GUIDE SHARE EUROPE
Deutsche Region

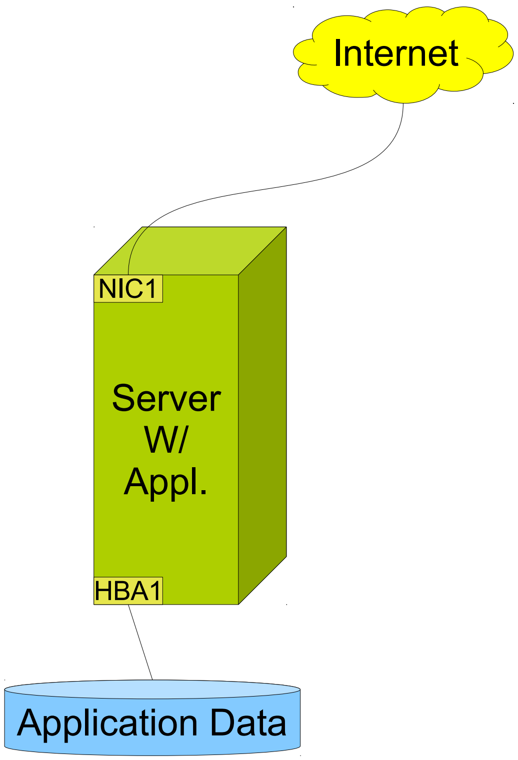
IBM

Data High Availability
RAID and Mirroring Technology

October 23, 2012

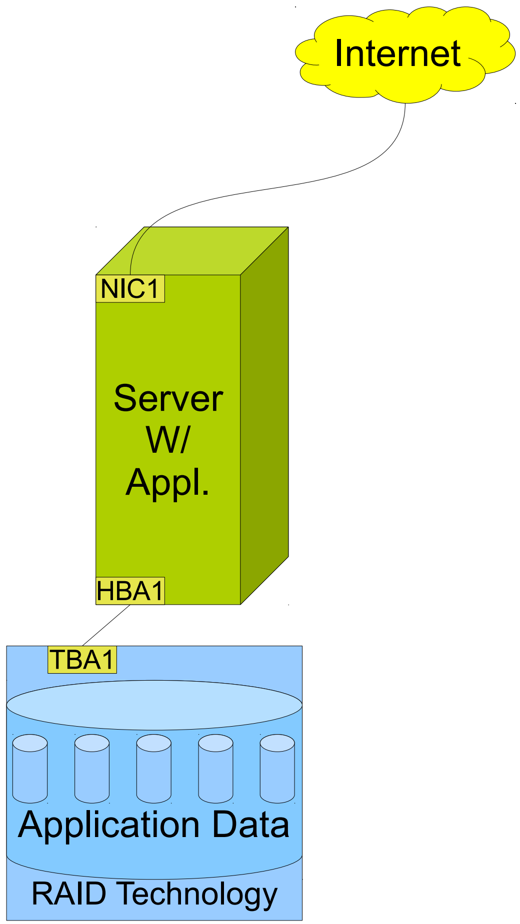
Dr. Holger Smolinski
smolinski@de.ibm.com
Global Client Center, IBM Germany R&D

Data Availability is critical to Application Availability



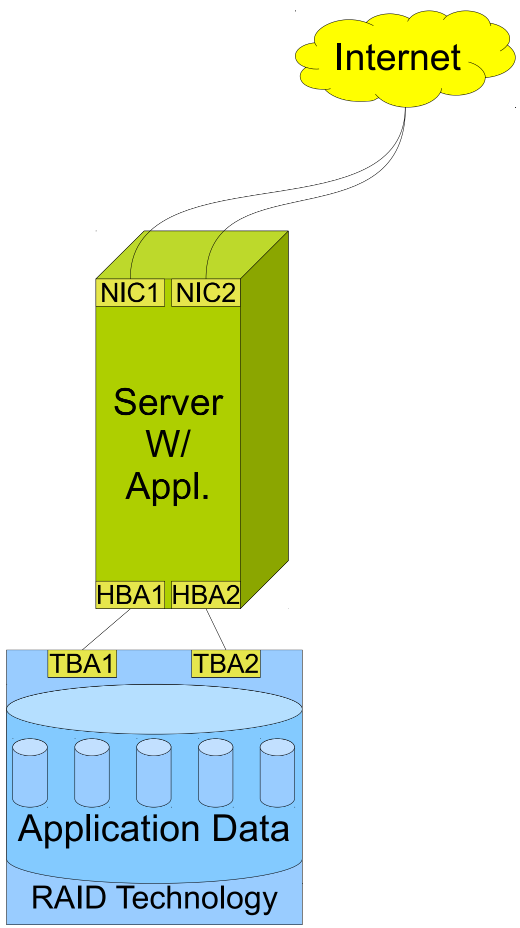
- **In an operational model, which does not include availability measures, there are many single points of failure:**
  - Internet connection
  - Network interface
  - Server node
  - Application instance
  - Storage connectivity
  - Data residing on disk
- **If any of the above fails, the entire application cannot run.**

