Persistent Reservations

Introduction to proposal 08-025

Kevin Butt
kdbutt@us.ibm.com

Persistent Reserve Out - Actions

- Register
  - REGISTER
  - REGISTER AND IGNORE EXISTING KEY
- Reserve
  - RESERVE
  - REGISTER AND MOVE
- Removing by reservation holder
  - RELEASE
  - REGISTER (service action reservation key=zero)
- Removing by non reservation holder
  - PREEMPT (and potentially establish new reservation)
  - PREEMPT AND ABORT (and potentially establish new reservation)
  - CLEAR (for error recovery scenario's)

NOTE: Reservation Key is used as a non-unique identifier

> "If an application client uses a different reservation key for each I_T nexus, the application client may use the reservation key to associate the persistent reservation with the I_T nexus that holds the persistent reservation. This association is done using techniques that are outside the scope of this standard."

Persistent Reserve In

- Report Registrations
  - READ KEYS
- Report Reservation
  - READ RESERVATION
- Report Registrations and Reservation
  - READ FULL STATUS

Persistent Reservation Types

- Exclusive to single I_T nexus (i.e., the pesistent reservation holder)
  - Write Exclusive
  - Exclusive Access
- Exclusive to all registered I_T nexuses (only one persistent reservation holder)
  - Write Exclusive – Registrants Only (RO)
  - Exclusive Access – Registrants Only
- Exclusive to all registered I_T nexuses (each registered I_T nexus is a persistent reservation holder)
  - Write Exclusive – All Registrants (AR)
  - Exclusive Access – All Registrants

NOTE: There is no protection against an unregistered I_T nexus registering when there is a reservation. For RO and AR types this means there is no protection from any host joining the reservation on the joining hosts whim.

Purpose of All Registrants and Registrants Only Types

- Response I received from Rob Elliot (HP) when I asked this question.

For disks, the reservation commands are used to manage access to a "quorum disk" by multiple hosts that are part of a cluster. The information on the quorum disk is then used to control how they access all the data disks.

Cluster members are loosely coordinated; they are assumed to not be aggressively trying to interfere with each other, but can inadvertently do so if their intercluster communication mechanism (e.g. Ethernet) is disrupted (called a "split brain" problem). They are all assumed to be running the same algorithm - if you mix Solaris clusters and Windows clusters, they might make a mess. The quorum disk provides a way for two compatible clusters to not both think they own the same set of drives.

The "Registrants only" type is the most used. A cluster designed around that type must declare one cluster owner (the persistent reservation holder) at a time; if the owner fails, another must be elected. If a non-owner fails (a host that is registered but not the persistent reservation holder), it's no big deal.

The "All registrants" type is an optimization that makes all cluster members equal, so if any member fails, the others just continue. I don't know if any cluster software has started to use this yet. It takes many years to deploy changes to this kind of software.

