Run-Time Monitoring and Formal Analysis of Information Flows in Chromium

Lujo Bauer, **Shaoying Cai***, Limin Jia,
Timothy Passaro, Michael Stroucken, and Yuan Tian

Carnegie Mellon University
* Institute for Infocomm Research

Carnegie Mellon University
CyLab

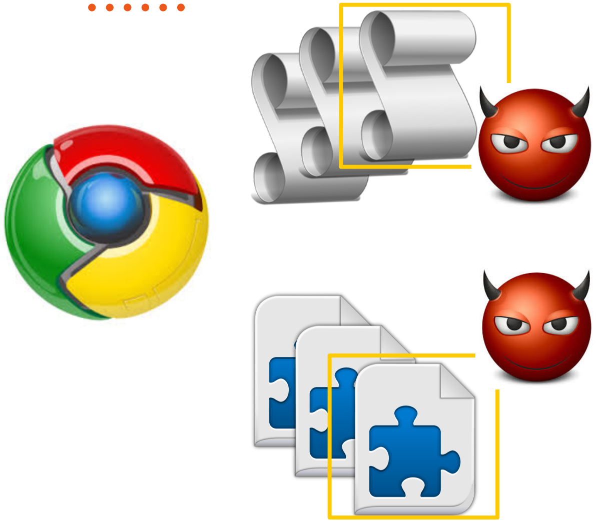
Agency for
Science, Technology
and Research
SINGAPORE

Websites increasingly host sensitive services

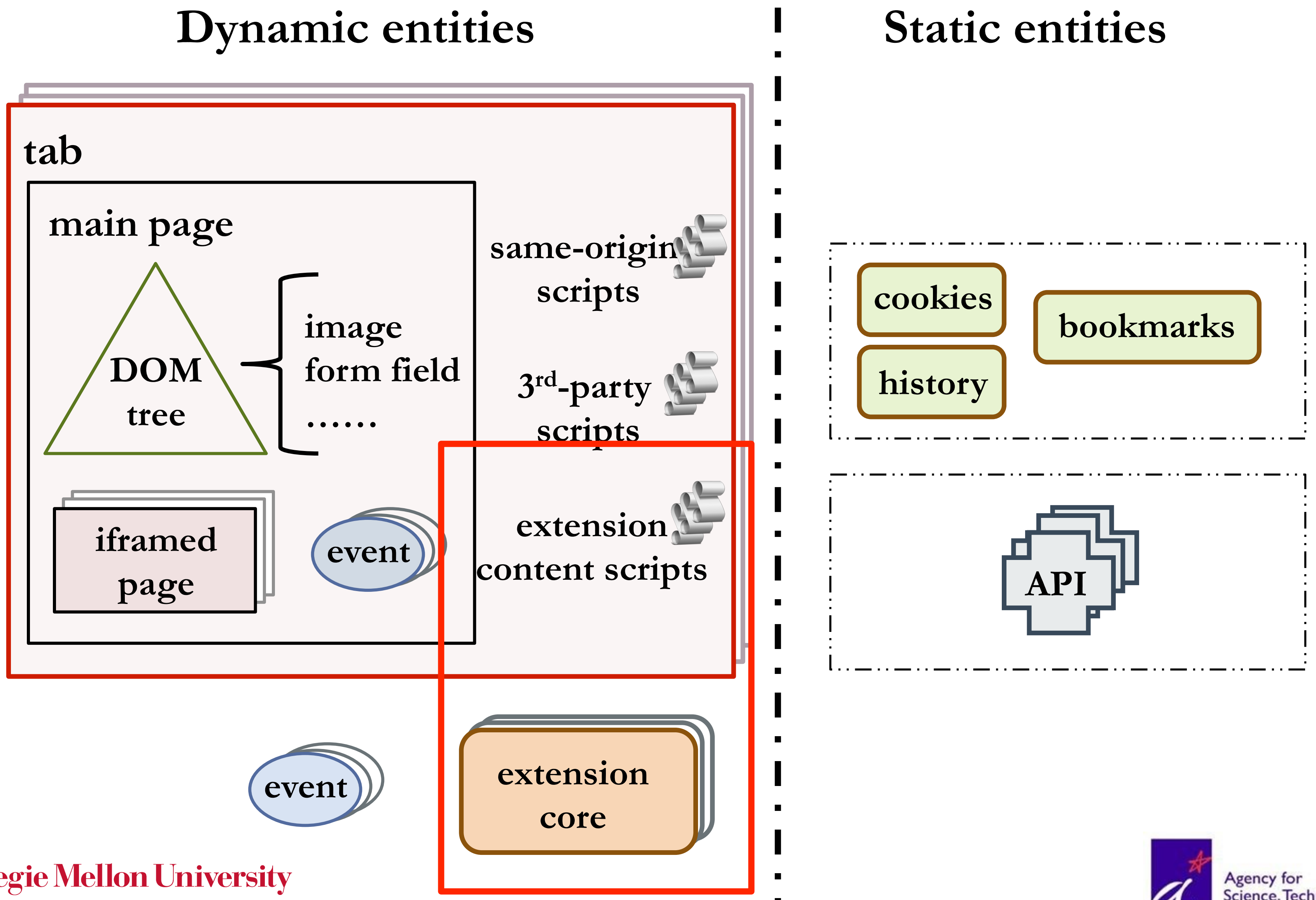
Passwords
Bank account numbers
Emails
……

Confidential data could be revealed to …

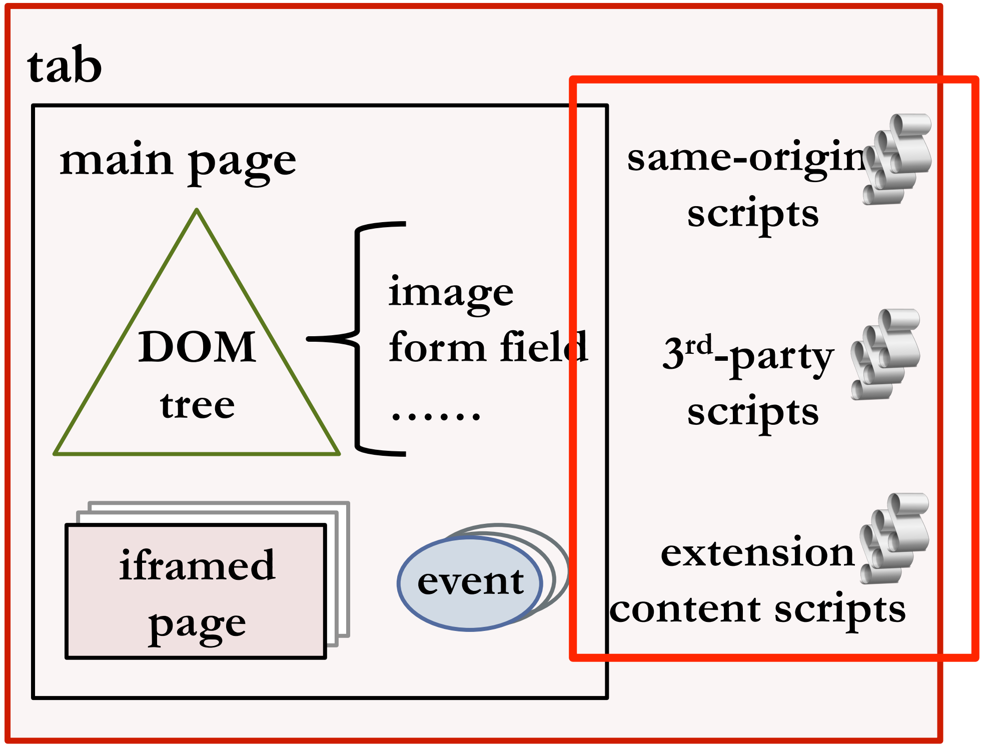
Passwords
Bank account numbers
Emails
……



