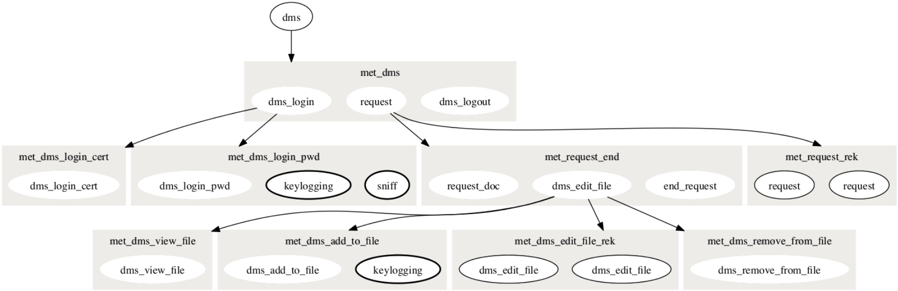


Hybrid Planning in Cyber Security Applications

Bachelor Thesis

Louisa Pragst
20. November 2013

Overview

- Transformation of a POCL domain into a hybrid domain
- Establishing a hierarchy
- Converting unsupported language features

Why Cyber Security Applications?

- Growing dependence on digital networks
- Protection is difficult
  - Little effort
  - Network size and complexity
  - Intruders outnumber defenders
  - Unknown attacks
- Methods of the artificial intelligence can be used to support defenders

BAMS

- Behavioral Adversary Modeling System
- Finding ways to attack a network
- Focus on malicious insiders
- Supporting network admins to identify vulnerabilities

Scope of BAMS

- Physical
  - Manages the interrelation between rooms, doors and keys
  - Keeps track of the position of persons and things
- Email
  - Sending and receiving emails
  - Email contains information, or programs as attachment
- Encryption
  - Encryption and decryption of files
  - Signing of emails

Scope of BAMS

- Keylogging
  - Keyloggers can read keyboard input when attached to a computer
- Malware
  - Defines different kinds of malware and its usage
  - Defines anti-virus program
- DMS
  - Document Management System
  - Enables changing of files by different users
  - Changing file permissions for files managed by the DMS

Scope of BAMS

- Network
  - Defines interconnectedness between computers
  - Defines firewalls
  - Enables sniffing
- Process
  - Login and logout
  - Starting and ending programs
  - Editing files
  - Changing file permissions

Hybrid Planning

- Causal reasoning as in POCL planning
- Task decomposition as in hierarchical planning
- Decomposition axioms
- Problem goals can be given as attributes that are to be achieved or as initial task networks

Why Hybrid Planning?

- Supporting admins with little modeling experience
- Using initial task networks:
  - To evaluate the vulnerability of a network against a specific kind of attack
  - To evaluate if a particular behavior facilitates attacks against a network

Bottom Up

- BAMS provides all primitive tasks
- Hierarchy can be deducted:
  - Uniting alternative courses of action
  - Integrating reoccurring actions
  - Abstracting an often used sequence of actions

Uniting alternative Courses of Action

dms_login_cert

PRE
running_prog(?dms_prog, ?chost, ?cuid)
¬dms_established()
running_prog(?dms_server, ?shost, ?suid)
reachable(?chost, ?shost, ?firewall)
fw_allows(?firewall, HTTPS)
has_cert(?cert, ?cuid)
cert_installed(?cert, ?chost)

POST
dms_established()

dms_login_pwd

PRE
running_prog(?dms_prog, ?chost, ?cuid)
¬dms_established()
running_prog(?dms_server, ?shost, ?suid)
reachable(?chost, ?shost, ?firewall)
fw_allows(?firewall, HTTPS)
knows(?human, ?info)
dms_password(?shost, ?info, ?cuid)

POST
dms_established()
keylog(?info, ?chost)
now_sniffing(?info, ?chost, ?shost)

Uniting alternative Courses of Action

dms_login

PRE
running_prog(?dms_prog, ?chost, ?cuid)
¬dms_established()
running_prog(?dms_server, ?shost, ?suid)
reachable(?chost, ?shost, ?firewall)
fw_allows(?firewall, HTTPS)
dms_login_possible(?chost, ?cuid, ?human)

POST
dms_established()

Uniting alternative Courses of Action

- Primitive tasks serve the same purpose
- Very few differences in pre-/postconditions
- Introduction of decomposition axiom and methods

Integrating reoccurring Actions

- Recursion
- Used to handle loops and fully integrate them in the hierarchy
- One method to end the recursion, one for the recursive call

Abstracting an often used Sequence of Actions

- Sequence of actions doesn’t allow variations or is often done in a specific way
- Ordering and variable constraints are important

Language Features in Question

- Conditional effects
- Existential quantifiers
- Universal quantifiers

Conditional Effects

- For $e_1 \Rightarrow e_2$, e_1 becomes part of the precondition and e_2 part of the postcondition
- If there were other effects, create a new task with $\neg e_1$ in its precondition. e_2 is no part of its postcondition.