Data on disk redundancy is the first step considered



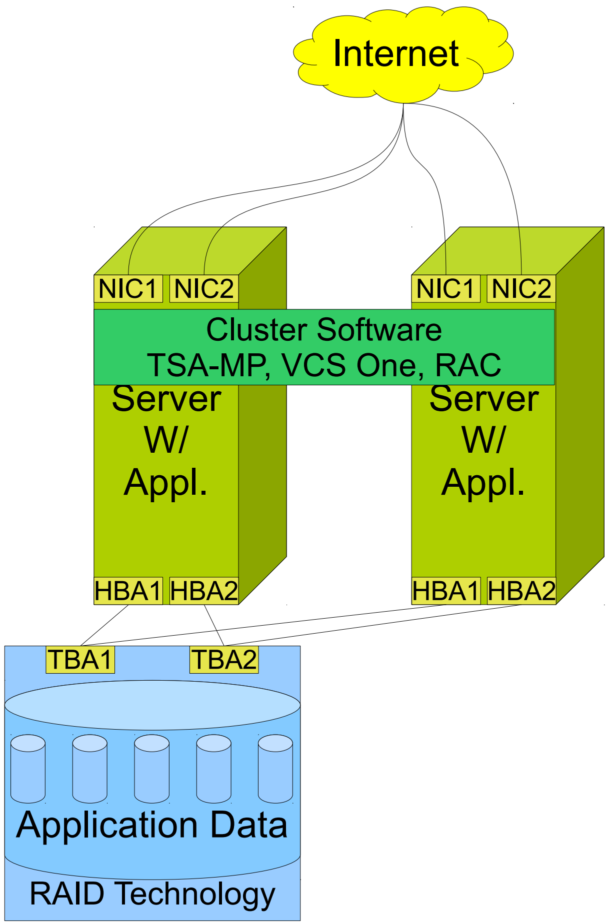
- **RAID (Redundant Array of Inexpensive Disks) protects data against single data drive failure**
  - RAID 0: linear concatenation – no protection
  - RAID 1: mirrored disks – all data is written to two disk drives. Single disk outage can be detected and recovered.
  - RAID 5: three units of data are distributed on 5 disks. Single and double disk outage can be detected and recovered. Single data corruption can be detected and corrected.
  - RAID 10: combination of RAID0 and RAID1 – characteristics of RAID1.

Infrastructure redundancy is the second step towards application availability



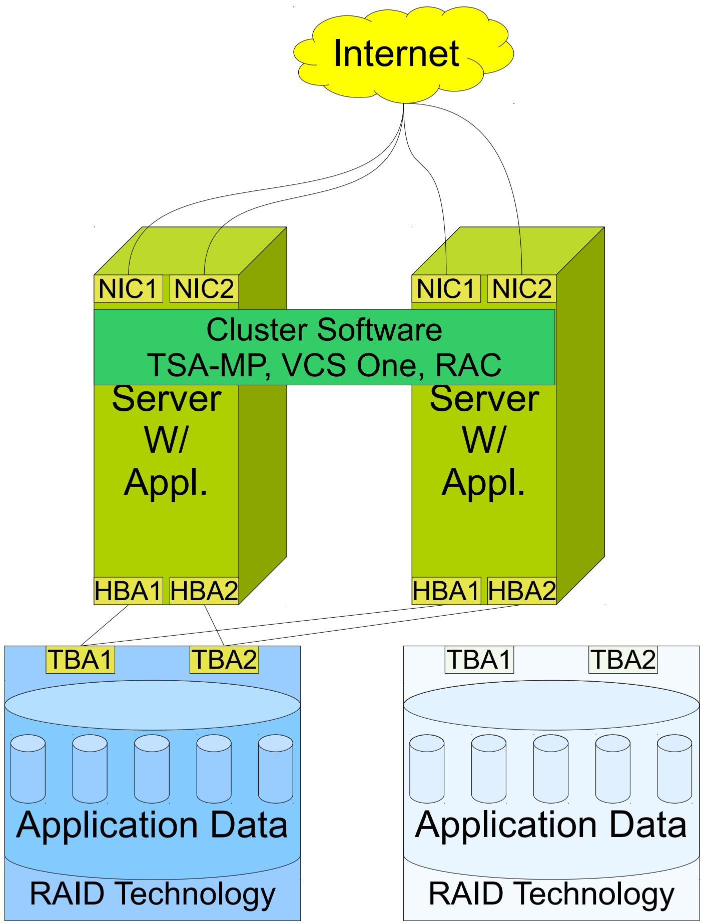
- **Paradigm: make all external connectivity redundant**
  - Mitigate risk of defective cabling
  - Mitigate risk of defective parts
- **Network**
  - Since IP addresses are 'the address' of the application we will learn about network HA techniques in the networking track of this class
- **Storage**
  - Use independent paths to the storage controllers

