DESIGN · MULTICAST · SSM

# One source, not any source: why the fabric runs source-specific multicast everywhere

A multicast join can ask one of two questions: "anyone sending to this group" or "this source, sending to this group". The first is any-source multicast, which needs a rendezvous point to find the senders and a second protocol to share them between networks, and which lets anyone send to a group. The second is source-specific multicast, in which a receiver names the one source it trusts and the tree forms from the receiver back toward that source, with no rendezvous point, no shared tree and no discovery in the routing plane. This paper draws both, explains why the IETF deprecated the first between networks, shows how a receiver on the fabric learns which source to name, lists what a router must support, and sets out why a source that is a routed address inherits the routing system's own security. It closes with the one place any-source still fits.

**Author:** Jeff Harris · Lightweb Inc. · jeff@lightweb.net

**Paper** 014 · **Version** 01

Audience: network engineers who ran PIM once and remember rendezvous points, and the engineer on the peer side of *Multicast over a unicast Internet*.

**We** = Lightweb Inc., author of this architecture and operator of the first fabric built on it; nothing here depends on that operator. **You** = an engineer deciding what the fabric asks of your routers.

Every figure uses the same colors: gray for a third party's network or a rendezvous point, green for the fabric's routers and trees, purple for receivers. A solid green line is a tree; a red line is traffic nobody asked for; a teal line is a check.

## The short version

- **A join names a source, or it does not.** Any-source asks for whoever sends to a group; source-specific asks for one named source. Everything else follows from that choice.
- **Any-source needs a meeting point.** Senders register with a rendezvous point, receivers join a shared tree rooted there, and routers later switch to a tree per source. Between networks a further protocol floods the list of active senders to every rendezvous point.
- **Anyone can send to an any-source group.** The network cannot tell a publisher from a flooder until the packets arrive. That is why the IETF deprecated any-source multicast between domains.
- **Source-specific has no rendezvous point.** The tree grows from each receiver back toward the one source it named, along the reverse path the unicast routing table already holds. There is nothing to register with and nothing to flood.
- **The receiver learns the source from a published announcement.** The shard manifest carries the sources with the group layout, and a handful of control groups are reached by name. Nothing in the routing plane discovers anything.
- **A source is a routed address, so routing security covers it.** A Route Origin Authorization says which network may originate it, and every peer can reject a forged origin before a tree is pulled toward it.
- **Any-source still fits one place.** A single site, one operator, one switch: the lab. The frames and the repair protocol are identical in both modes, so a lab built on one migrates to the other with no format change.

## 1. Two questions a join can ask

A receiver that wants multicast traffic tells its nearest router which traffic, and there are two ways to say it. It can name a group alone, which means "deliver whatever anyone sends to this group", or it can name a source and a group, which means "deliver what this one source sends to this group". The first is any-source multicast, ASM, and the join is written (*,G). The second is source-specific multicast, SSM, and the join is written (S,G). The difference looks small at the receiver. It decides the whole shape of the network behind it.




*Figure 1 - the two joins. On the left the network has to discover who is sending; on the right the receiver told it.*

## 2. What any-source needs

If a receiver does not name the source, the routers have to find it, and the way they find it is a rendezvous point. Every sender for a group registers with the rendezvous point, which is a router chosen by configuration; every receiver's join travels toward the same router; and the first copies of the traffic flow down a shared tree rooted there. Once a receiver's router sees traffic from a particular source it may switch to a shorter tree rooted at that source, which means the routers hold state for the shared tree and for every source tree at once. Between networks, each network's rendezvous point has to learn the senders known to every other network's rendezvous point, and a separate protocol, MSDP, floods that list of active sources to all of them.

Stated at its strongest, this works, and it has carried an operator's own television for twenty years. Inside one network, with sources the operator controls and nobody hostile able to send, a rendezvous point is a reasonable thing to run, and the shared tree is a reasonable way to start delivery before any source tree exists.