Browser architecture & security mechanisms

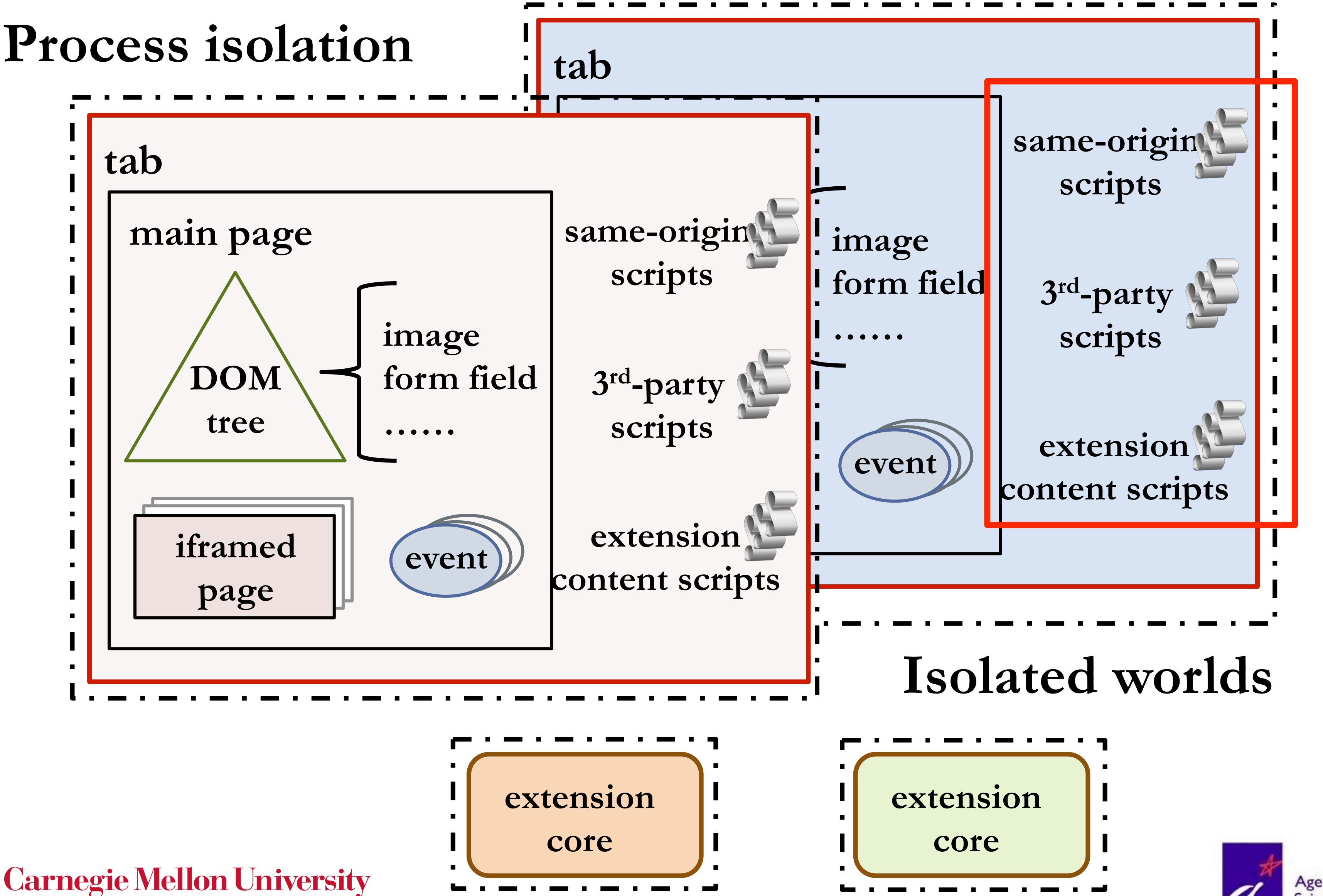


Browser architecture & security mechanisms



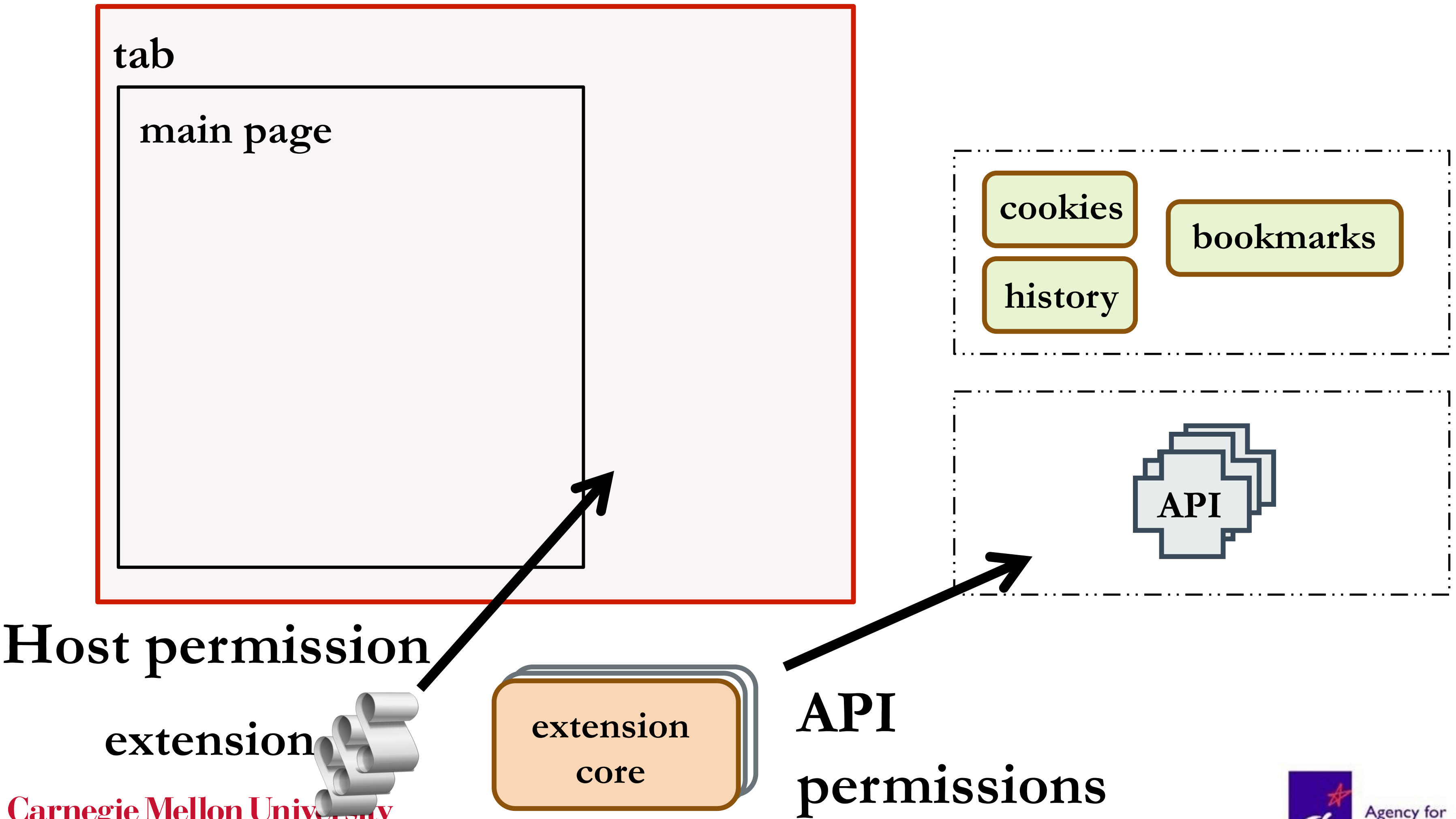
Same origin policy (SOP)

Browser architecture & security mechanisms

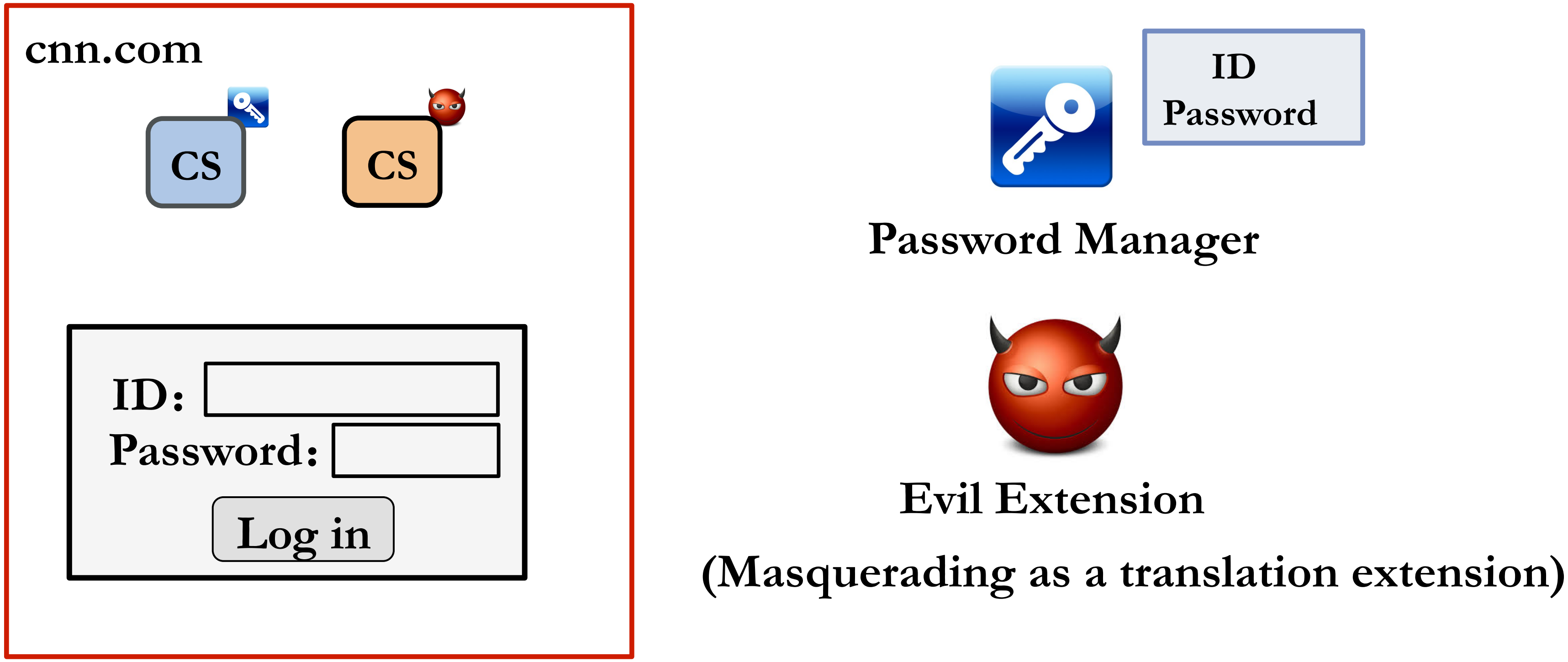


Browser architecture & security mechanisms

Permissions and content security policy (CSP)



