host
Mallory has access to an account
A Simple Exploit

Mallory creates a script (havoc)
A Simple Exploit

Bob

Mallory

Mallory creates a script *(havoc)*. If CGI enabled, server executes it.
A Simple Exploit

Bob

Mallory

HTTP

Server

File

System

read(bobsDoc)

havoc runs as www user, and can read Bob’s web files
A Simple Exploit

Bob

Mallory

HTTP Server

File System

7

8

5

6

> 

Locked Folder
A Simple Exploit

Bob

Mallory

resp(bobsDoc)

Mallory's request returns Bob's data
A Simple Exploit

**Lesson:**
Not sufficient to configure Apache
Must reason about interactions between multiple components!

Mallory’s request returns Bob’s data
Representation
Multiple heterogenous components, with their own configuration settings
Components interact through **operations**
Each operation is associated with a **data element**
A policy is in form:
\[ \forall c \in \text{Comp}, d \in [\text{Data}] \cdot \text{accesses}(c, d) \implies \text{cond}(c, d) \]

where cond is a domain-specific conditional predicate.
A policy is in form:

\[ \forall c \in \text{Comp}, d \in \text{[Data]} \cdot \text{accesses}(c, d) \implies \text{cond}(c, d) \]

where \text{cond} is a domain-specific conditional predicate.

e.g.

\[ \forall c \in \text{Client}, d \in \text{BobsDocs} \cdot \text{accesses}(c, d) \implies \text{ip}(c) \in \text{whitelist}(d) \]
Access Control

Each component allows/denies certain events

Component M is associated with access control list

$$ACL_M = \text{Comp} \times \text{Op} \times \text{Data}$$

where $$(c, o, d) \in ACL_M \iff c \text{ is allowed to perform } o \text{ on } d$$
The content of ACL depends on configuration parameters.

Model each component $M$ as a configuration function:

$$CF_M : CO_M \rightarrow ACL_M$$

where $o \in CO_M$ is a configuration object.
$o \in CO_{\text{Fsys}} \equiv \text{list of r/w/x permission bits for each file}$
e.g.
foo.png bob users -rwxr--r--
bar.txt www root -r-xr-xr-x

Config for UNIX File System

read(foo.png)

bob

X

exec(foo.png)

www

File System
0 ∈ CO_{Apache} ≡ \text{tree structure of httpd.conf & .htaccess}
$0 \in CO_{Apache} \equiv \text{tree structure of } \text{httpd.conf & .htaccess}$
Config for Apache Server

Local config can override parameters in global config

httpd.conf:

```html
<Directory /usr/*/www>
  Options -ExecCGI -FollowSymLinks
  AllowOverride all
</Directory>
```

.htaccess:

```c
Options +ExecCGI
```

```
/private docs
  foo.txt
  bar.png
```

```
/public meet notes
```
Config for Apache Server

Global config can prevent overrides

`.htaccess`:
Order Allow,Deny
Allow from mit.edu
Deny from all

`httpd.conf`:
```
<Directory /usr/*/www>
  AllowOverride None
  Order Deny,Allow
</Directory>
```

Diagram:
- `/usr/bob/www`
  - `private`
    - `docs`
      - `foo.txt`
      - `bar.png`
  - `public`
    - `meet`
    - `notes`
Encoding in Alloy

Alloy
Modeling language based on first-order logic
Declarative & modular, supports composition
Automated analysis for checking policies, generating & visualizing violations

Some Applications in Security
Access control policies
Web protocols (OAuth, origin header, CORS)
Program analysis (Ruby on Rails, Java)
abstract sig Settings {
  directives : set Directive
}
sig GlobalSettings, LocalSettings extends Settings {}

sig ApacheConfig {
  global : GlobalSettings, /* global settings */
  local : Dir -> lone LocalSettings /* local settings */
}

abstract sig Directive {}
sig Allow, Deny extends Directive {
  addr : set Addr
}
/* Definition of the configuration function for Apache */

fact ApacheConfigDefn {
    all cobj : ApacheConfig |
        let acl = ApacheServer.config[cobj] |
            all c : Client, q : Req, r : Resource |
                c -> q -> r in acl implies
                    validReq[cobj, c, q, r]
}

