Dynamic Connectivity Service

Oscar Koeroo
JRA3

Content

- **What's the problem?**
- **What do we need?**
- **How do we want to solve it**
- **Our prototype**
  - and how does it work

What's the problem?

- **Most to all WNs (in LCG-2) can make outbound connection to almost any machine on the Internet**
  - No Firewalls that limits a user
    - A few possibilities are:
      - *WN publicly addressable*
      - *Inbound is prohibited and outbound is still free to use*
        - NAT box
        - Firewall rules
      - *WNs are locked up for any Internet traffic*
  - VOs request ability for their users to connect to there own servers
    - Pulling VO specific data on a WN
      - *Packages*
      - *Data*
    - Push result on to VO specific machines
    - Interactive
      - *Database access*

- **This means that every (rogue) user can do harmful things like:**
  - Launch DDoS - Grid Jobs can aid or start a DDoS on a (web-)server
  - Share Warez - Each machine can serve as Warez servers
  - Make a pass-through for Worms & Viruses

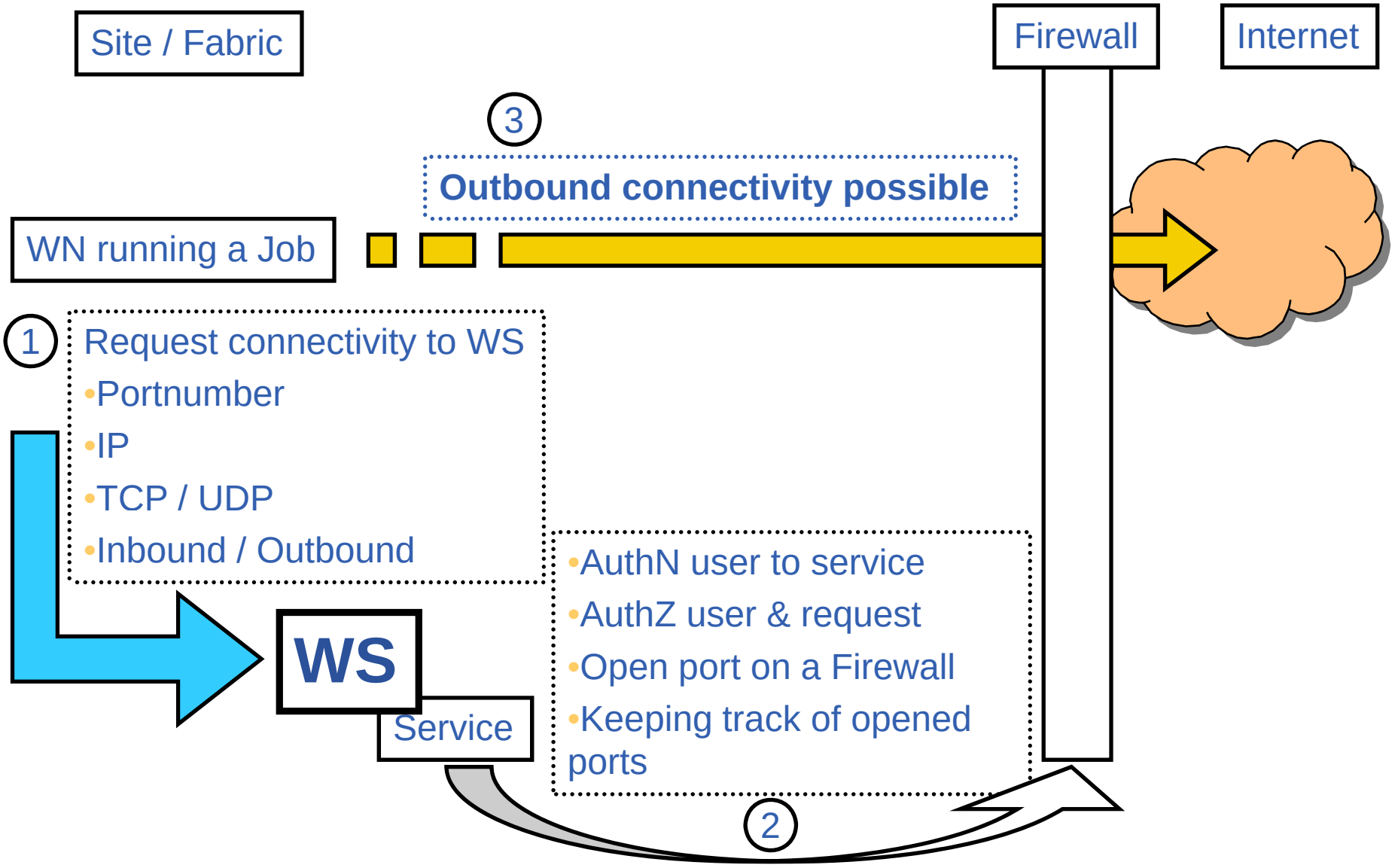
What we need…

- **Network containment**
  - We need to keep a user primarily contained inside the fabric
  - If users have a connectivity wish they can request it at the (concerning) resource centers
  - RCs need to be in full control of their (network) domain