Risks to users' data remain

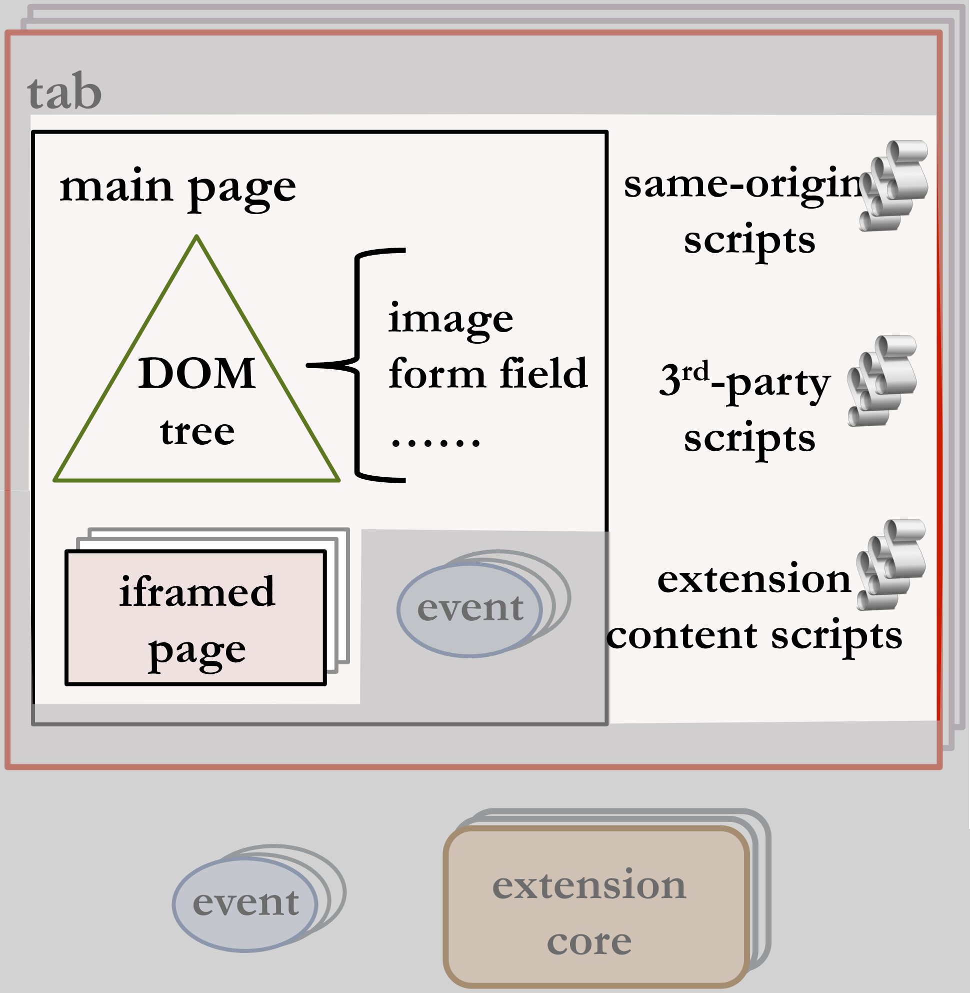


Password Manager

Evil Extension

(Masquerading as a translation extension)

Proposed solutions



JSFlow, …
[Arden et al. 2012,
Austin and Flanagan 2012,
Bichhawat et al. 2014,
Chugh et al. 2009,
Hedin et al. 2014,
Hedin and Sabelfeld 2012]

COWL, BFlow
[Stefan et al. 2014, Yip et al. 2009]

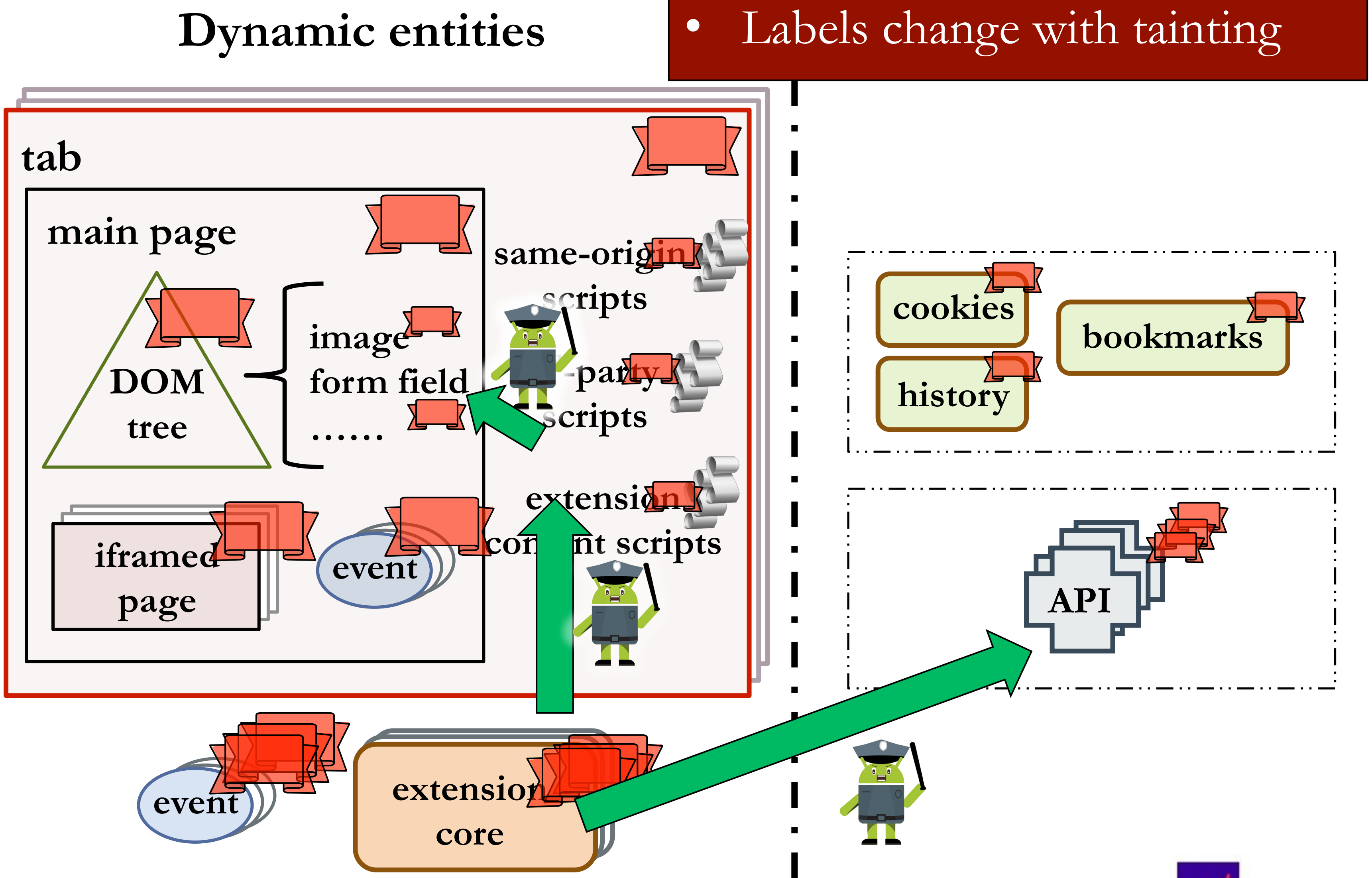
FlowFox
[Groef et al. 2012]

Our approach: Run-time information-flow control

- Uses coarse-grained dynamic taint tracking
- Encompasses wide range of browser entities
- Supports rich policy specification
- Formalized and proved noninterference
- Functional prototype implementation on Chromium

