Perun

Perun Description

Michal Procházka, Michal Šťava, Slávek Licehammer
History

- Perun v1
  - Supercomputing center in Brno
- Perun v2
  - Local Grid computing site
- Perun v3
  - National Grid Infrastructure
  - Identity Federations
  - Global AAI
  - Joint development: CERIT-SC and CESNET
Motivation

- National Grid infrastructure
  - users from different institutions
  - different resource providers
- Difficult to manage distributed entities
- User registration is needed
  - users already have some digital identity
- Delegation of the rights to manage entities
- Configuration of the access rights
- Fill the gap between users and end services
Perun Manages

- Virtual organizations
- Users
- Groups
- Resources
- Services
- Application forms
- Attributes, ...
What is it? (Shortly)

● IAM - Identity and Access Management

● Grab user identity -> categorize -> assign resources -> let them use the resources
Perun user interface

![Perun user interface screenshot](attachment:Perun_user_interface.png)
Perun

VO Administrator

Michal Procházka, Michal Šťava, Slávek Licehammer
Entities
User

• Represents physical person

• Ideally every person has only one user representation in Perun

• User can be identified using various digital identities
  ○ social/federated identity, digital certificate, ...
Virtual Organization (VO)

- Basic entity for users categorization
- Special type of a group
- Defined membership rules
- Defined purpose
- At least one VO administrator
- Entity which can have an agreement with service providers
Member

- Representation of user in VO
- Must obey VO membership rules
- Usually has limited lifetime
- One user can be member in several VOs
Group

- Categorization entity inside the VO
- Provides delegation support
- Basic entity used for access control
User lifecycle

1. Registration/import
2. Membership in VO
3. Membership in Groups
4. Access to the services
5. Membership renewal
6. Suspension/membership expiration
How to become a user

● Possess existing external identity supported by Perun
  ○ federated identity, social identity, digital certificate, ...
  ○ user’s enrollment

● Import from existing identity management system
  ○ direct connection to the external system
Enrollment management

- Every VO can define its own application form
  - request various information from the users

- Initial vs. extension application form
- Automatic vs. manual approval

- Text and notification customization

- Multilingual support
Example of registration form

Registration

Name * Pavel Procházka

E-mail * prochazka.pavel@gmail.com

Nickname * Nickname will be used by some of the ELIXIR services, e.g. wikis.

Submit

Fields with * are mandatory.
Import

- Users import from existing identity management system (external source)
- Periodic vs. one time
- Mapping rules between Perun and external source
- Various protocols supported
  - LDAP, SQL, XML, CSV, AD, ...
Account linking

- User can possess more identities
- Perun is able to link/unlink those identities
  - Heuristic search
- User can access Perun and its components with any of linked identity
- Identities can be transferred to end services
Account linking example

Your current identity is

RNDr. Michal Procházka at CESNET, z.s.p.o.

Join with

- Institutional identity
- Personal certificate
- Kerberos login
- Social identity

Powered by Perun -- © CESNET, CERIT

Support at perun@cesnet.cz
User’s roles

- Perun admin
  - God

- VO admin
  - manages whole VO including Group and all associated entities

- Group admin
  - manages group membership

- User
  - self-management
Live Demo

- Create a VO
- Invite member by an e-mail
- Add member from external source
- Create a group
- Add member to the group
Outline

● Attributes
● Facility
● Resource
● Group
● Relationship between F/R/G
● Owners, Perun services and Destinations
● gen/send/slave vs LDAP
● Examples
Attributes

- Piece of information attached to the entities and relations among entities
- Various formats (String, Integer, List, Map)
- Attribute modules
  - Syntax checks
  - Auto-fill
  - Dependency checking
- All data in Perun can be obtained via Attributes
Attributes

User

Member

Group

VO

Attributes

Attributes

Attributes
Attributes

- Preferred email
- User's email in VO
- Group email
- User's email in group
- VO email
Facility

- Real world entity managed by Perun
  - physical or virtual (cluster vs mailing list)
  - homogenous configuration
- Has a set of specific attributes
- Can provide Resources to VOs
- Managed by Facility Manager
Resource

- Access to specified part of Facility for VO
- Resources are VO specific
- May restrict usage conditions (e.g.: disk quotas)
- Groups are assigned to Resources
Facility and Resource

Facility → Resource → VO Group

Storage

Facility Manager: 5TB for your VO

VO Manager: I authorize group to use it
Relationship
Responsibilities

- **Facility Manager:**
  - creates and manages Facilities
  - creates Resources
  - assigns them to VOs
  - prepare services and other settings

- **VO Manager:**
  - chooses and assigns people on provided Resources
  - manages selected Resources Attributes
Owners

