Multicast in the Browser

Direct Sockets Enable Isolated Web Apps on TreeDN

IETF 124 MBONED Working Group

Omar Ramadan, Blockcast

TreeDN Efficiency – Even for Unicast

The Inefficiency Problem

- Traditional CDN: x86 servers replicate packets
- 1M viewers = 1M copies from origin
- High bandwidth costs, server sprawl, energy waste

TreeDN Solution

- Router card ASIC handles replication without x86
- Encourages In-ISP presence
- Incremental path towards multicast

Incremental Path: Phase 1 → Edge push • Phase 2 → Native multicast to edge • Phase 3 → Native multicast to Browser

Direct Sockets API – UDP in the Browser

Code Example

```
// Service Worker (IWA context)
const socket = new UDPSocket({
  localAddress: '0.0.0.0'
});
const info = await socket.opened;
socket.joinGroup('232.1.2.3');
const reader = info.readable
  .getReader();
const {value} = await reader.read();
```

Key Architecture

IWA: Sandboxed with elevated permissions

Service Worker: Only place Direct Sockets work

Chrome Canary: WIGC Experimental feature

```
- --enable-features=DirectSocketsInServiceWorkers
- --enable-features=MulticastInDirectSockets
```

Security Model

- No DOM access from Service Worker
- localhost-only for TCP servers
- Origin-based permissions

Critical Limitation – No SSM Support

Current Implementation Problem

```
// Chrome's joinGroup implementation
socket.JoinGroup(address); // Only group!
// System: IP_ADD_MEMBERSHIP (ASM only)
// NOT: IP_ADD_SOURCE_MEMBERSHIP (SSM)
```

What This Means

- ✅ ASM (*,G): Any-Source works
- ❌ SSM (S,G): Source-Specific NOT supported
- Modern networks prefer SSM

Workaround: AMT

- Browser ← UDP → AMT Relay

Call to Action: Chrome needs joinGroup(group, source)

Current Architecture (POC)

TreeDN → Multicast Source → Routers → AMT Relay

↓ UDP Tunnel (AMT)

Chrome Canary (IWA + Direct Sockets)

Service Worker
UDPSocket → AMT → UDP Multicast → MediaSource API

WASM Gateway
Shared sockets • (S,G) routing • Output servers

⚠️ **POC Limitation: Raw MPEG-TS not loss-resistant, no ABR**
Future: HTTP proxy for HLS/DASH + MoQ extensions

DEMO (POC)

What You'll See

Chrome Canary with experimental flags enabled

Raw MPEG-TS → MediaSource API (direct playback)

Live metrics – packets/sec, throughput, latency

Optional: VLC via localhost TCP (HTTP adaptation proxy)

Current POC Limitations

- ❌ No loss resistance (MPEG-TS corruption)
- ❌ No ABR (adaptive bitrate switching)
- ❌ No HLS/DASH compatibility

Future: HTTP Proxy

HLS/DASH + MoQ extensions

Open Capacity Marketplace Context

The Problem

- Providers need capacity in specific geographies
- ISPs + broadcasters have unused multicast + bandwidth infrastructure
- Manual deals don't scale • No trustless verification

OCM Solution

Provider → Marketplace → Suppliers

↓

Proof of Bandwidth → Smart Contract

MBone + Multicast Adaption for HTTP + Edge Cache

ISPs/Broadcasters as Suppliers

Edge PoP→ Home/Modem → Browser

Trustless Verification

Short network saturation tests

Multi-CDN Core

Use major providers + offload independent providers

SONAR: Statistical Coverage Verification

Three-Layer Architecture

Layer 1: ALTA
Content auth • 6% overhead • 1-10ms latency

Layer 2: Statistical
VRF sampling • 0.1% users • German Tank estimator

Layer 3: zkSNARK
Privacy (m>1000) • 328-byte proof

How It Works

VRF Selection: Unpredictable random sampling prevents gaming

Challenge Message: 32 bytes per user in sample (320KB per 10k)

User Attestations: 160 bytes each → packet stats + content hash

German Tank: $\hat{N} = max(packet\_max) + max/m - 1$

Confidence: 95% CI with m=10K: ±0.3% (for N=10M)

Economics & Privacy

Direct Submission
$0.0001/tx • $1/test (m=10K) • ❌ Privacy risk

zkSNARK Aggregation
$0.02/test (99.998% savings) • ✅ Privacy preserved

Fraud Detection: Unicast lag visible in <1s • Economic security via staking

Future Work: OTT Video

Multicast Adaptation HTTP Proxy (MAHP)

Similar to mABR, closer to MAUD – works for on-demand

Next: Media over QUIC Multicast Extensions

Benefits

Loss recovery • <2s latency • HTTP/3

Goal: Production OTT with multicast efficiency

Challenges & Path Forward

Current Limitations

❌ **No SSM support** – Chrome needs joinGroup(group, source)

⚠️ **Experimental API** – Not production-ready yet

⚠️ **IWA installation friction** – Only managed devices

Path Forward

Standardize Direct Sockets SSM: Chrome/WICG collaboration

OCM Integration: Integrate SONAR coverage verification browser extension

OTT Apps: Integrate HTTP/QUIC adaption

Conclusion & Call to Action

What We've Shown

- ✅ TreeDN makes multicast CDN-compatible
- ✅ Direct Sockets brings multicast to browsers
- ✅ SONAR enables OCM participation with trustless verification

Next Steps

Try it: github.com/blockcast/go-amt

Read drafts: draft-ramadan-mboned-sonar-01 • TreeDN RFC 9706

Join: MBONED WG • OCM Foundation

Questions?