Pacer: Comprehensive Network Side-Channel Mitigation in the Cloud

Aastha Mehta, Mohamed Alzayat, Roberta De Viti, Björn B. Brandenburg, Peter Druschel, Deepak Garg
Network side channels

Step 1: Observing victim’s traffic shape

A. Direct observation at a link

B. Contention at a shared link

Adversary’s timing measurements

Step 2: Inferences from traffic shape

- Traffic content (web pages\textsuperscript{[PETS02]}, VoIP\textsuperscript{[S&P08]}, videos\textsuperscript{[Security17]})
- Users’ medical, financial secrets\textsuperscript{[S&P10]}
Pacer: Key ideas

Cloaked tunnel abstraction ensures secret-independent traffic shape by design
Paravirtualized implementation for IaaS clouds
Batched transmission scheduling masks architectural side channels

In the paper:
Exploiting public inputs to efficiently shape bursty traffic
Graybox profiling to automatically compute efficient traffic shapes
Detailed evaluation results
Formal proof of security
Key abstraction

Key property: Shape of the tunnel traffic is independent of secrets

Unobservable payload
- Hide flow control
- Encrypt all packets to hide padding
- Elicit same response for padding and payload

Secret-independent timing
- Transmit on schedule
- Must not be delayed by secrets
Tunnel endpoints for Cloud tenants

Endpoint A integrated with VM host

Endpoint B on client device or premises
Endpoint architecture (conceptual)

- Install secret-independent schedule
- Generate fixed-sized packets on schedule

- e.g., IPSec
Challenge: timing delays

Application state or execution can affect stack execution

Performance isolation within guest is infeasible

Performance isolation in HV is impractical
- Hypervisor complexity
- Stack lock-in
- Still vulnerable to side channels
Pacer’s cloaked tunnel

- End-to-end padding
- All packets with valid TCP seq#
- Encryption after padding

- Transmit packets on a schedule
- Mask secret-dependent delays in HyPace
- Transmit fixed batch of packets in each interval

- Respects congestion control (practical requirement)

Guest

- Application
- Padding
- TCP
- IP
- Ethernet

Xen

- Dummy pkts (occasional)
- Encryption
- Pacing

Unobservable payload

Secret-independent timing

NIC \leftrightarrow Tunnel

dummy packets

performance isolation
Evaluation

Security:
• Accuracy of CNN attack classifier
• Formal model and proof of HyPace

Performance:
• Bandwidth vs privacy overheads for static content
• Client latency, throughput of a medical website
• Streaming experience from a multi-tier video service
### Application overheads

<table>
<thead>
<tr>
<th>Metric/Application</th>
<th>Medical</th>
<th>Video</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average bandwidth overhead</td>
<td>143%</td>
<td>4x</td>
</tr>
<tr>
<td>Latency overhead</td>
<td>10x – 18x</td>
<td>1-tier: 30x</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2-tier: 400x</td>
</tr>
<tr>
<td>Peak throughput overhead</td>
<td>14.4%</td>
<td>NA</td>
</tr>
</tbody>
</table>

No jitter at client-side

Overheads depend on application’s data, workloads, and public inputs
Summary

✓ Secure by design
  Secret-independent traffic shapes
  Masking secret-dependent delays

✓ Efficient and practical
  Public input-dependent shaping
  React to congestion control

✓ Usable
  Minimal changes to application
  Modest changes to hypervisor and guest OS

Code: https://gitlab.mpi-sws.org/pacer