A Framework for Risk-Aware Role Based Access Control

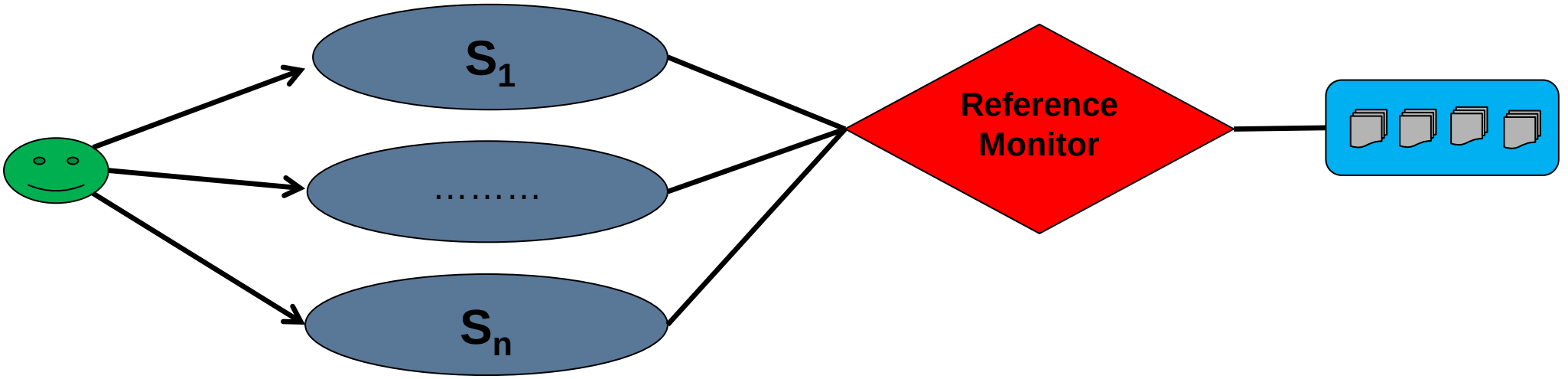
Khalid Zaman Bijon, Ram Krishnan and Ravi Sandhu
Institute for Cyber Security
University of Texas at San Antonio

October 16, 2013
SafeConfig 2013: IEEE 6th Symposium on Security Analytics and Automation

Traditional Organizations

Access Control Mechanism

General Access Control Systems



User	Subject/ Session	Mediates all access requests	Object
Alice, Bob, etc.	Process (e.g., *pid*), session (e.g., s*ip*), etc.	Implemented Access Control models, e.g., RBAC, DAC, MAC.	Resources to protect, e.g., mp3, doc, txt, directory.

Modern Organizations

Intro. & Motivation

Possible Solutions?

- Authenticate and grant same access everywhere
  - Is not sufficient
  - How do we know that the person in the other side is true employee
- Secure every place/situation by antivirus/firewalls
  - Not scalable/feasible
  - Impractical
- More dynamism in access control systems
  - Accept/Deny accesses based on security threats/risks involve in every situations/places instead of always giving same outcome for a user

Intro. & Motivation

Overall Strategy

- Risk-Awareness in Access Control Systems
  - Quantified Approach (Risk is represented as a metric)
  - Calculate risk value, involved in every situation
  - Grant access accordingly based on the estimated risk value

Intro. & Motivation (cont.)

