Channel Access

Kay Kasemir

kasemirk@ornl.gov

With material copied from

- Bob Dalesio (LANL)
- Ned Arnold (APS)
- Ken Evans (APS)

Sept. 2014

Channel Access: The EPICS Network Protocol

- Read and write Process Variables over the network.
- To many, CA is EPICS.
  - Especially to users of systems that have no IOC database.
  - "Integrate into EPICS" can mean:
    Talk CA on the network.

What is a Process Variable?

Good question!

"A named piece of data with attributes”

Consider this record:

```
record(calc, "t1:calcExample")
{
        field(DESC, "Sawtooth Ramp")
        field(SCAN, "1 second")
        field(CALC, "(A<10)?(A+1):0")
        field(INPA, "t1:calcExample.VAL")
}
```

What is a PV, given that record?

- "t1:calcExample"
  - PV for the current value of the record.
  - Number 0…10, changes each second.
- "t1:calcExample.DESC"
  - PV for the DESC (description) field of the record.
  - String "Sawtooth Ramp", static.
- "t1:calcExample.VAL"
  - Same as "t1:calcExample".
- “t1:calcExample.SCAN”
  - “1 second”, type enumerated, static.

Pretty much every field of a record can be a PV:

- "{record name}.{field name}"
- ".VAL" is implied when the field is left off.

'caget', 'caput'

'caget' command-line tool:

```
> caget t1:calcExample
t1:calcExample                  6
> caget t1:calcExample.VAL
t1:calcExample.VAL              9
> caget t1:calcExample.DESC
t1:calcExample.DESC             Sawtooth Ramp
```

'caput' allows writing:

```
> caput t1:calcExample.DESC "Howdy"
Old : t1:calcExample.DESC             Sawtooth Ramp
New : t1:calcExample.DESC             Howdy
```

'camonitor'