- Owner is an administrative contact for Facility
- Contact to:
  - Person (not need to exists in Perun)
  - Organization
  - other
- Description or name with contact information (email, telephone number etc.)
Perun services

- Representation of a service on the Facility
- Script (or program) obtaining data about Perun entities assigned to the Resource
  - gen, send, slave scripts (explain later)
  - data for authorization decision support
  - saved in Attributes (user, facility etc.)
- Data for authorization are propagated to end services
  - e.g.: set unix account for all authorized users
Destinations

- Describe way to transfer configuration from Perun to real world entities
- Target of authorization data propagation
- Assigned to the Facility
- It is pair:
  - Perun Service
  - definition of a target propagation
- Managed by Facility Manager
Perun to real world mapping

- **Facility**
  - Storage
- **Resource**
  - Facility Manager
- **VO Group**
  - VO Manager

- 5TB for your VO
- I authorize group to use it
Perun to real world mapping

![Diagram showing the mapping between Facility, Resource, VO Group, Storage, Facility Manager, and VO Manager.]

- **Facility**: Represents the physical or virtual resource.
- **Resource**: Indicates the allocation or access to the Facility.
- **VO Group**: Signifies the authorization group or VO having access.
- **Storage**: Represents the data storage component.
- **Facility Manager**: Manages the Facility.
- **VO Manager**: Manages the VO Group and its access permissions.

- **5TB for your VO**: Indicates the storage allocation for a specific VO.
- **I authorize group to use it**: Represents the authorization granted to a group for using the allocated storage.

This diagram illustrates the mapping process from Perun to the real-world scenario, highlighting the roles and permissions involved in managing access to storage resources.
Perun to real world mapping

Facility

Destination

Service
Account on tape storage

Service
Account on disk storage

Tape storage

Disk storage

Storage
• Most frequent propagation process
• There are 3 types of script:
  ○ **GENERATE**: generates authorization data (about Users, Groups etc.)
  ○ **SEND**: send already generated data to destinations
  ○ **SLAVE**: sits on destination, receives data from send script and process them (update web ACL, restart service etc.)
• It takes some time
gen/send/slave
Perun LDAP

- Another interface to Perun data
  - changes are proceeded in real time
- Consumers get rights to read specific information from the Perun LDAP
- Real time access to data in LDAP
Example 1 - Cluster management

- Facility = whole cluster (homogenous)
- Resources: (2 per VO)
  - 1 for users (unix account)
  - 1 for admins (k5login_root)
- Destinations = all nodes of cluster
- Default options and limitations defined by Facility Manager
- Preferred options defined by Users itself
  - Using gen/send/slave mechanism
Example 1 - Cluster management

- **Facility**
- **Resources**
- **Destinations**

- users
- admins
Example 2 - RT management

- RT - Request Tracker (issue tracking system)
- Facility = RT
- Resources = Queues
- Facility Manager provides queues to chosen VO
- VO Manager can assign Groups only to provided queues
- Members from Perun are strictly synchronized with RT system
  - Using Perun LDAP mechanism
Example 2 - RT management
Example 3 - Mailing lists

- Facility = Mailman or Sympa
- Resources = mailing lists
- Destinations = host with Mailman or Sympa
- Attributes = email of mailing list manager
  - Using gen/send/slave mechanism
Example 3 - Mailing lists
Perun

Additional Features

Michal Šťava, Michal Procházka, Slávek Licehammer
Outline

- Auditer
- Notifications
- API/Mini applications
- Service Users
- VO Observer
- Security Teams
- Facility contact groups
- Already managed by Perun
Auditer

- Every successful operation is audited
- Auditer produces audit logs: textual representation of operation and entities involved
- Audit log can be read by consumers
- Auditer tracks consumed messages by every consumer
Auditer

Real world

Perun

Auditer

Add member to our VO.

VO Manager

VO

Member

User

User[] created.

Member[] created in VO[].
Notifications

- Reads audit log
- Custom messages based on events occurred in audit log
- Multilingual support
- Notifications based on templates uses
  - Data from audit logs
  - Data from Perun
- Example:
  - Notification about membership expiration
Notifications

Perun generates messages

Auditer reads messages

Unimportant message.
User[] created.
Member[] created in VO[].
Unimportant message.
Unimportant message.

