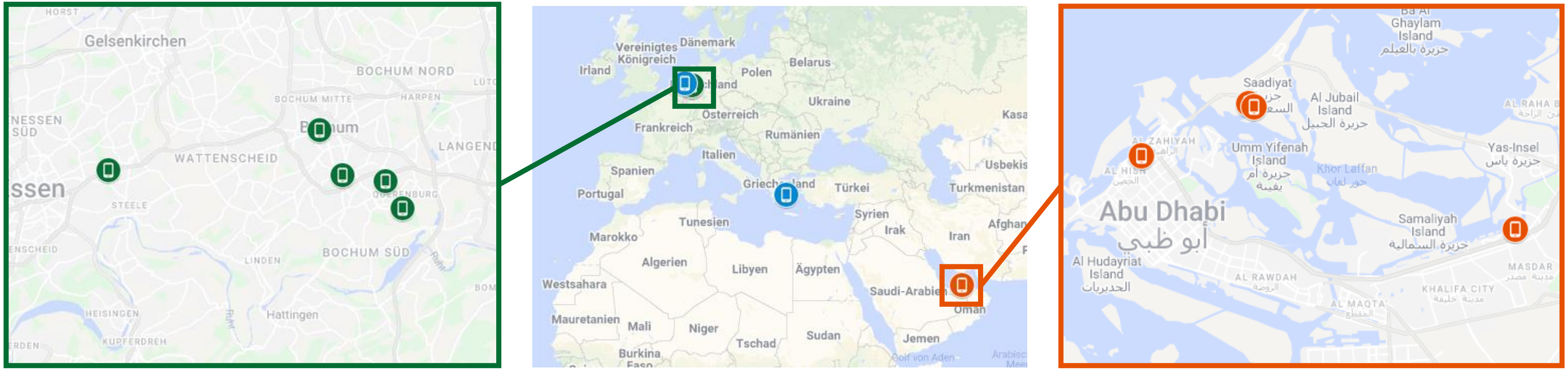


MEASURING MESSENGERS: ANALYZING INFRASTRUCTURES AND MESSAGE TIMINGS TO EXTRACT USER LOCATIONS IN INSTANT MESSENGERS

LASER 2023 | Learning from Authoritative Security Experiment Results
San Diego, CA, USA | March 03, 2023

Theodor Schnitzler
Research Center Trustworthy Data Science and Security, Germany
theodor.schnitzler@tu-dortmund.de

CENTER FOR TRUSTWORTHY DATA SCIENCE AND SECURITY
UA RUHR | RESEARCH ALLIANCE

NDSS 2023 PUBLICATION

Hope of Delivery: Extracting User Locations From Mobile Instant Messengers

Theodor Schnitzler[*,†], Katharina Kohls[‡], Evangelos Bitsikas[§,¶], and Christina Pöpper[¶]
[*]Research Center Trustworthy Data Science and Security, TU Dortmund, Germany [†]Ruhr-Universität Bochum, Germany
[‡]Radboud University, Netherlands [§]Northeastern University, USA [¶]New York University Abu Dhabi, UAE
theodor.schnitzler@tu-dortmund.de kkohls@cs.ru.nl bitsikas.e@northeastern.edu christina.poepper@nyu.edu

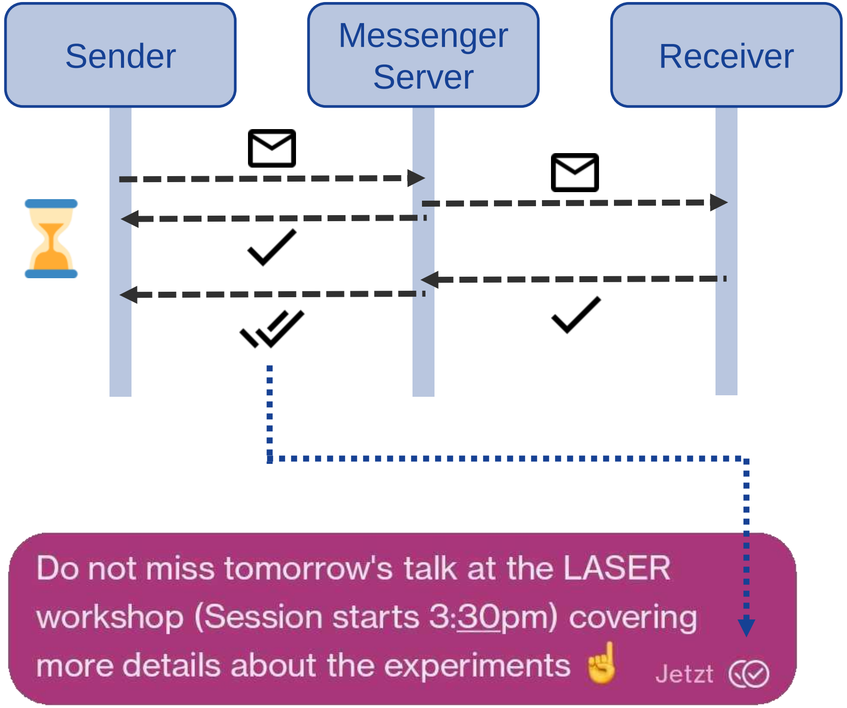
***Abstract*—Mobile instant messengers such as WhatsApp use delivery status notifications in order to inform users if a sent message has successfully reached its destination. This is useful and important information for the sender due to the often asynchronous use of the messenger service. However, as we demonstrate in this paper, this standard feature opens up a timing side channel with unexpected consequences for user location privacy. We investigate this threat conceptually and experimentally for three widely spread instant messengers. We validate that this information leak even exists in privacy-friendly messengers such as Signal and Threema.**

being in transit, processed and forwarded by the messenger server, to delivered to the recipient, and (if enabled) read by the recipient [2], often indicated by small symbols such as checkmarks. This is helpful information for users to track if a message has successfully reached its destination.

