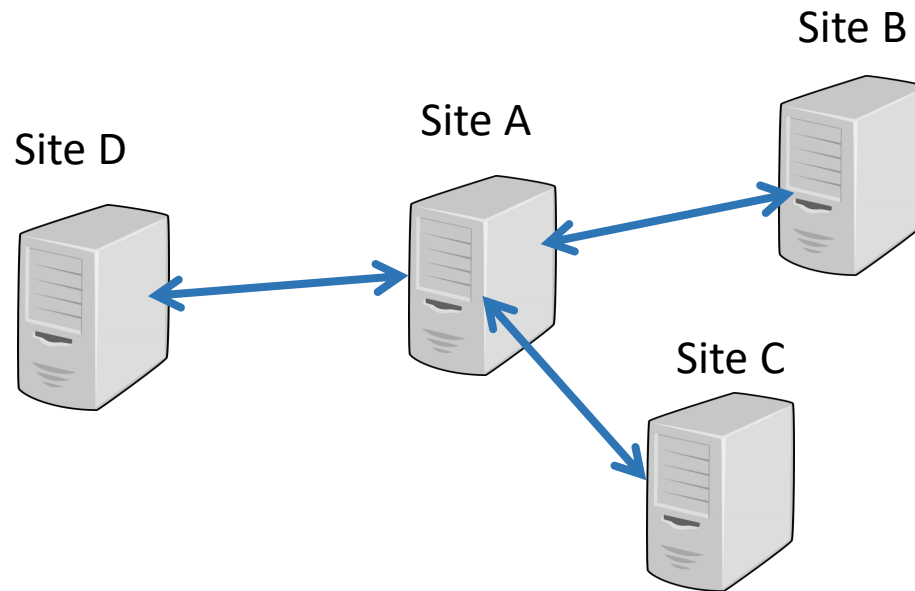


Samba KCC: Saying No to Full Mesh Replication

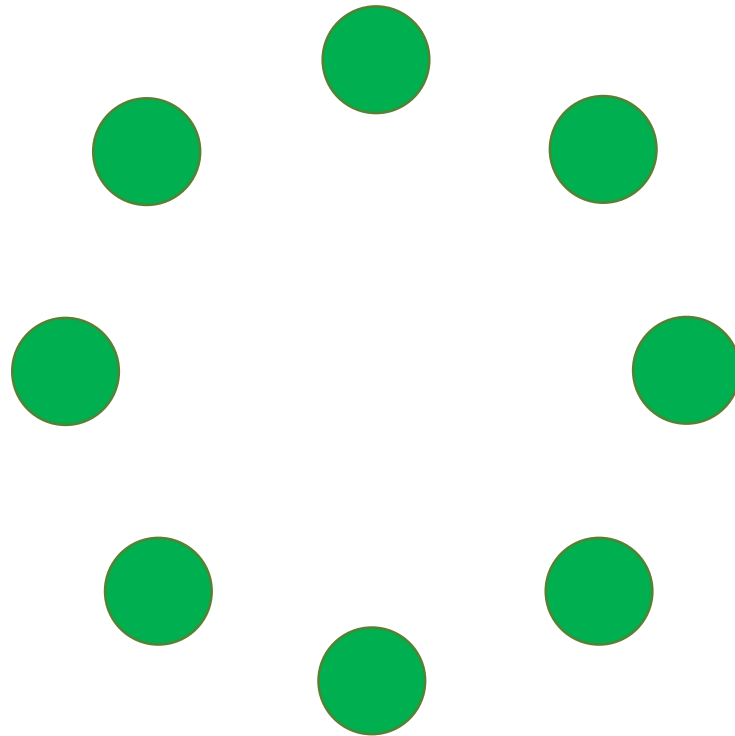
Garmin Sam
Catalyst IT, Samba Team

What is the KCC?