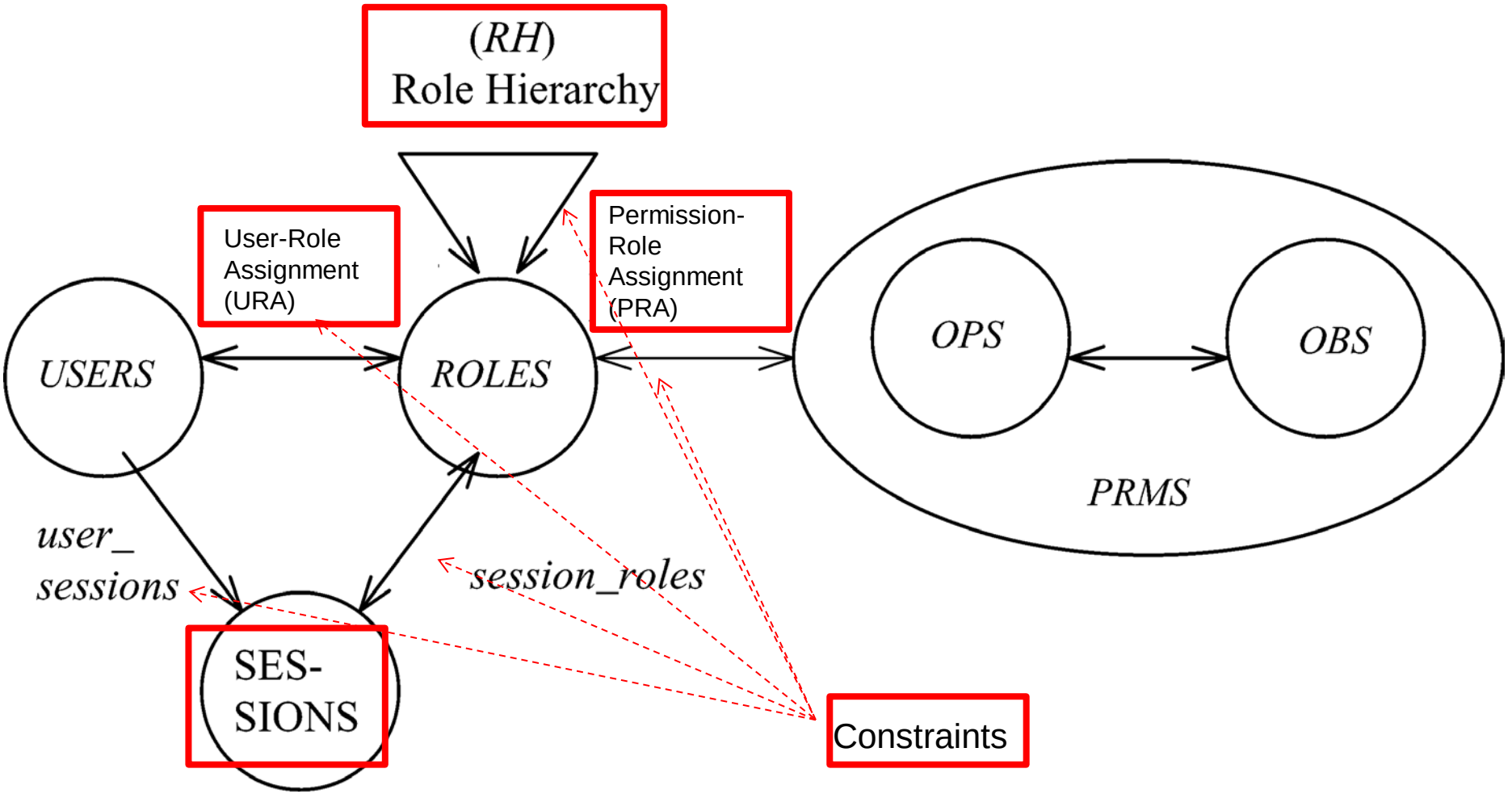
Conducted Research in this Arena

- **MITRE Corporation Jason Program Office.** *Horizontal integration: Broader access models for realizing information dominance (2004)*
  - *Pioneer work in quantified risk-aware access control systems*

- **Risk-awareness in Access Control Systems:**
  - E. Celikel et al (2009), F.Salim et al (2011), L. Chen et al (2011), N. Baracaldo et al (2012), K. Bijon et al (2012), S. Chari et al (2012) and others: **Risk-awareness in Role Based Access Control (RBAC) system (mainly focused on developing technique on risk-estimation and utilization)**
  - P. Cheng (2007), Q Ni (2010): **Risk-awareness in Lattice Based Access Control (LBAC)**
  - R. McGraw (2009), Kandala et al(2011): **Identify risk-factors for a risk-aware access control system**
  - H. Khambhammettu et al (2013): **a framework for various risk-assessment approaches in access control**

- The Framework
  - Identify the Risk-Aware RBAC Components
    - Faces different types of security risk while performing their operations
    - Need to develop additional functionalities to support a risk-awareness
  - Different Types of Risk-Awareness
    - Traditional Approaches
    - Quantified Approaches
      - Non-adaptive approach
      - Adaptive approach

