iLayer: Toward an Application Access Control Framework for Content Management Systems

Gorrell Cheek, Mohamed Shehab, Truong Ung, Ebonie Williams

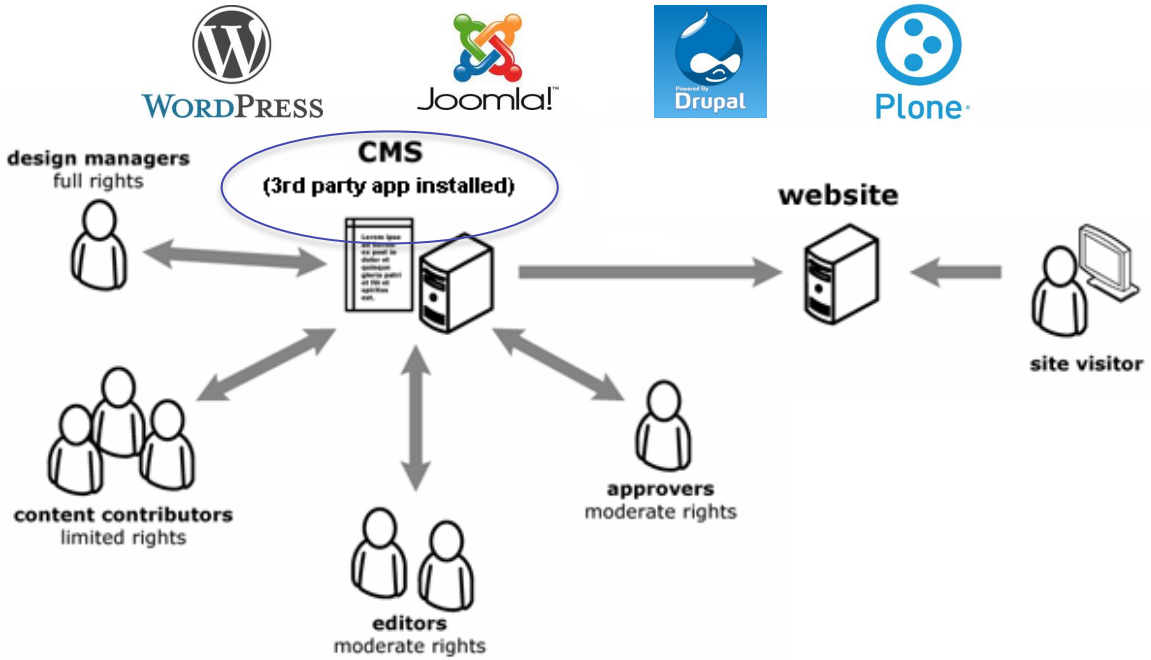
UNC CHARLOTTE
College of Computing and Informatics

Outline

- Preliminaries
- Motivation
- iLayer Framework
- Conclusion & Future Work

Content Management Systems (CMS)

- Online application that provides users the ability to easily create, design, publish and manage the content of a web site
- Multiple users with varying roles