However, as we will demonstrate in our paper, this feature can also serve as a side channel that allows to learn sensitive information about message recipients, such as revealing information about their current whereabouts, with undesired potential harm to location privacy.

Paper:

PROBLEM STATEMENT



Scenario

Sender: *San Diego*
Server: *Los Angeles*

$c = 299\,792\,458\ \mathrm{m/s}$

$v_{Internet} \leq \frac{2}{3}\, c$

Receiver:	$2 * dist_{e2e}$	RTT
San Diego	$\geq 660\,\mathrm{km}$	$\geq 3.30\,\mathrm{ms}$
Bochum	$\geq 9\,200\,\mathrm{km}$	$\geq 46.03\,\mathrm{ms}$

Side Channel

Time for delivery confirmation reveals information about the receiver's location

Does this work in practice?

DATA COLLECTION

Round 1

- Fixed Locations
- WiFi-only
- (Mostly) country-level

Round 2 (Germany + UAE)

- Local setups at city-area-level
- Rotating devices through locations
- WiFi + mobile data

RESULTS OVERVIEW

Receiver Country (Round 1)

	DE	GR	NL
DE	0.88	0.06	0.06
GR	0.07	0.63	0.29
NL	0.06	0.22	0.72

74%

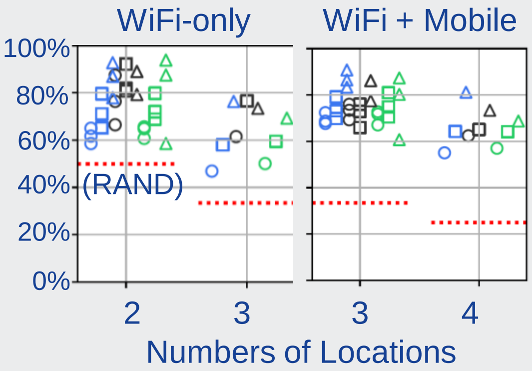
	DE	GR	NL
DE	0.90	0.02	0.08
GR	0.02	0.85	0.13
NL	0.09	0.13	0.77

84%

	AE	DE	GR	NL
AE	0.86	0.01	0.05	0.08
DE	0.04	0.81	0.06	0.09
GR	0.05	0.06	0.63	0.26
NL	0.09	0.06	0.18	0.67

74%

Device-at-Location (R2)

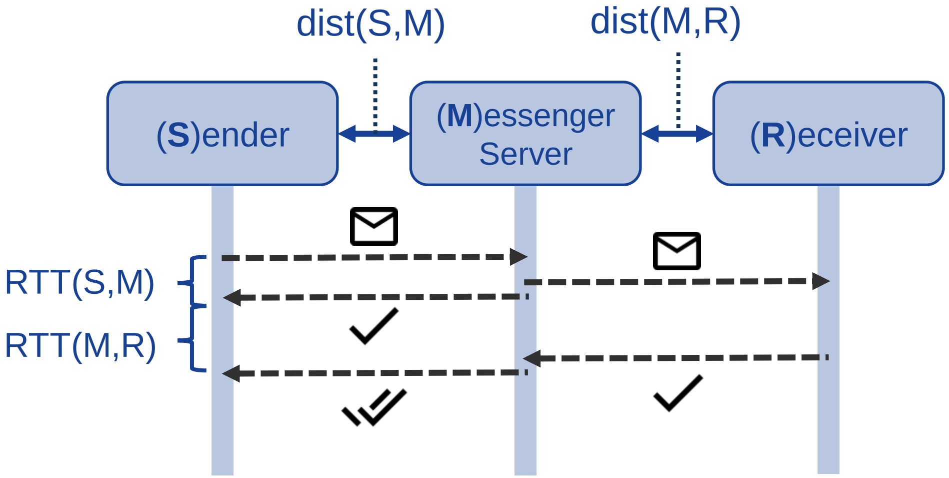


Network Connection (R2)

	Signal	Threema	WhatsApp
DE-22	92%	90%	92%
DE-23	90%	73%	89%
DE-24	94%	94%	92%

	Signal	WhatsApp
AE-22	56%	91%
AE-23	63%	82%
AE-24	76%	89%

MEASURING MESSENGERS



- Analyze Messenger Server Locations
- Identify Messages and Confirmations in Network Traffic
- Analyze Timings to Predict Location of Message Receivers

MESSENGER SERVER LOCATIONS

- No information provided
- Sources indicate **AWS US-East** (Ashburn, VA)

AWS EC2 North Virginia outage resolves but some issues linger

UPDATE: Signal falls over while Xero and Nest got a bit iffy when the main AWS EC2 region had degraded performance. Amazon Web Service says all is well but some users are still reporting trouble.

[zdnet.com]

- Servers located in Zurich area, CH

Where are the servers located?

Threema GmbH runs its own servers in two high-security data centers of an “ISO 27001”-certified colocation partner in the Zurich area (Switzerland).

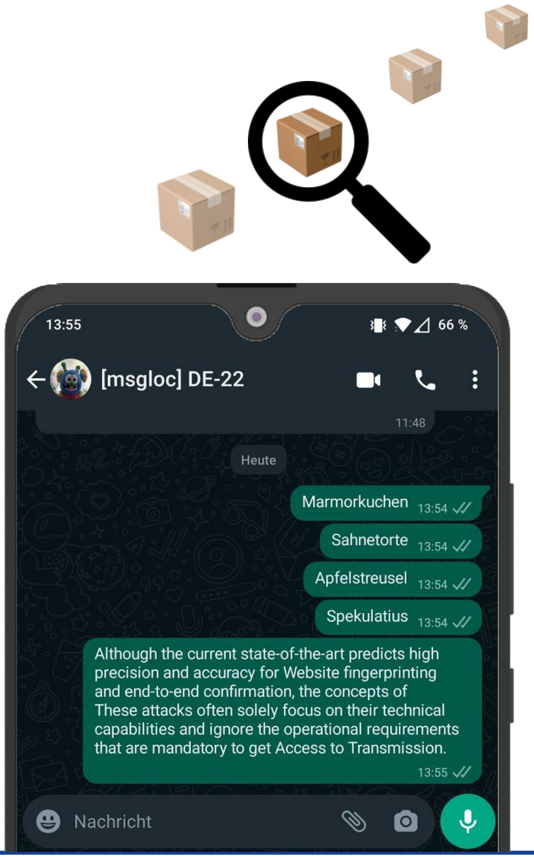
[threema.ch]

- No specific information
- Meta Data Centers (datacenter.fb.com)

?