Existential Quantifiers

- In precondition: Equivalent to parameter
- In postcondition: Not useful

Universal Quantifiers

- In precondition: Stepwise check relation for all matching constants
- In postcondition: Stepwise set relation for all affected constants
- Special cases allow different approaches:
  - Change model to avoid universal quantifiers
  - Relax universal quantifier

Universal Quantifiers - Stepwise

- Stop normal planning, set counter
- Mark processed constants, count down
- When all constants are processed, switch states
- Unmark constant, count second counter down
- Continue planning

Algorithms

- RePOP: POCL planning
- SHOP2: hierarchical planning
- Adjusted greedy algorithm
  - heuristic: 5* # abstract tasks flaws + # insert plan step modifications
  - flaw selection strategy: prefer abstract task, least cost flaw repair

Domain and Problem Configurations

- Problem without initial task network
- Problem with initial task network, domain without decomposition axioms
- Problem with initial task network, domain with decomposition axioms

Problems

- Storage of passwords
- Computer networks/Sniffing
- Vulnerability of programs

Findings

- Algorithm: Only greedy solves all configurations
- Expanded plans: Greedy expands plans faster
- Problem configuration determines, which algorithm works best
- Decomposition axioms aid in finding solutions

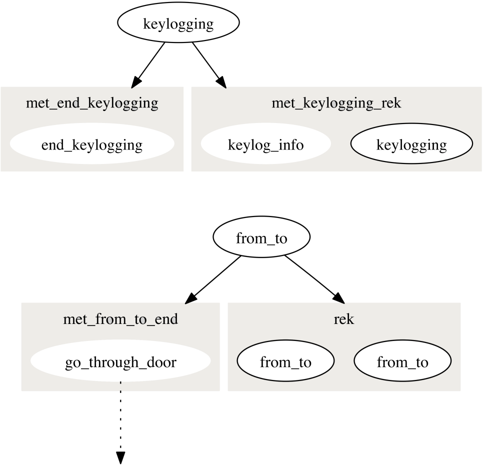
Network Security and Artificial Intelligence

- Modeling: Abstraction of the real world
- Determining relevant aspects is difficult: Unknown attacks
- Possible solution: Restrict represented scope

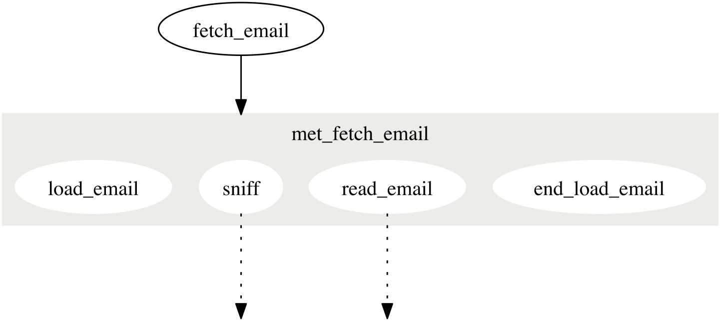
Conclusions

- Establish a hierarchy:
  - To unite alternative courses of action
  - To integrate reoccurring actions
  - To abstract an often used sequence of actions
- Convert unsupported language features:
  - Partially transfer conditional effects to the precondition
  - Rephrase to avoid universal quantifiers
  - Relax universal quantifiers
  - Process universal quantifiers stepwise
- In order to solve realistic network security problems the performance of problem solving needs to be improved

Integrating reoccurring Actions



Abstracting an often used Sequence of Actions



Universal Quantifiers - Change Model

- Try to change model in a way that avoids universal quantifiers
- For email:
  - Original relation: in_inbox(?email, ?uid) is set for all recipients, when email is sent
  - transmitted(?email) is set, when email is sent, is_recipient(?email, ?uid) is set earlier in the process
  - Eliminate in_inbox(?email, ?uid), use conjunction of transmitted(?email) and is_recipient(?email, ?uid)

Universal Quantifiers - Relax Universal Quantifier

- In postcondition the universal quantifier is not always necessary
- Set relation only for some constants, not necessarily all
- Counting and marking of constants no longer necessary

Importance of the chosen Algorithm

Average runtime in ms for "Storage of Passwords".

	no ITN, no DA	ITN, no DA	ITN, DA
Greedy	timeout	4748	3078

Average runtime in ms for "Computer Networks".

	no ITN, no DA	ITN, no DA	ITN, DA
Greedy	10908	13238	10188

Average runtime in ms for "Vulnerability of Programs".

	no ITN, no DA	ITN, no DA	ITN, DA
RePOP	2188	9726	7552
Greedy	160816	1102	664

Number of expanded Plans

Average number of expanded plans for “Storage of Passwords”.

	no ITN, no DA
RePOP	444111
SHOP2	25166
Greedy	1833023

Average number of expanded plans for “Vulnerability of Programs”.

	ITN, DA
RePOP	4393
SHOP2	8114
Greedy	247

Problem Configuration

Average runtime in ms for “Vulnerability of Programs”.

	no ITN, no DA	ITN, no DA	ITN, DA
RePOP	2188	9726	7552
Greedy	160816	1102	664

Average number of expanded plans for “Vulnerability of Programs”.

	no ITN, no DA	ITN, no DA	ITN, DA
RePOP	851	4208	4393
Greedy	529949	247	247

With or Without Decomposition Axioms

Average runtime in ms for "Storage of Passwords".

	ITN, no DA	ITN, DA
Greedy	4748	3078

Average runtime in ms for "Computer Networks".

	ITN, no DA	ITN, DA
Greedy	13238	10188

Average runtime in ms for "Vulnerability of Programs".

	ITN, no DA	ITN, DA
Greedy	1102	664