/* True iff request "q" is considered valid by the server */
pred validReq[cobj : ApacheConfig, c : Client, q : Req, r : Resource] {
    let d = relevantDirectives[cobj, r] |
        checkAllowDeny[c, allow[d], deny[d]]
        checkAuthentication[c, auth[d]]
        ... /* other constraints */
}

/* True iff client "c" is allowed by the server */
pred checkAllowDeny[c : Client, allow : Allow, deny : Deny] {
    c.addr in allow.addrs or c.addr not in deny.addrs
}
Specifying Config Function

/* Definition of the configuration function for Apache */

fact ApacheConfigDefn {
    all cobj : ApacheConfig |
    let acl = ApacheServer.config[cobj] |
    all c : Client, q : Req, r : Resource |
    c -> q -> r in acl implies
    validReq[cobj, c, q, r]
}

/* True iff request "q" is considered valid by the server */
pred validReq[cobj : ApacheConfig, c : Client, q : Req, r : Resource] {
    let d = relevantDirectives[cobj, r] |
    checkAllowDeny[c, allow[d], deny[d]]
    checkAuthentication[c, auth[d]]
    ...
    /* other constraints */
}

/* True iff client "c" is allowed by the server */
pred checkAllowDeny[c : Client, allow : Allow, deny : Deny] {
    c.addr in allow.addrs or c.addr not in deny.addrs
}
Specifying Config Function

/* Definition of the configuration function for Apache */

fact ApacheConfigDefn {
    all cobj : ApacheConfig |
    let acl = ApacheServer.config[cobj] |
    all c : Client, q : Req, r : Resource |
    c -> q -> r in acl implies 
    validReq[cobj, c, q, r]
}

/* True iff request "q" is considered valid by the server */
pred validReq[cobj : ApacheConfig, c : Client, q : Req, r : Resource] {
    let d = relevantDirectives[cobj, r] |
    checkAllowDeny[c, allow[d], deny[d]]
    checkAuthentication[c, auth[d]] 
    ... /* other constraints */
}

/* True iff client "c" is allowed by the server */
pred checkAllowDeny[c : Client, allow : Allow, deny : Deny] {
    c.addr in allow.addrs or c.addr not in deny.addrs
}
Analysis
Analysis

Config Values | Data Elements | Policy

Concrete Input

extract

specify

feedback

Component Model

FOL formulas in Alloy

embed

Violation | Policy OK

Output

SAT | UNSAT

SAT Solver

compile $S \land \neg P$

Boolean formula

extract

specify

feedback
Analysis

Does this really work?
No, not right away.
Analysis

Size of SAT formula depends on amount of data

Data Elements

Concrete Input

Component Model

FOL formulas in Alloy

embed

compile

SAT Solver

Boolean formula

extract

specify

feedback

Config Values

Policy

Violation

Policy OK

Output

Config Values

Policy

Violation

Policy OK

Output

Size of SAT formula depends on amount of data.
Policy-Specific Abstraction

No need to analyze all data!
Policy mentions only some subset of data

Policy:
Only clients with password can access www/private
Slicing

Policy mentions only some subset of data
Prune away irrelevant parts

Policy:
Only clients with password can access www/private

Diagram:
```
/usr/bob/www
  └── private
    └── docs
      ├── arc.zip
      ├── foo.txt
      └── ris.vid
  └── public
    ├── meet
    └── notes
```
Equivalence Partitioning

Some data objects are equivalent w.r.t. config parameters
Group objects into **equivalent classes**
Pick one from each class

Assume \{foo.txt, bar.png, ris.vid\} have the same file permissions
Equivalence Partitioning

Some data objects are equivalent w.r.t. config parameters
Group objects into **equivalent classes**
Pick one from each class

Pick a representative object from the equivalence class
Equivalence Partitioning

Some data objects are equivalent w.r.t. config parameters
Group objects into **equivalent classes**
Pick one from each class
Case Study
Apache Configuration Analyzer

Implementation
~2000 LOC Alloy, 4000 LOC Python
Security-related directives in Apache 2.2
AFS, UNIX file systems