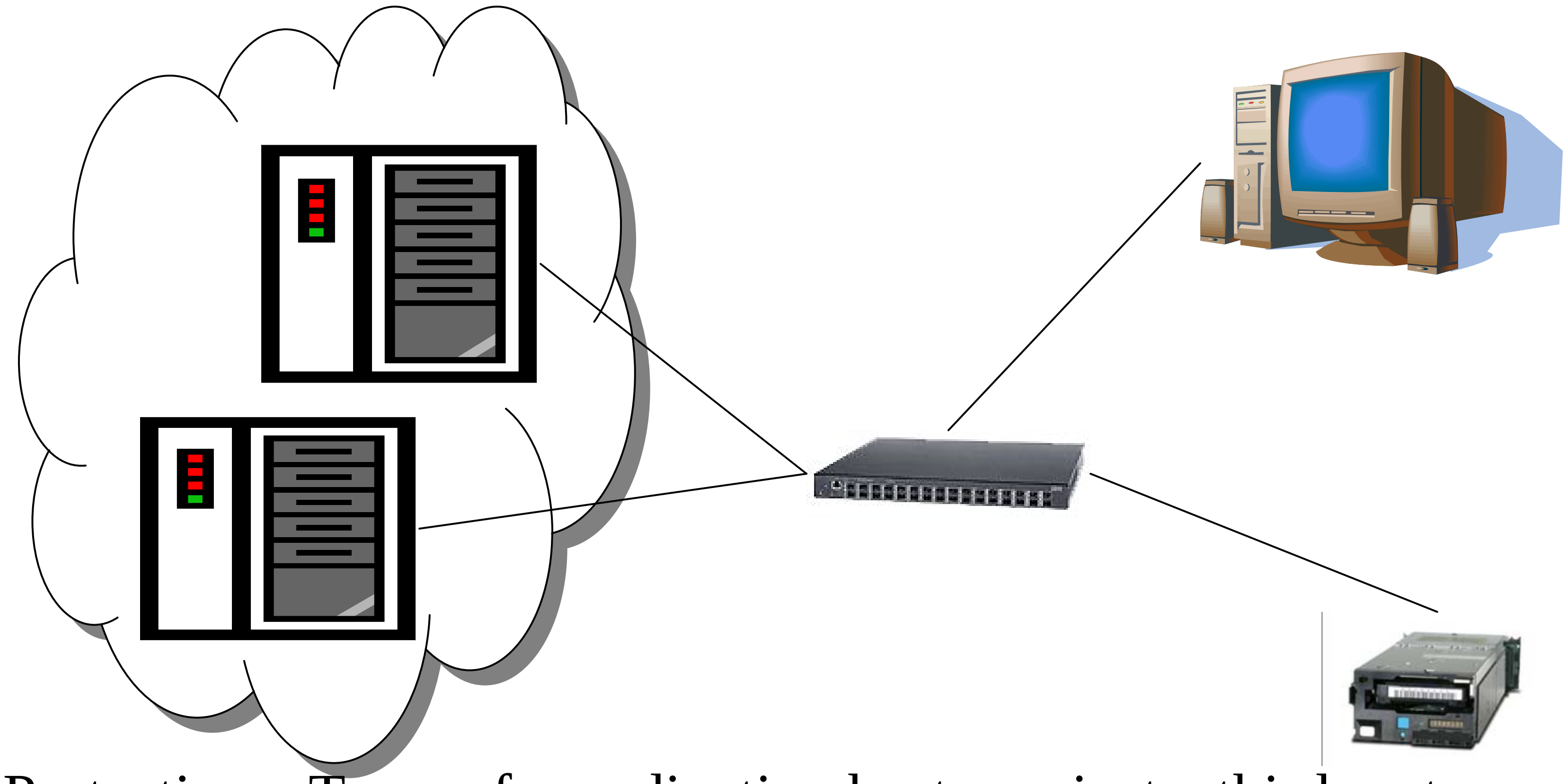
If an alien cluster is exposed to the drive, it can easily break through and start using the drive (the REGISTER AND IGNORE EXISTING KEY service action is a shortcut for doing this). You need to use zoning and access control features to separate the clusters and the logical units they intend to use from each other.

Some background information:

The January 1999 ENDL Letter had a good reservations tutorial.

Microsoft used to use RESERVE/RELEASE with periodic bus resets to perform a "challenge" and make sure the cluster owner is still active. They added support for persistent reservations in Windows 2003 SP1. Windows 2008 will require persistent reservations.

What is lacking?



Protecting a Team of coordinating hosts against a third party.

The following slides describe what the proposal (08-025) attempts to do.

Kevin Butt
kdbutt@us.ibm.com

New Persistent Reservation Types

- Exclusive to single I_T nexus (i.e., the pesistent reservation holder)
  - Write Exclusive
  - Exclusive Access
- Exclusive to all registered I_T nexuses (only one persistent reservation holder) (no protection from any host joining the reservation on the joining hosts whim)
  - Write Exclusive – Registrants Only (RO)
  - Exclusive Access – Registrants Only
- Exclusive to all registered I_T nexuses (each registered I_T nexus is a persistent reservation holder) (no protection from any host joining the reservation on the joining hosts whim)
  - Write Exclusive – All Registrants (AR)
  - Exclusive Access – All Registrants
- Exclusive to a Team of I_T nexuses (each I_T nexus in the Team is a persistent reservation holder)
  - Write Exclusive – Team
  - Exclusive Access – Team

NOTE: There is no protection against an unregistered I_T nexus registering when there is a reservation. Depending on the Type of reservation there may or may not be protection against any host joining the reservation on the joining hosts whim.

How to create team reservation

- Since the Reservation Key is not guaranteed unique, it cannot be used.
- TransportID is guaranteed unique, so it could be used.
- Create an additional ID called Team Key that is intended to be unique (only unique because of obfuscation)
- Use RESERVE service action
  - TYPE is one of the Team types
  - parameter list is basic parameter list:
    - Shall contain a Team Key (Team Key is never reported in PR In); and
    - may contain additional parameter data containing the TransportID descriptor used by the SPEC_I_PT (Table 133 SPC-4r11) filled in with TransportID's of participant initiator ports

How to remove Team Reservation

- Persistent Reserve Out from any reservation holder
  - RELEASE
- An unregister from any reservation holder removes only the I_T nexus on which it is received. The Team reservation stays intact unless it is the last member of the Team.
  - REGISTER (service action reservation key=zero)

    NOTE: allows for a hot swap of an HBA or the taking down of an initiator without disturbing reservation

How Team reservation is reported

- Persistent Reserve In
  - READ KEYS
    - No change
  - READ RESERVATION
    - RESERVATION KEY is set to zero
  - REPORT CAPABILITIES
    - Support Bit
      - Add Team Member List Capable (TML_C)
    - Persistent Reservation Type Mask format
      - Add EX_AC_T & WR_EX_T
  - READ FULL STATUS
    - No change. There will be one descriptor for each I_T nexus and each I_T nexus that is part of the Team will set the R_HOLDER bit to one.