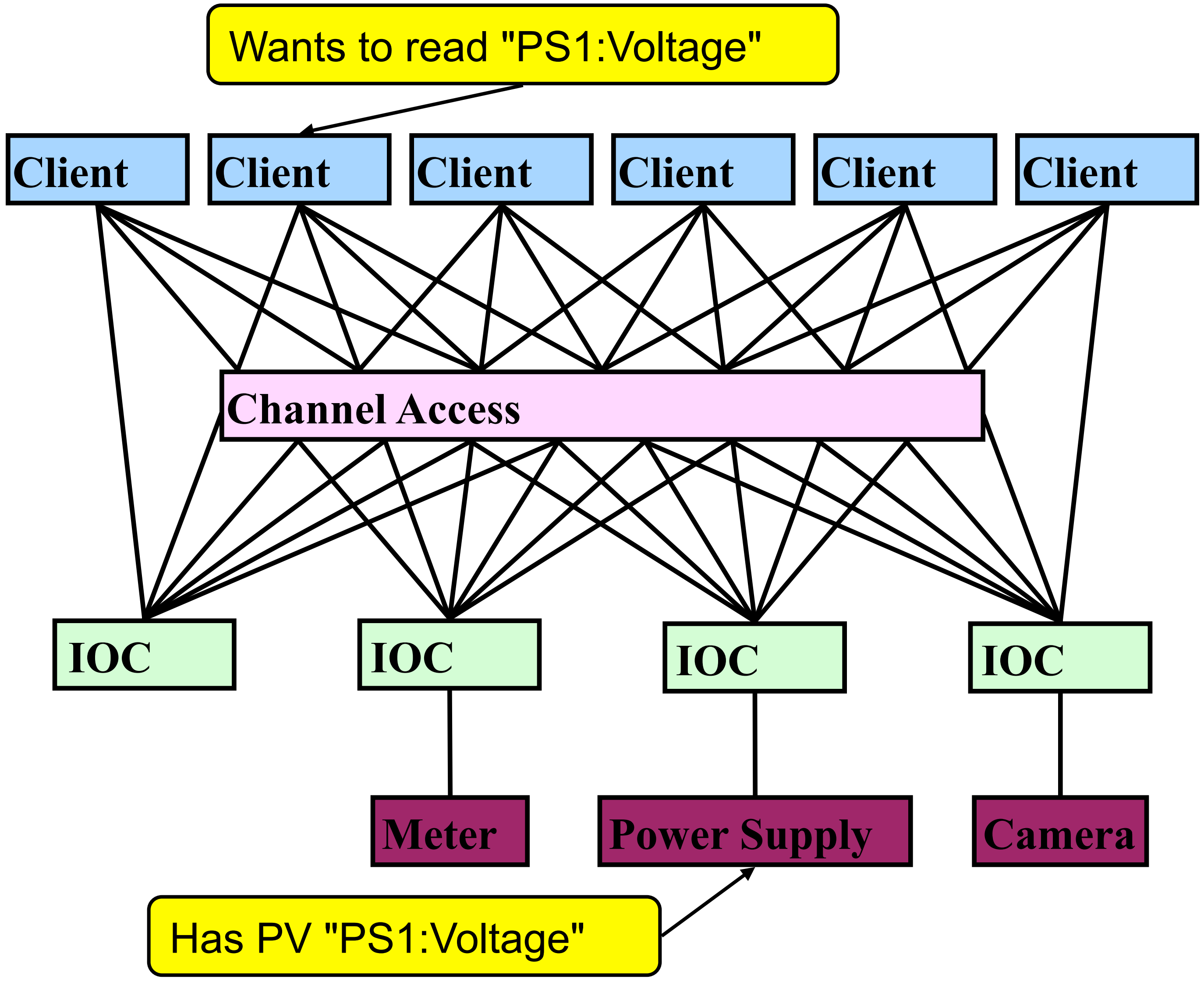
'camonitor' *monitors* value changes:

```
> camonitor t1:calcExample
t1:calcExample                 2006-10-06 13:26:03.332756 6
t1:calcExample                 2006-10-06 13:26:04.332809 7
t1:calcExample                 2006-10-06 13:26:05.332866 8
t1:calcExample                 2006-10-06 13:26:06.332928 9
t1:calcExample                 2006-10-06 13:26:07.332981 10
t1:calcExample                 2006-10-06 13:26:08.333034 0
t1:calcExample                 2006-10-06 13:26:09.333097 1
t1:calcExample                 2006-10-06 13:26:10.333143 2
… plus one more each second…
… press Ctrl-C to stop …


> camonitor t1:calcExample.DESC
t1:calcExample.DESC            2006-10-06 13:29:12.442257 Howdy
… and then nothing …
```

- Idea called *publish* and *subscribe*.