In a third step a SPOF is eliminated by clustering the server



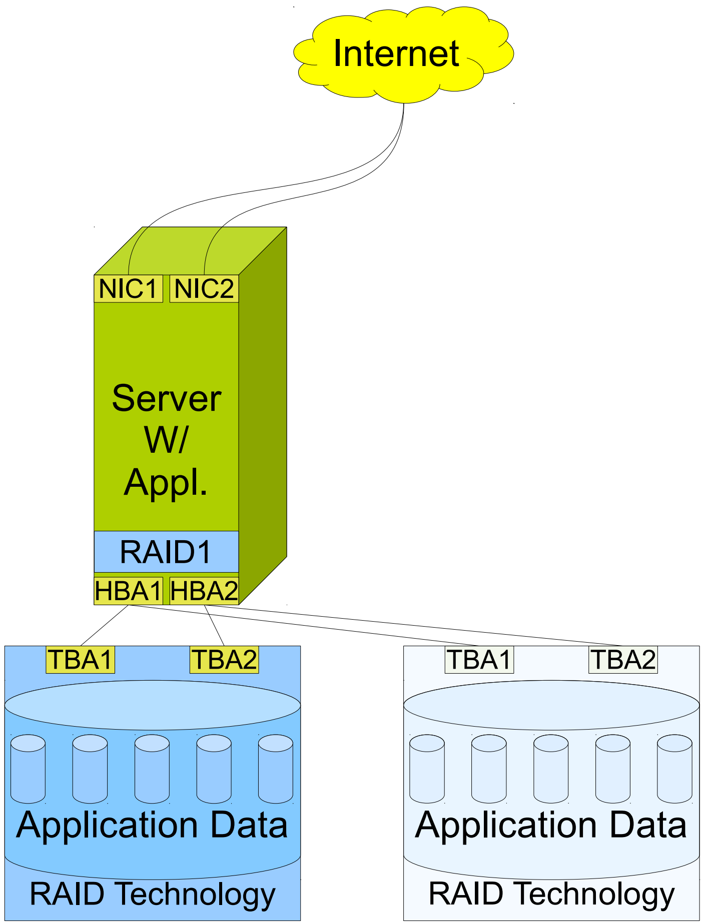
- **Sharing data across two servers is not a trivial problem**
  - Systems might cache data or file system metadata → both instances might see different contents or override their changes
  - Clustering Software must perform a selection of:
    - Lock data files and metadata to avoid colliding access
    - Use DIRECTIO to bypass operating system caches
    - Mandatorily assign ownership to data resources → STONITH in case of a cluster split