Risk-Aware RBAC Components

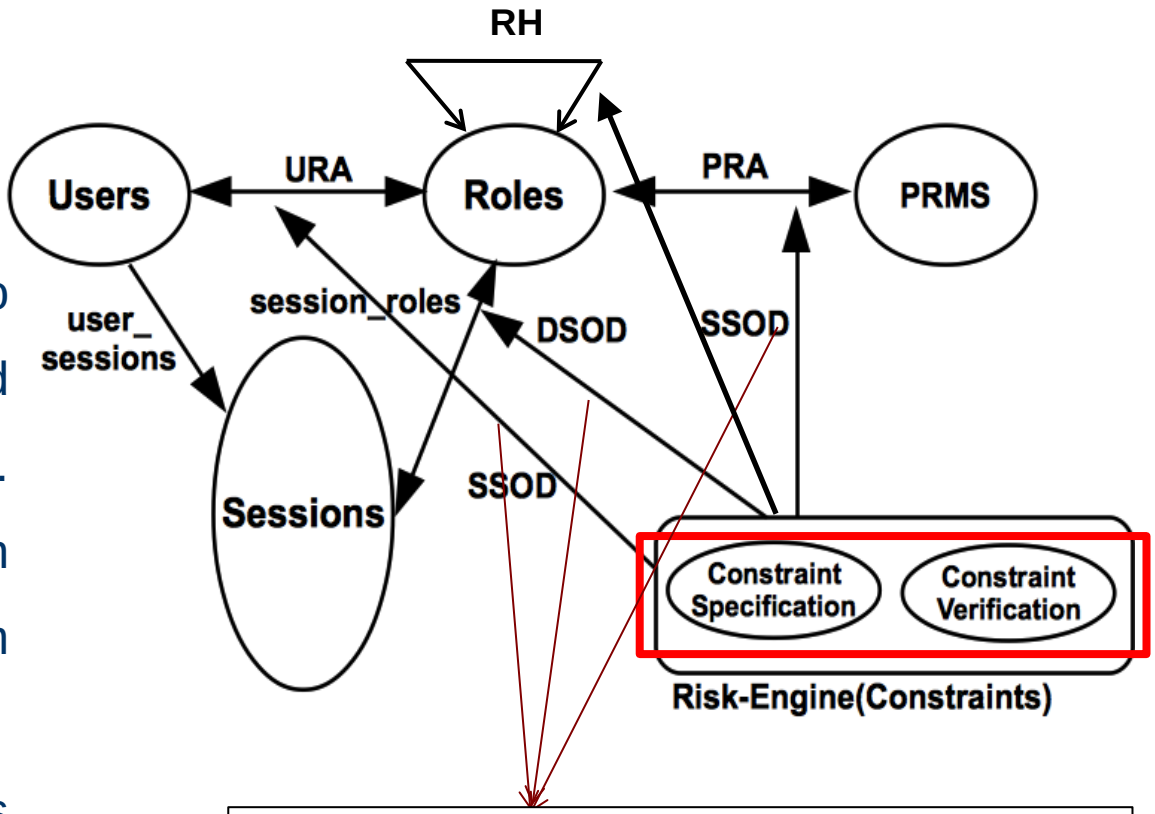


Risk-Awareness Types

- Traditional Approaches
  - Constraints driven risk mitigation
  - No explicit notion of risk value
- Quantified Approaches
  - Risk is explicitly represented as a metric
  - Risk is mitigated based on the estimated value