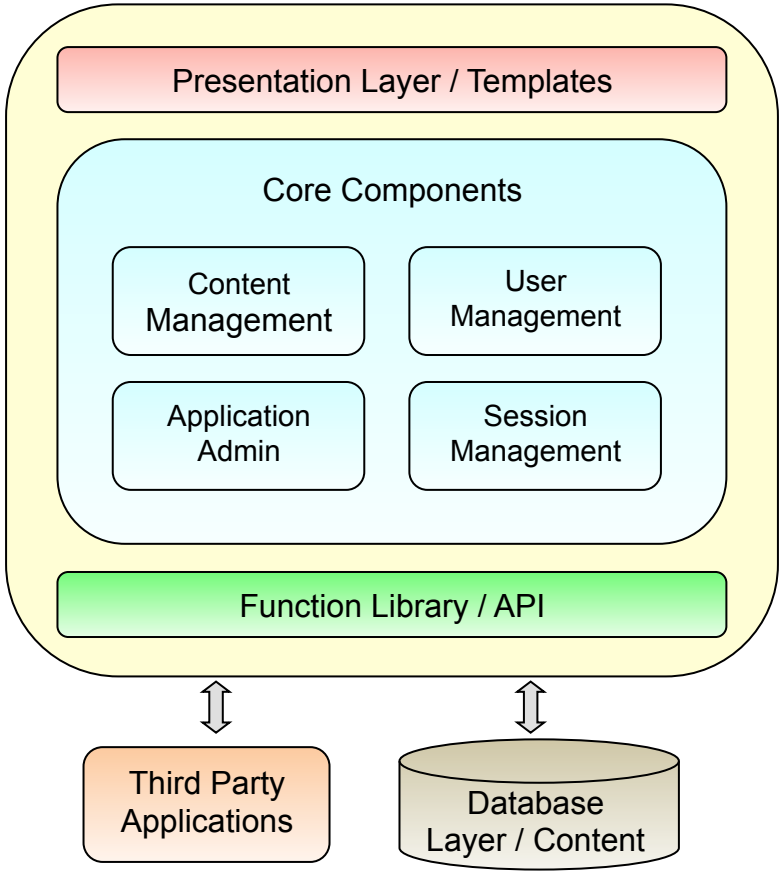


- Third party applications expand the capabilities and functionalities of content management systems
  - For example, a third party developed calendar application can provide schedule management

Image source: www.ubc.ca

Content Management Systems

- Presentation Layer
  - Displays to the visitor of the web site the output (or content) of the CMS
- Core Components
  - Provides foundational CMS functionality
- Database Layer
  - Stores all content
- Function Library / API
  - Provides interface for integrating third party applications



Outline

- Preliminaries
- Motivation
- iLayer Framework
- Conclusion & Future Work

CMS Third Party Application Access Control

- Third party application (TPA) access control functionality is not well developed
- TPA's typically have full administrator level access to the CMS and its content
- File permissions are the primary way to restrict access to TPA's
  - Difficult to implement
    - CMS administrator must be able to translate access requirements of TPA's into file permissions
    - Average CMS administrator may not have sufficient skill or experience
  - Implemented post installation
- More often than not, CMS administrators take minimal or no action to secure the CMS and its content from TPA's

CMS Third Party Application Access Control

- Conducted study of popular CMS – Drupal
- Analyzed 412 third party applications' database calls

Table Name	Table Description	Potential Impact	% of 3rd Party Apps That Require Access
sessions	Contains user session information, e.g., userID, sessionID, user IP address, etc.	Session hijacking	2%
users_roles	Lists the assignments between users and roles	Privilege escalation	5%
node_revisions	Contains edits / revisions of node content	Content compromise	7%
permissions	Lists each user role's permissions	Privilege escalation	7%
users	Contains usernames, passwords, profile information, etc.	Account compromise	23%

We believe that CMS users need additional tools and mechanisms to protect their online information from attacks via third party applications.