There is only one SPOF left: The storage controller



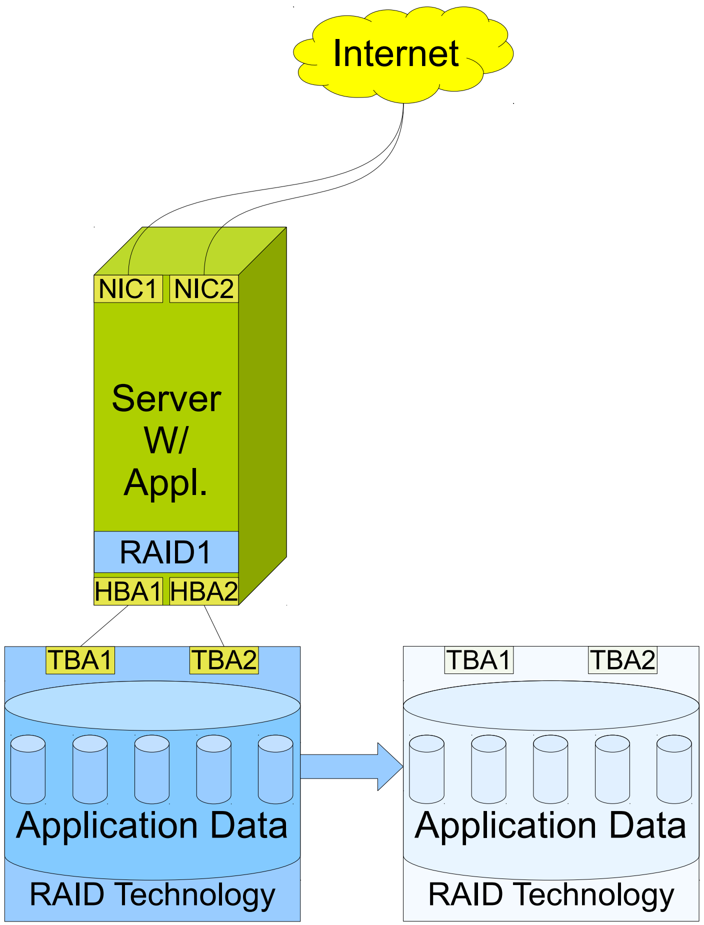
- **All previous components could be made redundant by just adding identical resources**
  - They either are stateless or the state becomes meaningless in case of an error.
- **But the disk subsystem has a state: 'the data on the disk' needs to be the same even if storage is duplicates**
- **Storage replication techniques are required!**

Option 1: Host Based RAID 1 – Disk Mirroring



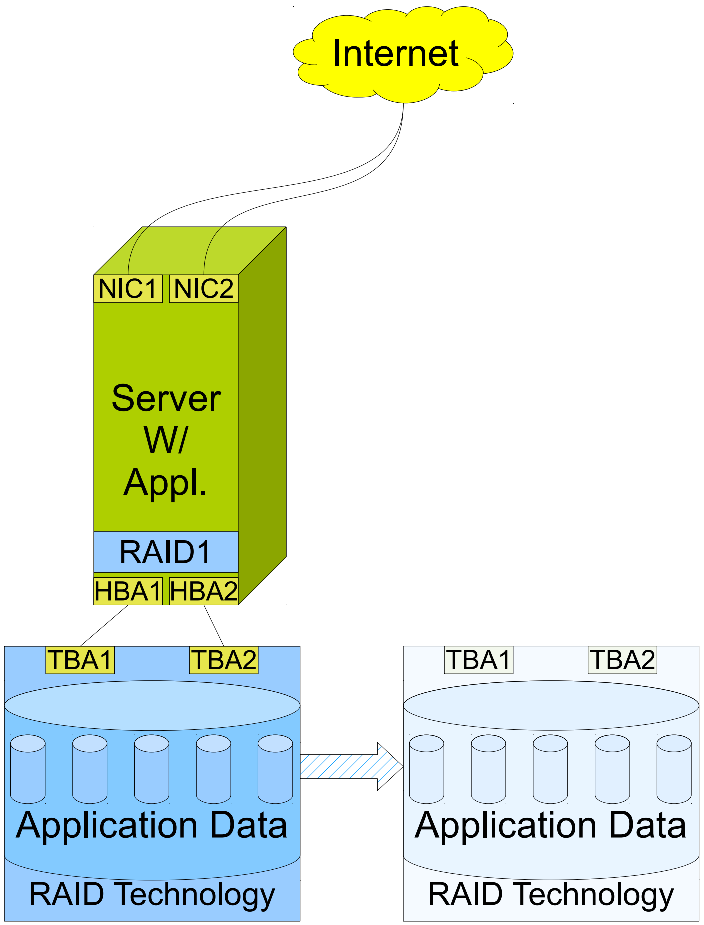
- **Linux brings two (3) host based RAID1 implementations**
  - 1. device mapper (LVM) based
  - 2. MD device driver (mdadm) based
  - 3. IBM LVM based real-time mirror
- **Properties**
  - Symmetric configuration
  - Host controls completion of request, data integrity policy and recovery procedure
- **First 2 implementations suffer from some weaknesses e.g.:**
  - Transient failure (timeout) handled as error → RAID array breaks or blocks.