Analyze Phone‘s Network Traffic
to verify and/or aggregate more information

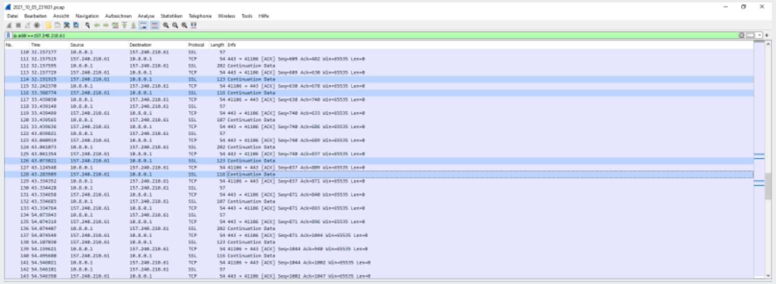
HOW TO ANALYZE NETWORK TRAFFIC ON ANDROID?



Packet Capturing

- tPacketCapture app
- Uses Android‘s VPN mechanism
- Monitor and collect (encrypted) traffic
- No root required

PCAP Analysis



FEDERATED ANALYSIS OF MESSENGER SERVERS

IP addresses

Manually checking
similar names/addresses

Federated DNS resolution

Reverse DNS lookup
`dig -x 123.123.123.123`

Domain names

IATA Location Codes

VMs in different countries

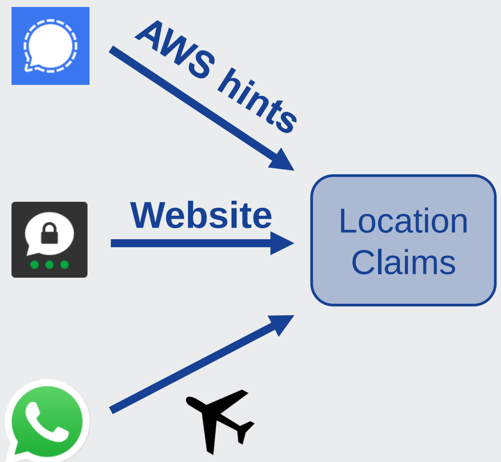
Online services

e.g., `check-host.net`

```
whatsapp-chatd-edge-shv-02-lax3.facebook.com.   (Los Angeles)
whatsapp-chatd-edge-shv-01-lhr8.facebook.com.   (London)
whatsapp-chatd-edge-shv-01-cdg2.facebook.com.   (Paris)
whatsapp-chatd-edge-shv-01-nrt1.facebook.com.   (Tokyo)
whatsapp-chatd-edge-shv-01-gru1.facebook.com.   (Sao Paolo)
…
```

LOCATION PLAUSIBILITY CHECK

Information Aggregation



Federated Pings

whatsapp-chatd-edge-shv-01-cdg2.facebook.com. Test

LOCATION	REQ	MIN	MAX	AVG	STD DEV	LOSS
Frankfurt 179.60.192.49	3	8.79 ms	8.83 ms	8.82 ms	0.02 ms	0%
Amsterdam 179.60.192.49	3	13.19 ms	14.18 ms	13.53 ms	0.46 ms	0%
London 179.60.192.49	3	16.21 ms	16.93 ms	16.49 ms	0.31 ms	0%
New York 179.60.192.49	3	76.96 ms	77.89 ms	77.32 ms	0.4 ms	0%
Dallas 179.60.192.49	3	112.52 ms	112.6 ms	112.56 ms	0.03 ms	0%
San Francisco 179.60.192.49	3	148.9 ms	149.38 ms	149.09 ms	0.2 ms	0%
Singapore 179.60.192.49	3	164.78 ms	165.79 ms	165.13 ms	0.47 ms	0%
Sydney 179.60.192.49	3	235.84 ms	235.86 ms	235.85 ms	0.01 ms	0%
Tokyo 179.60.192.49	3	232.98 ms	233.09 ms	233.04 ms	0.05 ms	0%
Bangalore 179.60.192.49	3	169.96 ms	170.87 ms	170.27 ms	0.42 ms	0%

[keycdn.com/ping]

Timings and Distances

- Calculate distances between location claim and probe locations
- Compare orders
- Compare transmission speeds
- No formal verification

MESSENGER SERVERS AND LOCATIONS

- 2 IPv4, both the same domain name

  `textsecure-service.whispersystems.org`
  `76.223.92.165    13.248.212.111`
  `ac88393aca5853df7.awsglobalaccelerator.com.`
- Pings < 3ms from each location
- Additional traceroutes from Europe point towards the US (East Coast)
- No certainty