Outline

- Preliminaries
- Motivation
- iLayer Framework
- Conclusion & Future Work

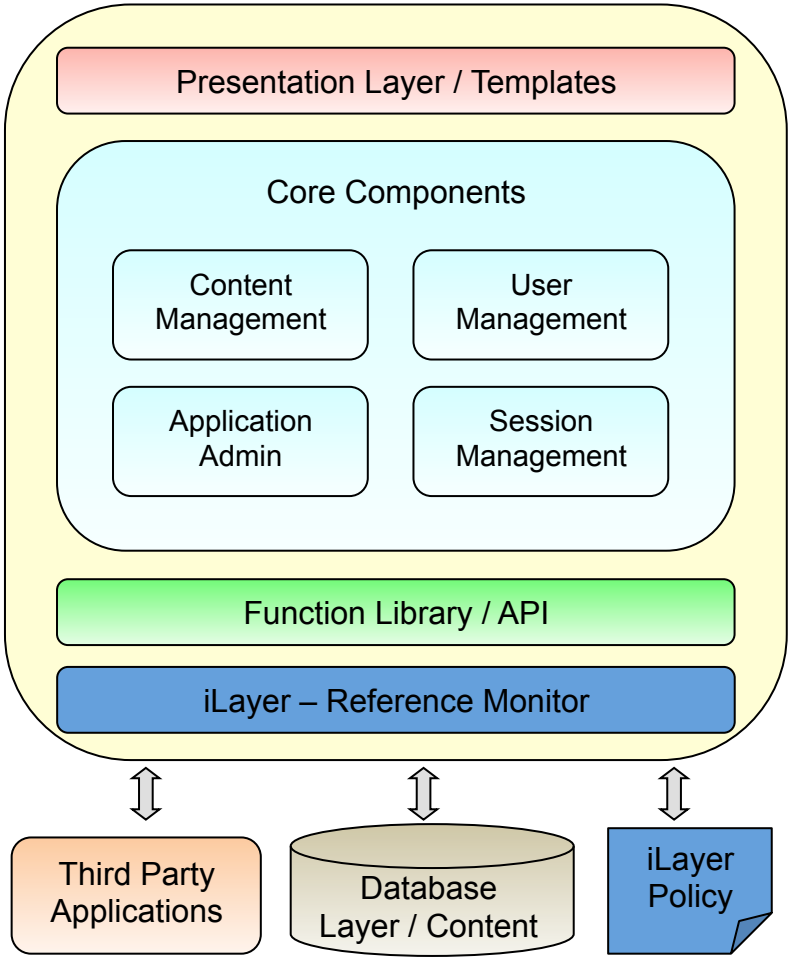
iLayer – TPA Access Control Framework

- Manages user to third party application policies in Content Management Systems

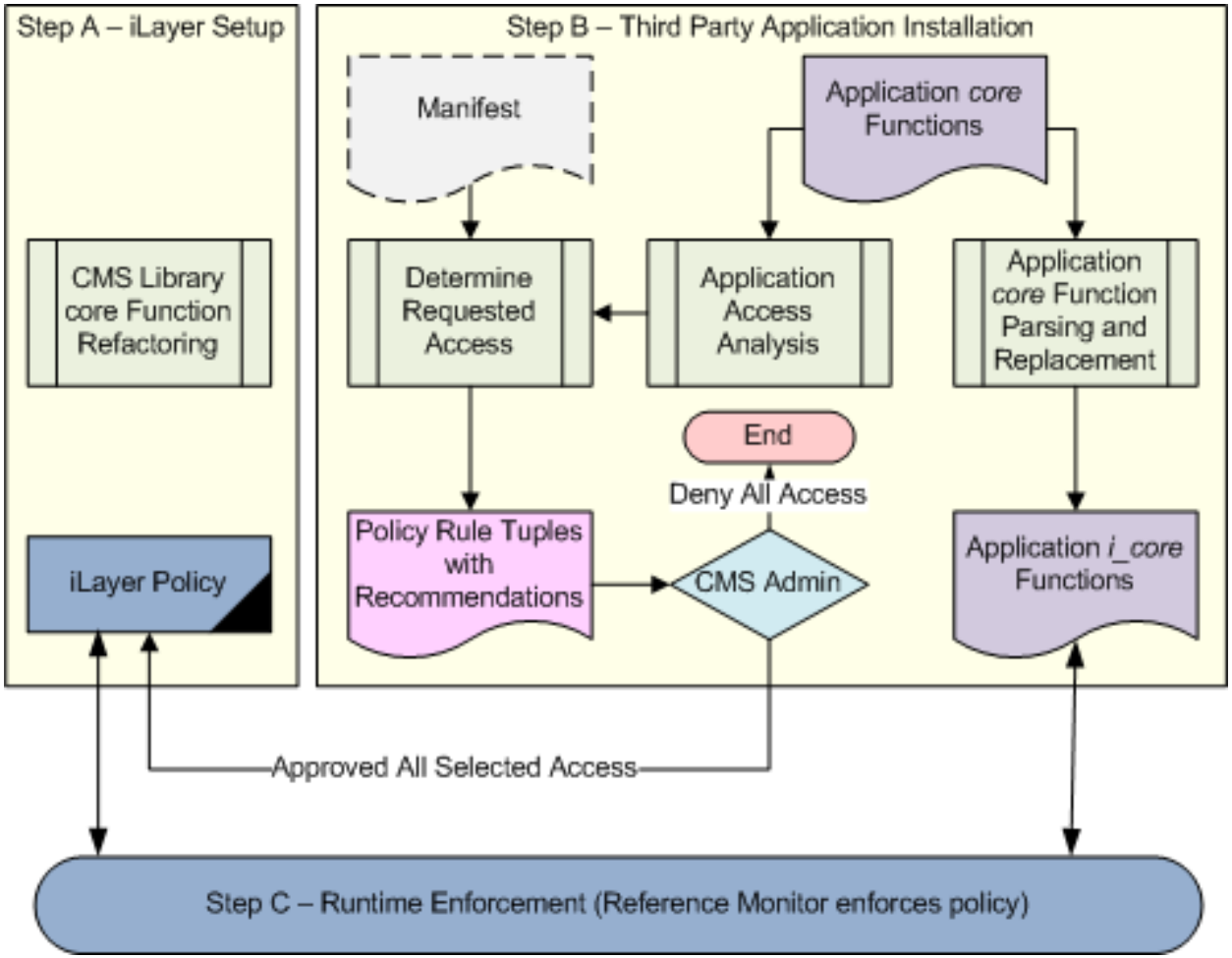
Makes policy recommendations to CMS administrative user for third party applications.