Templates
T1: ^User.*created
T2: ^Member.*created

Predefined mails

controls against templates
uses and fills message

send email

VO Manager

User
API/Mini applications

- Perun provides REST-like interface over HTTPs
- CLI
- Perl and PHP binding
- JavaScript library
  - Mini applications - dedicated web based applications
  - Example: user-profile
Service Users

- Special variant of normal user
- Usually used for automatic systems
  - backuping, nagios etc.
- Don’t want to lose this settings with a person is leaving (e.g.: nagios administrator)
- Normal Users are assigned to this Service User
  - they have rights to work with it, use it
  - have responsibility for this service User
Service Users

Perun admin

VO manager

Select VO

CESNET

Members

Groups

Resources

Applications

Application form

Resource tags

Settings

Managers

External sources

Group manager

Facility manager

User

Create service member

1. Create service identity

Member's name: Name

Member's email: email@cesnet.cz

Namespace: EINFRA

Login: service-user-for-test

User cert DN: /DN/OF/CERT

CA cert DN: /CA/DN/CERT

Optional

Continue
VO Observer

- Role in the Perun system
- Similar to VO Manager
- Can read the same data
- Can’t modify anything
- For the User support purpose
  - e.g.: bad settings of User’s attributes
- For the supervisors
  - statistics, overview etc.
Security Teams

- Entity in Perun
  - has managers
  - publish blacklist of users
- Every Facility can assign one or more Sec. Teams
  - has to trust in the Team
- Blacklisted users are:
  - not propagated by Services to Destinations
  - or marked there
Security Teams (2)
Facility contact groups

- For evidence and information purpose
- ‘group’ of contacts with description assigned to the Facility
- Contacts about:
  - Users
  - Groups
  - Owners
- Will enhance Owners (better linking with Perun Users)
Managed by CESNET’s Perun

- Attribute Authority
- Mailing lists
- MetaCentrum (Czech NGI)
- DÚ
- VŠB VMware
- Alternative passwords
- Meetings
- EGI fedCloud
- RT
Global Schema

Apache (authentication) -> Tomcat -> Perun (authorization) -> Database
Internal Schema

Perun WebApp

GUI

RPC

REGISTRAR

CORE

DISPATCHER

CLI

ENGINE
Perun WebApp contains

- Base (object definitions, utils)
- Core (users, groups, resources, services logic)
- Cabinet (publications management)
- Registrar (user enrollment management)
- Dispatcher (ACL provisioning planning)
- RPC (REST-like interface to Perun)
GUI component

- JavaScript based web application
- GUI contains
  - administration GUI
  - Registrar GUI
  - password reset GUI
- Mini-applications
Configuration

- Defined on build (/etc/perun/), can be overridden on runtime
- Each module can have own config (/etc/perun/module-name.properties)
Logging

- Logging defined in `/etc/perun/log4j.xml`
- Default log files are in `/var/log/perun/module-name.log`
Development and sustainability

Michal Šťava, Michal Procházka, Slávek Licehammer
Outline

● Team development
● Methodology
● Development
● Deployment
● Documentation
● Bug reports and feature requests
Team development

● CESNET and Masaryk University cooperation
● 6 core team developers and some MU students
● Sharing responsibilities
● Service development with other people
  ○ Zdeněk Šustr, František Dvořák, Jiří Ráž, Michal Strnad, Jan Horníček etc.
Methodology

● Agile development
  ○ Iterative development
  ○ Extreme programming
  ○ Task/Feature Driven Development
● Rolling updates
● Weekly meetings (Jiří Bořík, Michal Voců)
Development

- Driven by GIT
- Open GitHub repository
- Everyone can send pull request
  - validated by Perun core team
- Testing every pull request
  - connection between GIT and Jenkins
- Usage of advanced GIT features
  - branches, cherry picking, pull requests etc.
Deployment

- Driven by Jenkins
- Code is tested automatically
- Easy testing every build against:
  - Oracle DB
  - PostgreSQL
  - HSQLDB
- Automatic deployment
- Notify about failed builds
- Build history
Development diagram

Core develops manages

Developers works with

GitHub public

Jenkins communicates

builds and tests deployes

GitHub private

CESNET Perun

MU Perun

PT Perun
Documentation

- For Users (perun.cesnet.cz)
  - basic use cases
  - advanced use cases
- For Technical purpose (perun.cesnet.cz)
  - API (CLI, GUI etc.)
- Internal (redmine)
Bug reports and feature requests

- Request Tracker (rt.cesnet.cz)
  - for users requests and issues
  - need of quick reaction
  - automatic reports of errors from GUI
- Redmine
  - internal tasks
  - development plan
Perun in the World

Slávek Licehammer
Michal Procházka, Michal Šťava
Production deployments

- CESNET’s eInfrastructure
- Masaryk University
- EGI fedCloud
- ELIXIR AAI
- SAGrid
Testing deployments

- Portuguese NREN
- VŠUP
- GARR CloudIdP
- Eko-Connect Nigeria
- SIFULAN Malaysia