Our approach

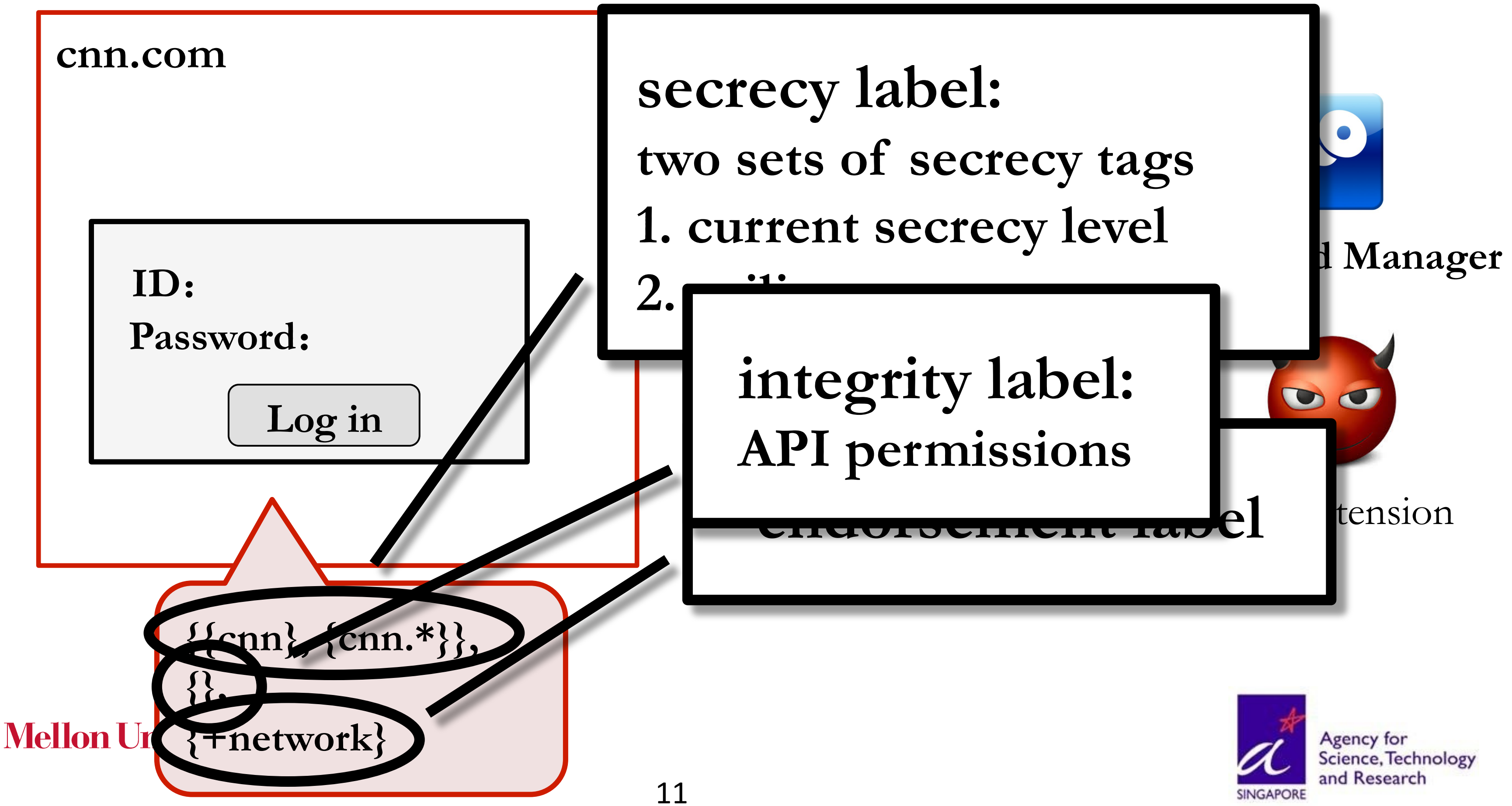
- Labels represent policy
- Communications are mediated
- Labels change with tainting



Carnegie Mellon University
CyLab

Agency for Science, Technology and Research
SINGAPORE

Example walkthrough



cnn.com

ID:

Password:

Log in

{{cnn}, {cnn.*}},

{},

{+network}

secrecy label:

two sets of secrecy tags

1. current secrecy level

2.

integrity label:

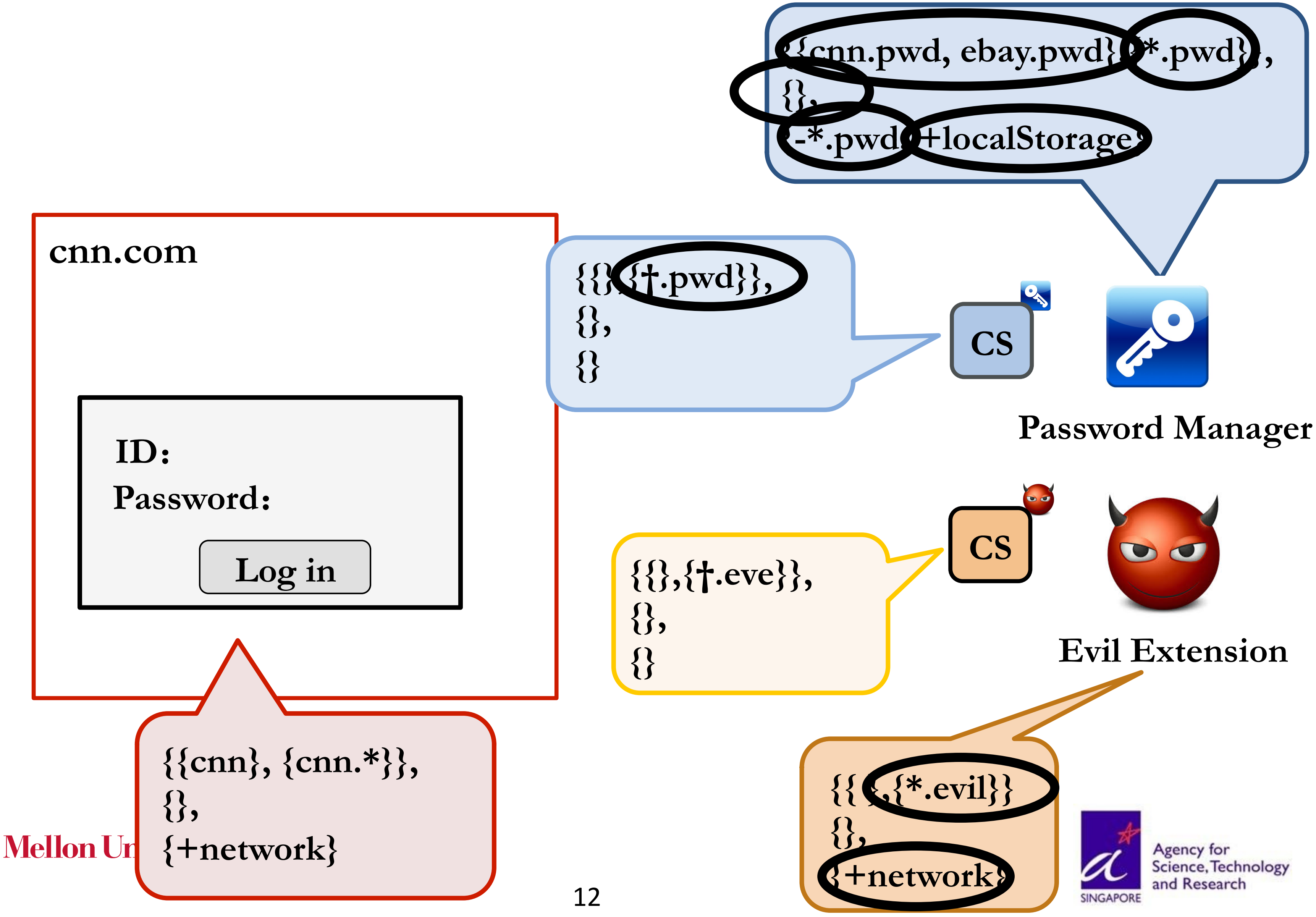
API permissions

el

d Manager

tension

Example: before injecting scripts



{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd,+localStorage}

cnn.com

ID:
Password:
Log in

{{},{†.pwd}},
{},
{}

CS

Password Manager

CS

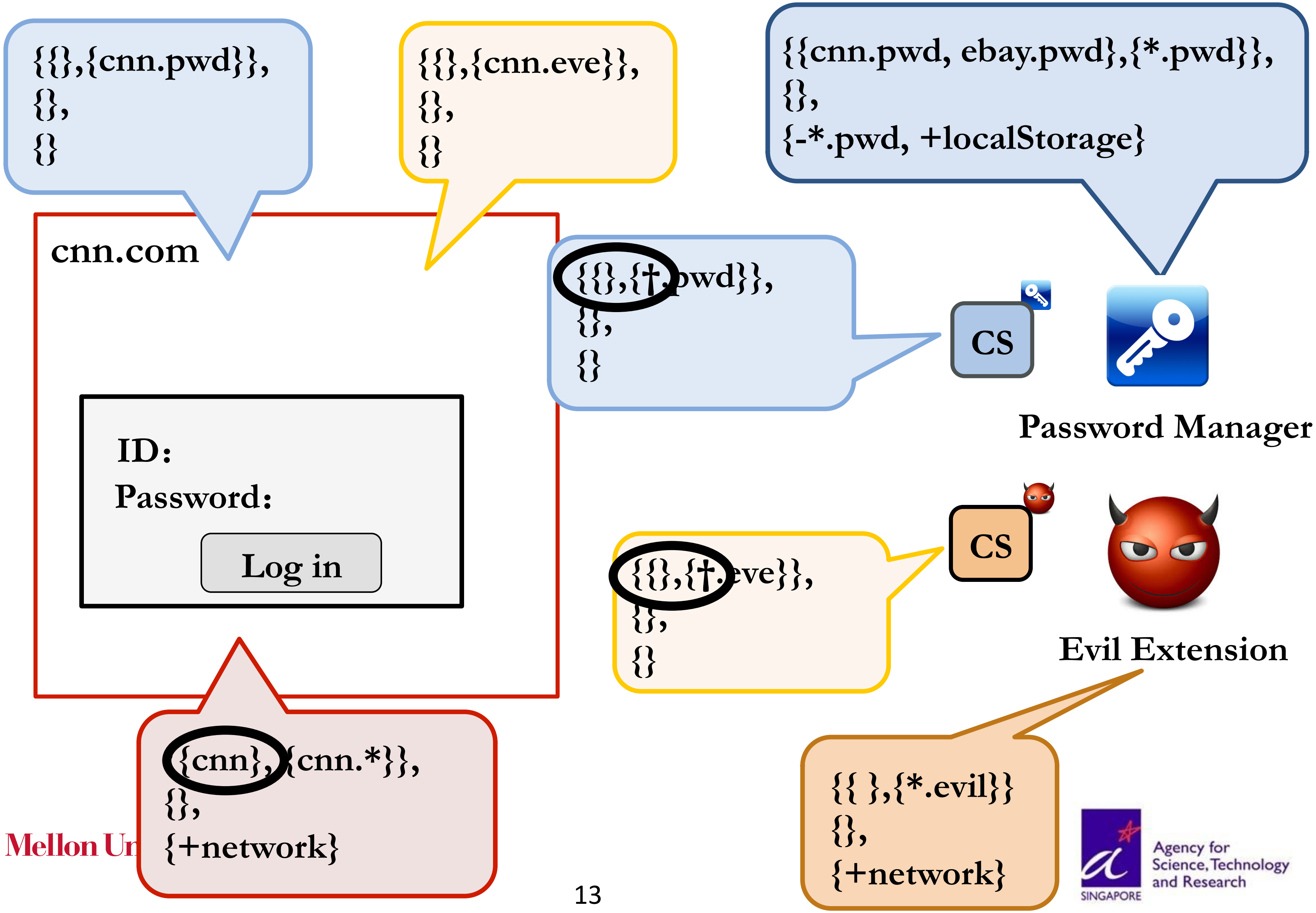
{{},{†.eve}},
{},
{}

Evil Extension

{{cnn}, {cnn.*}},
{},
{+network}

{{},{*.evil}}
{},
{+network}

Example: content scripts injected



{{},{cnn.pwd}},
{},
{}

{{},{cnn.eve}},
{},
{}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

cnn.com

ID:
Password:
Log in

{{},{†.pwd}},
{},
{}

CS

Password Manager

{{},{†.eve}},
{},
{}

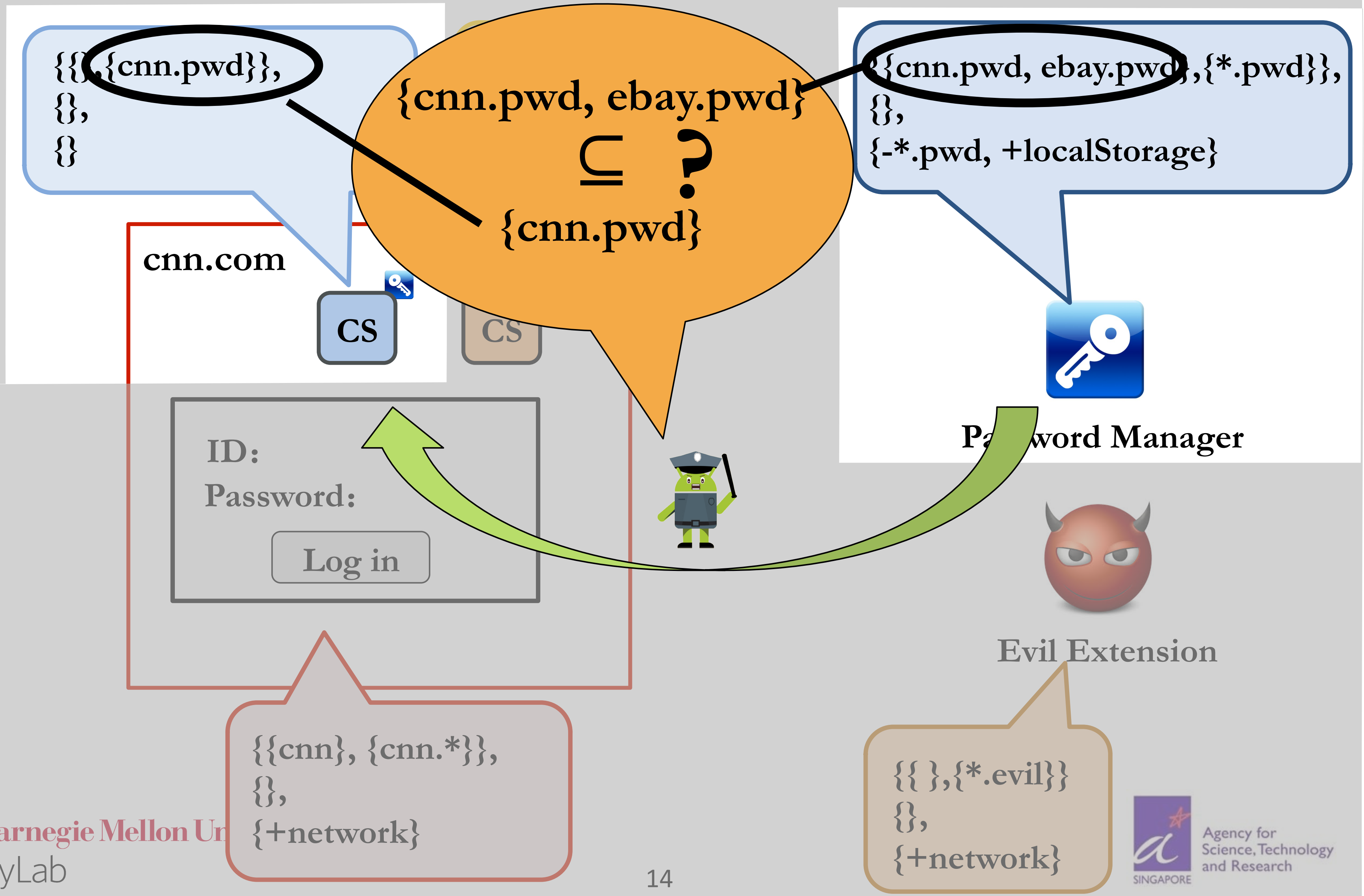
CS

Evil Extension

{{cnn},{cnn.*}},
{},
{+network}

{{ },{*.evil}}
{},
{+network}

Example: password sent to content script



{{},{cnn.pwd}},
{},
{}

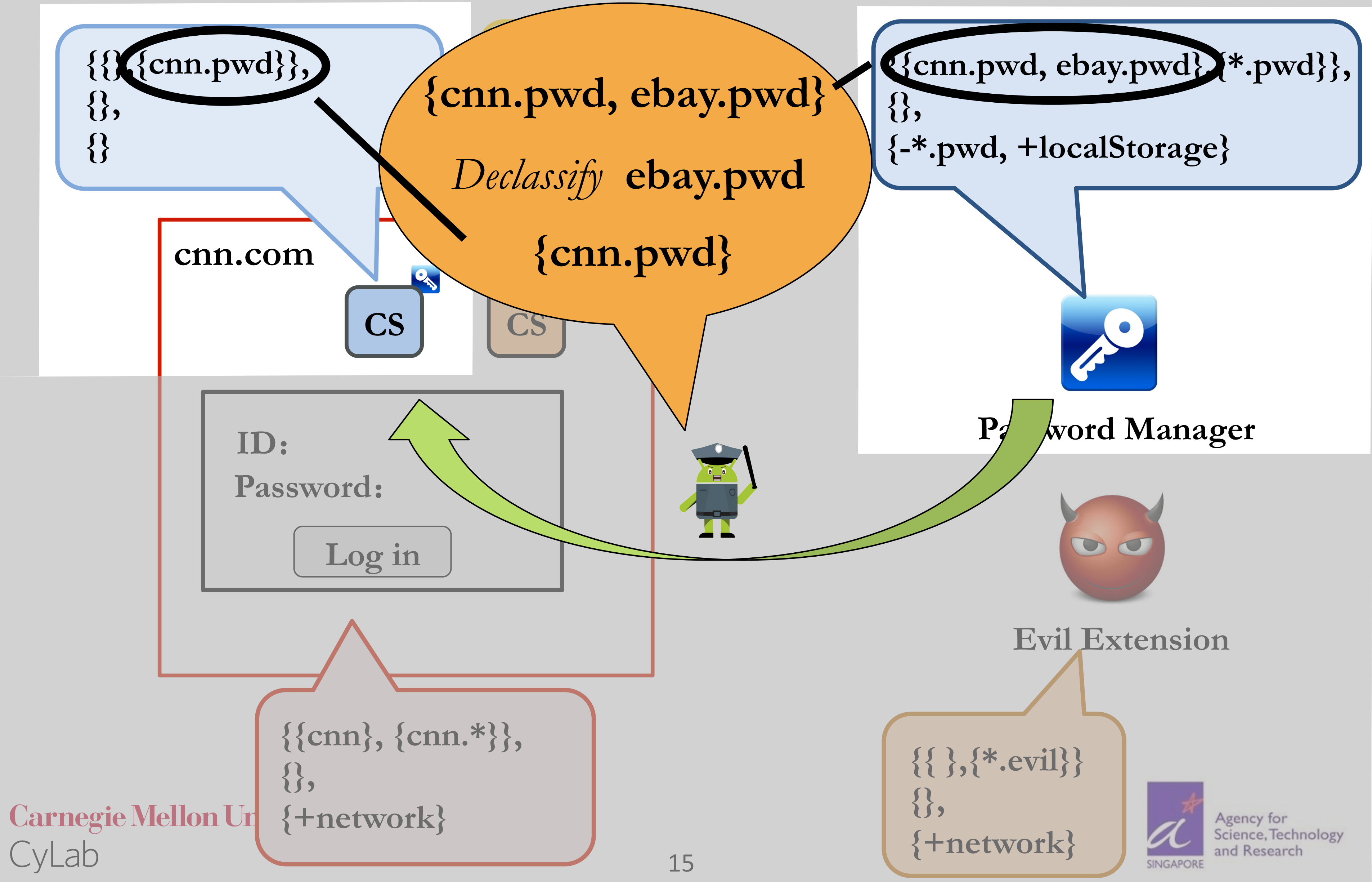
{cnn.pwd, ebay.pwd}
⊆ ?
{cnn.pwd}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