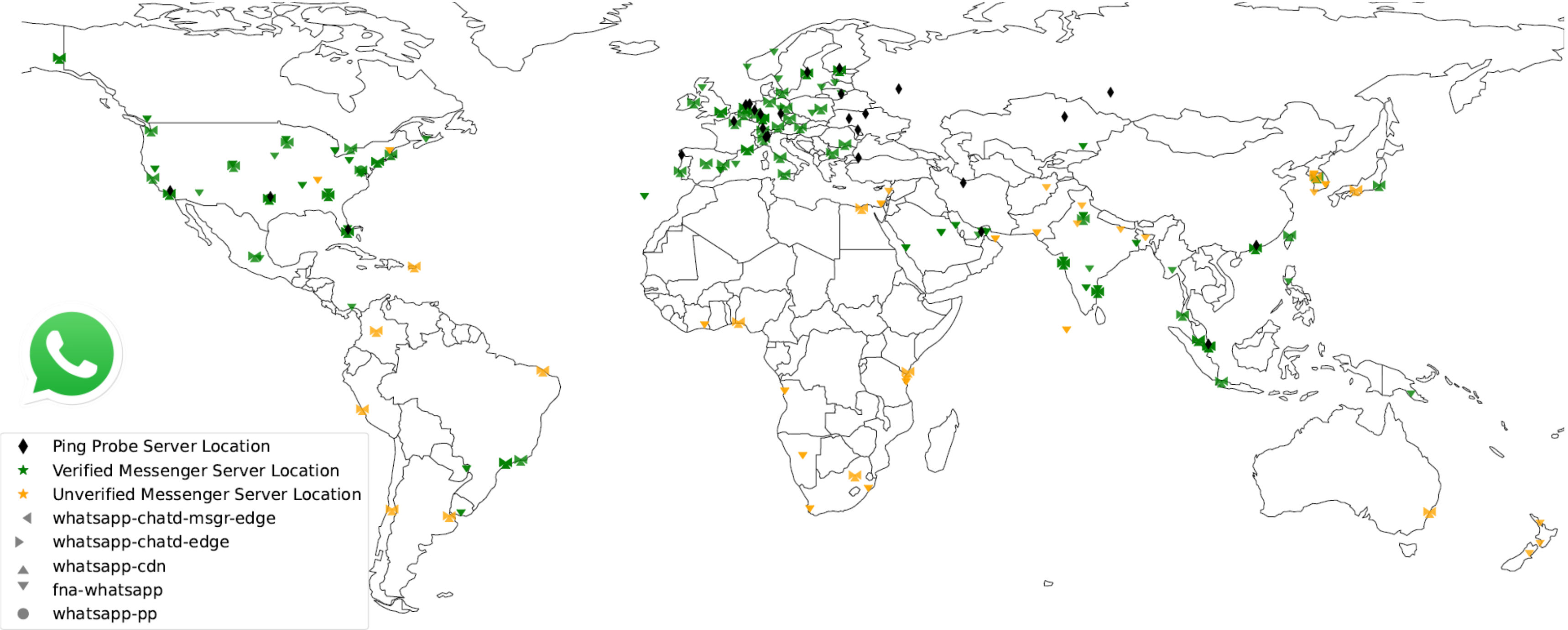
Virginia/US (?)

- 11 consecutive IPv4 addresses

  `msgapi.threema.ch`
  `185.88.236.xxx`
  `currently no response`
- Pings quite plausible
  - Frankfurt (DE) – Zurich: 300 km
  - Milan (IT) – Zurich: 220 km
  - Linear distance vs. Topology
  - Connectivity differences

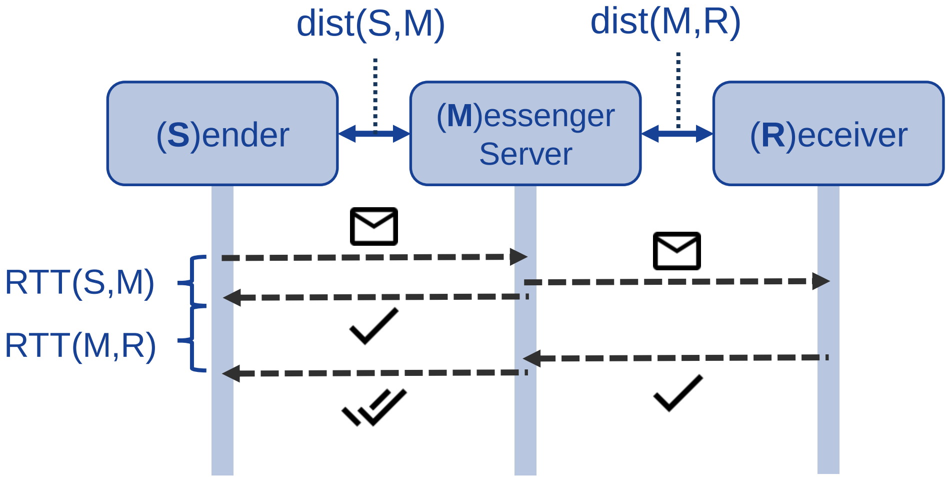
Zurich/CH

MESSENGER SERVERS AND LOCATIONS



5 domain namespaces, 409 total domains / IPv4, 142 different locations (US/EU mostly plausible)

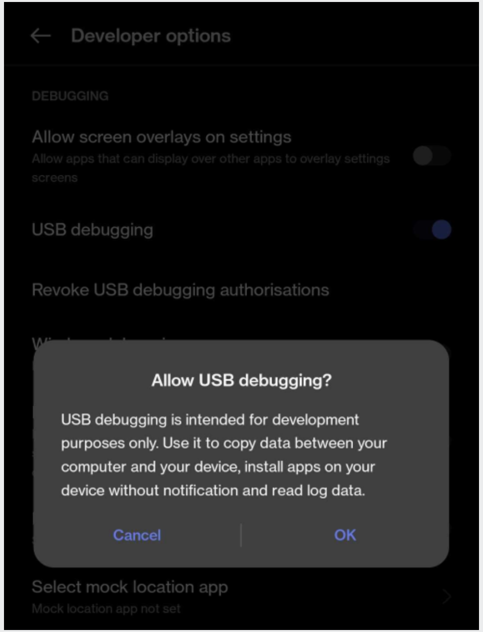
MEASURING MESSENGERS



- [x] Analyze Messenger Server Locations
- [ ] Identify Messages and Confirmations in Network Traffic
- [ ] Analyze Timings to Predict Location of Message Receivers

ANDROID DEBUG BRIDGE

Android Device



Controller (Laptop)

- Install ADB and start ADB server instance

```
apt-get install android-tools-adb
adb start-server
```

- Send commands to phone
  (confirm prompt on phone upon sending the first command)
  - Wake up phone

    ```
    adb shell input keyevent KEYCODE_WAKEUP
    ```

  - Start App

    ```
    adb shell am start -n
    jp.co.taosoftware.android.packetcapture/.PacketCaptureActivity
    ```