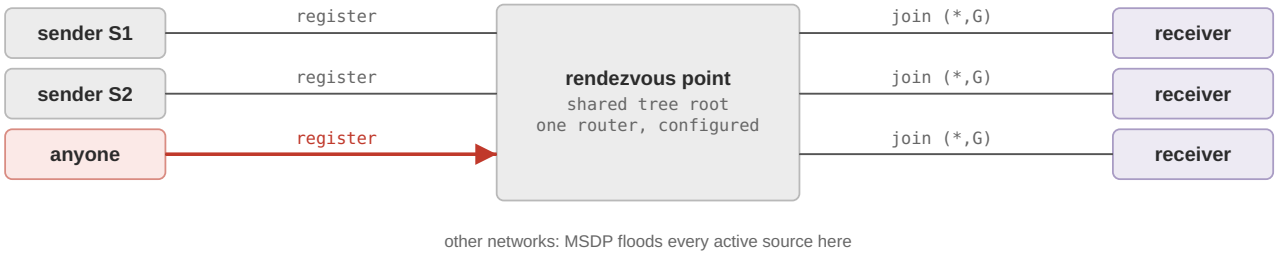



*Figure 2 - any-source multicast. Senders register with a rendezvous point, receivers join a shared tree rooted there, and between networks the list of senders is flooded to every rendezvous point.*

## 3. Why it does not cross domains

The strength of any-source is also its exposure. Anyone who can reach a group's rendezvous point can register as a sender, and the traffic flows down the shared tree to every receiver before anything can judge it. The rendezvous point is a single, well-known router that every sender must reach and every join must pass, which makes it the obvious target of a flood. And MSDP, the protocol that shares active sources between networks, floods state about every sender on every group to every peer; a burst of new senders, whether a legitimate event or a worm scanning multicast addresses, becomes a burst of state in every rendezvous point on the Internet. The IETF recorded all of this and in 2020 published a best current practice deprecating any-source multicast for use between domains, with source-specific multicast as the replacement.

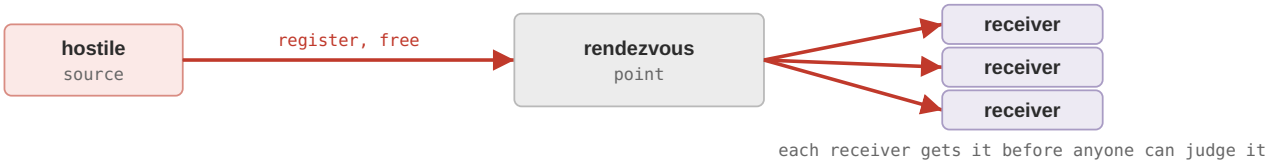



*Figure 3 - a hostile source on a shared tree. Registration is open, so unwanted traffic reaches every receiver on the tree before any filter can act.*

## 4. Source-specific, drawn

A receiver that names its source removes every piece of that machinery. Its join carries the source address, and the router nearest it forwards the join one hop toward that source, which it finds by looking the source up in its ordinary unicast routing table and taking the reverse path. The next router does the same, and so on, until the join reaches a router already on a tree for that source and group, or the source itself. The tree is built from the leaves toward the root, exists only where a join asked for it, and ends where interest ends. There is no rendezvous point, no shared tree, no register message, no switch from one tree to another, and no protocol to flood sources between networks, because every router learned the source from the join.

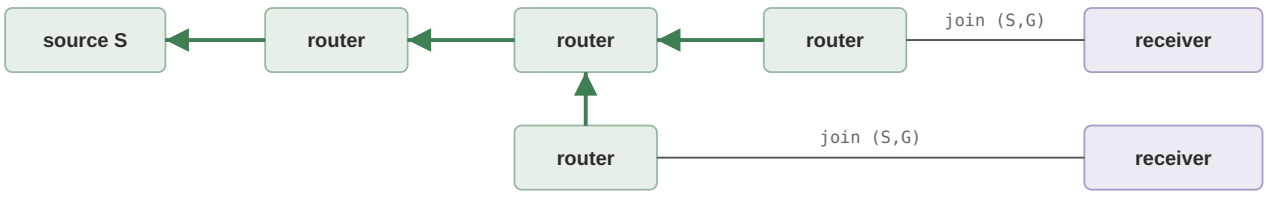



*Figure 4 - a source-specific tree. Each join is forwarded one hop toward the named source along the unicast reverse path, and the tree grows from the receivers back to the source.*