Policy Templates
Only clients on [whitelist] can access [resources]
Only clients not on [blacklist] can access [resources]
Only clients with [password/certificate] can access [resources]
No clients should be able to access [resources]
Sample Output

The Apache Configuration Analyzer started successfully.
An Apache config file successfully 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
Experiment on CSAIL Sites

Experiment
Analyzed 10 sites hosted on CSAIL servers
Targeted “sensitive” data (courses, department page, etc.)
24 policy instances: Asked if available, and guessed otherwise

(TIG & CSAIL members notified prior to the experiment)
### Results

<table>
<thead>
<tr>
<th>Site</th>
<th>Policy</th>
<th># Obj. Original</th>
<th># Obj. Reduced</th>
<th>Extraction Time (s)</th>
<th>Translation Time (s)</th>
<th>Solving Time (s)</th>
<th>Total Time (s)</th>
<th>Violation Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P3</td>
<td>954</td>
<td>58</td>
<td>8.721</td>
<td>56.996</td>
<td>2.006</td>
<td>67.723</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>954</td>
<td>89</td>
<td>12.129</td>
<td>270.01</td>
<td>31.486</td>
<td>313.625</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>P3</td>
<td>272</td>
<td>27</td>
<td>2.717</td>
<td>3.437</td>
<td>0.224</td>
<td>6.378</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>272</td>
<td>33</td>
<td>2.904</td>
<td>8.869</td>
<td>2.181</td>
<td>13.954</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>P3</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.895</td>
<td>0.690</td>
<td>3.997</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.970</td>
<td>0.782</td>
<td>4.164</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>P3</td>
<td>895</td>
<td>60</td>
<td>5.304</td>
<td>82.274</td>
<td>2.582</td>
<td>90.16</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>895</td>
<td>76</td>
<td>5.463</td>
<td>172.799</td>
<td>5.750</td>
<td>184.012</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>895</td>
<td>44</td>
<td>2.947</td>
<td>28.452</td>
<td>2.128</td>
<td>33.527</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>P3</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.112</td>
<td>0.361</td>
<td>2.793</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.255</td>
<td>0.463</td>
<td>3.038</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>P3</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>1.741</td>
<td>0.228</td>
<td>2.421</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>192</td>
<td>39</td>
<td>4.605</td>
<td>18.624</td>
<td>1.821</td>
<td>25.050</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>0.362</td>
<td>0.153</td>
<td>0.967</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>P3</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.652</td>
<td>3.010</td>
<td>23.730</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>364</td>
<td>40</td>
<td>8.522</td>
<td>19.931</td>
<td>4.868</td>
<td>33.321</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.592</td>
<td>3.241</td>
<td>23.901</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>P3</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>18.162</td>
<td>5.124</td>
<td>25.259</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>17.912</td>
<td>4.750</td>
<td>24.635</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>P1</td>
<td>338</td>
<td>27</td>
<td>1.958</td>
<td>18.070</td>
<td>1.807</td>
<td>21.835</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P3</td>
<td>338</td>
<td>82</td>
<td>9.073</td>
<td>211.899</td>
<td>6.695</td>
<td>227.667</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>P3</td>
<td>163</td>
<td>35</td>
<td>2.238</td>
<td>12.376</td>
<td>1.491</td>
<td>16.105</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>163</td>
<td>74</td>
<td>4.8465</td>
<td>177.124</td>
<td>17.597</td>
<td>199.568</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>163</td>
<td>31</td>
<td>1.668</td>
<td>7.189</td>
<td>2.685</td>
<td>11.542</td>
<td>No</td>
</tr>
</tbody>
</table>
Three of the site owners have applied the fixes, one of which had a symbolic link that was pointing directly to a directory traversal, allowing anyone to access the files under the directory that were intended to be used by students to submit their information.