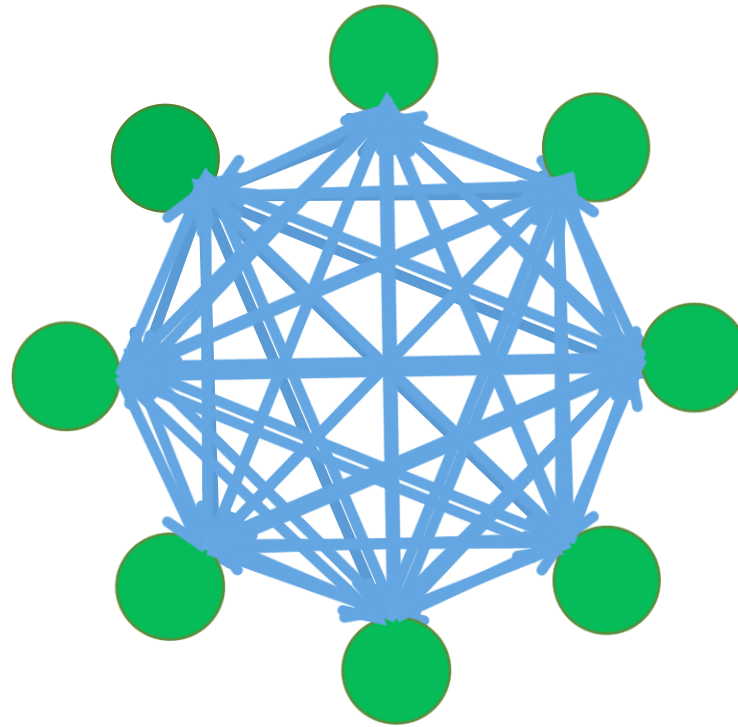
- Knowledge consistency checker
- Used to manage replication connections in AD
- Set of algorithms to produce efficient network topologies



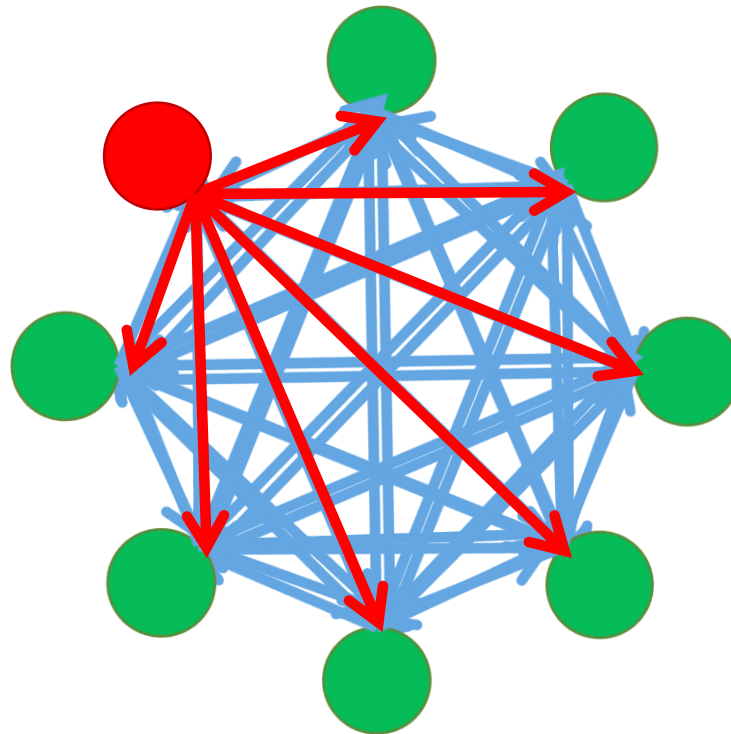
What is the KCC?



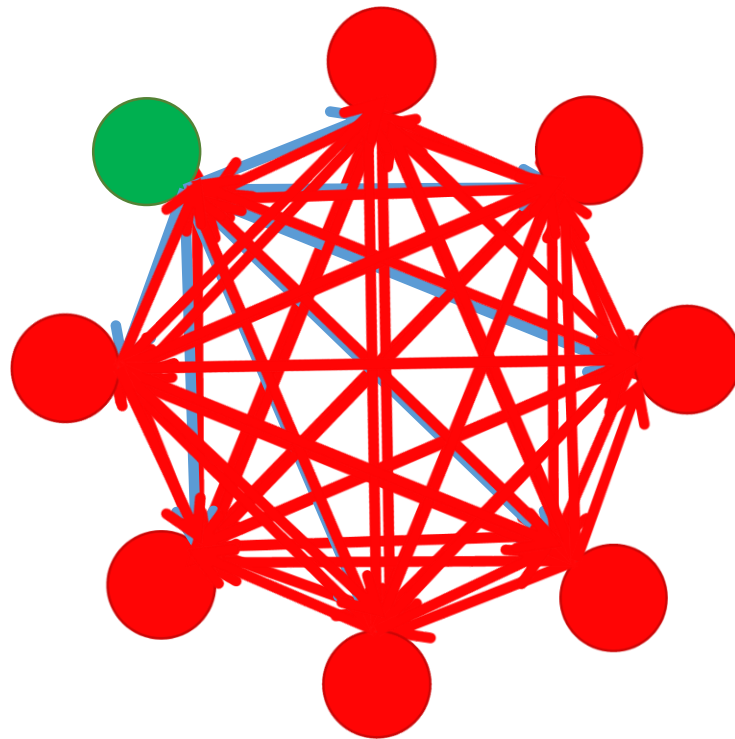
What is the KCC?