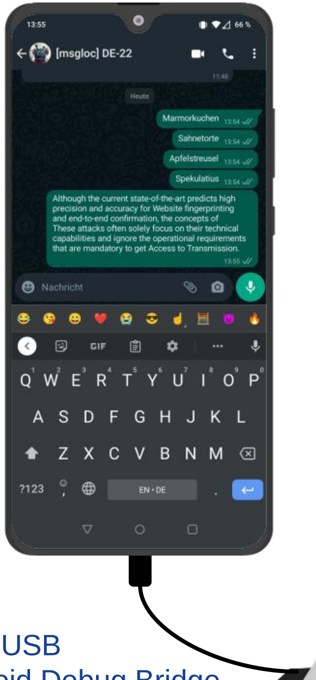
  - Interact with UI

    ```
    adb shell input tap <x> <y>
    adb shell input swipe <x1> <y1> <x2> <y2>
    ```

NO ROOT REQUIRED

[https://developer.android.com/studio/command-line/adb]

MEASUREMENT SETUP



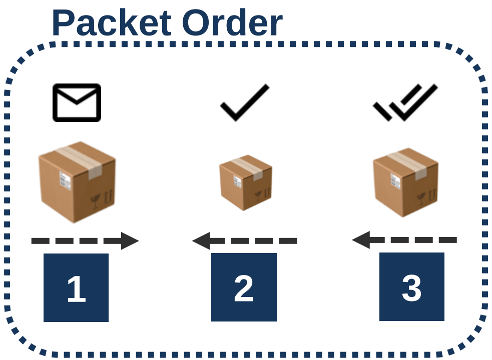
ADB-USB
Android Debug Bridge

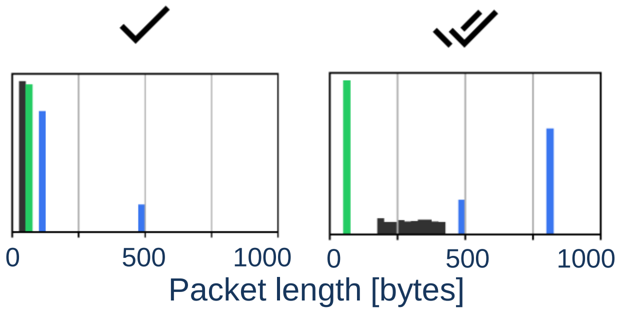
Sending Messages

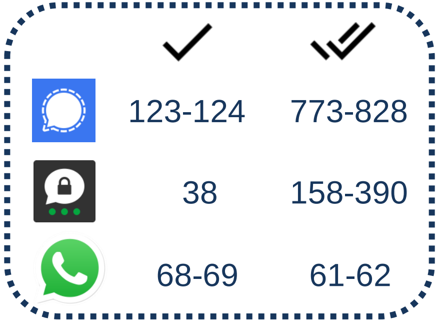
- Iterate through messengers + receivers
- Capture network traffic on the phone
- Open chat + send messages
  - 5 messages, 10s pause
- Continuously repeated (CronJob)

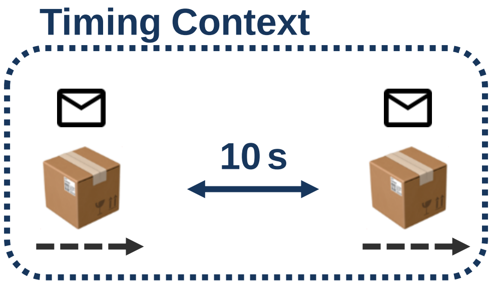
Receiving Messages

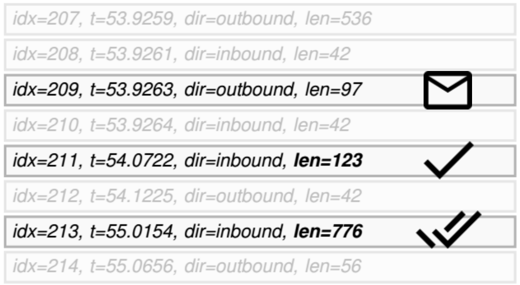
MESSAGES AND CONFIRMATIONS IN NETWORK TRAFFIC







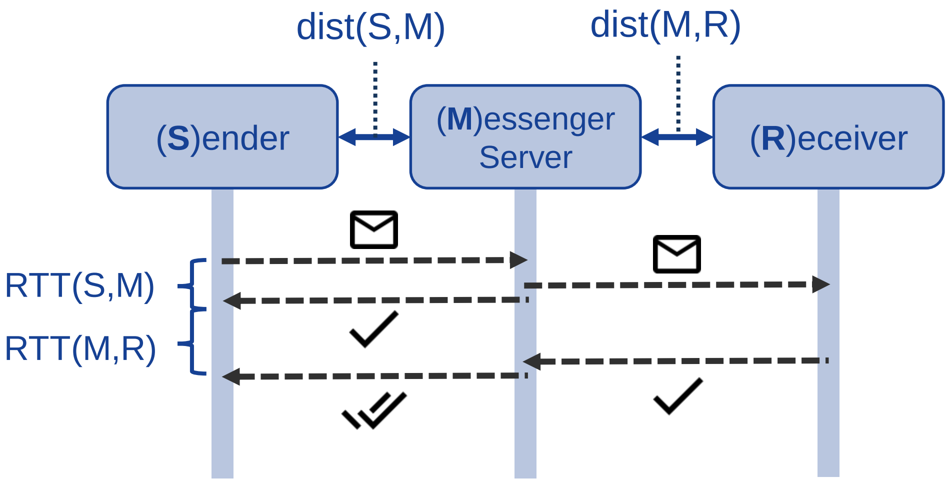




PCAP analysis
in Python: **dpkt**

```
pip install dpkt
import dpkt
frames = dpkt.pcap.Reader(open(pcap_file,'rb'))
```