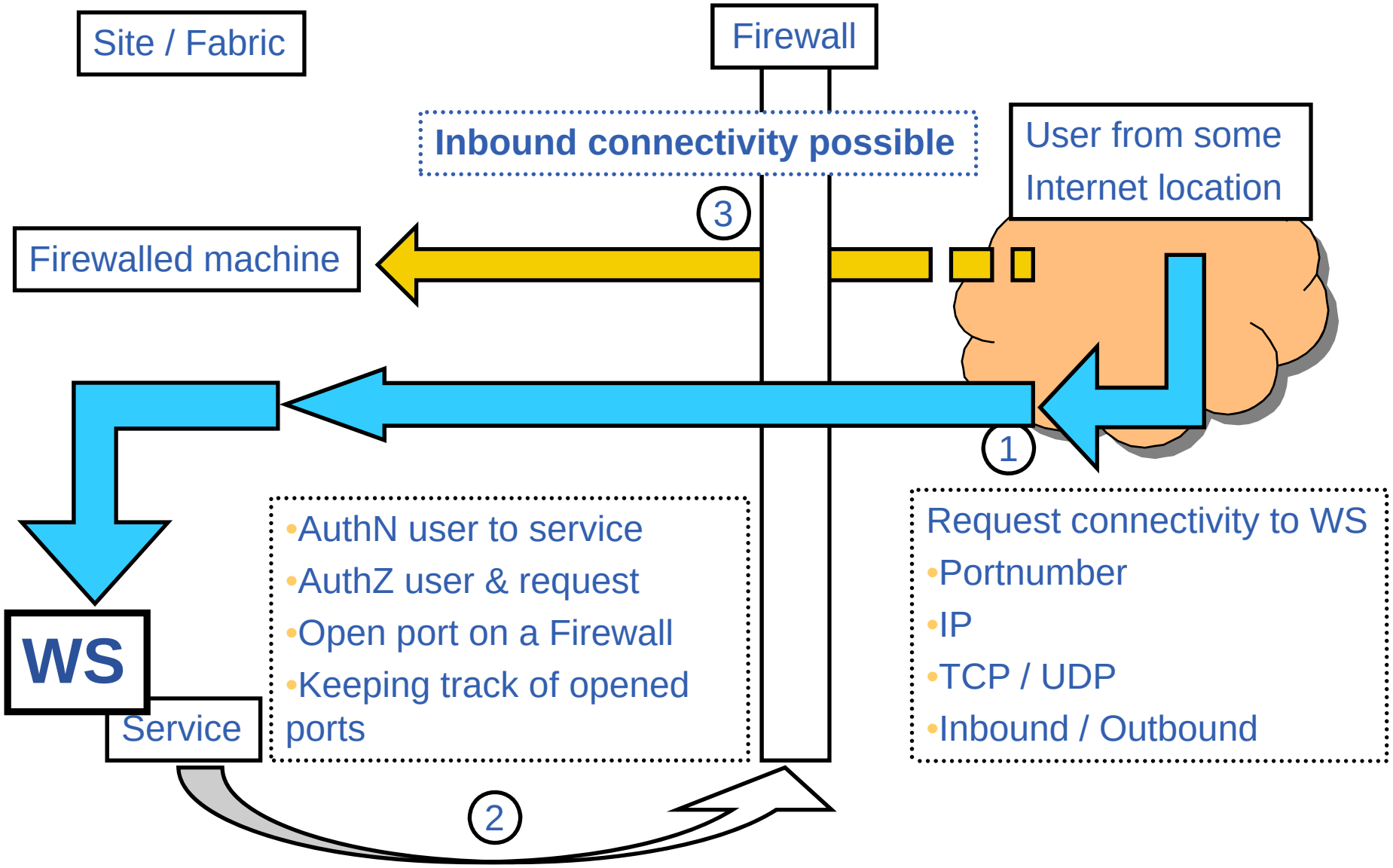
How do we want to solve it

- **Lockup a site tight**
  - From a WN's perspective and the running job:
    - No (direct) inbound connectivity
      - *Achieved by setting up a router, NAT box or Firewall (or some combination) prohibiting these connections*
    - No outbound connectivity
      - *The router, NAT box or firewall (or a combi.) prohibit these connections*
  - Narrow the static firewall rules for all Grid Services as much as possible
    - Grid services mutual authenticate themselves to other services with some kind of access control so they can be regarded as safe(r) connections
- **Only when needed open-up a port to make a (controlled) connection available**