Policies are reviewed and set by the CMS administrative user and enforced by the iLayer Framework.

iLayer Framework Overview

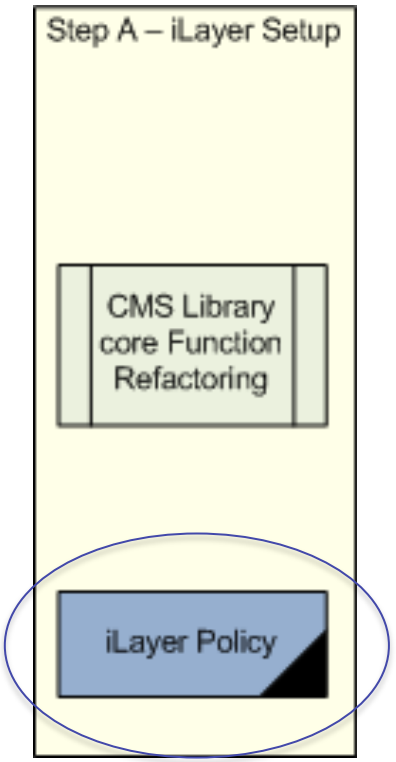


iLayer Framework Overview



A. iLayer Setup – Policy Table Setup

- Create **iLayer Policy** table
- Policy is made up of three components:
  - subject: third party application that will be granted access
  - object: database table being given access to
  - permission: access privilege that is granted which could be either:
    - read (select)
    - write (delete, insert, update)
    - read & write

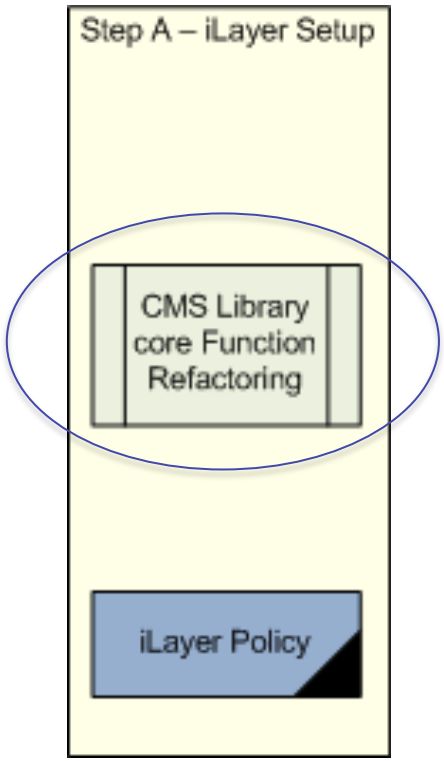


A. iLayer Setup – Refactoring

- **Library core Functions Refactoring**

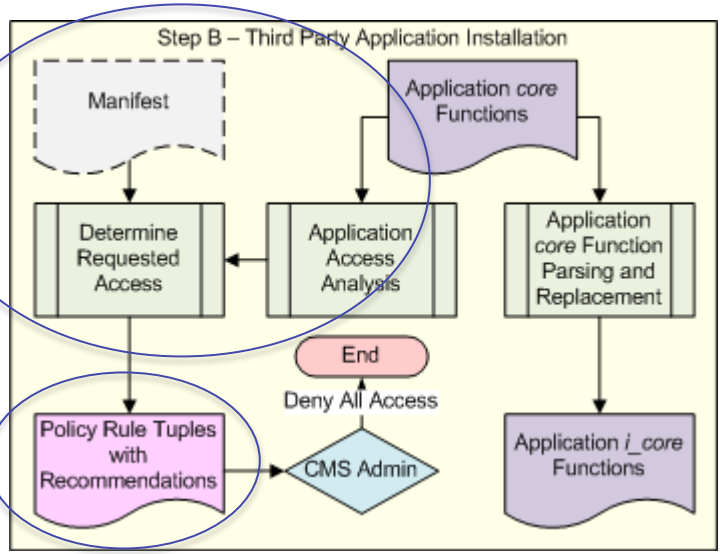
Original core()	**Refactored core() to i_core()**

function core(arg) {
…
//extract table name and
action from the arg
…
}
	function i_core(3PA_Params, arg) {
	…
	//extract table name and
	action from the arg
	//loop for all table names
	if(matchPolicy(3PA_Params, table, action) = null)
	errorHandler();
	else
	core(arg);
	}