Adding/Removing an I_T nexus to a Team Reservation?

- An I_T nexus sends a Reserve with a matching Team Key
  - Adds this I_T nexus to the team
- Any persistent reservation holder may send a Persistent Reserve Out command with a new set of TransportID's. This performs in one atomic action:
  - Changes the team to contain only the I_T nexuses in the newly sent list of TransportID's.
    - **NB** if the new TransportID list does not contain a pre-existing I_T nexus, then any I_T nexus that loses it's reservation as a result of this shall receive a UA.

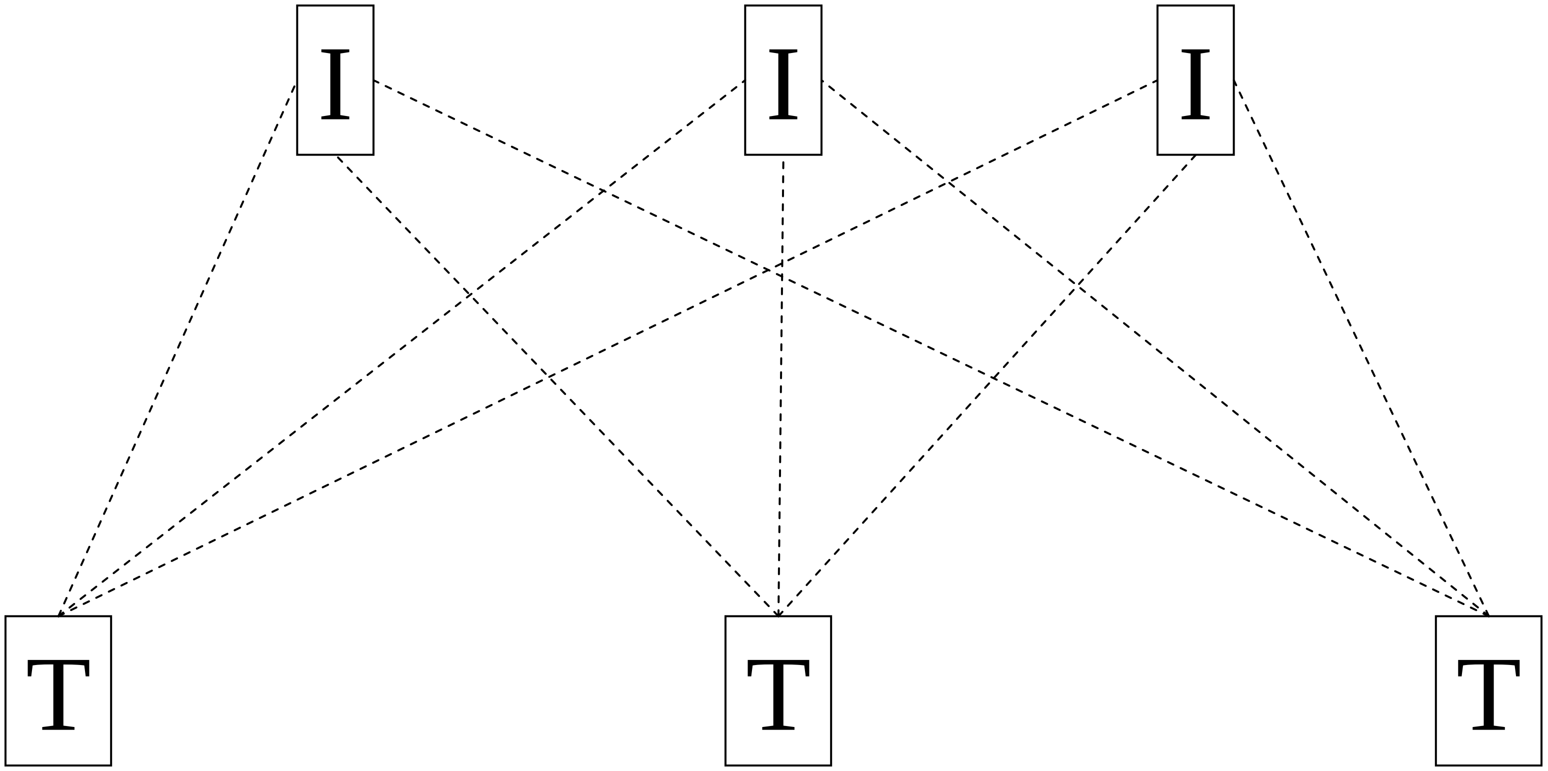
See the Proposal

- T10/08-025

Kevin Butt
kdbutt@us.ibm.com

Backup Slides

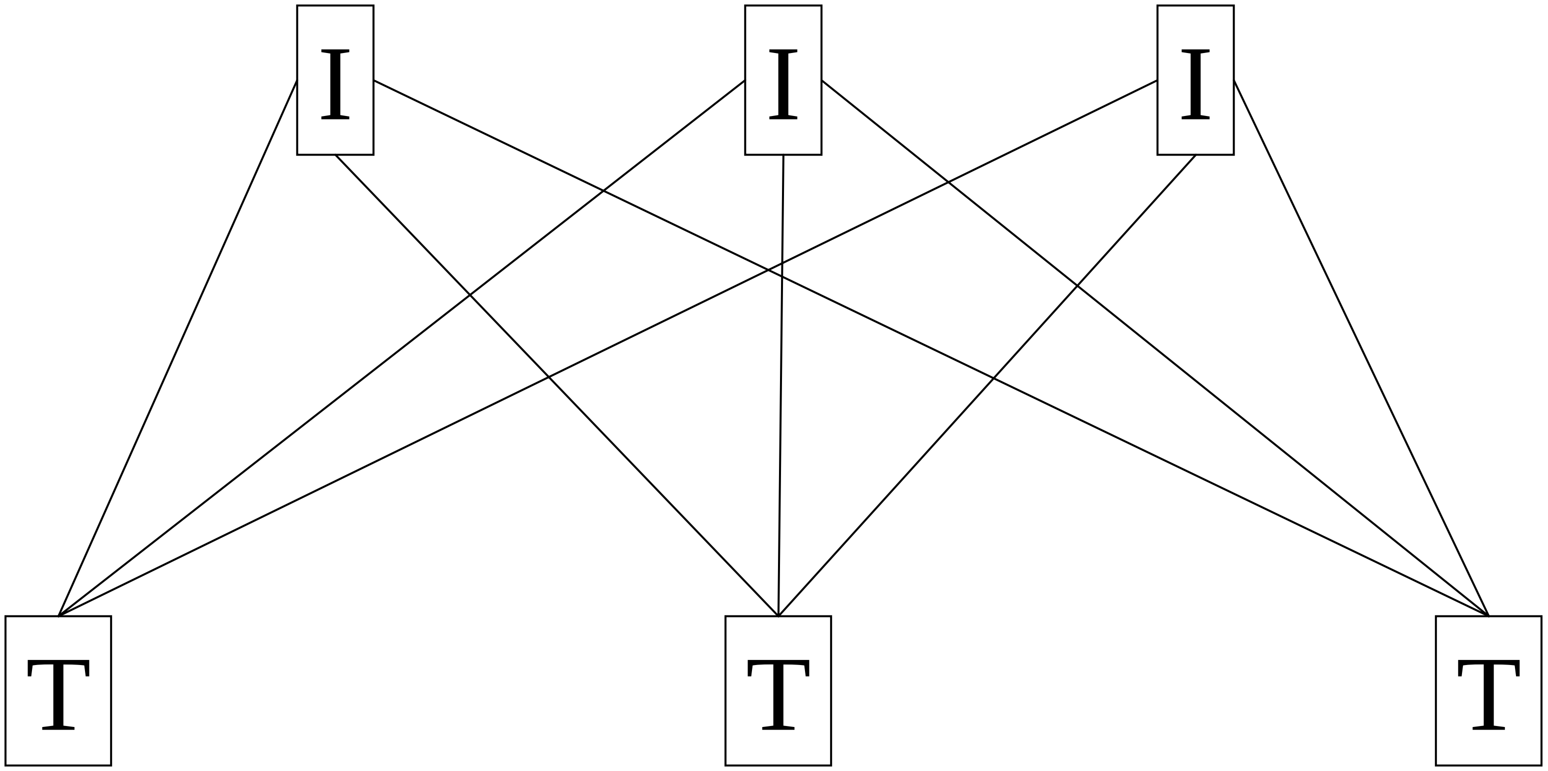
Registration I_T nexuses



I = Initiator Port

T = Target Port

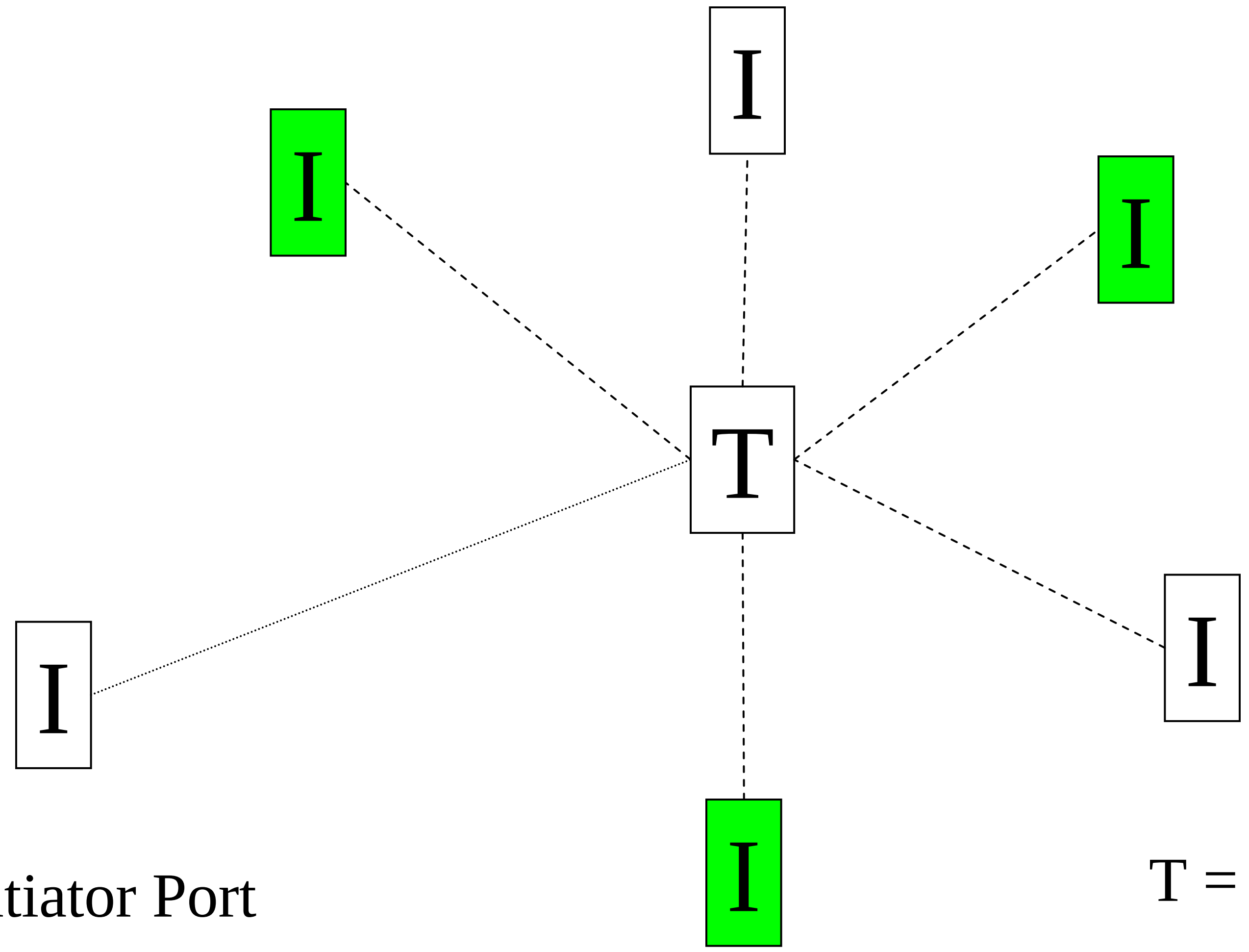