The address range makes the mode unmistakable. Source-specific groups live in a reserved range of the IPv6 multicast space, so a router that sees a group in it knows a join must name a source, and a join without one is simply invalid. The fabric's groups are allocated in that range, as the published address plan sets out.

## 5. How a receiver knows the source

Source-specific multicast moves one question out of the routing plane: how does a receiver know which source to name? The answer has to come from somewhere, and on the fabric it comes from a published announcement. The shard manifest (BRC-139) is a small, periodically announced record that carries the current group layout, how many groups there are and how a transaction maps to one, together with the sources that feed them. A receiver that has the manifest has everything a join needs. The manifest itself, and a few other control groups, are reached through bootstrap sources the receiver is configured with by name, resolved when it starts and refreshed as it runs. That is the whole of discovery: one record, a few names, and no protocol in the routers.

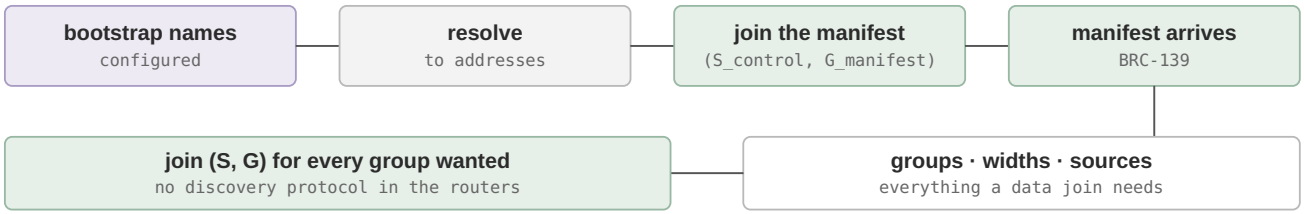



*Figure 5 - discovery. A receiver resolves a few bootstrap names, joins the manifest group from them, and reads the sources it then names in every data join.*

What the manifest does not say is as important. It carries the sources the operator publishes and nothing about where they sit or how they are arranged; a source is an address, and a receiver needs no more than that. Operators arrange their sources as they see fit, and a receiver's joins do not change when they do.

## 6. What your routers need

The requirements are standard features of a router operating system, and all of them are already present on a router that runs any multicast today.

| Requirement | Standard | Where it runs |
|---|---|---|
| Source-filtered joins from receivers | RFC 3810 (MLDv2) | the router nearest each receiver |
| Source-specific tree building | RFC 7761 in SSM mode, RFC 4607 | every router on the path |
| A route to each source for the reverse path | the unicast table, or RFC 4760 multicast address family between domains | every router on the path |
| The source-specific address range | RFC 4607, BRC-129 | configuration |
| No rendezvous point, no MSDP | | nothing to run |

*1. The last row is the point: source-specific removes configuration rather than adding it.*

Between networks the reverse path needs a route to the fabric's sources, and the practice is to carry those routes in the multicast address family of multiprotocol BGP, so that a peer takes the few source prefixes it needs for reverse-path lookup and no general unicast routing with them. A router that resolves reverse paths from its unicast table alone can take the same few prefixes there instead. Either way the exchange is a handful of aggregates, as *Multicast over a unicast Internet* sets out for the peering layers.