B. Third Party Application Installation

- **Determine Requested Access** by the third party application:
  1. **Manifest** provided by the third party developer
  2. **Application Access Analysis**



- **Policy Rule Tuples** presented to administrative user:
  - (subject, object, permissions)
    - (application_name,
    - database_table_name,
    - read)

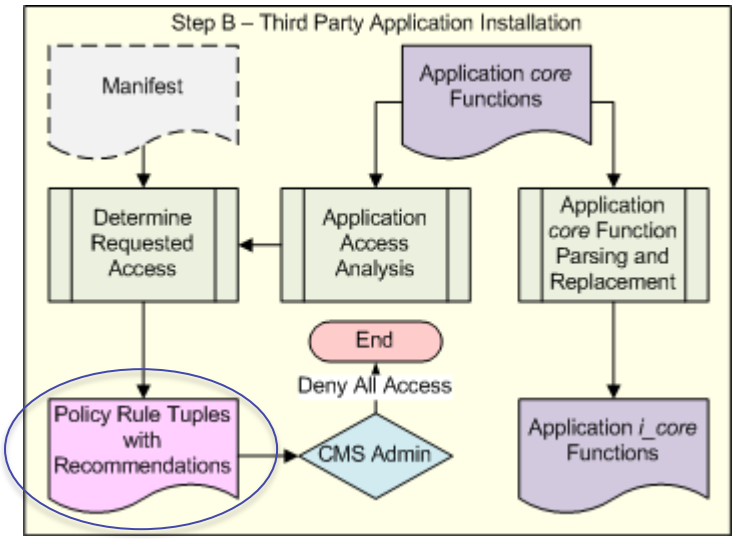
```
<manifest>
    <policy_rule id=``pr1">
        <subject>appName</subject>
        <object>birthday_table</object>
        <permission>select</permission>
        <required_flag>0</required_flag>
        <comments>Access is not required; but...</comments>
    </policy_rule>
    <policy_rule id=``pr2">
    …
    </policy_rule>
</manifest>
```

B. Third Party Application Installation

- Thumbs up or thumbs down **Policy Rule Recommendation** is presented
- Indicator of the community's usage of the policy rule tuple

$$x_i = \begin{cases} 1 & \text{if } a_i \text{ is granted} \\ 0 & \text{if otherwise} \end{cases}$$

Application ID	Granted Accesses / a_i (object - permission)								
	sessions - read	*Sessions - write*	*user_roles - read*	*user_roles - write*	*node_revs - read*	*node_revs - write*	...	*files - read*	*files - write*
001	0	1	0	0	0	1	x_i	0	0
002	0	0	0	0	0	0		1	0
003	1	1	1	0	0	0		0	1
...									
412	0	1	0	0	0	1		0	1
413	0	0	0	0	0	0		?	?

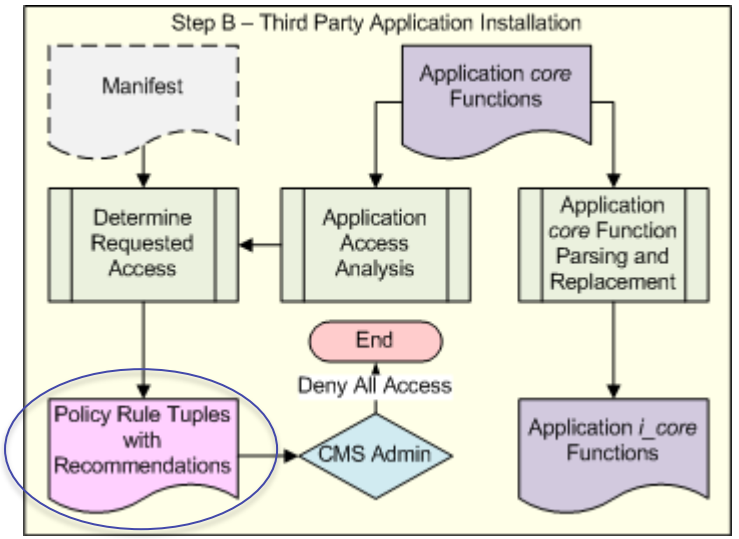


B. Third Party Application Installation

- Thumbs up/down **Policy Rule Recommendation** is based on the maximum likelihood of the set of possible permission combinations for all requested objects based on historically granted accesses