Reservation I_T nexuses



I = Initiator Port

T = Target Port

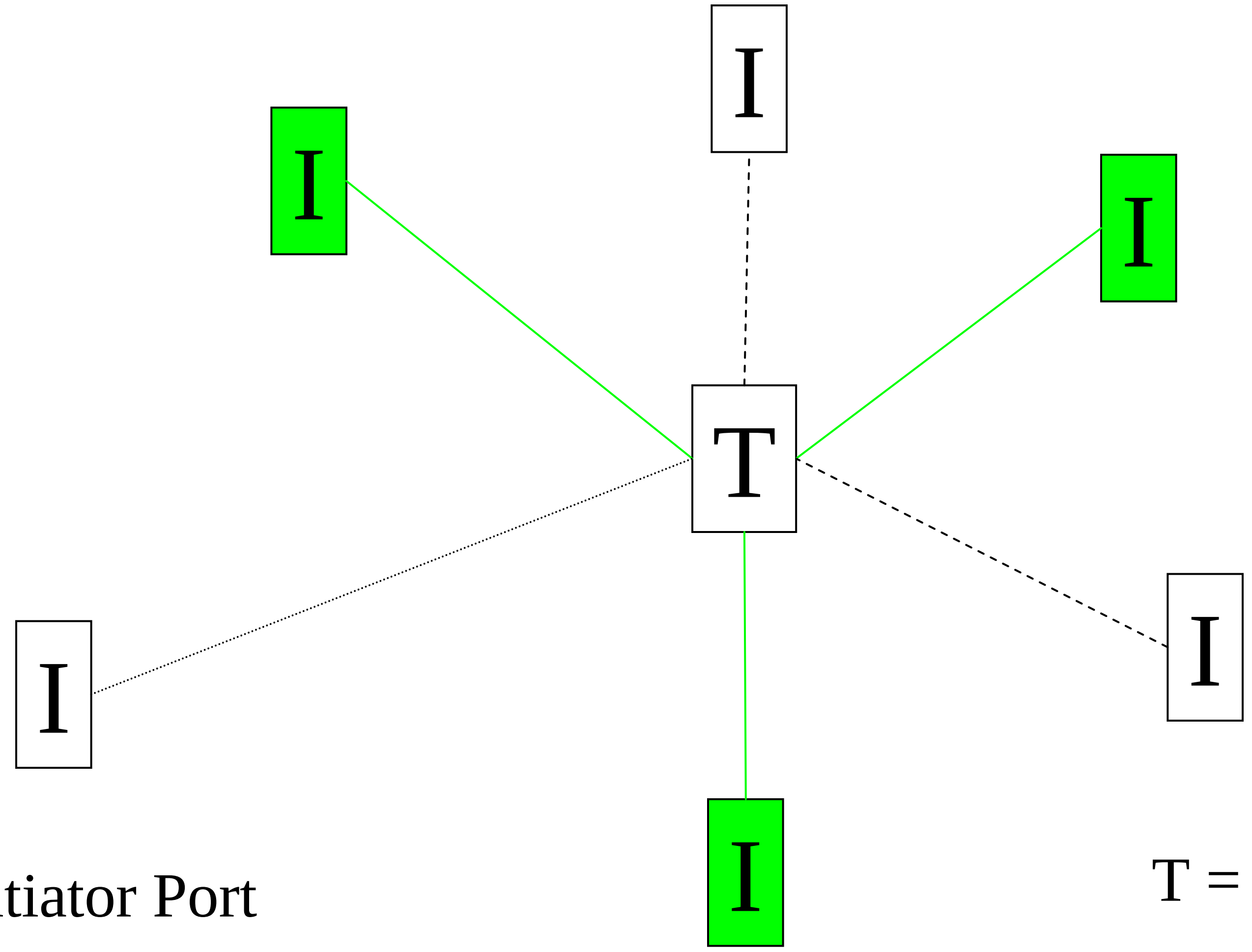
Green I’s desire to share an exclusive access



I = Initiator Port

T = Target Port

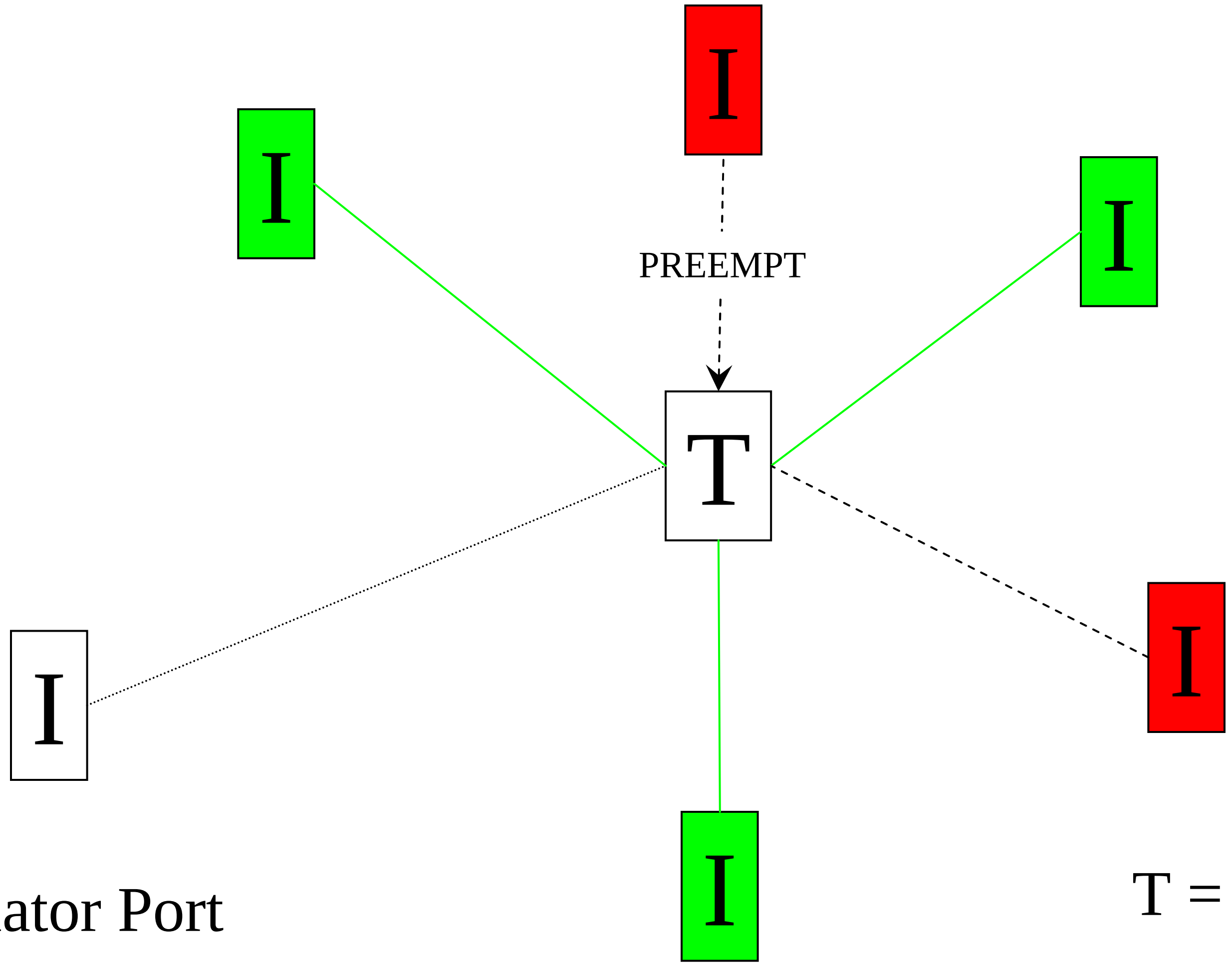
Green I’s share an exclusive access reservation