Option 2: Controller Based Replication - synchronous



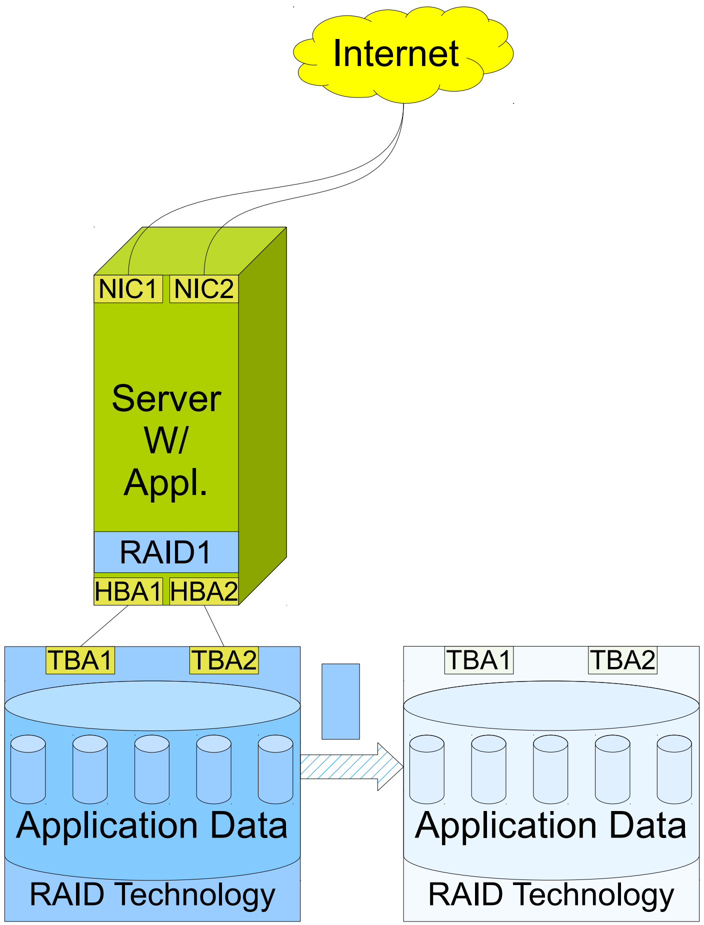
- **Example:**
  - IBM Metro Mirror (synchronous) aka synchronous PPRC
- **Properties**
  - Asymmetric configuration
  - Write request is reported complete after successful replication
  - Identical data on both sides guaranteed
  - Any failure on local/remote storage and/or replication link will block replication and IO requests

