**Step 1: PowerShell Real-Time Stats Collector**

Save this as BridgeMonitor.ps1:

powershell

CopyEdit

param(

[int]$RefreshSeconds = 2

)

# Detect network adapters (modify to target specific one)

$adapters = Get-NetAdapter | Where-Object {$\_.Status -eq "Up"}

# Run monitoring loop

while ($true) {

Clear-Host

Write-Host "=== Tableau Bridge Real-Time Network Throughput ===" -ForegroundColor Cyan

Write-Host "Time: $(Get-Date -Format 'HH:mm:ss')"

foreach ($adapter in $adapters) {

$stats = Get-NetAdapterStatistics -Name $adapter.Name

Write-Host "Adapter: $($adapter.Name)"

Write-Host (" Sent: {0:N2} MB" -f ($stats.OutboundBytes / 1MB))

Write-Host (" Received: {0:N2} MB" -f ($stats.InboundBytes / 1MB))

}

# Optional: Per-process (Bridge only)

$bridgeProcs = Get-Process -Name tableau\_bridge, tabprotosrv, tableau\_bridge\_service -ErrorAction SilentlyContinue

if ($bridgeProcs) {

Write-Host "`nTableau Bridge Processes:"

foreach ($proc in $bridgeProcs) {

Write-Host (" {0} (PID {1}) - CPU: {2:N2}%" -f $proc.ProcessName, $proc.Id, $proc.CPU)

}

}

Start-Sleep -Seconds $RefreshSeconds

}

**▶️ Run It:**

powershell

CopyEdit

.\BridgeMonitor.ps1 -RefreshSeconds 3

This will give you a real-time console display every 3 seconds.

**▶️ Step 2 (Optional): Real-Time Web Dashboard (HTML/JS)**

If you want a simple web-based dashboard (localhost only):

1. Install a lightweight PowerShell web server (Polaris, Pode, or just use Start-Job + Out-File HTML).
2. Example simple static HTML with auto-refresh:

html

CopyEdit

<html>

<head><title>Bridge Throughput</title></head>

<body>

<h2>Tableau Bridge Network Throughput</h2>

<pre id="stats">

Loading...

</pre>

<script>

function refresh() {

fetch('/bridge\_stats.txt')

.then(res => res.text())

.then(txt => document.getElementById('stats').innerText = txt);

}

setInterval(refresh, 3000);

refresh();

</script>

</body>

</html>

1. Modify the PowerShell script to **write the stats to bridge\_stats.txt**, then serve it via IIS or a simple PowerShell web server.

**▶️ Step 3: (Optional) Chart View (React/JS)**

If you'd prefer a more advanced frontend (React + Chart.js), I can generate the full code for:

* Live updating line graphs of Mbps send/receive
* Min/max/average throughput
* Historical timeline (e.g., past 5 min)