ditto: WAN Traffic
Obfuscation at Line Rate

Roland Meier\(^{(1)}\),
Vincent Lenders\(^{(2)}\),
Laurent Vanbever\(^{(1)}\)

NDSS 2022
Traffic volume and timing allows to determine which video somebody is watching.
Traffic volume and timing allows to identify characteristics of the endpoint
... and there are many more
This kind of attacks is concerning for Wide Area Network operators too.

Diagram:
- Datacenter / Campus / …
- Dedicated high-throughput links
Major WAN operators acknowledge the risk and already use link-layer encryption.

Data-link Layer encryption in Azure

Whenever Azure Customer traffic moves between datacenters—outside physical boundaries not controlled by Microsoft (or on behalf of Microsoft)—a data-link layer encryption method using the IEEE 802.1AE MAC Security Standards® (also known as MACsec) is applied from point-to-point across the underlying network hardware. The packets are encrypted and decrypted on the devices before being sent, preventing physical "man-in-the-middle" or snooping/wiretapping attacks. Because this technology is integrated on the network hardware itself, it provides line rate encryption on the network hardware with no measurable link latency increase. This MACsec encryption is on by default for all Azure traffic traveling within a region or between regions, and no action is required on customers’ part to enable.

AWS Security Solutions

- Physical Layer: Secure facilities and optical encryption using AES-256
- Data link layer: MACsec AES-256 (IEEE 802.1AE)
- Network Layer: VPC Encryption | Cross-Region Peering | Amazon VPN
- Transport Layer: Amazon s2n | NLB-TLS | ALB | CloudFront | ACM integration
- Application Layer: AWS Crypto SDK | Server-side encryption with KMS integration
Three challenges for a practical WAN traffic-analysis prevention system

- **Security**
  - Traffic does not leak information
  - ditto makes observed traffic independent from the actual traffic

- **Performance**
  - WANs run at 100s of Gbps
  - ditto reduces overhead by using efficient traffic patterns

- **Deployability**
  - Infeasible to change all servers
  - ditto runs in the network data plane at line rate
Existing countermeasures do not satisfy the requirements of WANs

- Incomplete security
  Still allow traffic analysis attacks

- Low throughput
  Megabits to few gigabits per second

- Difficult to deploy
  Require changes at end-hosts
ditto protects against an eavesdropper and provides three security properties

- **Volume anonymity**
  Attacker cannot determine the size of packets and flows

- **Timing anonymity**
  Attacker cannot determine the timing between two packets

- **Path anonymity**
  Attacker cannot track packets across multiple protected links

Can record all (encrypted) traffic and metadata
Packet sizes and timing allow traffic-analysis attacks in unprotected traffic

size

$t_0$ $t$
The high-level idea behind ditto is to make the observed traffic independent from the real traffic.
The high-level idea behind ditto is to make the observed traffic independent from the real traffic.
While secure, “constant” traffic can be inefficient.
ditto shapes traffic according
to an efficient pattern
ditto runs in the network data plane
Computing efficient traffic patterns

Traffic shaping in the data plane

Experimental results
Computing efficient traffic patterns
Traffic shaping in the data plane
Experimental results
An efficient pattern minimizes padding and chaff packets
An efficient pattern minimizes padding and chaff packets

Minimize

- Padding

\[ t_0 \quad t \quad \text{ditto} \]

\[ t_0 \quad t \]
An efficient pattern minimizes padding and chaff packets.

- Minimize:
  - Padding
  - Chaff packets

Diagram: A visual representation showing the reduction of padding and chaff packets over time (t0 to t).
An efficient pattern minimizes padding and chaff packets

Minimize

- Padding
- Chaff packets

with a repeating pattern
The pattern should fit to the traffic that is expected in the protected network

- **Input:** Traffic distribution e.g., from own recording
- **Output:** Pattern states for a given pattern length $L$

$$P_i = \text{percentile}_{(i+1) \cdot 100/L} \mathcal{D}$$

$$i \in [0,1,\ldots,L - 1]$$
Computing efficient traffic patterns

Traffic shaping in the data plane

Experimental results
ditto uses three operations to enforce the pattern at line rate

- **Buffering**
  until a packet fits in the pattern

- **Padding**
  to make packets larger

- **Chaff packet insertion**
  to fill gaps without real traffic
At a high level, ditto consists of 4 building blocks

real packets → chaff packet insertion → buffering → padding → encryption → protected traffic
ditto sends traffic over encrypted tunnels (e.g., using MACsec)
ditto pads packets by adding custom headers
ditto uses round-robin scheduling to enforce the pattern
ditto uses priority queues to mix real and chaff packets
ditto generates chaff packets by recirculating and cloning them
ditto runs entirely in the data plane of programmable switches
Current switches do not support 2-level queueing — the paper explains how we solved it.
ditto protects against an eavesdropper and provides three security properties

- **Volume anonymity**
  Attacker cannot determine the size of packets and flows

- **Timing anonymity**
  Attacker cannot determine the timing between two packets

- **Path anonymity**
  Attacker cannot track packets across multiple protected links

  Traffic always follows the pattern, which makes the volume constant

  Traffic is sent at a fixed rate according to the pattern

  Traffic is encrypted per link and the volume is always the same
Computing efficient traffic patterns

Traffic shaping in the data plane

Experimental results
Our evaluation shows that ditto performs well and is secure

- Experiments on hardware
  Intel Tofino switches

- Simulations in software
  to show potential of future hardware
This experiment measures how much throughput ditto can achieve.
Ideally, the output rate equals the input rate
ditto reaches up to 78 Gbps with Internet backbone traffic on a 100 Gbps link
ditto performs significantly better than (idealized) related work
This experiment measures how much impact ditto has on applications.
ditto does not affect the website load time up to 60 % load
Longer patterns achieve better performance

![Graph showing load time vs. input rate with patterns of different lengths.]
Longer patterns achieve better performance
ditto: WAN Traffic Obfuscation at Line Rate

- **Security**
  Traffic does not leak information

- **Performance**
  WANs run at 100s of Gbps

- **Deployability**
  Infeasible to change all servers

**ditto** makes observed traffic independent from the actual traffic

**ditto** reduces overhead by using efficient traffic patterns

**ditto** runs in the network data plane at line rate

Roland Meier
meierrol@ethz.ch

github.com/nsg-ethz/ditto