Option 3: Controller Based Replication - asynchronous



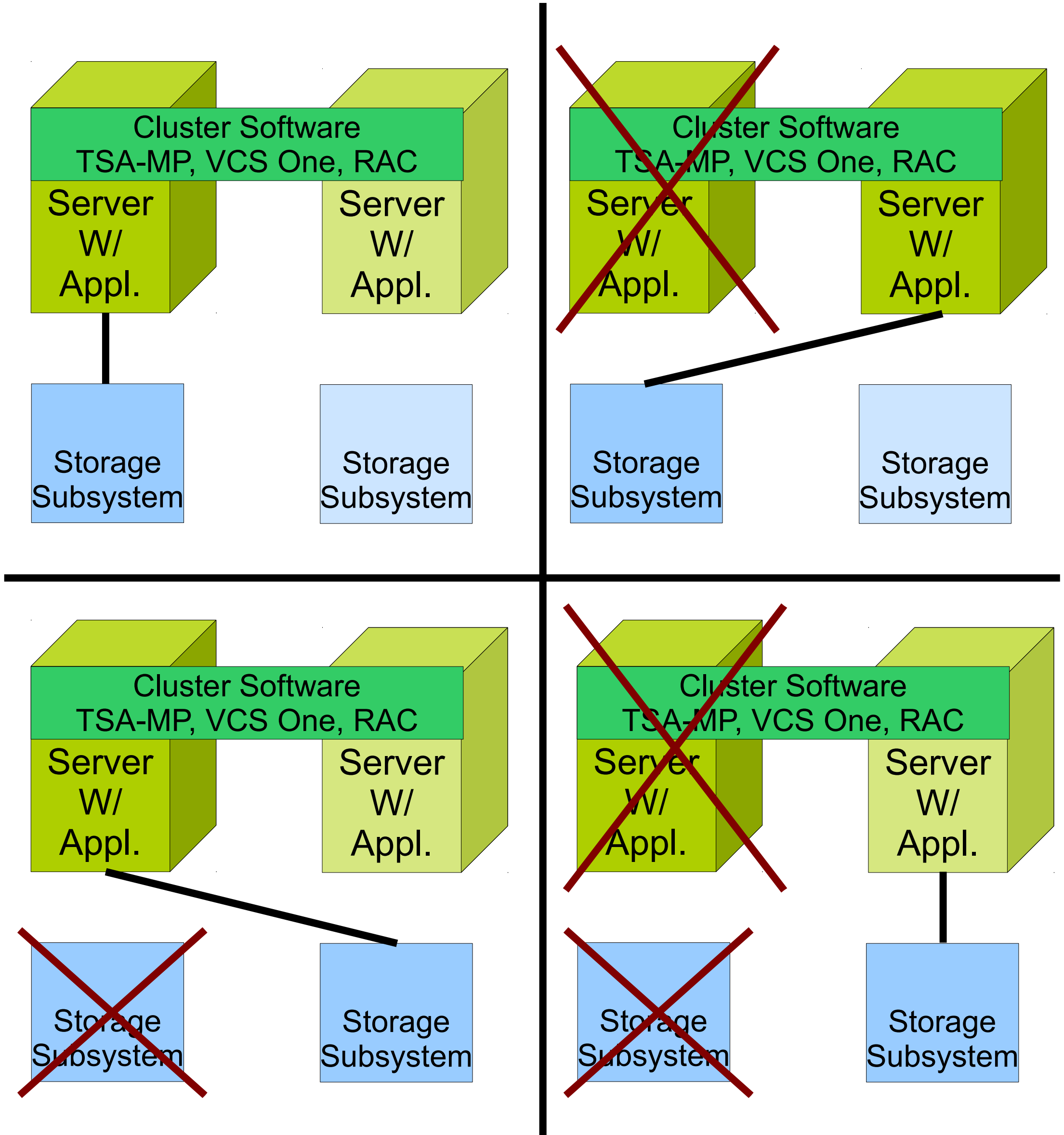
- **Example:**
  - IBM Metro Mirror (asynchronous) aka asynchronous PPRC
- **Properties**
  - Asymmetric configuration
  - Write request is reported complete after successful replication on primary side
  - Identical data on both sides guaranteed within size of backlog

Option 4: Controller Based Replication - batched



- **Example:**
  - IBM Global Mirror aka XRC
- **Properties**
  - Asymmetric configuration
  - Write request is reported complete after successful replication on primary side
  - Identical data on both sides guaranteed within size of backlog

Everything is redundant and disk storage replicated – let's build a HA solution!



