The Plan

- Breaks found
- Secure development reflection
- Security best practices
- In-class survey time!
Break Review

- Lots of crashes!
- Incorrect implementation of spec
- Security properties not specifically stated
- Non-logic problems
Incorrect Implementation

- Admin password not checked
- Exit and set default delegator by non-admin
- Default delegator rights provided after creation
- Too many rights for the hub (e.g., read)
- Incorrect delegation checks
- No rollback on failure
- Rule permission checking
Incorrect Implementation

- Admin password not checked
- Exit and set default delegator by non-admin
- Default delegator rights provided after creation
- Too many rights for the hub (e.g., read)
- Incorrect delegation checks
- No rollback on failure
- Rule permission checking

Who do you think this is?
Un(der)specified

• Anyone login
• Leak of variable type information
• Circular delegation
• Multiple delegation paths
Circular Delegation

admin
Circular Delegation

admin

bob                charlie

alice
Circular Delegation

admin

bob  alice  charlie
Circular Delegation

```python
hasPermission(var, principal, right):
    if principal is "admin>:</p>
        return true
    else:
        delegator = getDelegator(var, principal, right)
        if there is a delegator:
            return hasPermission(var, delegator, right)
        else:
            return false
```
Multiple Paths

admin

bob

alice

charlie
Multiple Paths

admin

bob

alice

charlie
Multiple Paths

bob

admin

alice

charlie
Multiple Paths

admin

bob

alice

charlie

DENIED
Non-logic problems

- Bugs in libraries used
- Integer overflow
- Code injection
as principal admin password "admin" do
  create principal bob "bob"
  set count = 1
  set delegation count admin write -> bob
  return count

***
as principal admin password "admin" do
  create principal bob "bob"
  set count = 1
  set delegation count admin write -> bob
  return count
***
as principal bob password "bob" do
  set count = 2147483647
  return x
***
as principal admin password "admin" do
create principal bob "bob"
set count = 1
set delegation count admin write -> bob
return count
***
as principal bob password "bob" do
set count = 0b1111111111111111111111111111111
return x
***
as principal admin password "admin" do
create principal bob "bob"
set count = 1
set delegation count admin write -> bob
return count
***
as principal bob password "bob" do
set count = 0b1111111111111111111111111111111
return x
***
as principal admin password "admin" do
set count = count + 1
if count < 0 then set door = 0
return count
***
as principal admin password "admin" do
create principal bob "bob"
set count = 1
set delegation count admin write -> bob
return count
***
as principal bob password “bob" do
set count = 0b11111111111111111111111111111111
return x
***
as principal admin password "admin" do
set count = 0b10000000000000000000000000000000 (-1)
if count < 0 then set door = 0
return count
***
Code Injection

```python
os.system("<call compiled executable> " + port + " " + config + " " + admin + " " + hub)

output = eval(<expr>)
```
Code Injection

```
import os

os.system("<call compiled executable> " + port + " " + config + " " + admin + " " + hub)

./server 1024 config.json ";ls " ";rm -rf /"

output = eval(<expr>)
```
Code Injection

```
os.system("<call compiled executable> " + port + " " + config + " " + admin + " " + hub)

./server 1024 config.json "; ls "; rm -rf /");

output = eval(<expr>)

set x = __import__('os').system('rm -rf /') + 1
```
Secure Development Reflection

• What worked?
• What didn’t work?
• What would you do differently?
Secure Development Reflection

• What worked?
• What didn’t work?
• What would you do differently?
Secure Development Reflection

• What worked?

• What didn’t work?

• What would you do differently?
Secure Development Reflection

• What worked?
• What didn’t work?
• What would you do differently?
Best Practices

• Limit security-specific code
• Graceful error handling (and logging)
• In-depth design/threat modeling
• Test cases are important!
• Code review!
In-class Survey Time!

- End-of-course survey
  - link emailed to you
- Work on final design document