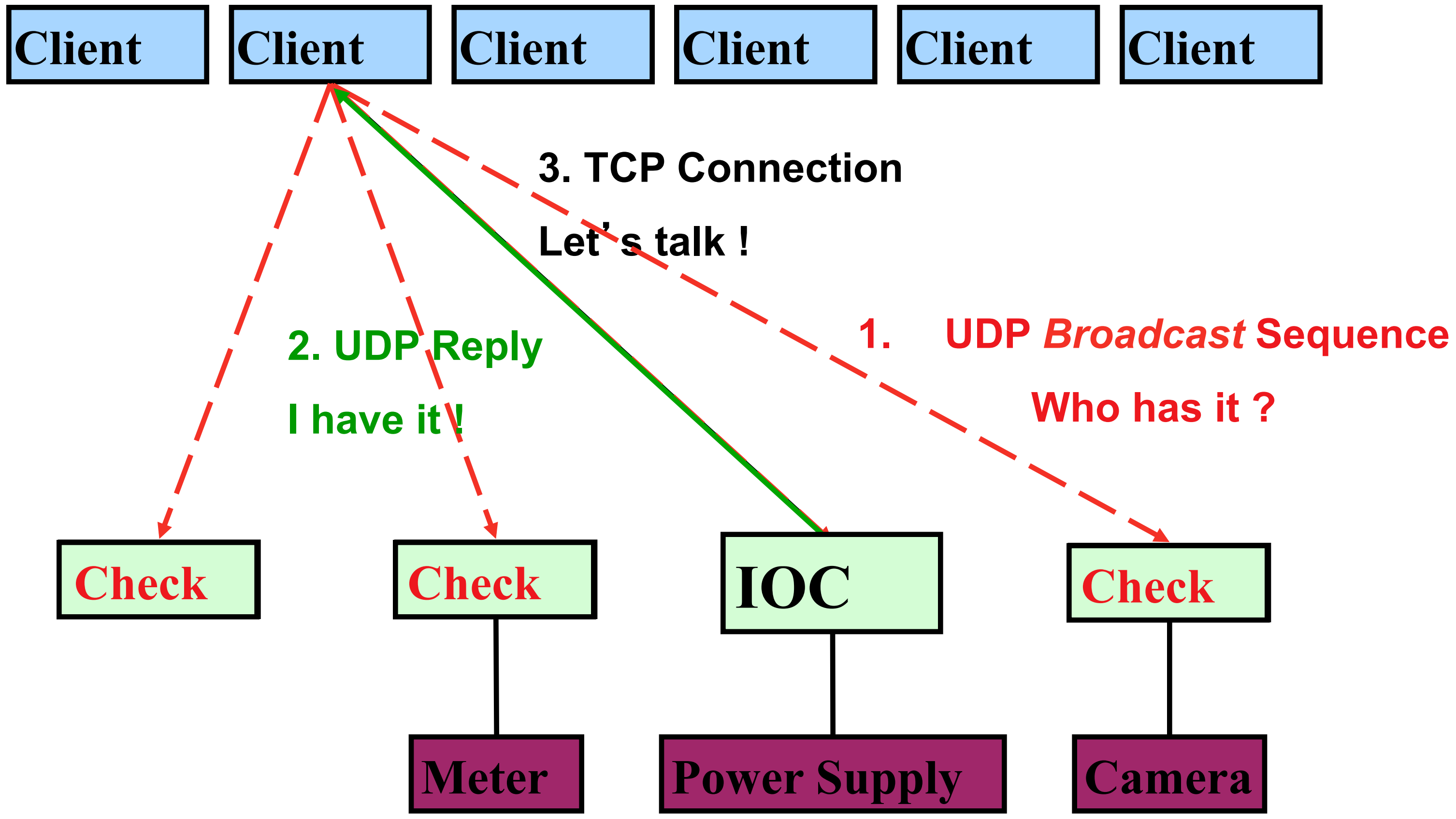
How Clients find Channels



Internet 101

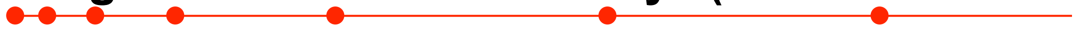
- The Internet Protocol (IP) consists of UDP and TCP
  - We ignore the very low-level Internet Control & Message Protocol (ICMP).
- User Datagram Protocol (UDP)
  - Sends a network packet
    - from one port on one computer
    - to one or more ports on one or more other computers.
  - Fast!
  - Checksum: If the packet arrives, it's OK.
  - But not reliable: Packets get lost, arrive out-of-order, arrive more than once.
- Transmission Control Protocol (TCP)
  - Sends a stream of bytes from one port on one computer to another port on another computer.
  - Reliable: Bytes arrive at the receiver in the correct order.
    - Basically adds serial numbers to UDP packets, requesting repeats for missing packages.
  - But slower, and message boundaries get lost:
    - "Hello Fred!" might arrive as "Hel" <pause> "lo F" <pause> "red!"