Results

<table>
<thead>
<tr>
<th>Site</th>
<th>Property</th>
<th># Obj. Original</th>
<th># Obj. Reduced</th>
<th>Extraction Time (s)</th>
<th>Translation Time (s)</th>
<th>Solving Time (s)</th>
<th>Total Time (s)</th>
<th>Violation Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(P_3)</td>
<td>954</td>
<td>58</td>
<td>8.721</td>
<td>56.996</td>
<td>2.006</td>
<td>67.723</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>954</td>
<td>89</td>
<td>12.129</td>
<td>270.01</td>
<td>31.486</td>
<td>313.625</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>(P_3)</td>
<td>272</td>
<td>27</td>
<td>2.717</td>
<td>3.437</td>
<td>0.224</td>
<td>6.378</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>272</td>
<td>33</td>
<td>2.904</td>
<td>8.869</td>
<td>2.181</td>
<td>13.954</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>(P_3)</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.895</td>
<td>0.690</td>
<td>3.997</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_6)</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.970</td>
<td>0.782</td>
<td>4.164</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>(P_3)</td>
<td>895</td>
<td>60</td>
<td>5.304</td>
<td>82.274</td>
<td>2.582</td>
<td>90.16</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>895</td>
<td>76</td>
<td>5.463</td>
<td>172.799</td>
<td>5.750</td>
<td>184.012</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_6)</td>
<td>895</td>
<td>44</td>
<td>2.947</td>
<td>28.452</td>
<td>2.128</td>
<td>33.527</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>(P_3)</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.112</td>
<td>0.361</td>
<td>2.793</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_6)</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.255</td>
<td>0.463</td>
<td>3.038</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>(P_3)</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>1.741</td>
<td>0.228</td>
<td>2.421</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>192</td>
<td>39</td>
<td>4.605</td>
<td>18.624</td>
<td>1.821</td>
<td>25.050</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>(P_6)</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>0.362</td>
<td>0.153</td>
<td>0.967</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_3)</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.652</td>
<td>3.010</td>
<td>23.730</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>364</td>
<td>40</td>
<td>8.522</td>
<td>19.931</td>
<td>4.868</td>
<td>33.321</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_5)</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.592</td>
<td>3.241</td>
<td>23.901</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>(P_3)</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>18.162</td>
<td>5.124</td>
<td>25.259</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_5)</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>17.912</td>
<td>4.750</td>
<td>24.635</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>(P_1)</td>
<td>338</td>
<td>27</td>
<td>1.958</td>
<td>18.070</td>
<td>1.807</td>
<td>21.835</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>(P_3)</td>
<td>338</td>
<td>82</td>
<td>9.073</td>
<td>211.899</td>
<td>6.695</td>
<td>227.667</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>(P_3)</td>
<td>163</td>
<td>35</td>
<td>2.238</td>
<td>12.376</td>
<td>1.491</td>
<td>16.105</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_4)</td>
<td>163</td>
<td>74</td>
<td>4.8465</td>
<td>177.124</td>
<td>17.597</td>
<td>199.568</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(P_6)</td>
<td>163</td>
<td>31</td>
<td>1.668</td>
<td>7.189</td>
<td>2.685</td>
<td>11.542</td>
<td>No</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>Site</th>
<th>Property</th>
<th># Obj. Original</th>
<th># Obj. Reduced</th>
<th>Extraction Time (s)</th>
<th>Translation Time (s)</th>
<th>Solving Time (s)</th>
<th>Total Time (s)</th>
<th>Violation Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P3</td>
<td>954</td>
<td>58</td>
<td>8.721</td>
<td>56.996</td>
<td>2.006</td>
<td>67.723</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>954</td>
<td>89</td>
<td>12.129</td>
<td>270.01</td>
<td>31.486</td>
<td>313.625</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>P3</td>
<td>272</td>
<td>27</td>
<td>2.717</td>
<td>3.437</td>
<td>0.224</td>
<td>6.378</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>272</td>
<td>33</td>
<td>2.904</td>
<td>8.869</td>
<td>2.181</td>
<td>13.954</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>P3</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.895</td>
<td>0.690</td>
<td>3.997</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>88</td>
<td>21</td>
<td>1.412</td>
<td>1.970</td>
<td>0.782</td>
<td>4.164</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>P3</td>
<td>895</td>
<td>60</td>
<td>5.304</td>
<td>82.274</td>
<td>2.582</td>
<td>90.16</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>895</td>
<td>76</td>
<td>5.463</td>
<td>172.799</td>
<td>5.750</td>
<td>184.012</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>895</td>
<td>44</td>
<td>2.947</td>
<td>28.452</td>
<td>2.128</td>
<td>33.527</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>P3</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.112</td>
<td>0.361</td>
<td>2.793</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>55</td>
<td>17</td>
<td>1.32</td>
<td>1.255</td>
<td>0.463</td>
<td>3.038</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>P3</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>1.741</td>
<td>0.228</td>
<td>2.421</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>192</td>
<td>39</td>
<td>4.605</td>
<td>18.624</td>
<td>1.821</td>
<td>25.050</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>192</td>
<td>11</td>
<td>0.452</td>
<td>0.362</td>
<td>0.153</td>
<td>0.967</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>P3</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.652</td>
<td>3.010</td>
<td>23.730</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>364</td>
<td>40</td>
<td>8.522</td>
<td>19.931</td>
<td>4.868</td>
<td>33.321</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>364</td>
<td>35</td>
<td>9.068</td>
<td>11.592</td>
<td>3.241</td>
<td>23.901</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>P3</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>18.162</td>
<td>5.124</td>
<td>25.259</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>64</td>
<td>38</td>
<td>1.973</td>
<td>17.912</td>
<td>4.750</td>
<td>24.635</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>P1</td>
<td>338</td>
<td>27</td>
<td>1.958</td>
<td>18.070</td>
<td>1.807</td>
<td>21.835</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>P3</td>
<td>338</td>
<td>82</td>
<td>9.073</td>
<td>211.899</td>
<td>6.695</td>
<td>227.667</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>P3</td>
<td>163</td>
<td>35</td>
<td>2.238</td>
<td>12.376</td>
<td>1.491</td>
<td>16.105</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>163</td>
<td>74</td>
<td>4.8465</td>
<td>177.124</td>
<td>17.597</td>
<td>199.568</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>163</td>
<td>31</td>
<td>1.668</td>
<td>7.189</td>
<td>2.685</td>
<td>11.542</td>
<td>No</td>
</tr>
</tbody>
</table>
Lessons from Experiment

