Powershell-ICMP

Minimal ICMP-based file exchange utilities (**sender + listener**) — forked and adapted from the original Powershell-ICMP by Oddvar Moe (@oddvarmoe).

Current release: `v2.0.0`
License: BSD 3-Clause
Maintainer / Fork: kaestnja (Jan Kästner)
Contributions: ChatGPT ("Kati") — refactors, diagnostics, and robustness improvements.

Overview

This repository contains two small PowerShell utilities intended for **controlled lab/test environments**:

- **Powershell-ICMP-Listener.ps1**
  Binds to a configured local IPv4 address, captures ICMP Echo **Request** payloads (via an **IP raw socket** with `IOControl ReceiveAll`), reassembles simple chunked transfers, performs integrity checks, and writes received files atomically into a `Dashboard` folder.

- **Powershell-ICMP-Sender.ps1**
  Creates (by default) a small host-info file `<ComputerName>.txt`, prepends the **UTF-8 filename**, computes a **SHA-256 checksum** of the content, and sends everything in **ICMP-safe chunks** to a static listener IP. Optionally awaits a best-effort **ACK** from the listener when running elevated.

Both scripts are intentionally small and self-contained so they can be run in an isolated lab without external dependencies. Runtime configuration is read from a **PowerShell Data File** `Powershell-ICMP.config.psd1` placed **next to the scripts**.

What's new in v2

Compared to v1.x and the original PoC, this fork adds:

- **PSD1 configuration** (no JSON): `Powershell-ICMP.config.psd1` overrides sane defaults.
- **Extended header**: Transfer ID, total chunks, filename length, checksum length (SHA-256), optional HMAC slot.
- **Integrity check**: Listener verifies SHA-256 over the **raw content** before saving.
- **Atomic saves**: Write to `.part`, then move/replace final file → no half-written results.
- **Deduplication**: Completed `(srcIP, transferId)` cached with TTL to skip repeated transfers.
- **Rate control**: Sender delay between chunks (default 15 ms) to reduce burstiness.
- **ACK (best-effort)**: After successful save, listener emits an ICMP control payload; elevated sender can wait.
- **Diagnostics**: Clear host/PS info, local IPs, firewall helper, route hints, consistent timestamps.
- **Graceful exit**: `Ctrl+C` stops the loop; RCVALL is disabled and socket disposed.
- **Consistent file naming**: Receiver writes `Name_YYYYMMDD_HHMMSS.ext` preserving the **original extension**.

Protocol (compact)

ICMP Echo **payload** starts with the magic `"IC"` and **version** `0x02` in both first and continuation chunks.

First chunk layout

```
+0   : 'I' (0x49)
+1   : 'C' (0x43)
+2   : 0x02 (version)
+3   : flags (bit0 = FirstChunk = 1)
+4..7: TransferId (UInt32, little-endian)
+8..9: TotalChunks (UInt16, little-endian)
+10  : FileNameLen (byte)
+11  : ChecksumLen (byte)    # SHA-256 = 32 (expected)
+12  : HmacLen (byte)        # reserved, 0 in v2.3.0
+13.. : FileName (UTF-8, FileNameLen bytes)
+..   : Checksum (ChecksumLen bytes — SHA-256 over file content only)
+..   : Content (first slice; remainder of ICMP payload)
```

Continuation chunk layout

```
+0   : 'I' (0x49)
+1   : 'C' (0x43)
+2   : 0x02 (version)
+3..6: TransferId (UInt32, little-endian)
+7..8: Sequence (UInt16, 1-based, little-endian)
+9..10: TotalChunks (UInt16, little-endian)
+11.. : Content slice (raw file bytes for this sequence)
```

ACK payload (listener → sender)

```
+0 : 'I' (0x49)
+1 : 'C' (0x43)
+2 : 0x02
+3 : 0x80   # ACK marker
+4..7 : TransferId (UInt32, little-endian)
```

> The listener runs an **IP raw socket** with `IOControl ReceiveAll` (Windows NDIS) to capture the IP frame and strip the IP header manually to reach the ICMP payload. We do not rely on OS-level ICMP parsing in order to carry a custom app-level framing reliably.