MEASURING MESSENGERS



- [x] **Analyze Messenger Server Locations**
- [x] **Identify Messages and Confirmations in Network Traffic**
- [ ] **Analyze Timings to Predict Location of Message Receivers**

DATA COLLECTION

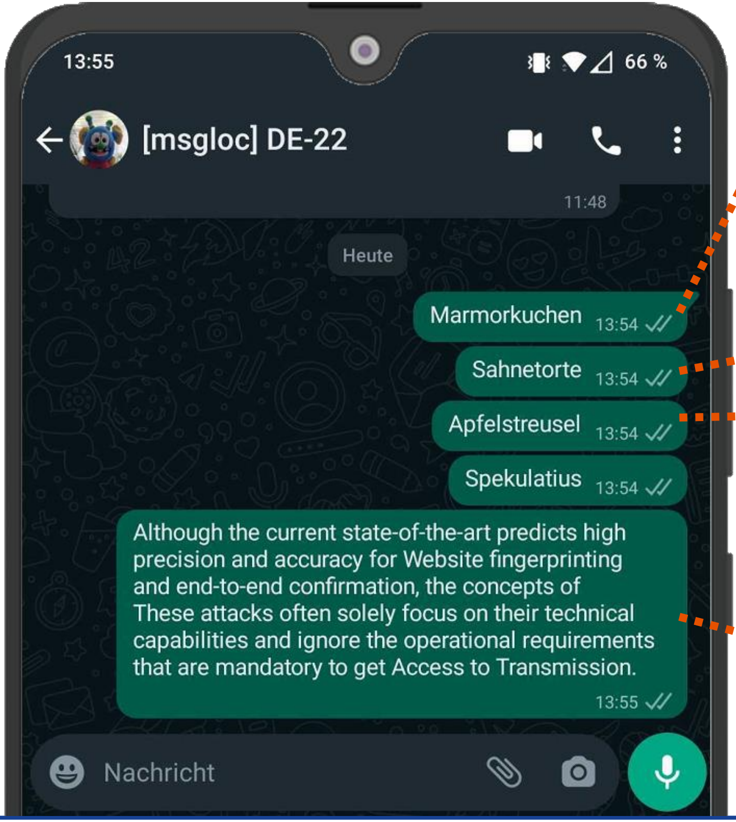
Round 1

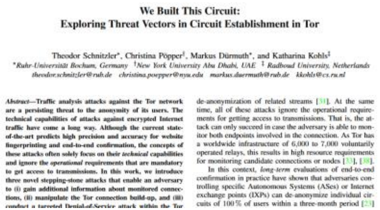
- Fixed Locations
- WiFi-only
- (Mostly) country-level

Round 2 (Germany + UAE)

- Local setups at city-area-level
- Rotating devices through locations
- WiFi + mobile data

MESSAGE CONTENTS





We Built This Circuit:
Exploring Threat Vectors in Circuit Establishment in Tor

Theodor Schnitzler*, Christina Pöpper†, Markus Dürmuth*, and Katharina Kohls‡
*Ruhr-Universität Bochum, Germany †New York University Abu Dhabi, UAE ‡ Radboud University, Netherlands
theodor.schnitzler@rub.de christina.poepper@nyu.edu markus.duermuth@rub.de kkohls@cs.ru.nl

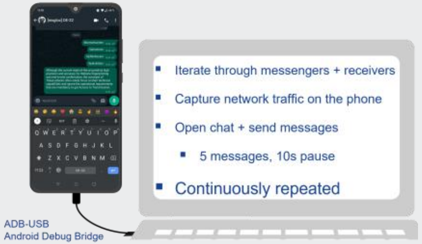
***Abstract*—Traffic analysis attacks against the Tor network are a persisting threat to the anonymity of its users. The technical capabilities of attacks against encrypted Internet traffic have come a long way. Although the current state-of-the-art predicts high precision and accuracy for website fingerprinting and end-to-end confirmation, the concepts of these attacks often solely focus on their *technical* capabilities and ignore the *operational* requirements that are mandatory to get access to transmissions. In this work, we introduce three novel stepping-stone attacks that enable an adversary to (i) gain additional information about monitored connections, (ii) manipulate the Tor connection build-up, and (iii)**

de-anonymization of related streams [31]. At the same time, all of these attacks ignore the operational requirements for getting access to transmissions. That is, the attack can only succeed in case the adversary is able to monitor both endpoints involved in the connection. As Tor has a worldwide infrastructure of 6,000 to 7,000 voluntarily operated relays, this results in high resource requirements for monitoring candidate connections or nodes [33], [38].

In this context, *long-term* evaluations of end-to-end confirmation in practice have shown that adversaries controlling specific Autonomous Systems (ASes) or Internet exchange points (IXPs) can de-anonymize individual circuits of 100 % of users within a three-month period [23]

DATA PREPARATION

Setup



Messenger Infrastructures

Packet Captures

Sender + Receiver

- Device ID
- Network
- Location

Distances

- dist(S,M)
- dist(M,R)

Messenger

- Name
- Server IP
- Server Loc.

Timings

- Date+Time
- Sent time
- RTT(S,M)
- RTT(S,R)
- RTT(M,R)