I = Initiator Port

T = Target Port

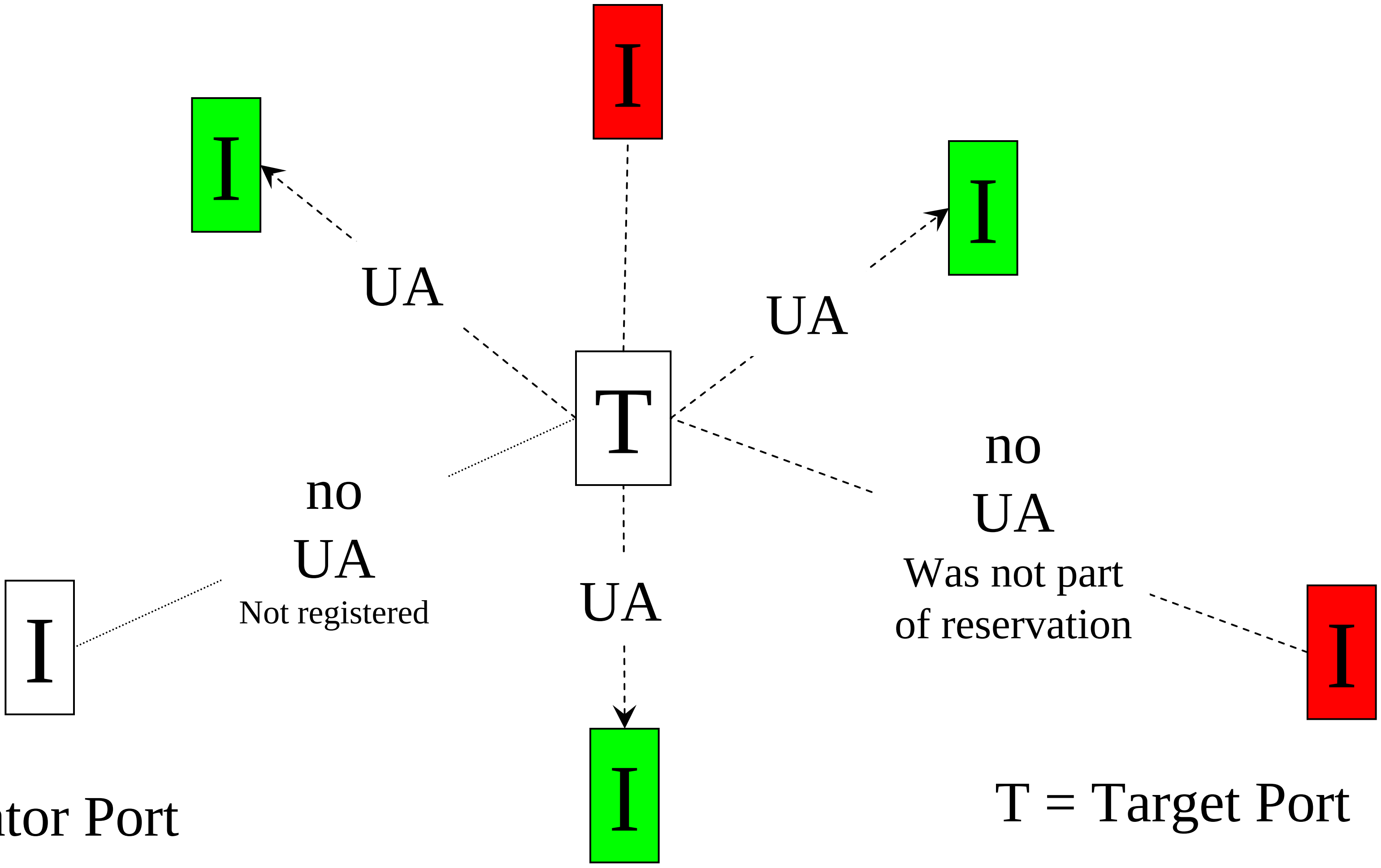
A Red I takes the reservation away (i.e., PREEMPT)