{{cnn}, {cnn.*}},
{},
{+network}

{{ },{*.evil}}
{},
{+network}

Example: password sent to content script



{{},{cnn.pwd}},
{},
{}

{cnn.pwd, ebay.pwd}

Declassify ebay.pwd

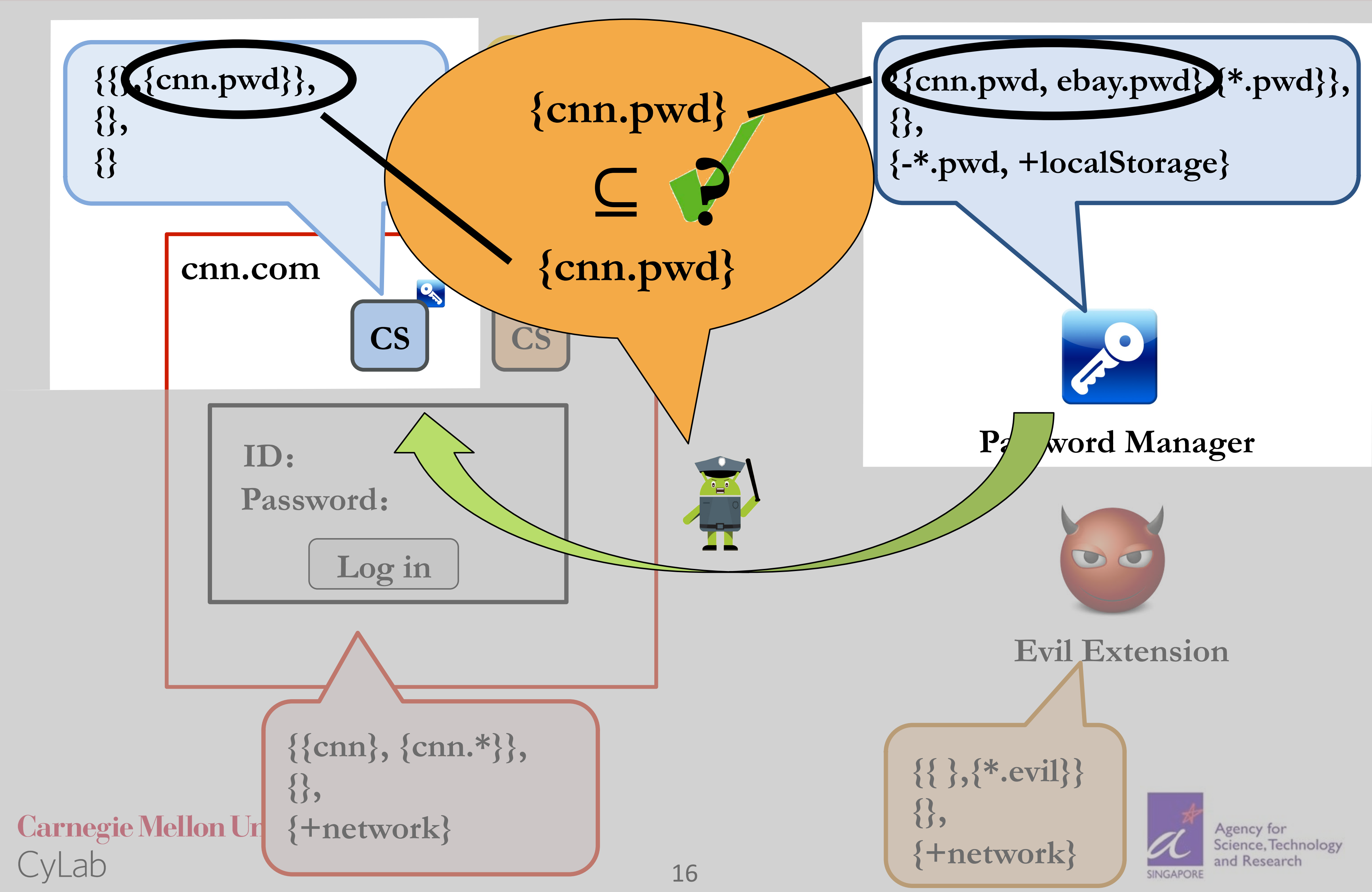
{cnn.pwd}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