What is the KCC?



What is the KCC?

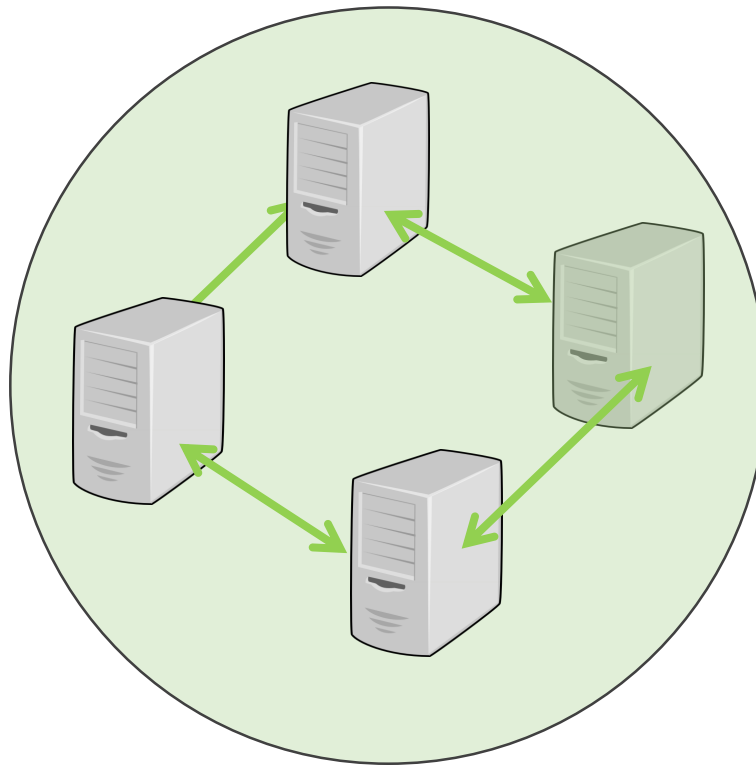


History of the KCC

- Original full-mesh C code
- Attempt at MS-ADTS algorithms in C
- Dave Craft (2011) in Python inter-site algorithms
- Late 2014—Early 2015 Douglas and myself
- Samba 4.3 introduced, Samba 4.5 set as default

Intra-site algorithm

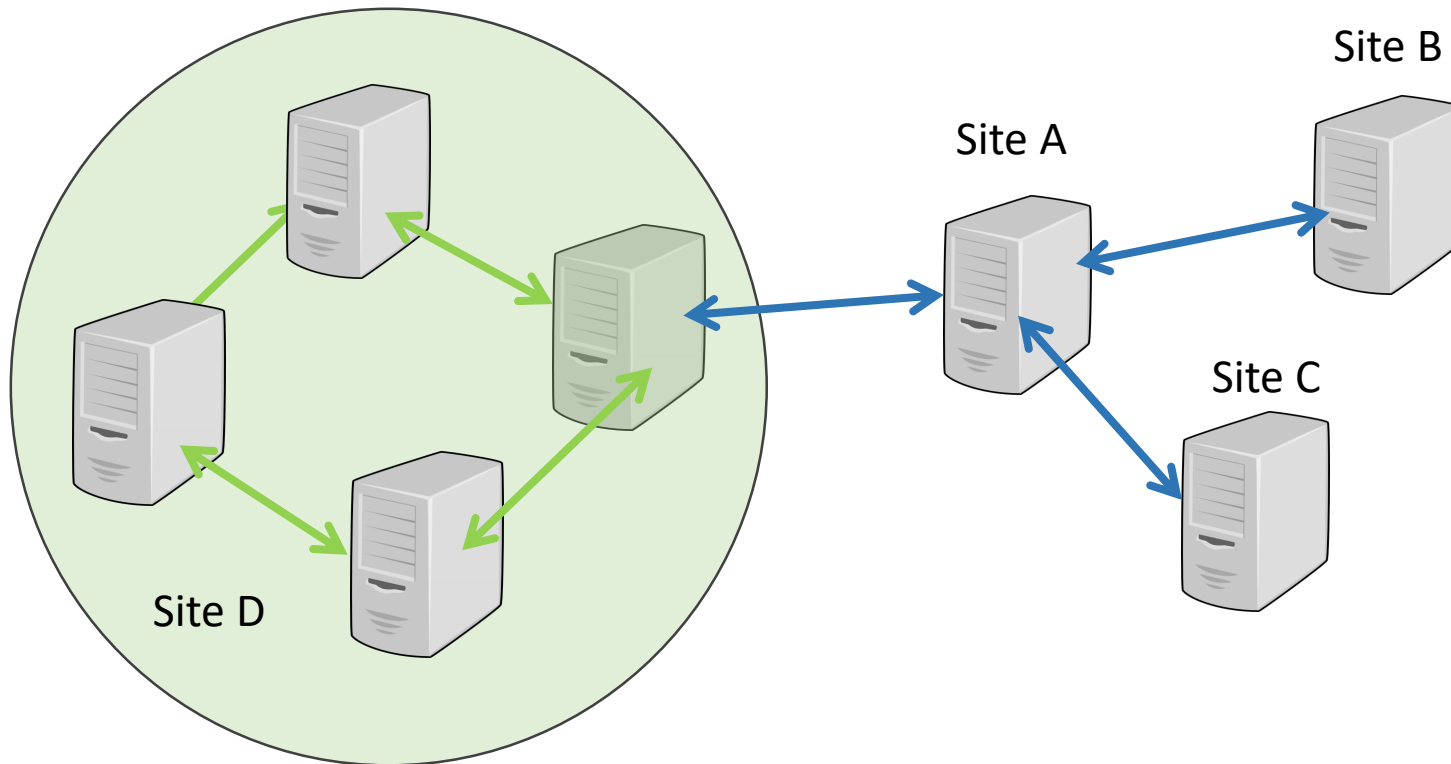
- Ring topology, with a few extra connections