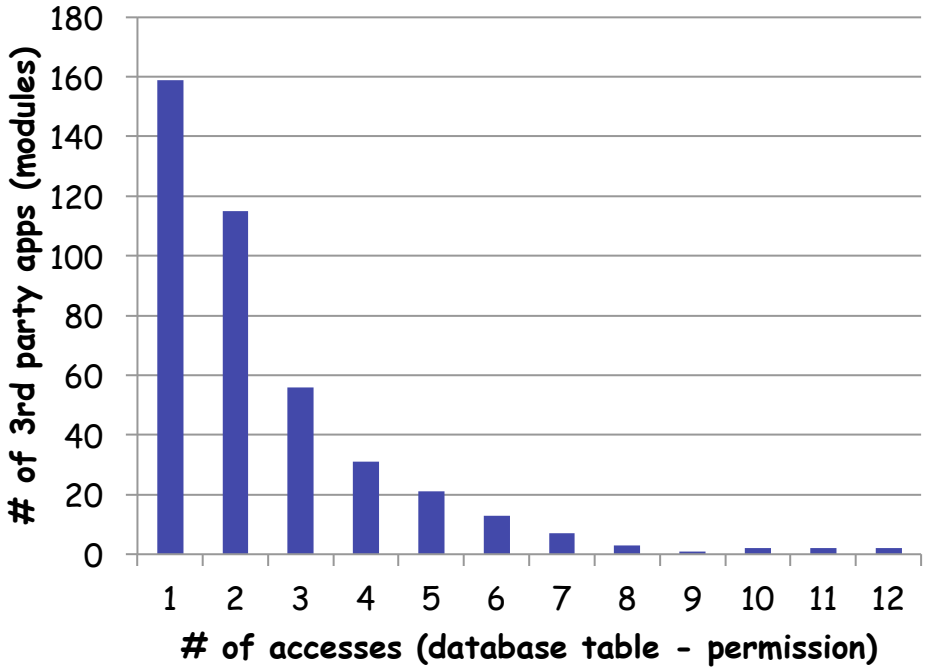
$$X = \arg \max_{x_1,\ldots,x_{r-1}} P(x_r = 0,\ldots,x_n = 0 \mid x_1,\ldots,x_{r-1})$$

files - read	*files - write*	X	$P(\bar{R} \mid X)$	Recommen-dation
deny	deny	$\{x_1 = 0, x_2 = 0\}$	0	
deny	allow	$\{x_1 = 0, x_2 = 1\}$	0	
allow	deny	$\{x_1 = 1, x_2 = 0\}$	.2	
allow	allow	$\{x_1 = 1, x_2 = 1\}$	.5	



B. Third Party Application Installation

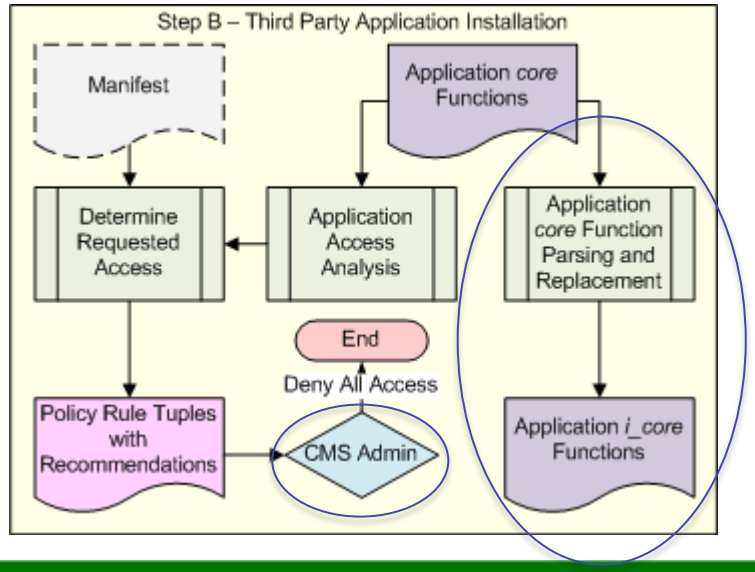
- Number of conditional probability computations equal 2^n , where *n* equal the number of requested accesses