Configuration (`Powershell-ICMP.config.psd1`)

Create this file **next to the scripts** to override defaults:

```
@{
  ListenIP               = '192.168.6.50'
  DashboardFolderName    = 'Dashboard'
  SharedSecret           = 'ICMP-LAB-SECRET'
  EnableFirewallRule     = $true
  CompletedTtlMinutes    = 10
  MaxConcurrentTransfers = 64
  MaxBytesPerTransfer    = 10485760
  DebugVerbose           = $false
  AckEnabled             = $true
  InterChunkDelayMs      = 15
  AckWaitTimeoutMs       = 5000
  IcmpMtuPayload         = 1472
  UseAckIfElevated       = $true
}
```

> Any missing keys fall back to built-in defaults in each script.

Quick start

1. Place the three files in one folder:
   `Powershell-ICMP-Listener.ps1`, `Powershell-ICMP-Sender.ps1`, `Powershell-ICMP.config.psd1`

2. On the **listener host** (the machine that owns the configured `ListenIP`, default `192.168.6.50`), run:

   ```
   pwsh .\Powershell-ICMP-Listener.ps1
   ```

   If run elevated and `EnableFirewallRule = $true`, the script will create/ensure an inbound **ICMPv4 Echo Request** rule for the listen IP.

3. On a **sender host** in the same network (adjust `TargetIP` in the PSD1 if needed), run:

   ```
   pwsh .\Powershell-ICMP-Sender.ps1
   ```

   The sender will create/update `Dashboard\<ComputerName>.txt`, send it in ICMP chunks, and (if elevated and `UseAckIfElevated = $true`) wait for a best-effort ACK.

Example output (sender)

```
```

```
Powershell-ICMP-Sender v2.0.0 starting...
Sender host: W051P11 | PS: 7.5.3 | Elevated: True
Local IPv4: 192.168.6.51
Target IP: 192.168.6.50
Dashboard folder exists: C:\...\Dashboard
Info file created/updated: C:\...\Dashboard\W051P11.txt
Reachability check: success to 192.168.6.50, rtt=0ms, ttl=128
Preparing transfer: ID=1907884005, File='W051P11.txt', Size=116 bytes, Chunks=1
Sent chunk 1/1
Sender elevated: waiting up to 5000 ms for ACK from 192.168.6.50...
ACK received: SUCCESS
Sender finished on host W051P11.
```

Receiver result: file saved as `W051P11_YYYYMMDD_HHMMSS.txt` in `Dashboard` (extension preserved).

Troubleshooting

- **No packets / no files**: Ensure the listener is run **as Administrator** to open raw IP sockets and enable `ReceiveAll`.
- **Firewall**: The listener will attempt to add a rule for **ICMPv4 Echo** to the configured IP if elevated. You can also create it manually.
- **ACK not received**: ACK is best-effort. Sender must be **elevated** to open a raw socket for listening. The transfer still succeeds without ACK.
- **Duplicates**: The listener caches completed `(srcIP, transferId)` for `CompletedTtlMinutes` to skip repeats.
- **High CPU / burst**: Increase `InterChunkDelayMs` (e.g., 30–50 ms).
- **Large files**: `MaxBytesPerTransfer` caps acceptable size on the listener.

Security & scope

These tools are for **lab/testing** only. ICMP data exfiltration can trigger IDS/IPS or violate policy. Use on **trusted hosts** and networks you control.
No confidentiality is provided by default. If you need authentication/confidentiality, add a shared secret (HMAC) in a future version (see roadmap).

Roadmap / TODO

- Optional **HMAC** over header+content using a shared secret from PSD1 (mutual authenticity).
- **Selective retransmission** / stronger ACK semantics.
- Optional **persistence** of in-flight state to survive listener restarts.
- Configurable **logging** to file and structured eventing.
- Simple **CLI parameters** to override PSD1 keys at runtime.

License & credits

- **License:** BSD 3-Clause (see original repository).
- **Original author:** Oddvar Moe (@oddvarmoe).
- **Fork maintainer:** kaestnja (Jan Kästner).
- **Contributions:** ChatGPT ("Kati").