Traditional Risk-Awareness

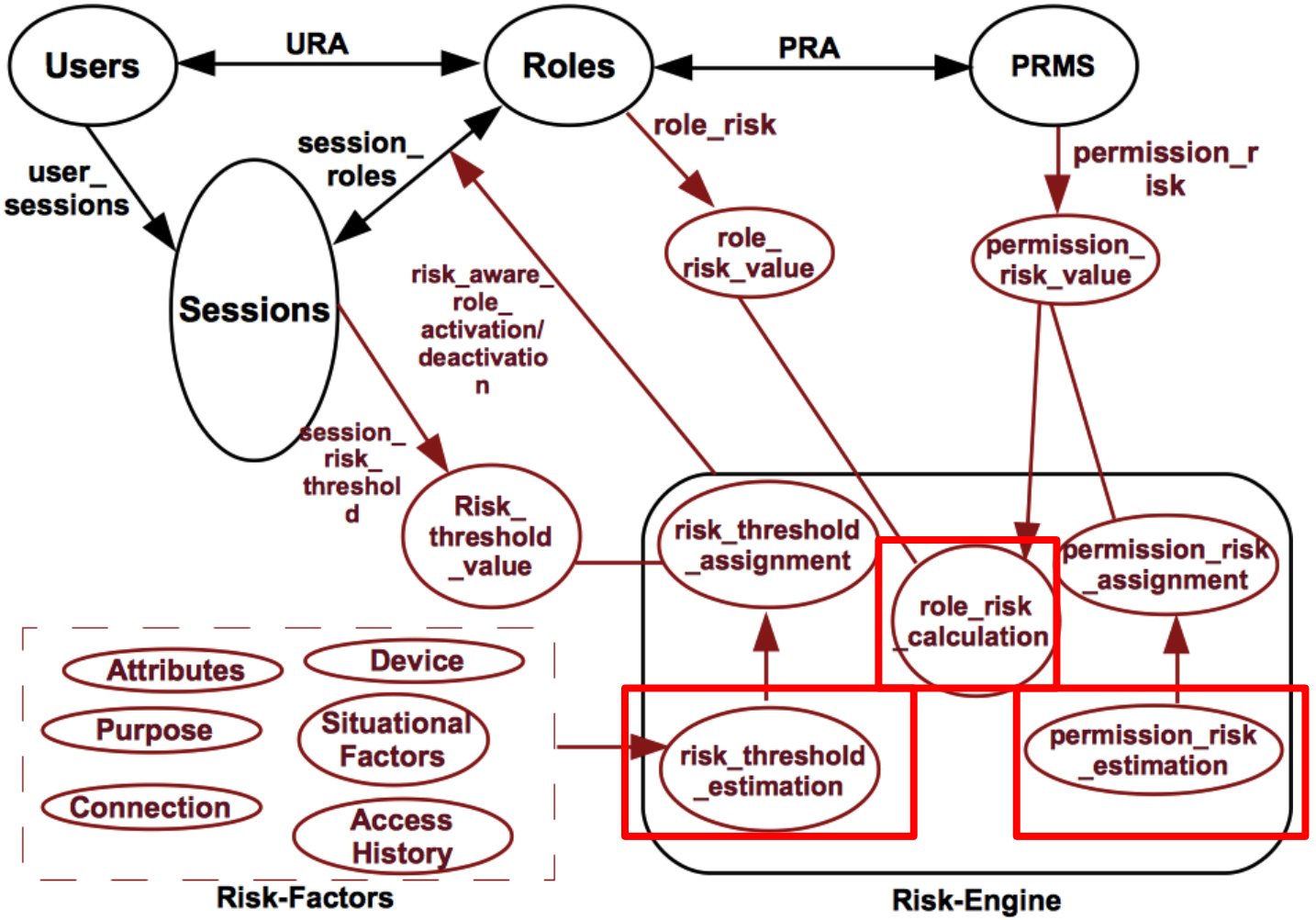
1. Administrative user needs to identify risky operations and generate constraints accordingly. (For example, a constraints can restrict two risky roles from assigning to same user (SSOD).
2. Static in nature (a constraint always gives same outcome, unless modified)



Quantified Risk-Awareness
(Non-Adaptive)