Inter-site algorithm

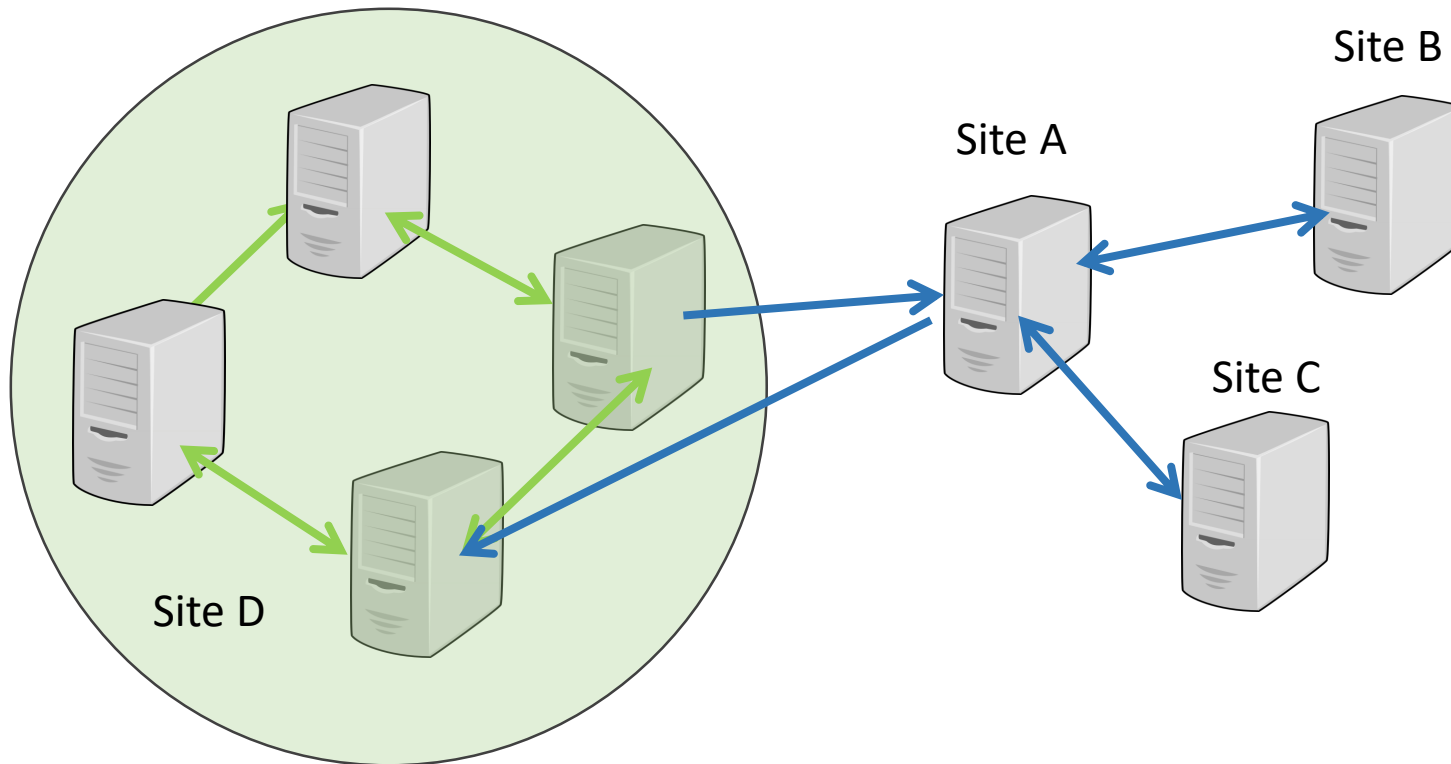
- Stable answer across entire DC network