Search and Connect Procedure



Search Request

- A search request consists of a sequence of UDP packets
  - Per default: Broadcast to the local subnet.
    - Basically plug-and-play when you get started.
  - Or to IP addresses listed in EPICS_CA_ADDR_LIST
    - Routers do not forward broadcasts!
    - You have to add 'other' subnets or specific IOCs off the local subnet to that environment variable!
  - Starts with a small interval (30 ms)
    - Doubles each time, until reaching 5 second intervals.
    - Stops after 100 packets (~8 minutes) or when it gets a response
    - Wakes again on "beacon anomaly" (details follow later)

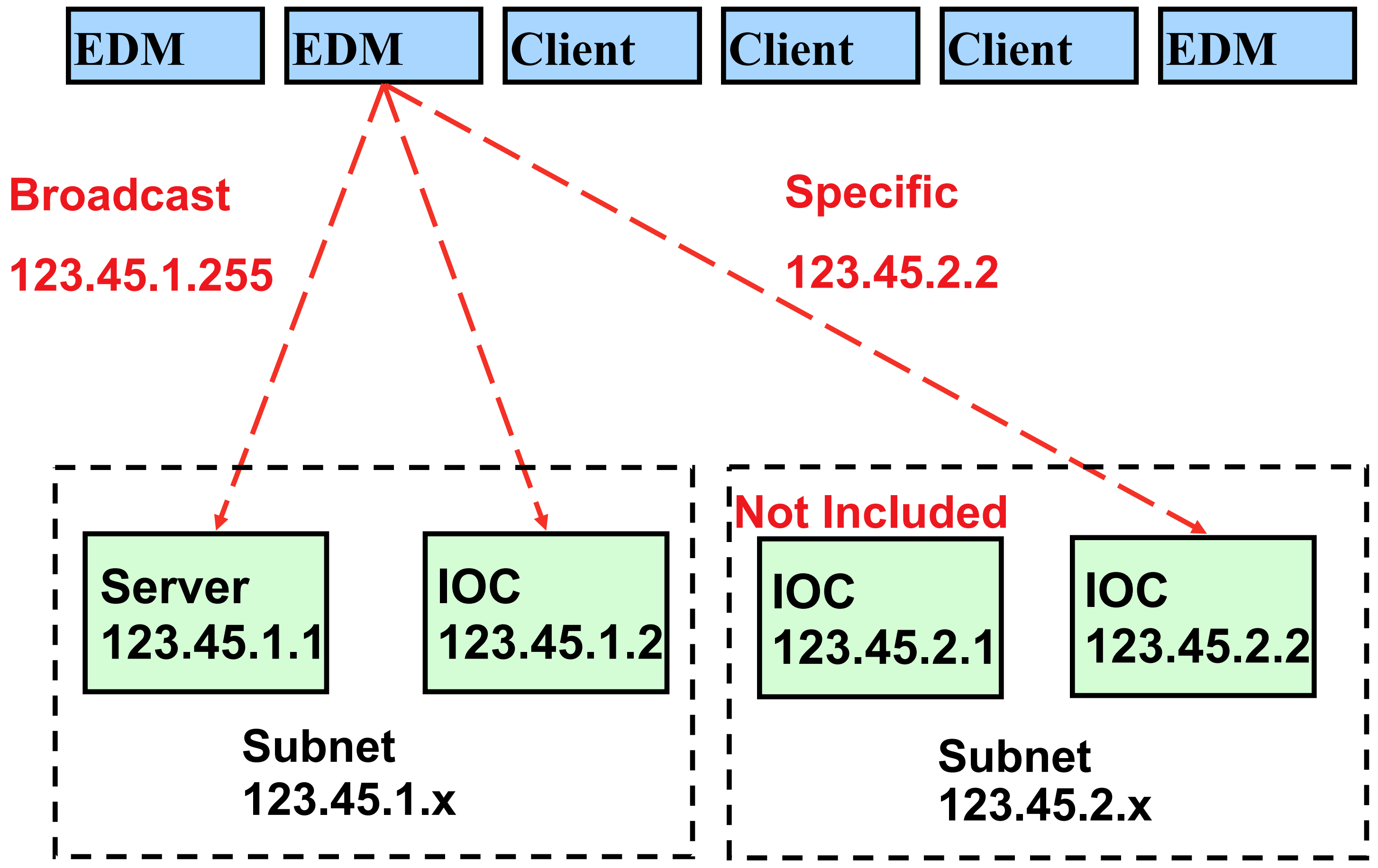


- CA Servers check each search packet
- Usually connects on the first packet or the first few
  - But non-existent PVs cause a lot of traffic
  - Try to eliminate them