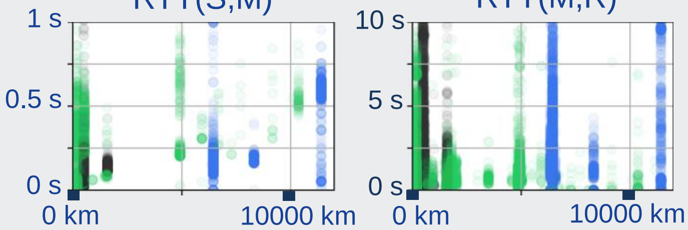
Evaluation Dataset

ANALYSIS OF TIMINGS AND DISTANCES

Timings and Distances

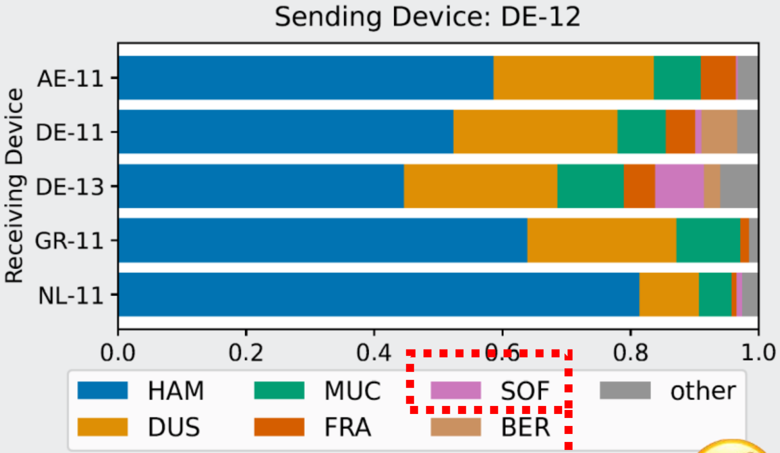
Goal: Find a function

$f: S \rightarrow T$ with $f(s)+\varepsilon = t,\ \varepsilon \rightarrow 0$



- Large time variances at similar distances
- Slow transmission speeds (M→R)
- Low variety in distances (generalizability!)
- Similar for individual senders and servers

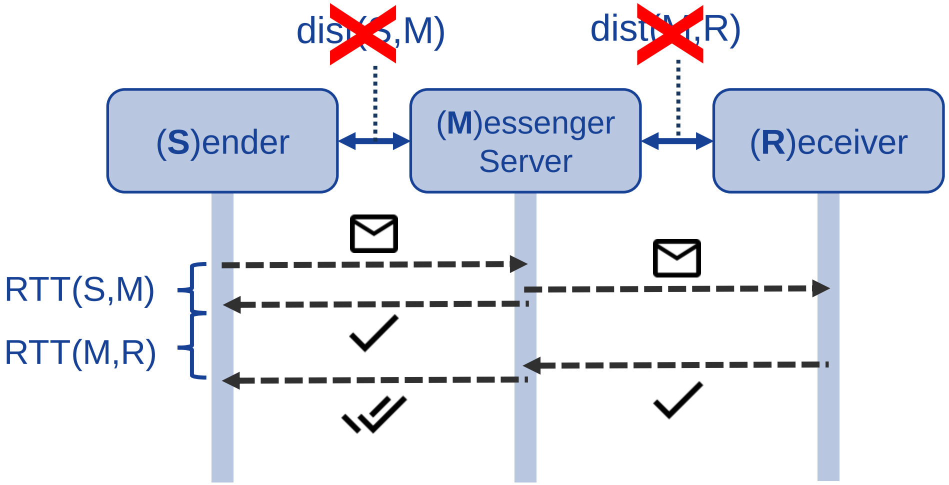
Server Selection



Bulgaria(?)

- Sender GR-11: Sofia, Bulgaria
- Sender AE-21: Marseille, France

MEASURING MESSENGERS

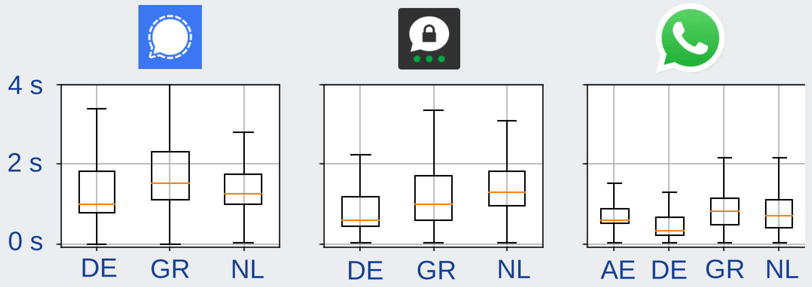


- [x] **Analyze Messenger Server Locations**
- [x] **Identify Messages and Confirmations in Network Traffic**
- [ ] **Analyze Timings to Predict Location of Message Receivers**

ANALYSIS OF TIMINGS AND RECEIVER LOCATIONS

Timing Distributions

Message Sender: DE-11 Bochum, DE



RTT(M,R) of 5 messages

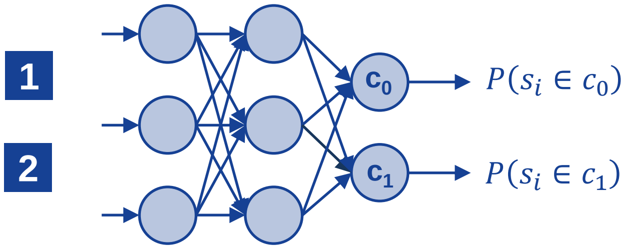
s	RTT_1(M,R)	RTT_2(M,R)	RTT_3(M,R)	RTT_4(M,R)	RTT_5(M,R)	c
s0	0.161045	0.367807	0.189508	0.133215	1.086010	1
s1	0.139126	0.263945	0.208273	0.318427	1.050682	0
s2	0.116070	0.959320	0.371446	0.075188	0.972167	0
s3	0.588105	0.432598	0.116624	0.217052	0.882888	0
s4	0.352139	0.093173	0.207296	0.184161	0.847522	0
s5	0.888563	0.149882	0.209223	0.175710	0.238975	1
s6	0.321202	0.267288	0.204692	0.152205	0.972913	1
s7	0.211452	0.156785	0.421123	0.165585	1.115668	0
s8	0.320205	0.650930	0.125180	0.784062	0.125119	0
s9	0.155052	0.177442	0.148592	0.078013	0.822601	1
s10	0.181755	0.196456	0.156299	0.203927	0.991780	0
s11	0.174066	0.307921	0.226345	0.322114	0.949903	1
s12	0.225167	0.150083	0.128277	0.178671	1.010559	0
s13	0.128531	0.217139	0.133994	0.269631	0.778859	1
s14	0.120790	1.006174	0.199258	0.094544	1.823422	0
s15	0.223729	0.199927	0.216786	0.145953	0.912231	1
s16	0.151150	0.182758	0.119122	0.197469	1.011616	1
s17	0.228764	0.313403	0.213551	0.427457	0.940652	1
s18	0.146101	0.182869	0.213168	0.201455	0.842262	1
s19	0.565934	0.404749	0.526175	0.218871	1.288376	0