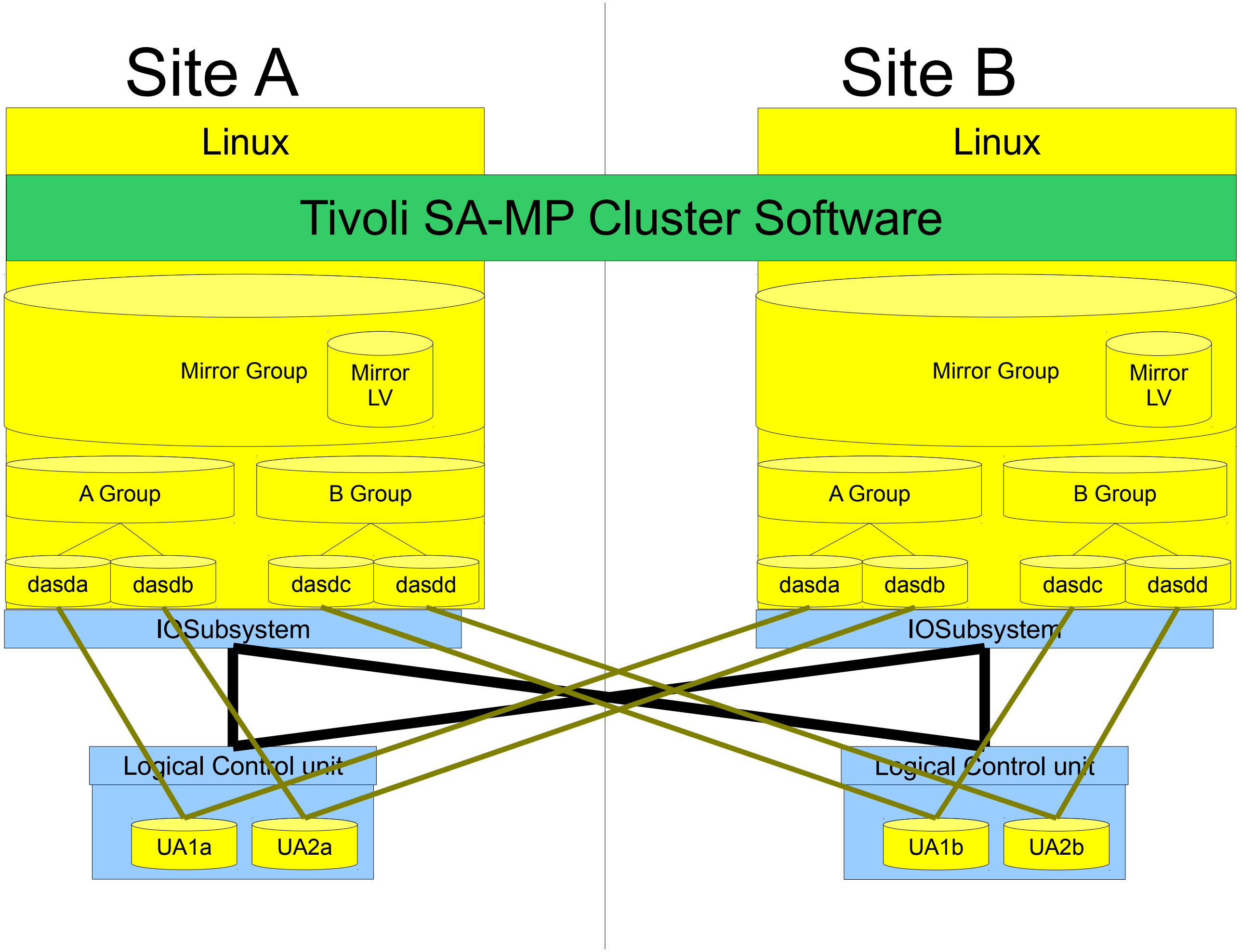
- **Left to Right: Failure of a server**
  - Cluster SW takes care of it
- **Top to Bottom: Failure of a storage subsystem**
  - Asymmetric solutions require the secondary to become primary
- **Top-Left to Bottom-Right can also be considered a site-failover:**
  - Replication latency might become an issue

However – it's not that simple...

- **Disk Storage does not only hold application data but – often overlooked:**
  - Swap space, holding user heap and stack data paged out from memory
  - Backing store for binary program text, loaded on demand, when code needs to be executed
  - Backing store for shared memory segments paged out on memory shortage.
- **In consequence for a multi-site HA solution:**
  - Synchronous replication may cause application stalls or outages
    - Delayed load of data, text, or heap&stack
    - Inbound network traffic may get queued and fill up memory
    - Outbound data buffers, too
  - Asynchronous replication may cause inconsistency of data on both sides
  - Symmetric Host based replication requires cabling from both hosts to both sites
  - Asymmetric Controller based replication requires identity switch (pri → sec → pri)
    - Usually some controlling software is required – JavaCL Tool, z/OS Sysples et.al.
- **Chose, which meets your NFRs best!**