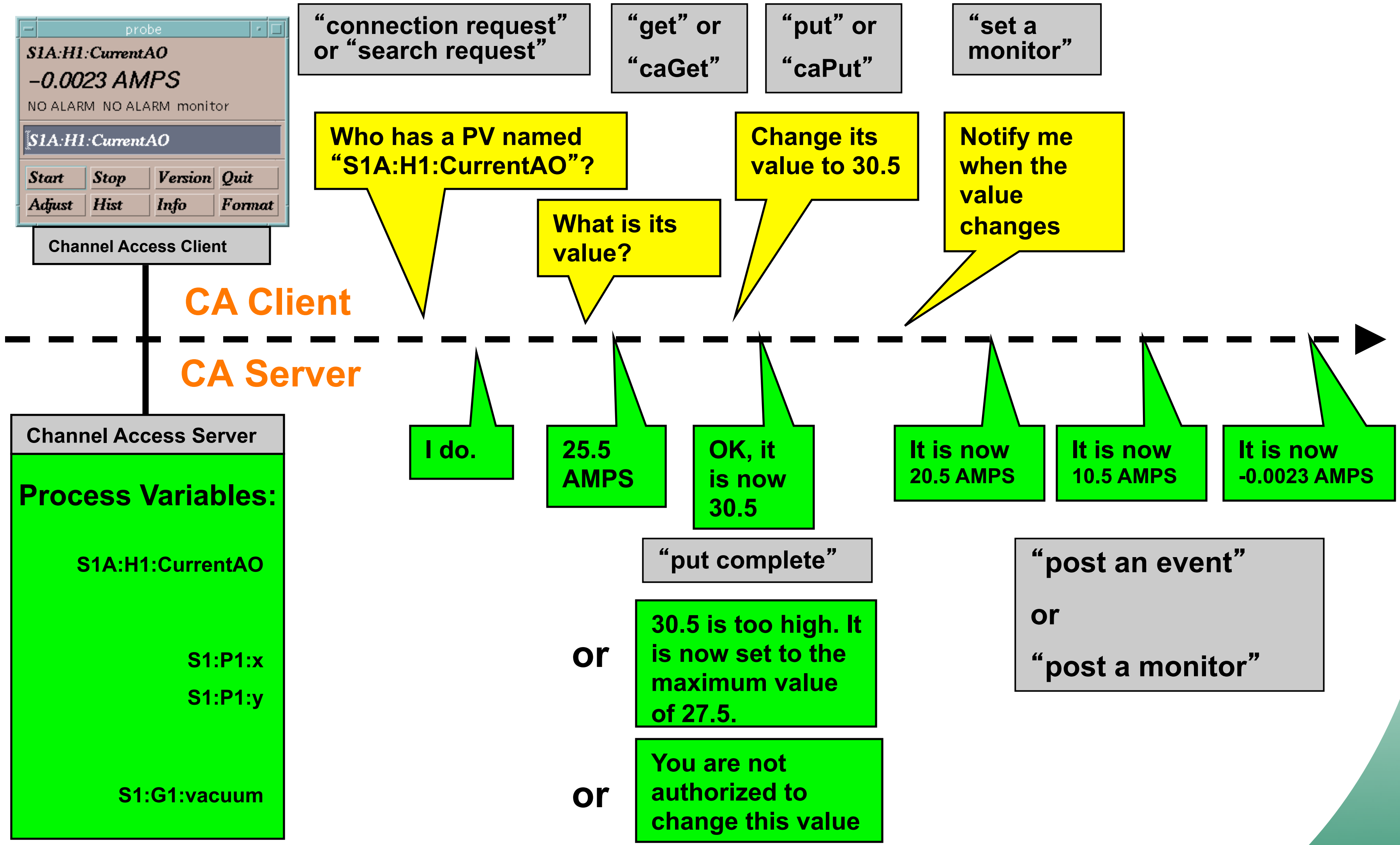
Important Environment Variables

- EPICS_CA_ADDR_LIST
  - Determines where to search
  - Is a list (separated by spaces)
    - "123.45.1.255 123.45.2.14 123.45.2.108"
  - Default is broadcast addresses of all interfaces on the host
    - Works when servers are on same subnet as Clients
  - Broadcast address
    - Goes to all servers on a subnet
    - Example: 123.45.1.255
    - Use ifconfig –a on UNIX to find it (or ask an administrator)
- EPICS_CA_AUTO_ADDR_LIST
  - YES: Include default addresses above in searches
  - NO: Do not search on default addresses
  - If you set EPICS_CA_ADDR_LIST, usually set this to NO

EPICS_CA_ADDR_LIST



Channel Access in One Slide



Multiple IOCs on same Host

- First IOC:
  - UDP port 5064, listening to searches
  - TCP port 5064 to serve data
  - Receives search? “Talk to me on 5064!”
- Second IOC:
  - UDP port 5064, listening to searches
  - TCP port {whatever} to serve data
  - Receives search? “Talk to me on {whatever} !”

So far, so good…

Multiple Subnets

- IOCs on IP 1.1.1.1, subnet 1.1.1.0
  - UDP 5064, TCP 5064
  - UDP 5064, TCP ???
- Try to reach from other subnet
  - EPICS_CA_ADDR_LIST=1.1.1.1
    - Won't work!
    - Quirk in network kernels:
      Only the IOC started LAST will get anything on UDP 5064
  - EPICS_CA_ADDR_LIST=1.1.1.255