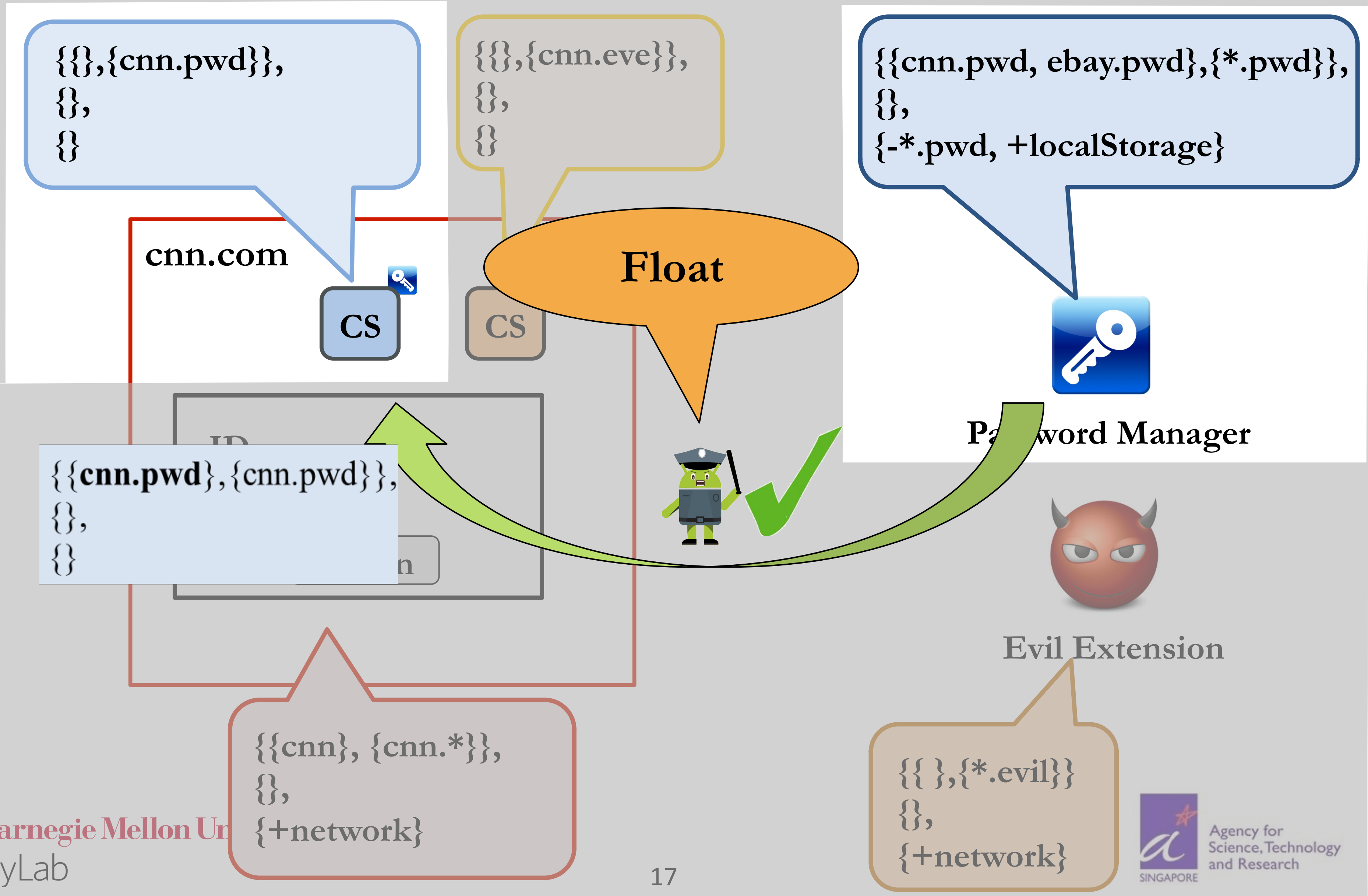
{{cnn}, {cnn.*}},
{},
{+network}

{{ },{*.evil}}
{},
{+network}

Example: password sent to content script



Example: password sent to content script



{{},{cnn.pwd}},
{},
{}

{{},{cnn.eve}},
{},
{}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

cnn.com

CS

CS

Float

Pa word Manager

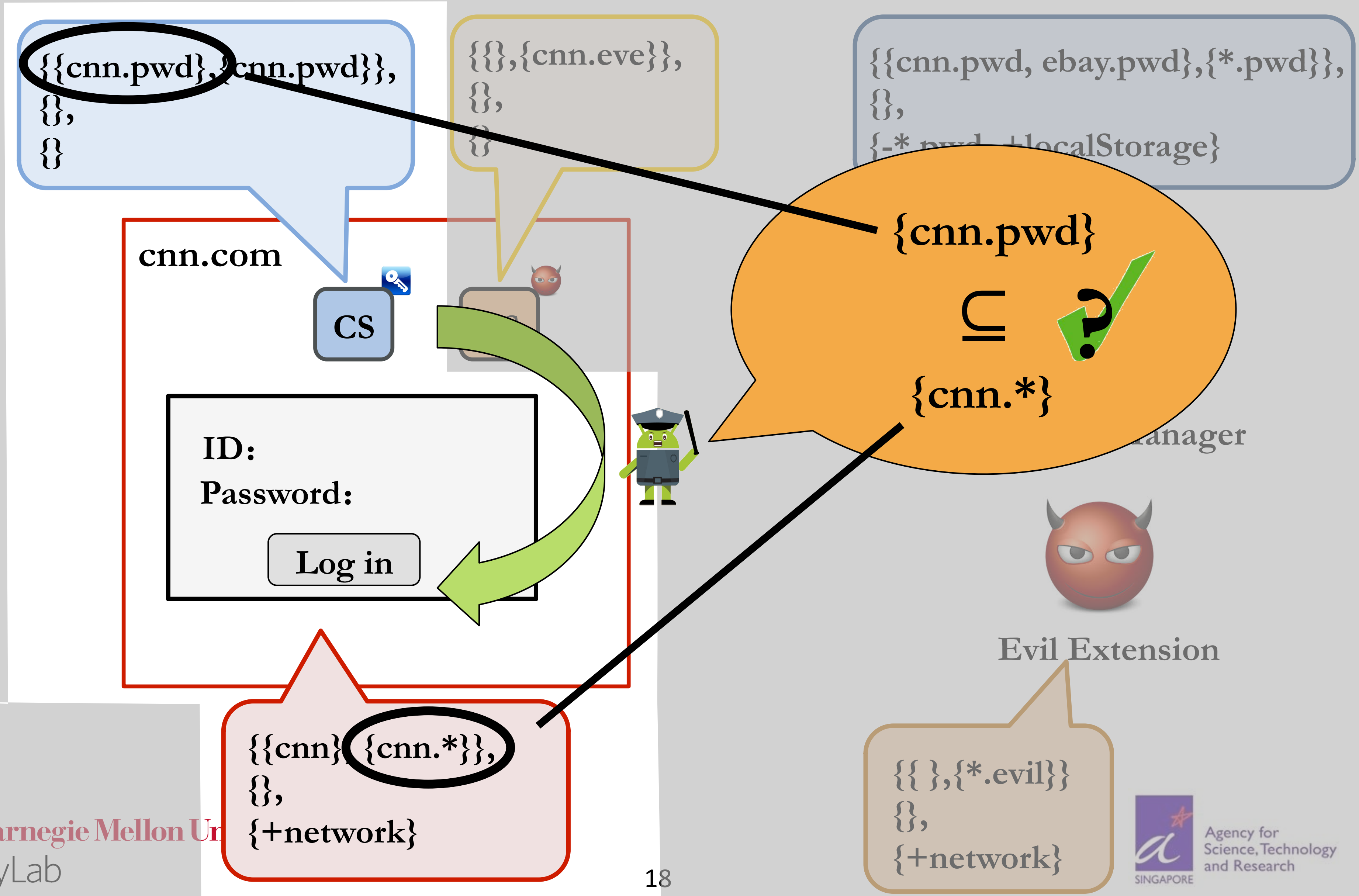
{{**cnn.pwd**},{cnn.pwd}},
{},
{}

Evil Extension

{{cnn}, {cnn.*}},
{},
{+network}

{{ },{*.evil}}
{},
{+network}

Example: password filled in



{{cnn.pwd},{cnn.pwd}},
{},
{}

{{},{cnn.eve}},
{},
{}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

cnn.com

CS

ID:

Password:

Log in

{cnn.pwd}

⊆ ? ✓

{cnn.*}

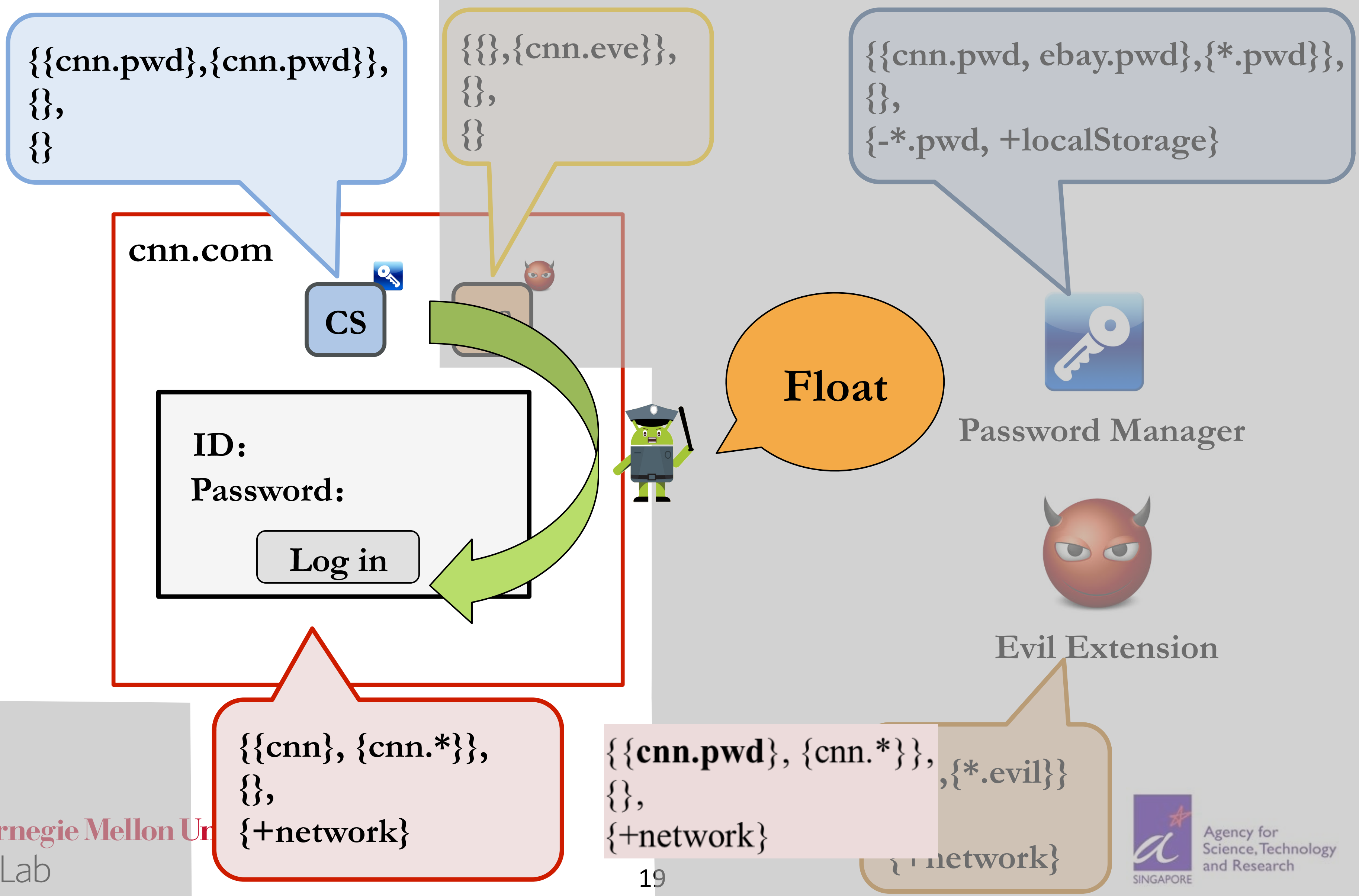
anager

Evil Extension

{{cnn},{cnn.*}},
{},
{+network}

{{ },{*.evil}}
{},
{+network}

Example: password filled in



{{cnn.pwd},{cnn.pwd}},
{},
{}

{{},{cnn.eve}},
{},
{}

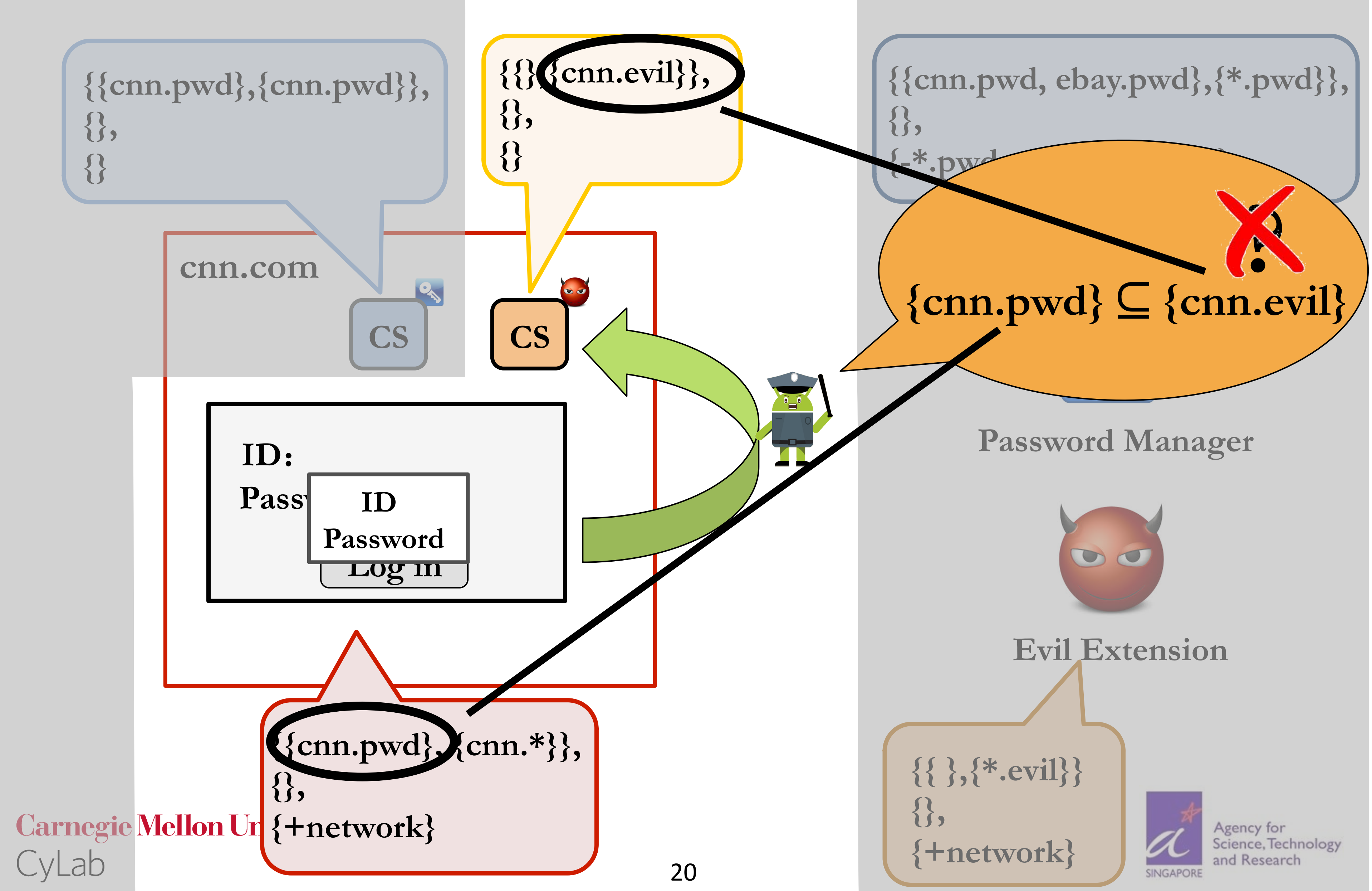
{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd, +localStorage}

{{cnn}, {cnn.*}},
{},
{+network}

{{**cnn.pwd**}, {cnn.*}},
{},
{+network}

,{*.evil}}
{+network}

Example: password stealing blocked



{{cnn.pwd},{cnn.pwd}},
{},
{}

cnn.com

CS

{{},{cnn.evil}},
{},
{}

CS

ID:

Pass

ID

Password

Log in

{{cnn.pwd},{cnn.*}},
{},
{+network}

{{cnn.pwd, ebay.pwd},{*.pwd}},
{},
{-*.pwd

{cnn.pwd} ⊆ {cnn.evil}

Password Manager

Evil Extension

{{ },{*.evil}}
{},
{+network}

Carnegie Mellon University CyLab

Agency for Science, Technology and Research SINGAPORE

Approximating existing browser policies

- SOP
- CSP
- postMessage
- iframe policies
- Domain relaxation

- Interesting composition issues when representing them all in one framework
  - E.g., conflicting policies of iframed page and parent page

Formal proof of security

- **Model enforcement mechanism**
  - In an extended version of Chromium
- **Specify security property – noninterference**
  - Attacker cannot learn any information about secrets prohibited by policies
- **Proof of noninterference**
  - Provides assurance of the model’s correctness

Limitations

- **Trace-based noninterference**
  - Attacker may have more knowledge than traces
  - Allows certain implicit flows
- **To achieve stronger formal security guarantees:**
  - Make scheduler less predictable
  - Non-determinism or probabilistic execution
  - Secure multi-execution
  - Stronger notions of noninterference
  - …

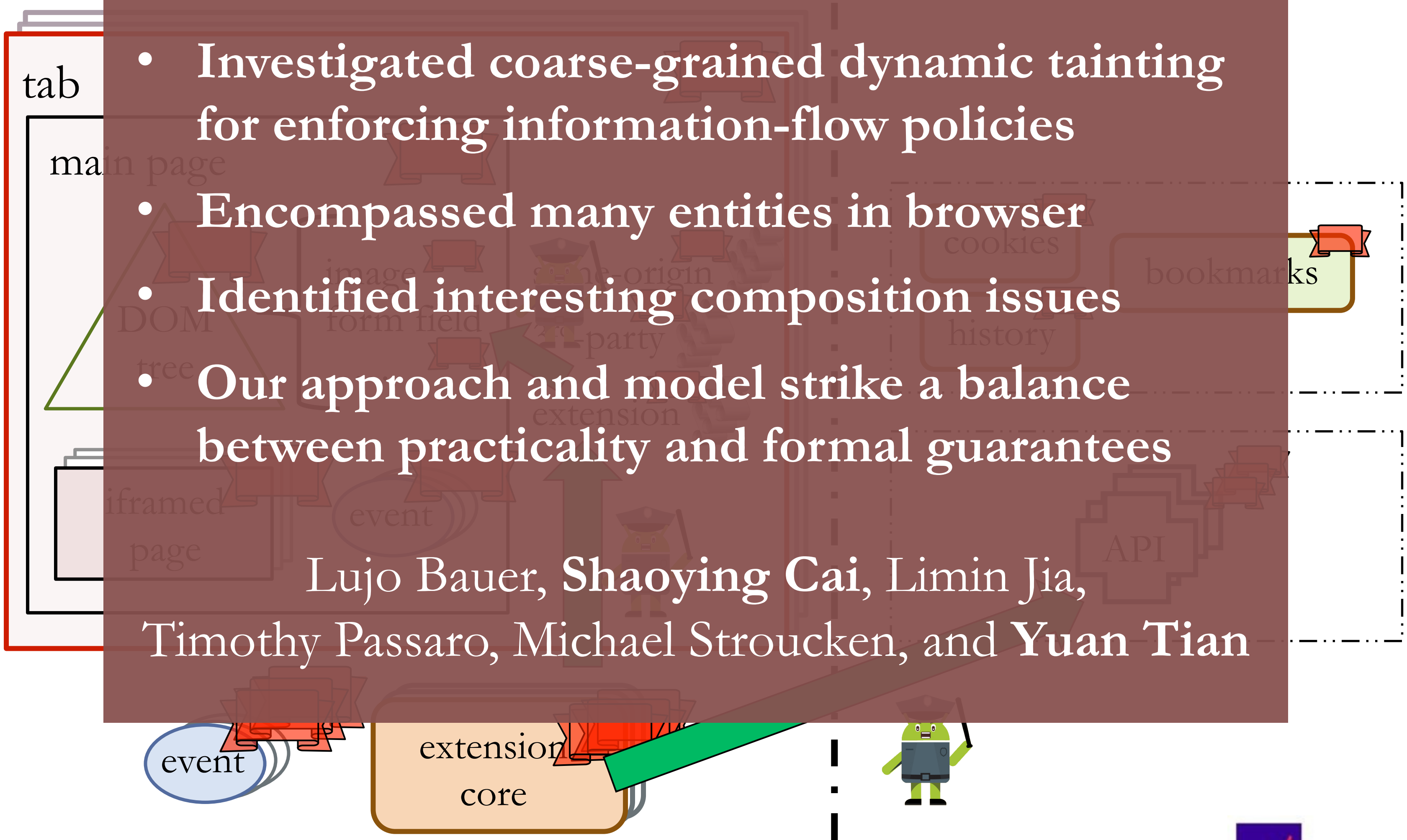
Prototype implementation

- **Built on Chromium version 32.0.1660.0**
- **Front pages of Alexa global top-10 web sites (40 runs each)**
- **29% overhead to page load time added (unoptimized)**
  - E.g., **Google.com**: 6 web requests, 28 label checks, 17% overhead
  - E.g., **Amazon.com:** 212 web requests, 639 label checks, 25% overhead

Summary

Dynamic entities

Static entities



- Investigated coarse-grained dynamic tainting for enforcing information-flow policies
- Encompassed many entities in browser
- Identified interesting composition issues
- Our approach and model strike a balance between practicality and formal guarantees

Lujo Bauer, **Shaoying Cai**, Limin Jia,
Timothy Passaro, Michael Stroucken, and **Yuan Tian**