## 7. Why it is more securable

A source on the fabric is an address inside a prefix the operator originates in BGP. That makes it a route, and routes have a security system. A Route Origin Authorization in the RPKI states which autonomous system may originate the prefix, and prefix origin validation lets every peer reject an announcement from any other, before a single join can be forwarded toward it. A forged source would need a forged route, and the forged route fails validation at the first peer that checks. The unicast default-free zone carries on the order of a million prefixes and securing all of them is a long project; a fabric's sources are tens of aggregates, every one can carry an authorization, and every peer can validate every one. *Sharding the object plane* sets this in the wider context; here the point is that none of it is available to any-source multicast, whose senders are not routes and whose rendezvous point is reachable by anyone.

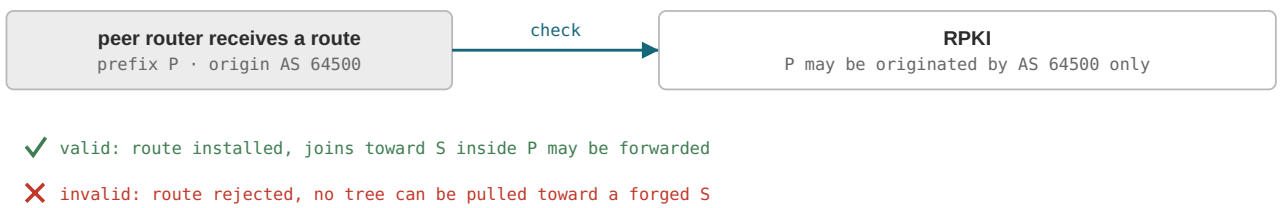



*Figure 6 - origin validation on a source. A peer checks the route that carries a source against the RPKI, and a forged origin is rejected before any join is forwarded.*

Two further properties follow from the shape. A receiver that named a source is fed by that source and nothing else, so there is no equivalent of the hostile sender of §3; traffic from an address a receiver did not name has no tree to travel. And the content on the tree proves itself, which is the subject of *The data carrier* and of *Multicast over a unicast Internet* §14: a frame is checked by its signature and an object by its proofs, so even a source that was somehow forged would deliver nothing a receiver would accept.

## 8. Where any-source still fits

One site, one operator, one switch. A lab or a single-room deployment with a switch that snoops multicast listener reports has no domain border, no hostile sender and no need for a rendezvous point beyond the switch itself, and any-source multicast at site scope is the simplest thing that works there. The fabric's frames, its sequence numbers, its group derivation and its repair protocol are identical in both modes; only the join and the address range differ. A lab built any-source becomes a source-specific deployment by changing the mode and naming the sources, and nothing that was built on it has to change.

| Posture | Data plane | Control groups | Rendezvous point | Crosses a domain border |
|---|---|---|---|---|
| one site, any-source | ASM, site scope | ASM | the switch, or one router | no |
| one domain, source-specific | SSM | SSM | none | no |
| **between domains** | **SSM, global** | **SSM** | **none** | **yes** |

*1. Only the last row peers. The frames and the repair protocol are the same in all three.*

## 9. Why it scales

Of the properties *Multicast over a unicast Internet* lists as the reason this multicast scales where earlier attempts did not, source-specific multicast is the one the network itself contributes: no one can send to a group. A receiver pulls a tree from a source it named, the network holds state only for trees that were asked for, and the list of sources is a published record rather than a protocol that floods. Between domains that is what makes a peering a configuration on routers the peer already owns, with a few source prefixes exchanged and nothing to defend but routes, which the routing system already knows how to defend.

## 10. Assumptions and limits

| Claim | Basis | Note |
|---|---|---|
| Any-source needs a rendezvous point and shared trees | **verified** | RFC 7761 |
| MSDP floods active sources between rendezvous points | **verified** | RFC 3618 |
| Any-source is deprecated between domains | **verified** | RFC 8815 (BCP 229) |
| Source-specific joins name a source and build reverse-path trees | **verified** | RFC 4607, RFC 3810 |
| The fabric's groups are in the source-specific range | **verified** | BRC-129 |
| The manifest carries sources with the group layout | **verified** | BRC-139 |
| Source prefixes travel in the multicast address family | **verified** | RFC 4760, RFC 8313 |
| Origin validation rejects a forged origin | **verified** | RFC 6811, RFC 9319 |
| The default-free zone holds about a million prefixes | **stated** | order of magnitude only |
| Frames and repair are identical in both modes | **structural** | the mode is a join and an address range, not a format |