    - OK. When using broadcast into subnet, all IOCs on UDP 5064 will see search requests.

Firewall?!

- IOCs on IP 1.1.1.1, subnet 1.1.1.0
  - UDP 5064, TCP 5064
  - UDP 5064, TCP ???
  - EPICS_CA_ADDR_LIST=1.1.1.255
- Firewall cannot open unpredictable TCP ???
- Likely to block broadcasts

- Need to run CA Gateway:
  - Firewall allows access to CAGateway
  - CAGatway uses broadcast inside subnet

Handling of Network Interruptions

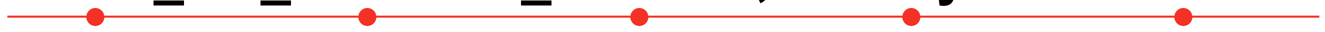
- No Network is up 100%, so CA was designed to handle this:
  - TCP connection closed by server?
    - Notify client code about problem
      - Operator displays tend to indicate this.
    - Client sends new search requests.
  - No data nor beacon from server for 30 sec.?
    - Client sends “Are you there?” query
    - If no response for 5 sec, also notify client code, but TCP connection is kept open to avoid network storms.
    - If server eventually sends data: OK. Otherwise we're waiting until the OS cuts the TCP connection (~hours).

Beacons

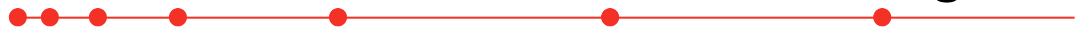
- Assume all is fine, we are connected, but the data simply doesn't change.
  - How do we know the server is still OK?
- Assume we searched for a PV, didn't get any response for ~8 minutes.
  - How do we learn about a new CA server starting up which might have the missing PV? What triggers renewed search requests?