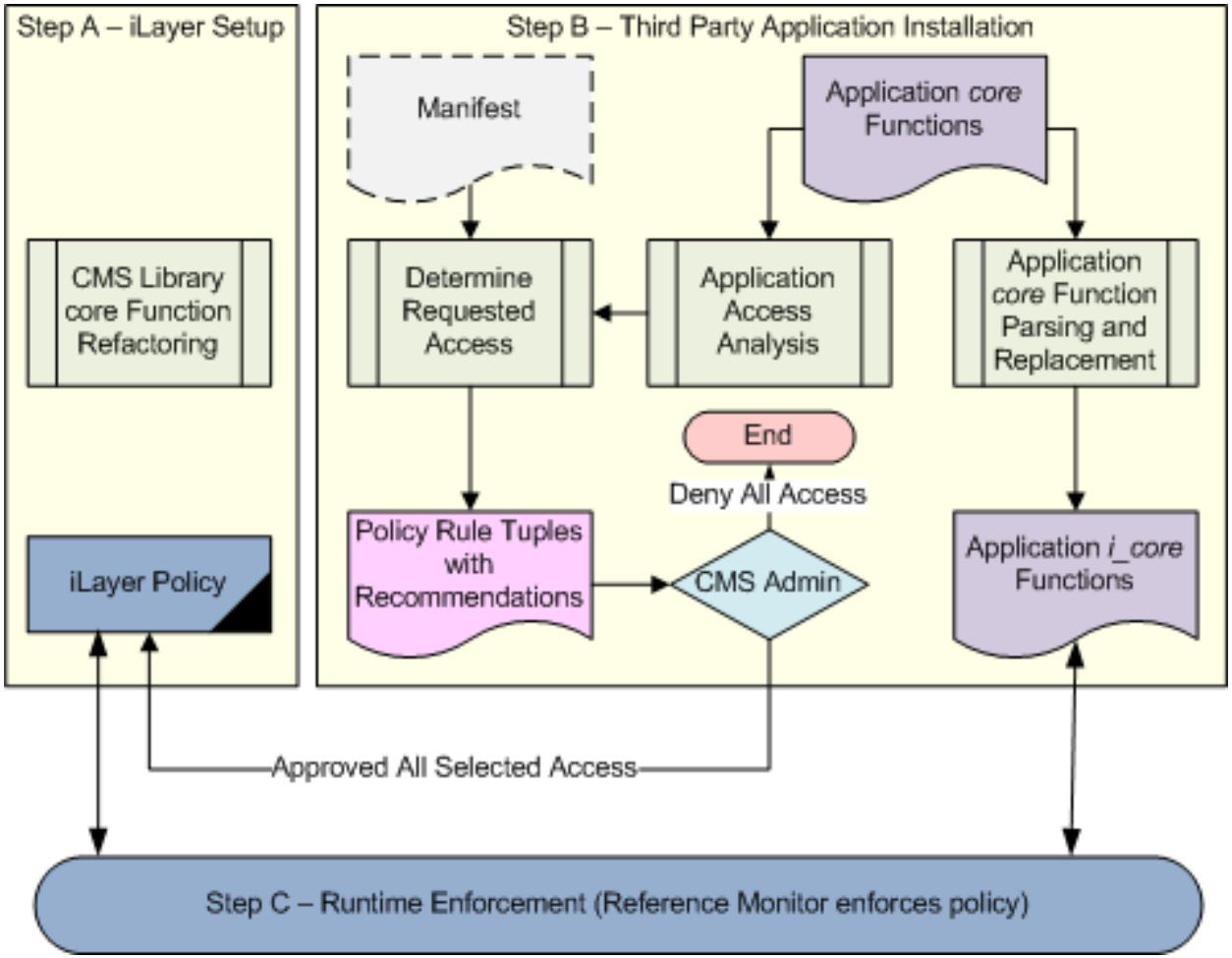
- Distribution of number of accesses for 412 Drupal TPA's (modules) – average 2.45 and median 2.0

B. Third Party Application Installation

- All the policy rule tuples are presented to the CMS administrative user for a **Policy Decision:**
  - Approve all selected access
  - Deny all access
- **Function Parsing and Replacement**
  - Third party application code is parsed and all instances of *core* functions are replaced with their corresponding *i_core* functions