Limits, stated plainly:

1. **Discovery moved, it did not vanish.** A receiver needs the manifest and a few bootstrap names before its first data join, and a receiver with a wrong or stale manifest joins nothing useful.
2. **Reverse-path lookup needs a route to the source.** A peer that imports no source prefixes builds no tree; the exchange of a few aggregates is not optional at the network-layer peering.
3. **Origin validation protects the route, not the content.** It stops a forged origin; the content on a valid tree is checked by the receiver, per *The data carrier*.
4. **Any-source at one site assumes a cooperative site.** The posture table's first row has no defense against a sender inside the site, and is not offered beyond it.
5. **Join state scales with sources times groups at each receiver.** A receiver's budget of joins is finite; *Sharding the object plane* shows how the group plan keeps it bounded.

## References


*BRC-124: Multicast Transaction Frame Format*: https://github.com/bsv-blockchain/BRCs/blob/master/transactions/0124.md
*BRC-126: Multicast Transaction NACK Retransmission Protocol*: https://github.com/bsv-blockchain/BRCs/blob/master/transactions/0126.md
*BRC-129: IPv6 Multicast Group Address Assignments*: https://github.com/bsv-blockchain/BRCs/blob/master/transactions/0129.md
*BRC-139: Multicast Shard Manifest Announcement Protocol*: https://github.com/bsv-blockchain/BRCs/blob/master/transactions/0139.md
IETF, *RFC 3618: Multicast Source Discovery Protocol (MSDP)*: https://www.rfc-editor.org/rfc/rfc3618
IETF, *RFC 3810: Multicast Listener Discovery Version 2 (MLDv2) for IPv6*: https://www.rfc-editor.org/rfc/rfc3810
IETF, *RFC 4607: Source-Specific Multicast for IP*: https://www.rfc-editor.org/rfc/rfc4607
IETF, *RFC 4760: Multiprotocol Extensions for BGP-4*: https://www.rfc-editor.org/rfc/rfc4760
IETF, *RFC 6811: BGP Prefix Origin Validation*: https://www.rfc-editor.org/rfc/rfc6811
IETF, *RFC 7761: Protocol Independent Multicast - Sparse Mode (PIM-SM)*: https://www.rfc-editor.org/rfc/rfc7761
IETF, *RFC 8313 (BCP 213): Use of Multicast across Inter-Domain Peering Points*: https://www.rfc-editor.org/rfc/rfc8313
IETF, *RFC 8815 (BCP 229): Deprecating Any-Source Multicast (ASM) for Interdomain Multicast*: https://www.rfc-editor.org/rfc/rfc8815
IETF, *RFC 9319 (BCP 185): The Use of maxLength in the RPKI*: https://www.rfc-editor.org/rfc/rfc9319
*Sharding the object plane* (Lightweb Inc., Paper 010): https://1bsv.net/papers/object-plane-sharding.pdf
*The data carrier* (Lightweb Inc., Paper 011): https://1bsv.net/papers/data-carrier.pdf
*Multicast over a unicast Internet* (Lightweb Inc., Paper 012): https://1bsv.net/papers/multicast-over-unicast-internet.pdf


---

**1BSV - BSV Layered Multicast** - cash moves fast on 1BSV. · 1bsv.net

---


**Lightweb Inc.** · Jeff Harris · jeff@lightweb.net · github.com/lightwebinc


*Unbounded Solutions for a Small World™ · © Lightweb Inc.*


© 2026 Lightweb Inc. All rights reserved. 1BSV and Unbounded Solutions for a Small World are trademarks of Lightweb Inc. This paper describes an architecture in rapid development: the design, its interfaces and this document are subject to change without notice, and later versions supersede this one. Nothing in this paper is a commitment to ship any feature, an offer of service, or professional advice.