Beacons

- UDP broadcast packet sent by a CA Server
- When it is healthy, each Server broadcasts a UDP beacon at regular intervals (like a heartbeat)
  - EPICS_CA_BEACON_PERIOD, 15 s by default



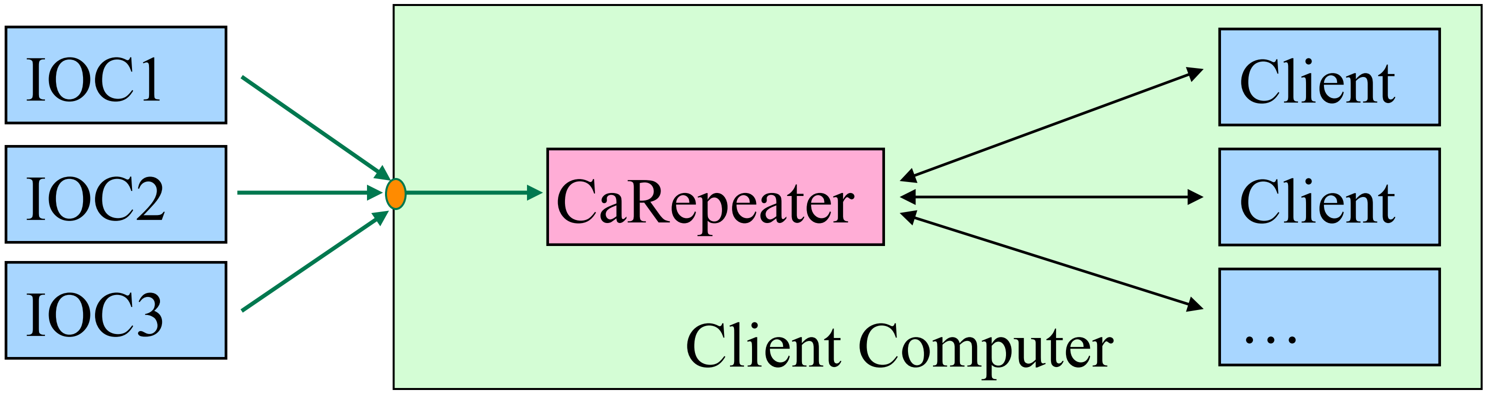
- When it is coming up, each Server broadcasts a startup sequence of UDP beacons
  - Starts with a small interval (25 ms, 75 ms for VxWorks)
  - Interval doubles each time
  - Until it gets larger than 15 s, then it stays at 15 s
    - Takes about 10 beacons and 40 s to get to steady state



- Clients monitor the beacons
  - Receive beacons: Server is OK.
  - Receive new beacons at changing intervals: Beacon anomaly, new CA server, re-issue searches.

caRepeater?

- Older OSs didn't allow multiple programs to listen to the same UDP port
  - They didn't see the beacons (UDP broadcasts)!
- caRepeater solves this problem
  - There is one caRepeater process per workstation
  - Clients make a TCP connection to it when they start up
  - caRepeater receives the beacons
    - EPICS_CA_REPEATER_PORT[usually 5065]
  - .. and forwards them to clients.



Issues

- **CA Client does not connect**
  - Check basic network connectivity.
    - Can server and client machines 'ping' each other?
    - Check EPICS_CA_ADDR_LIST if server is on different subnet.
- **CA Client does not re-connect after network issue or IOC reboot**
  - Use casw, wireshark: Does the client computer receive the (anomal) beacons of the rebooting IOC?
  - Check EPICS_CAS_BEACON_ADDR_LIST,
  - since routers will not forward beacons across subnets.
  - Check if 'caRepeater' is running on the client.

What is a PV (Channel)?