Users make simple mistakes
Most likely because they never thought about them

Users do care about security
Given a demonstration of an exploit
Discussion
Previous Works

**COPS** [Farmer & Spafford 90], **Kuang** [Baldwin 94]
Database of security rules, extensible
Common vulnerabilities in OS/networks

**MulVal** [Ou & Appel 05]
Network configuration analysis with Datalog

**Margrave** [Fisler, et al. 05]
Analysis of complex access control policies with SAT

**Network security management** [Guttman & Herzog 05]
Composition of models for end-to-end security analysis
Isn’t it a lot of work to build Alloy models? Is it worth it?
Yes, but it needs to be done only once.
Discussion

Isn’t it a lot of work to build Alloy models? Is it worth it?
Yes, but it needs to be done only once.

What if the model is wrong?
Possible (likely); guarantee as strong as the model.
Isn’t it a lot of work to build Alloy models? Is it worth it?
Yes, but it needs to be done only once.

What if the model is wrong?
Possible (likely); guarantee as strong as the model.

Why not use a checklist?
Often too strict!
Discussion

Isn’t it a lot of work to build Alloy models? Is it worth it?
Yes, but it needs to be done only once.

What if the model is wrong?
Possible (likely); guarantee as strong as the model

Why not use a checklist?
Often too strict!

Why not just write a script?
You could (in principle), but it’s more work & less robust.
Discussion

Isn’t it a lot of work to build Alloy models? Is it worth it?
Yes, but it needs to be done only once.

What if the model is wrong?
Possible (likely); guarantee as strong as the model

Why not use a checklist?
Often too strict!

Why not just write a script?
You could (in principle), but it’s more work & less robust.

How applicable is this approach to others?
We’re still figuring out; need more case studies.
Ongoing Work

**Multi-Level Security Analysis**
Not just configuration, but issues in code too

**Generating Explanations for Security Failures**
Analysis not enough, users can’t interpret output

**Evolving Threat Model**
How to keep the models up-to-date?