Examples

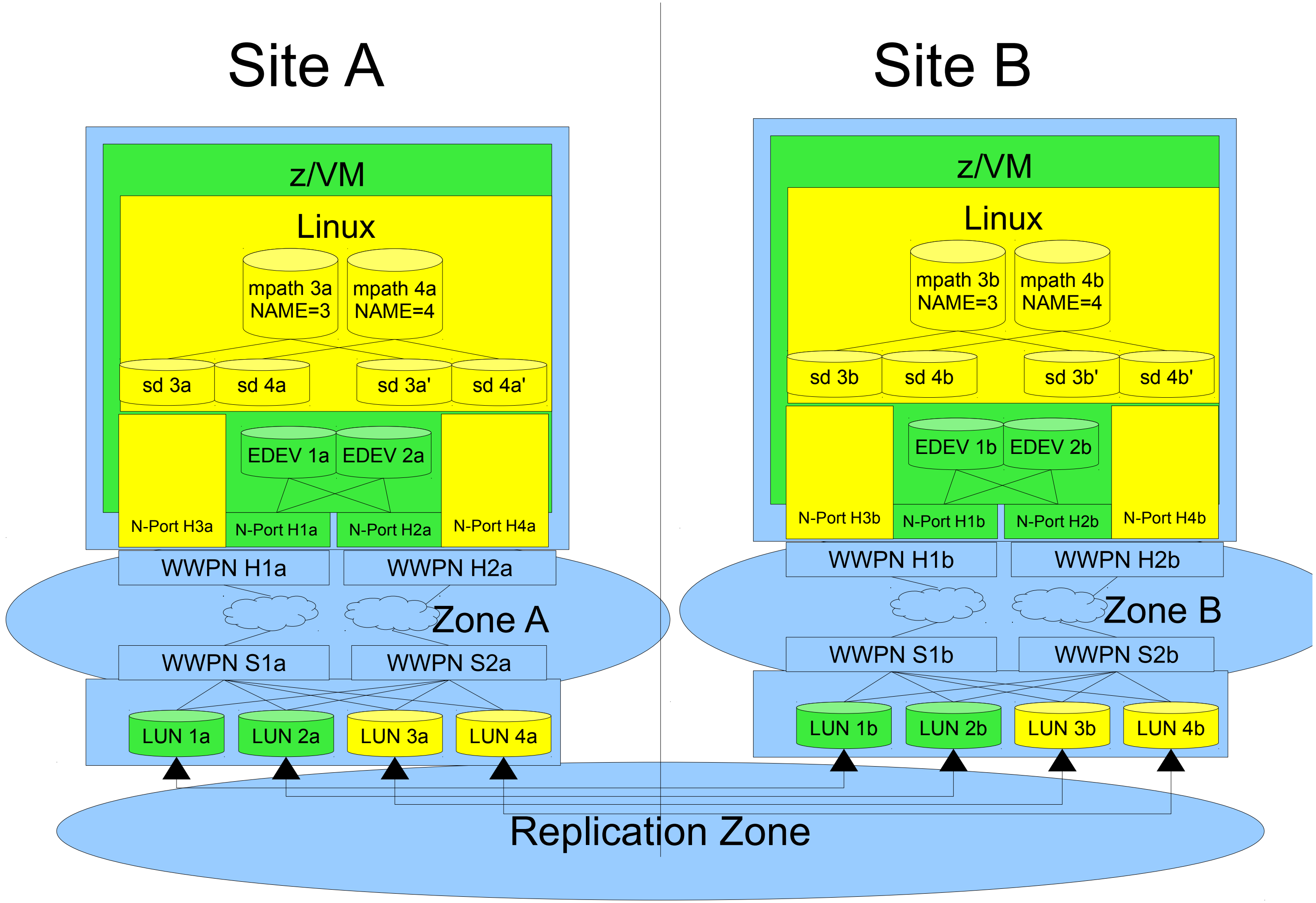
Example 1: LPAR Linux system, host based replication, LVM volumes



Example 1: Characteristics

- **Used IBM Real-Time Disk Mirroring (LVM based)**
  - Hybrid between synchronous and asynchronous mirroring
    - Synchronous until grace period has expired
    - Asynchronous then – up to 100% difference
      - automatic resynch after transient error
  - Tunable grace period guarantees application availability
  - Supports any type of data storage, swap space, journalling file systems
    - Concurrent access of cluster members prohibited
- **Tivoli System Automation for Multiplatform controls all resources**

Example 2: SCSI only z/VM & Linux system, controller based replication, 2 site configuration



Example 2: Configuration Actions required

- **Create three Zones for Site A, Site B, and Replication**
- **Configure z/VM to attempt to access both sides' FCP devices**
  - For z/VM LUN 1 define two statements:

```
EDEV 2a0 TYP FBA ATTR typ
     FCP_DEV 10a0 WWPN S1a LUN 1a
     FCP_DEV 11a0 WWPN S2a LUN 1a
EDEV 2b0 TYP FBA ATTR typ
     FCP_DEV 10b0 WWPN S1b LUN 1b
     FCP_DEV 11b0 WWPN S2b LUN 1b
```

  - Selection of the right EDEV will be performed automagically by VOLSER
- **Configure Linux to attempt to access both sides**
  - Add other side WWPNs manually to Linux system
  - Create 2 initial ramdisks – one for each site
    - Might be tricky, because not supported in standard `mkinitrd` tool
  - Specify 2 sections in /etc/zipl.conf – one for each site with `initrd` and `kernel command` line
  - Use boot program specifier in '`SET LOADDEV`' command to select site/configuration

Thank you ! Feel free to ask questions!

Questions?

Holger Smolinski

Certified IT Specialist

Linux on System z

Schönaicher Strasse 220
71032 Böblingen, Germany

Phone +49 (0)7031-16-4652
Holger.Smolinski@de.ibm.com