Inter-site algorithm



Bridgehead servers are the end-point connections between sites.

Inter-site algorithm



There is not necessarily a single bridgehead server.

Remove unneeded connections

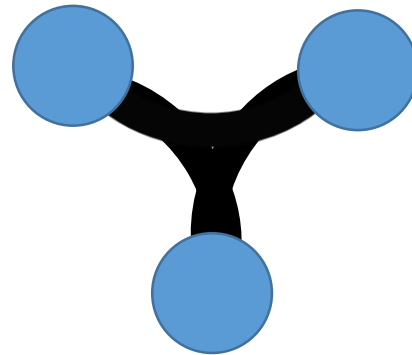
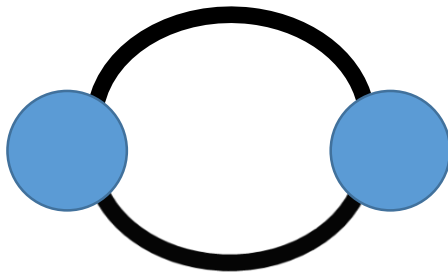
- Stable answer across entire DC network
- Used to manage replication connections in AD
- Set of algorithms to calculate efficient network topologies

Translate connections

- Stable answer across entire DC network
- Used to manage replication connections in AD
- Set of algorithms to calculate efficient network topologies

Challenges

- Stable answer across entire DC network
- Used to manage replication connections in AD
- Set of algorithms to calculate efficient network topologies



History of the Samba KCC

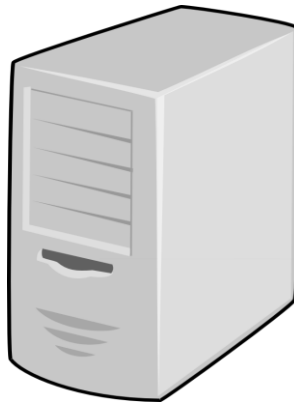
- Easy to debug your own bugs, not someone else
- Dot graph
- Avert some of these bounces outside of KCC

•With Samba 4.5, the new site-aware Samba Knowledge Consistency Checker (KCC) has been turned on by default. Instead of using full mesh replication between every DC, the KCC will set up connections to optimize replication latency and cost (using site links to calculate the routes). Although there is more effort required in establishing effective site topologies, it has enabled users to create larger and more distributed networks without any of the previous replication penalties. It has also meant that Samba AD can be aware of particular details of a network (such as satellite links or certain firewall restrictions) to ensure that information flows through the network in a reasonable way. The aim is to look at how sites in AD generally work, what role the KCC

.Hyperedge and hypergraphs



•Internal consistencies, it's hard to say what actually needs to be corrected.



```
.2637      # Step 1
.2638      self.refresh_failed_links_connections(ping)
.2640      # Step 2
.2641      self.intrasite()
.2643      # Step 3
.2644      all_connected = self.intersite(ping) # overlay a failed
network ontop of the expected
.2646      # Step 4
.2647      self.remove_unneeded_ntdsconn(all_connected)
```

.i1929 The KCC does not create more than 50 edges directed to a
.1930 single DC. To optimize replication, we compute that each
node
.1931 should have $n+2$ total edges directed to it such that (n) is
.1932 the smallest non-negative integer satisfying
.1933 $(\text{node_count} \leq 2*(n*n) + 6*n + 7)$

.Select ITSG

- .1140 An intersite graph has a Vertex for each site object, a
- .1141 MultiEdge for each SiteLink object, and a MutliEdgeSet for
- .1142 each siteLinkBridge object (or implied siteLinkBridge). It
- .1143 reflects the intersite topology in a slightly more abstract
- .1144 graph form.

.White, red black color vertex