I = Initiator Port

T = Target Port

Reservation was taken away (i.e., PREEMPT)

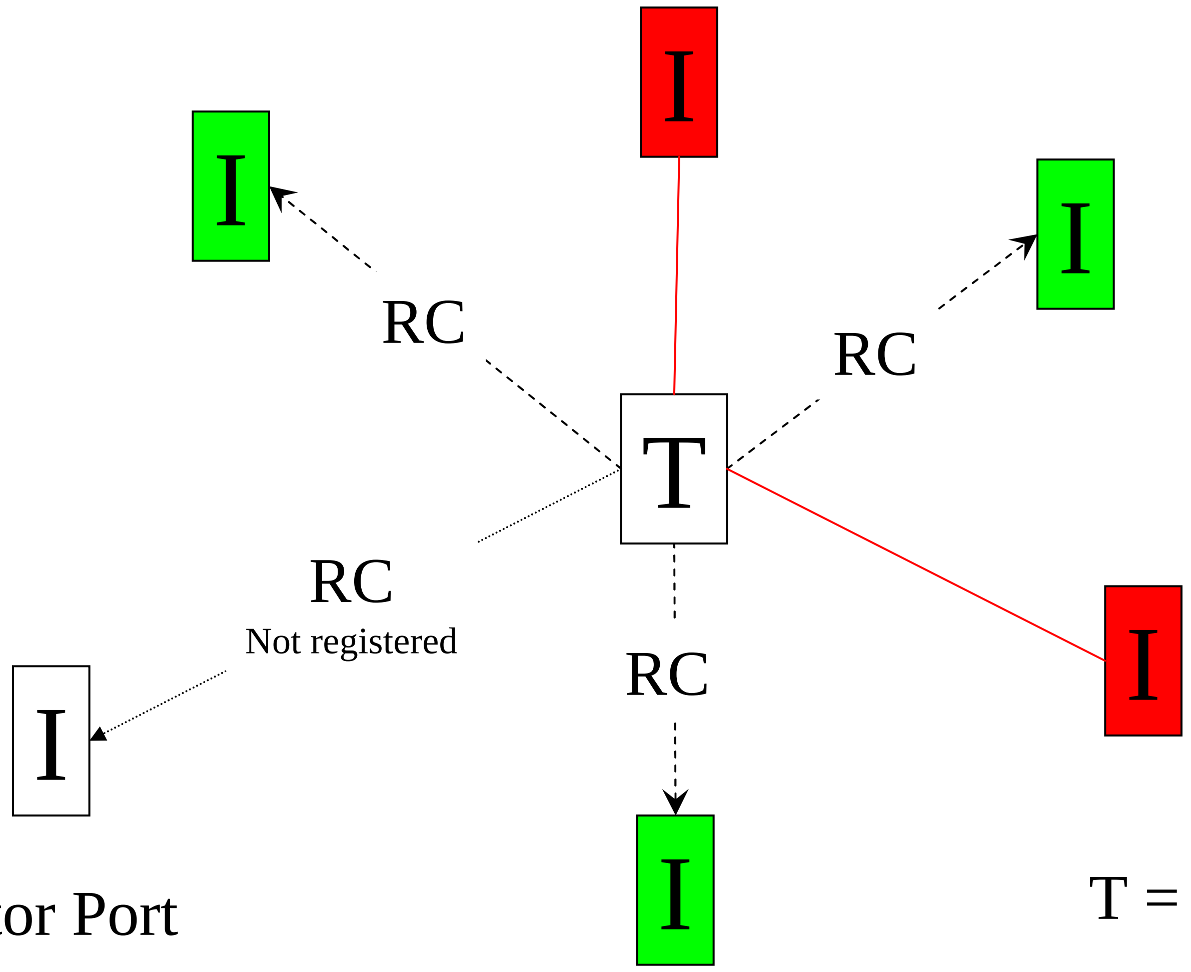


I = Initiator Port

T = Target Port

Kevin Butt
kdbutt@us.ibm.com

Red I’s establish reservation



I = Initiator Port

T = Target Port

Kevin Butt
kdbutt@us.ibm.com