1. Risk-threshold should vary across sessions (e.g. a session from office vs. session from home pc)

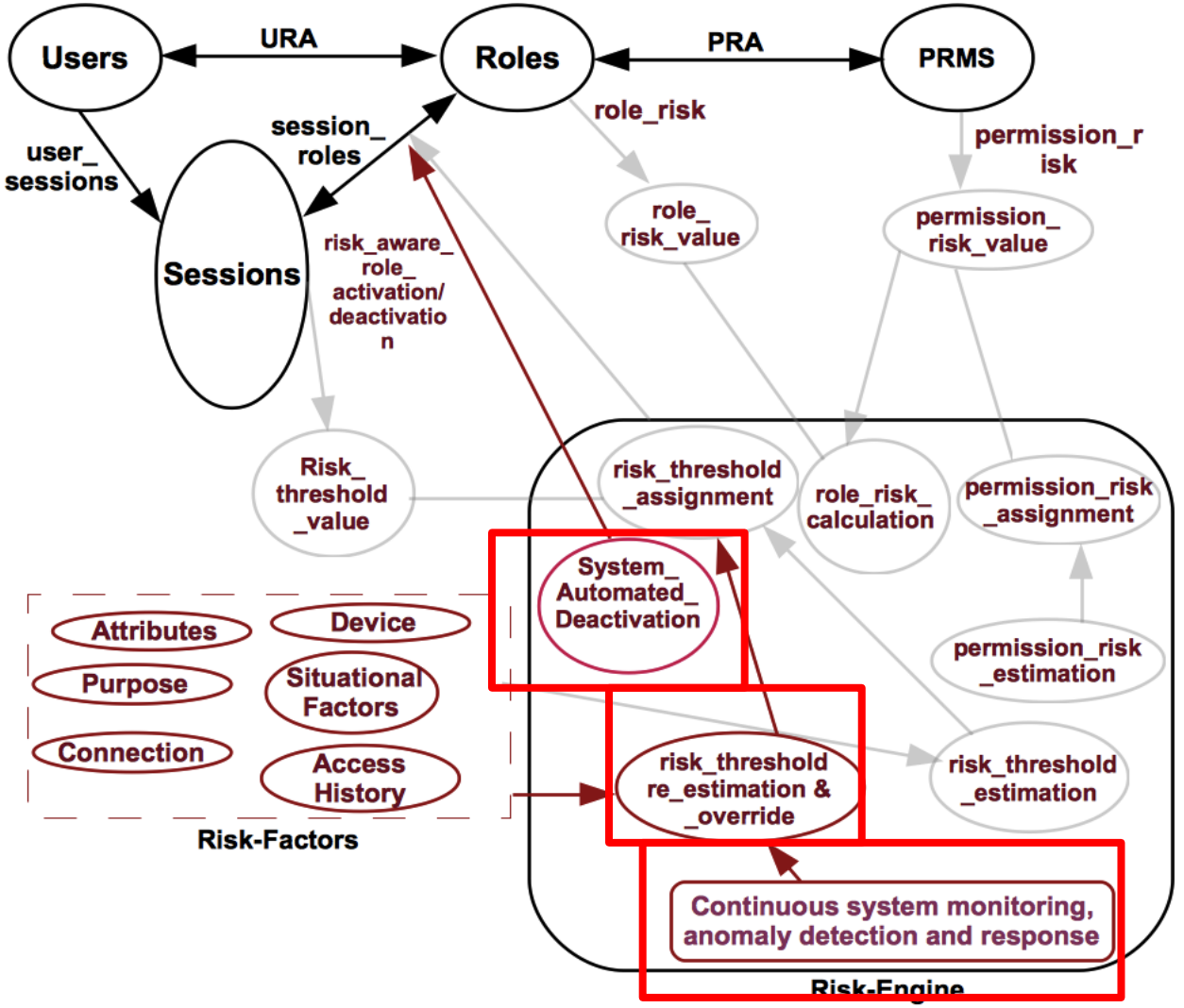
2. Risk-threshold limits user activities by restricting role-activation



Quantified Risk-Awareness
(Adaptive)