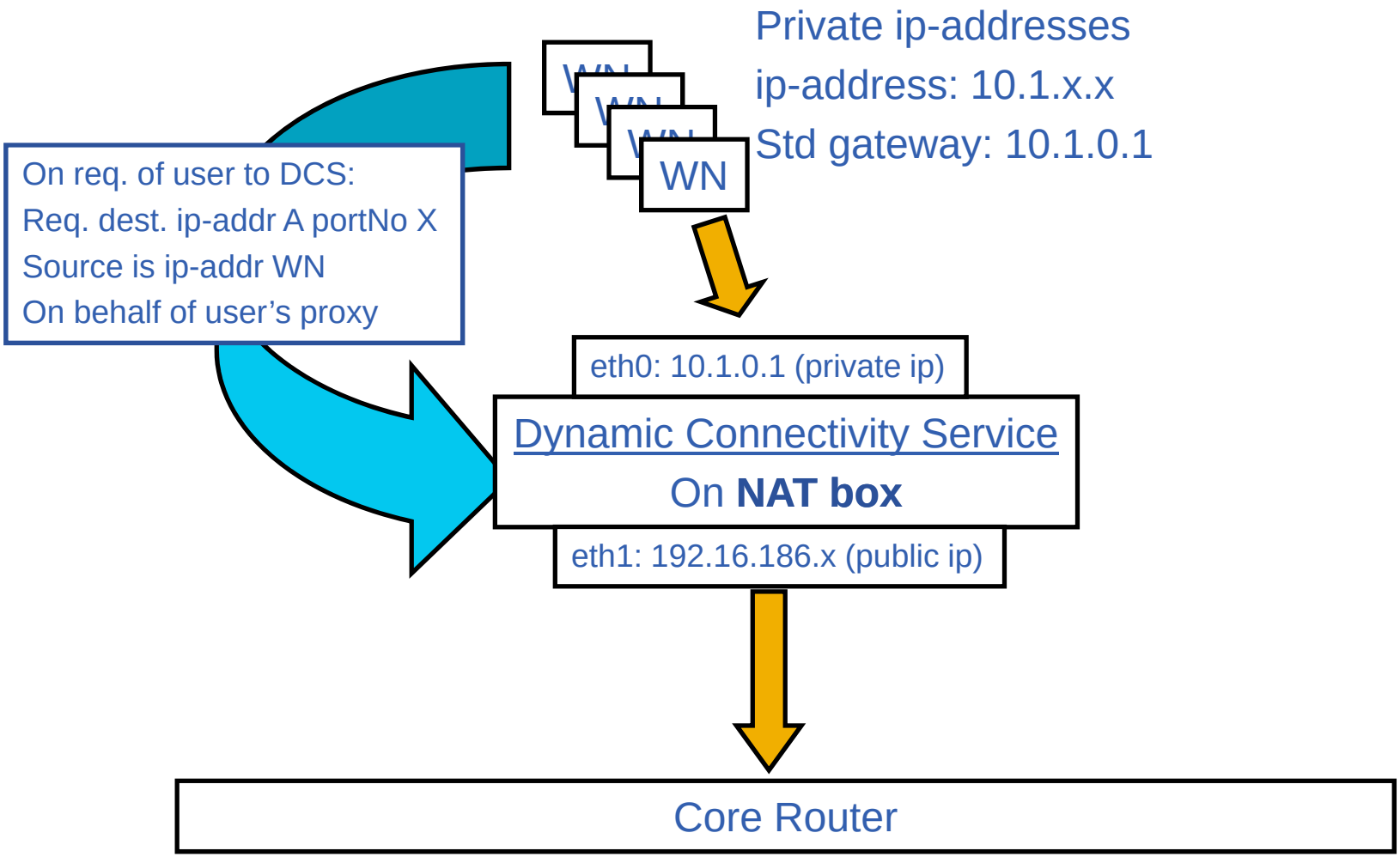
How does it work: Use Case #1



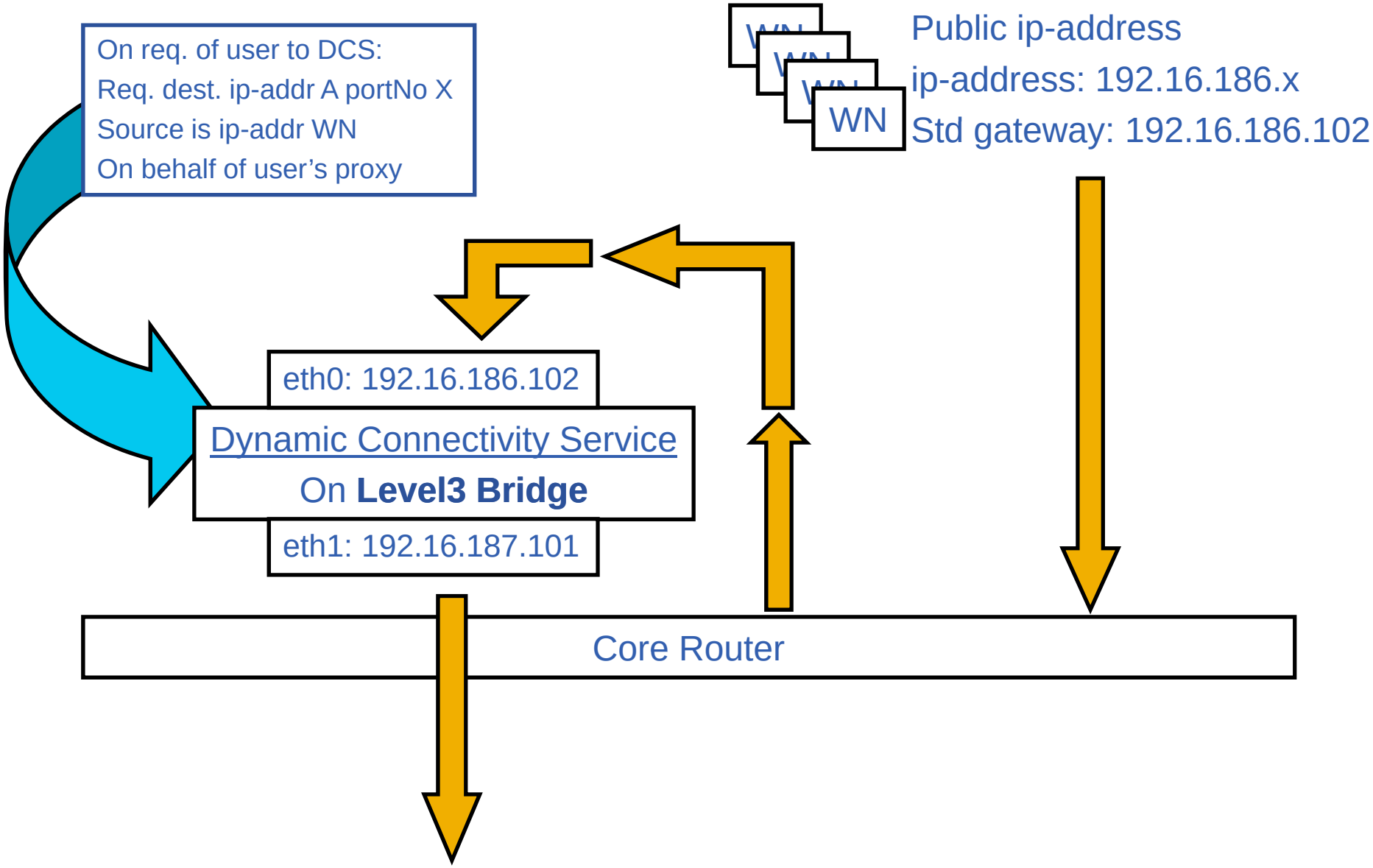
How does it work: Use Case #2



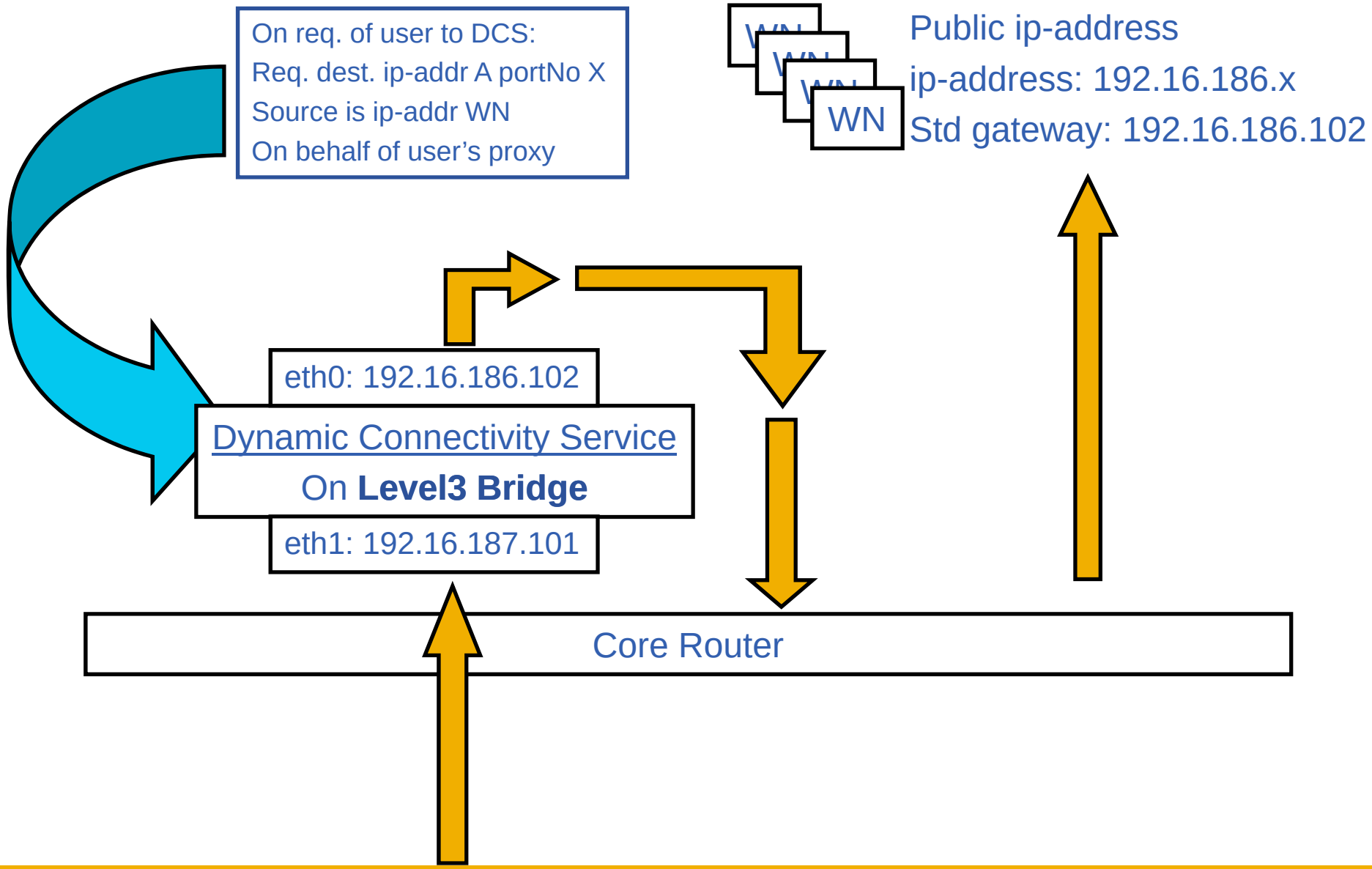
How to Deploy? – with NAT



How to Deploy? – with routers #1



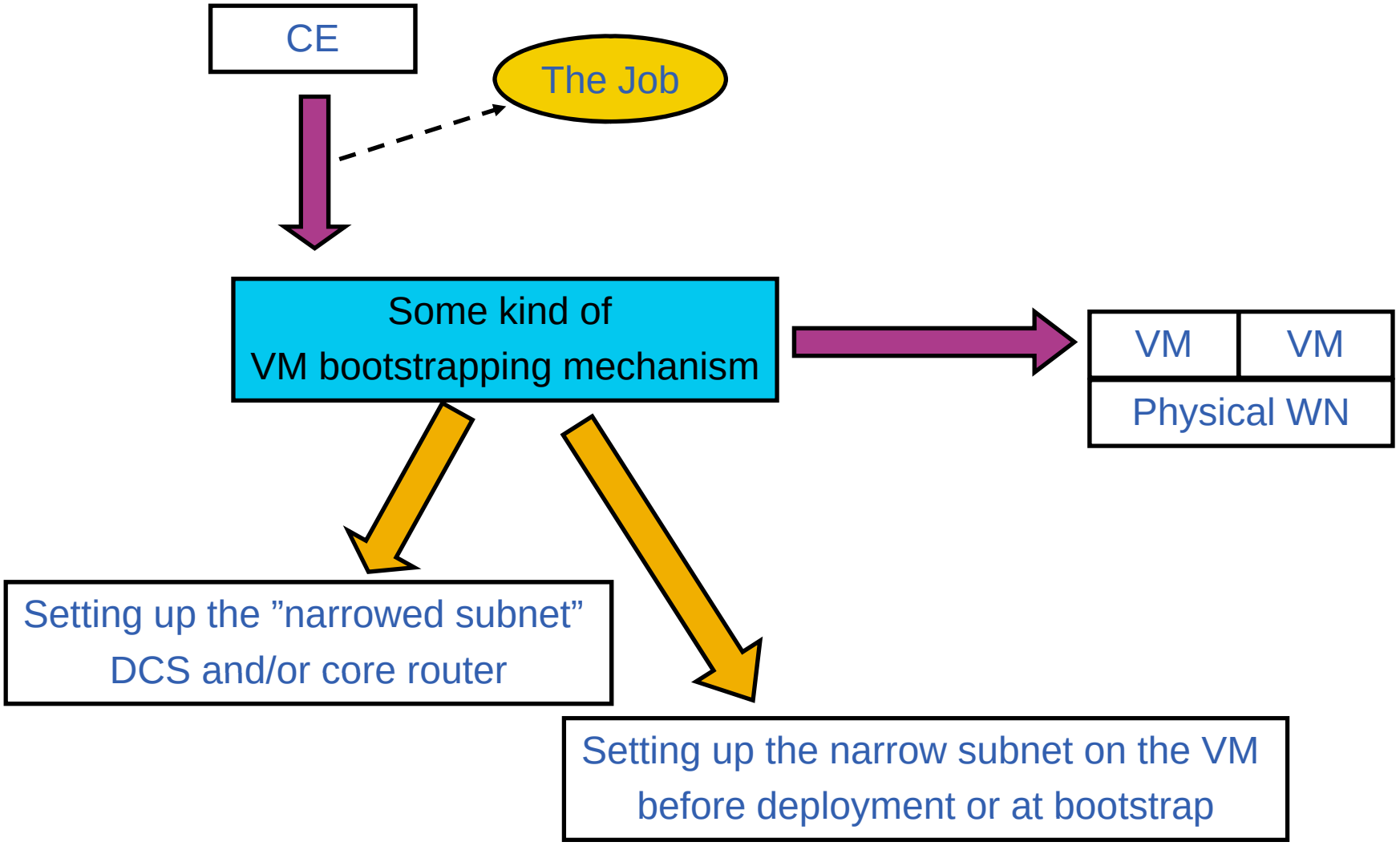
How to Deploy? – with routers #2



Sandboxing the network

- **When using Virtual Machines in your cluster as WNs:**
  - Separate the WNs in two subnets, divide by real and virtual WNs
    - Creates the ability to separate the physical WN and its virtual self(s) on a network basis
      - *Example: use 10.x.x.x for the VMs and 192.168.x.x for the real hardware and never allow a VM to connect with the Physical hardware*
    - Gains the ability to be very flexible within the virtual subnet without disturbing the physical hardware network even though they share the same wire
  - On the fly VLANs (or narrowed subnets) per job (by control of the **Site Admin, not** the **user**)
    - The ultimate sandboxing is to create a network sandbox
      - *Very narrow/small subnet, partitioning the VM subnet*
      - *Use VLANs*
    - If a job needs only one CPU, then setup the network for one VM
    - If a job needs multiple nodes (MPI) then broaden the VLAN to be include the requested amount of VMs (equal to CPU)
- **When using a DCS between the core firewall it can realize firewall rulesets per On-the-Fly subnet instead of per node**

The Virtual Idea...



Past, Present & Future

- **Past (Current pre-prototype implementation Feb 2005):**
  - No AuthN and AuthZ security elements
  - Only portnumber requests
  - Based on iptables
- **Present**
  - Currently no manpower to work on this
- **Future**
  - AuthN & AuthZ
  - Fine & coarse grained connectivity policy description
  - Implementation of all the Virtual Machine ideas concerning the network utilizations