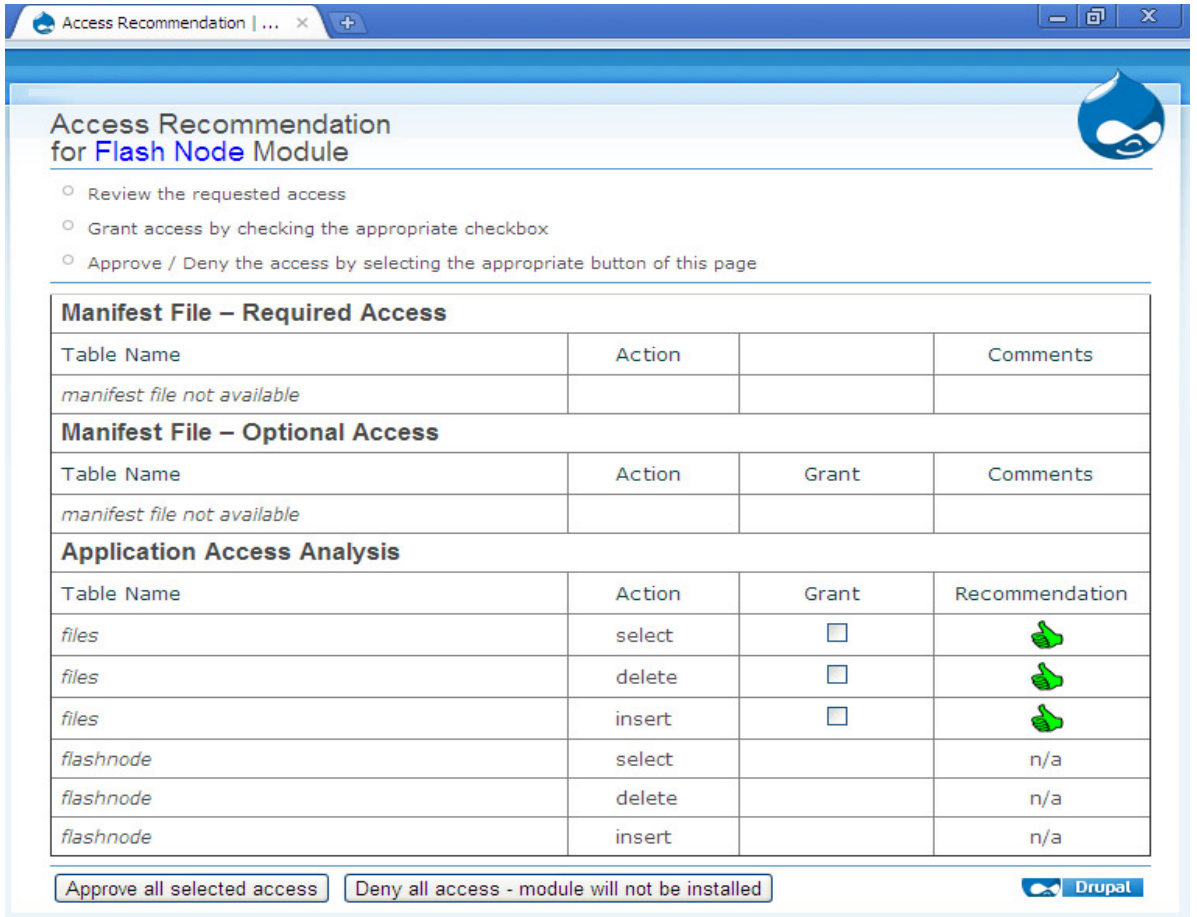


C. Runtime Enforcement



Prototype – Drupal CMS Platform

- Limited proof of concept prototype



Outline

- Preliminaries
- Motivation
- iLayer Framework
- Conclusion & Future Work

Conclusion

- Presented a third party application access control framework for content management systems
- Provides policy setting functionality, including a policy setting recommendation capability
- Policies are reviewed and set by the CMS administrative user and enforced by the iLayer Framework
- Implemented a prototype of the iLayer Framework on the Drupal Content Management System

Future Work

- Expanding the current Drupal prototype from proof of concept to fully functional system
- Providing the ability to review and update third party application policies post installation
- Conducting a benchmark study measuring the performance impact of iLayer on a content management system
- Conducting a detailed survey and analysis of the threat vectors
- Conducting an extensive user study

Questions

Gorrell Cheek
University of North Carolina at Charlotte
gcheek@uncc.edu