1. Continuous user-activities monitoring and anomaly detection

2. Response mechanism by automatic revocation of privileges (e.g. system automated role deactivation)



Formal Specification

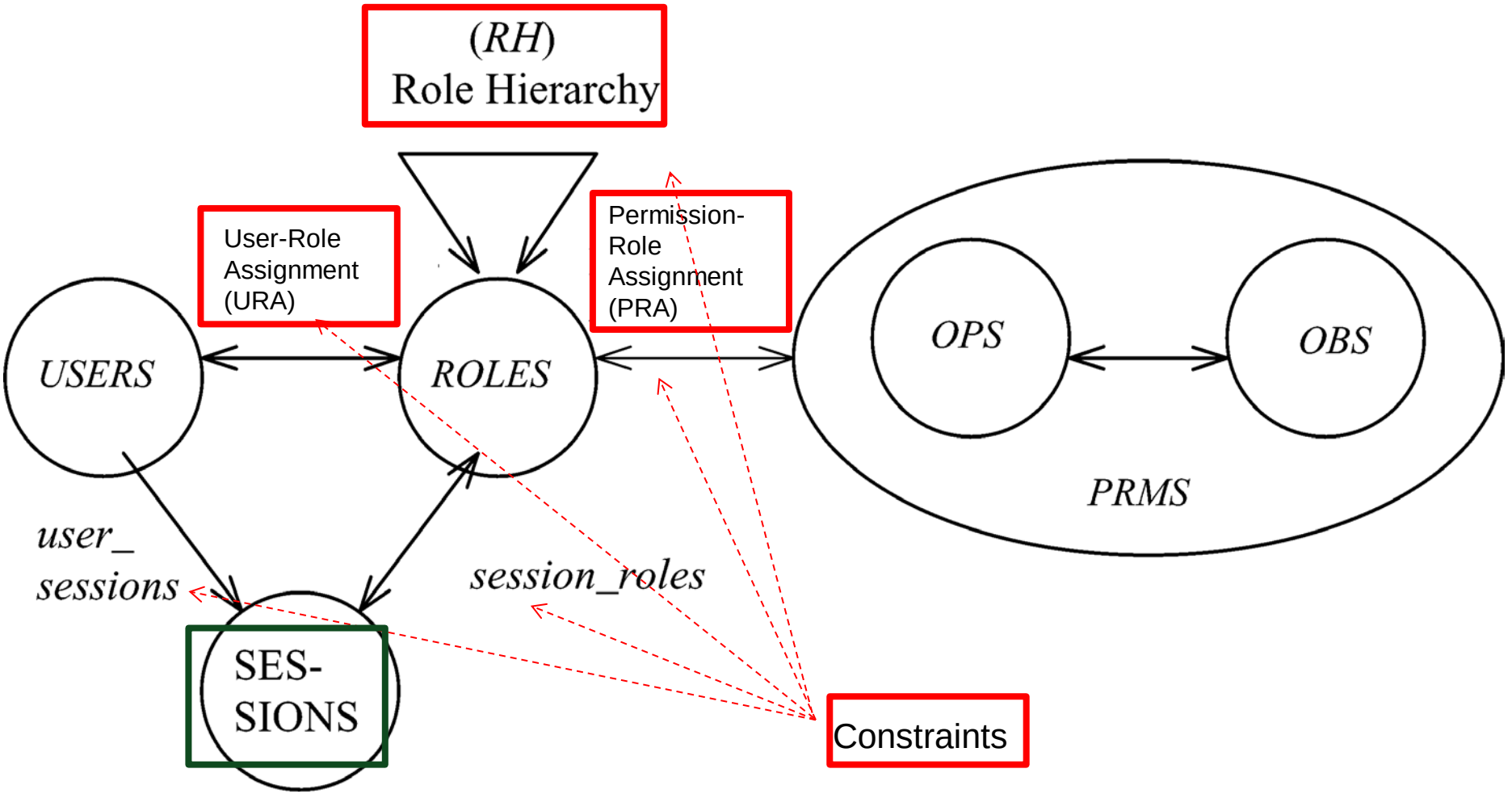
- Formally enhance NIST Core RBAC model
  - To support a session with adaptive risk-threshold

- Functions of the adaptive quantified risk-aware sessions
  - **AssignRisk**: assigns a risk value to a permission
  - **RoleRisk**: returns estimated risk of a role
  - **CreateSession**: user creates a session and system calculate risk-threshold for the session
  - **AddActiveRole**: called by users, tries to activate a particular
  - **Deactivation**: called by AddActiveRole to deactivate some already activated roles in order to activate that role
  - **SActivityMonitor**: This function monitors user sessions, if something is wrong it calls system automated deactivation (SADeactivation) function.
  - **SADeactivation**: This function automatically identifies which roles need to deactivate and asks user to deactivate them.

Conclusion

- To Summarize the framework:
  - The Risk-Aware RBAC Components are identified
    - Sessions, User-Role assignments, Permission-Role assignments, Role Hierarchy, Constraints
    - Each components should have different functionalities (need to be developed to support a Risk-Awareness)
  - Different Types of Risk-Awareness Approaches
    - Traditional Approaches
      - Constraints specific (implicit risk and static in nature)
    - Quantified Approaches
      - Non-adaptive approach (explicit notion of risk that varies across different situations)
      - Adaptive approach (need run-time monitoring capabilities and additional system functions for automatic response)

Future Work


(RH)
Role Hierarchy
User-Role Assignment (URA)
Permission-Role Assignment (PRA)
USERS
ROLES
OPS
OBS
PRMS
user_ sessions
session_roles
SES-SIONS
Constraints

The End

Questions?

Backup

http://www.forbes.com/sites/danschawbel/2013/03/29/david-heinemeier-hansson-every-employee-should-work-from-home/