Dynamic RDMA Credentials
James Shimek, James Swaro
Cray Inc.
Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.
Agenda

- What need does DRC address?
- Background
- What is the Dynamic RDMA Credentials feature?
- Architecture
- Usage
- Scalability
- Future Work
- Summary
- Q&A
What need does DRC address?

- Existing sharing functionality unavailable on SLURM
- System and User Protection Domains do not provide fine grain access control
- System Services
  - ADIOS
- Applications
  - Live data injection/offload
  - Live Debugging
  - Visualization
Background

- Seastar interconnect allowed communications between any application running on System
- Gemini interconnect added concept of “Protection Tags (ptags)” to secure communication
- Aries ptags are per-node with pkey protecting network resources
- Protection domains (pdomains) allow for shared network access between all applications on the system (system pdomain) or all applications of a specific user (user pdomain)
  - Not available under non-ALPS WLMs (SLURM)
  - No ACL like capabilities
What is the Dynamic RDMA Credentials feature?

- System for providing shared network access between applications belonging to different users, groups, or jobs
- Provides full control of access permissions to applications and administrators
- Applications can request credentials that can be shared between applications at runtime through the *libdrc* API

\[1\] Does not require interaction with work load manager
Benefits of using Dynamic Credentials

- **Managed Credentials**
  - Credentials can be managed inside/outside of WLM context

- **Shared Network Access**
  - Applications can communicate through shared cookies

- **Security**
  - Applications can only access credentials if they are authorized

- **Ease of Use**
  - Existing uGNI applications can use DRC with minor changes
Architecture

Node-local

Remote Server

Application

LibDRC  uGNI

Application Requests

User App Service

(DRCC)

Client Requests

Job Exit Events

Job expiration

Request Cookie

DRCC

DRCS

DRCJEDI

DRC

Work Load Manager

Node-local

Remote Server

Application

LibDRC  uGNI

Application Requests

User App Service

(DRCC)

Client Requests

Job Exit Events

Job expiration

Request Cookie

DRCC

DRCS

DRCJEDI

DRC

Work Load Manager
Architecture

Node-local

Application Requests

LibDRC  uGNI

Application

DRCC

Remote Server

Client Requests

Job Exit Events

Job expiration

Request Cookie

DRC

Work Load Manager

(compute)

(apply)

(analysis)
Architecture

Node-local

Application

LibDRC uGNI

User App Service

Application Requests

DRCC

Client Requests

Job Exit Events

DRCS

Job expiration

Request Cookie

DRCJEDI

Remote Server

DRC Work Load Manager

(ALPS/SLURM)
Architecture

Node-local

Application
LibDRC  uGNI

User App Service

Application Requests

DRCC

Job Exit Events

Remote Server

Client Requests

Job expiration

DRCS

Request Cookie

DRCJEDI

DRC Work Load Manager

(ALPS/SLURM)
Architecture

Node-local

Application
LibDRC | uGNI

User App Service

DRCC

Application Requests

Client Requests

Job Exit Events

Job expiration

Request Cookie

Remote Server

DRC
Work Load Manager

(DLPS/SLURM)

Computes

User App

Service
Architecture

Node-local

Application

LibDRC  uGNI

DRCC

Application Requests

User App Service

Remote Server

Client Requests

Job Exit Events

Job expiration

Request Cookie

DRC

Work Load Manager

(DLPS/SLURM)

Client

Requests

Job

expiration

Request

Cookie

DRCS

DRCJEDI

(AlPS/SLURM)
What is a Credential?

- Credential internally contains everything that application needs to configure HSN
- Cookie is used to configure uGNI
Features

- Job Expiration Detection
- Administrative Limits on Credentials
Features

- Persistent Credentials
- Node-local Credential Caching
Features

- Credential Tokenization
- Node-Insecure Mode
How do I use Dynamic Credentials

● Applications can directly interface with the DRC system via libdrc
  ● Full set of API calls necessary for basic interaction with the system
  ● New API features can be implemented without affecting existing applications (eg no need to recompile)

● Administrators and authorized users may use drccli to interface with the system
  ● Expanded set of functionality to provided security, administrative, and extended functionality for long-running system services
DRC Application Programming Interface

- **Main Library Functions**
  - `drc_acquire`
  - `drc_access`
  - `drc_access_with_token`
  - `drc_grant`
  - `drc_revoke`
  - `drc_release`

- **Helper Library Functions**
  - `drc_get_wlm_id`
  - `drc_get_first_cookie`
  - `drc_get_second_cookie`
  - `drc_get_credential_token`
drccli

- **Capabilities**
  - List
    - Retrieves credentials currently under the control of DRC
  - Limits
    - Show, add, update, remove limits on drcs controlled credentials
  - Acquire
    - Acquires a credential for an existing, running application
  - Release
    - Releases an existing credential
  - Grant
    - Revokes access to an existing credential based on UID, GID, WLM_ID
  - Revoke
    - Revokes access to an existing credential based on UID, GID, WLM_ID
DRCCLI Example Usage

- Acquiring Credential through CLI
  >: # drccli acquire
  
- Releasing Credential through CLI
  >: # drccli release 1

- Granting Access through CLI
  >: # drccli grant –w <some_wlm_id> <credential_id>
Scalability

- **Access has to be called from every node in the job**
  - Tokenization optimization reduces calls to drcs for access request to one per job

- **Acquire, grant, and release, revoke only need to be called from one node in the job**
Future Work

● Scalability
  ● Multi-process DRCS
  ● Multi-threaded DRCC

● Additional drccli enhancements
  ● Non-administrative use of drccli
  ● User requested features
Summary

- DRC provides capabilities to provision inter-application HSN communication.
- Provides secure and dynamic access control capabilities
- Built on top of existing capabilities – Nothing is going away.
- Available with CLE 6.0UP01
Q&A

James Shimek
jshimek@cray.com