- Whenever there's a CA server out there which decides to respond to a search request, that's a PV!
- iocCore responds to "{record}.{field}" searches if
  - the {record} is for a record on this IOC,
  - and the {field} is an accessible field of that record,
  - or it's the pseudo-field "RTYP" (record type).
- So (almost) every field of every record is a PV.
- But you can implement your own CA server based on the CAS library (for C++), and then you decide when to respond!
  - This used to be important when iocCore only ran on vxWorks, so the 'portable' CAS lib was used to serve data from Linux, Win32, VMS, …
    Now it's probably better to implement device support and run iocCore.

OAK RIDGE National Laboratory

Channel Properties

- Each channel comes with properties:
  - Value
    - of type string or double or int or …
    - Scalar or array
  - Time stamp
    - Up to nanosecond precision
  - Severity code
    - OK, MINOR, MAJOR, or INVALID
  - Status code to qualify the severity
    - OK, READ error, WRITE error, at HIGH limit, …
  - units, suggested display range, control limits, alarm limits.

Client interface to properties

- The available properties are fixed.
  - One cannot add a new 'color' property.
- The request types are fixed.
  - "DBR_…" types.
  - Available:
    - Just value.
    - Value with status and severity.
    - Value with status, severity and time stamp.
    - "Everything:" value, units, time, status, limits, …
  - Not available:
    - Custom combinations like value with units.

Records & Fields vs. Channels & Properties

- A CA client asks for the properties of a channel.
- The implementer of the CA server decides how to answer.
- The iocCore implementation maps the fields of a record to the properties of a channel.
  - Details are in the source code for the respective record type. Not always predictable or meaningful!

Example: AI record "fred"

- PV "fred" or "fred.VAL"
  - value property of channel = VAL field of record.
    - Type double, one element (scalar).

– time property	= TIME field
– status	= STAT
– Severity	= SEVR
– units	= EGU
– Precision	= PREC
– display limit low, high	= LOPR, HOPR
– control limit low, high	= LOPR, HOPR
– alarm limits	= LOLO, LOW, HIGH, HIHI

- Makes a lot of sense.
  - GUI can display the value together with units, formatted according to the precision, as e.g. "12.37 volts”.

Example: AI record "fred"

- PV "fred.SCAN"
  - value property of channel = SCAN field of record.
    - Type enumerated, values: "Passive", "1 second", ...
  - time property = TIME field
  - status = STAT
  - Severity = SEVR
  - units = EGU
  - Precision = 0
  - display limit low, high = 0, ??
  - control limit low, high = 0, ??
  - alarm limits = 0, 0, 0, 0

- Makes some sense, but
  - Units don't really apply to the SCAN field.
  - It's value range is really limited by the available SCAN choices, not 0..??.

When will ‘camonitor’ receive new value?

- When the CA server (IOC) sends a new value!
  - Analog records: VAL change >= MDEL
  - Binary records: Every change

- Assuming Client uses ‘DBE_VALUE’ subscription
  - DBE_LOG
    - Meant for archive systems. Analog record change >= ADEL
  - DBE_ALARM
    - Meant for alarm systems

Database Channel Access Link Flags

- CA: Force CA link, even though target in same IOC
- CP: For INP link, process on received CA monitor
- CPP: CP, but only if SCAN=Passive

Allows for “process record if inputs change”

Points to remember

- In 99% of the cases, CA "just works"
  - If not, check EPICS_CA_ADDR_LIST
  - If that's not it, there could be a subnet/router issue with UDP search broadcasts and beacons.
- Channel/property and Record/field are different things!
  - This decouples the CA clients from the IOC database and its record types, allowing EPICS collaborators to share CA client tools for vastly different records and databases.
  - But also means that CA clients have no idea about records nor fields.
    - Client can't know that there might be a "readback" AI that goes with a "setpoint" AO record.
    - The archiver stores channels and their properties, not a whole AI or motor record.
    - Important properties for dealing with waveform data is definitely missing (sample rate, type of data).