1 80% data for training (s0–s14)

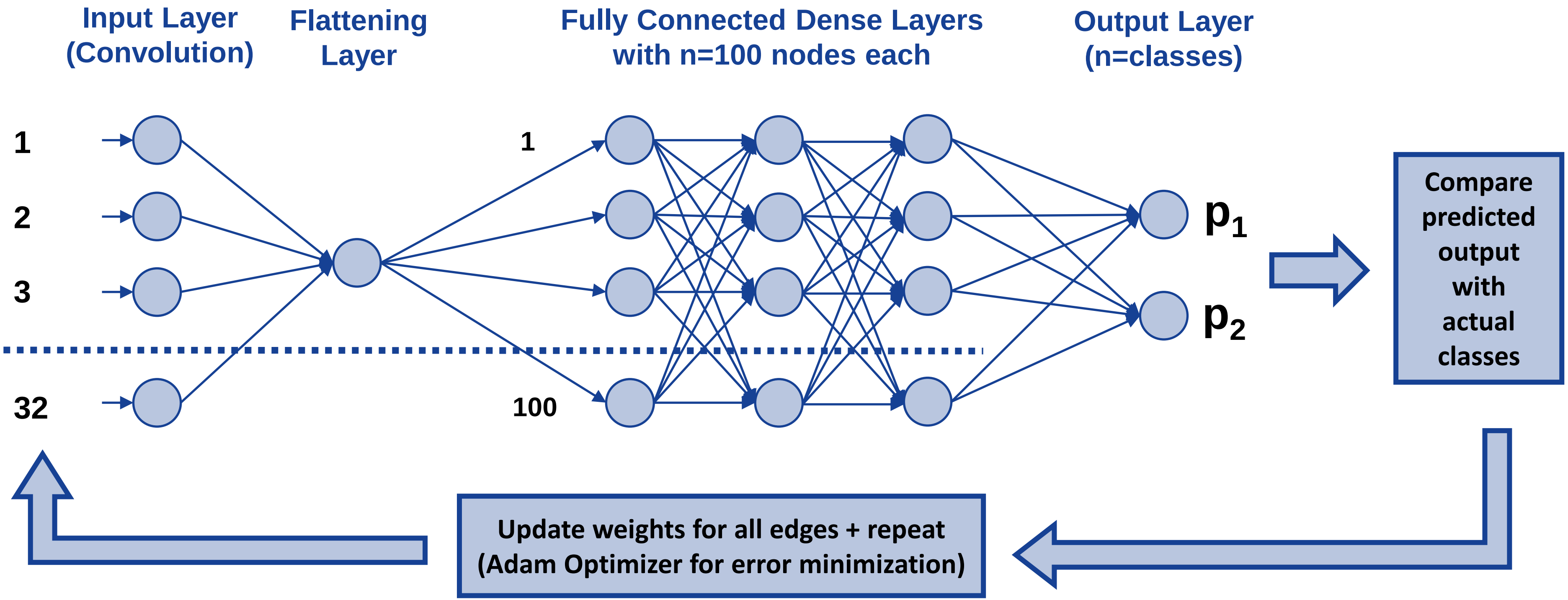
2 20% data for testing (s15–s19)

Classification

Assign newly measured RTTs a location based on previously observed data



NEURAL NETWORK ARCHITECTURE



NEURAL NETWORK IMPLEMENTATION

ML framework for Python (and other languages):
tensorflow (v2.11.0) -> Keras API
[https://www.tensorflow.org/api_docs/python/tf/keras]

Parameter Tuning

```
# build the nn
model = tf.keras.Sequential()
model.add(tf.keras.layers.Conv1D(parameters['filters'], kernel_size=(2), activation='relu',input_shape=input_shape))
model.add(tf.keras.layers.Flatten())
for d in range(parameters['dense_layers']):
    model.add(tf.keras.layers.Dense(parameters['neurons'], activation='relu'))
model.add(tf.keras.layers.Dense(num_classes,activation=tf.keras.layers.Softmax()))

# compile and run
model.compile(optimizer='adam',
    loss = tf.keras.losses.categorical_crossentropy,
    metrics=['accuracy'])

model.fit(train_data,train_labels,epochs=parameters['n_epochs'])

# predict test data
predictor = tf.keras.Sequential([model,tf.keras.layers.Softmax()])
predictions = predictor.predict(test_data)
```

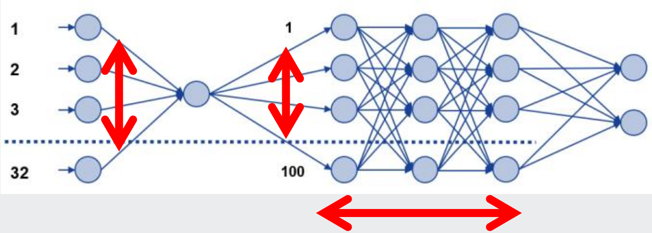
NETWORK LAYERS

TRAINING

PREDICTION

EVALUATION RUNS

Parameter Tuning



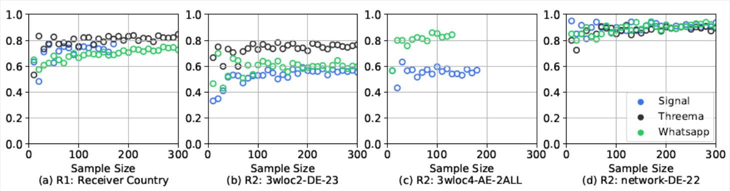
Actual Evaluation

- Receiver country
- Within a country (yes/no)
- Locations of a single receiver
- Network connection (WiFi/Mobile)

\+ *varying timing sequence lengths*

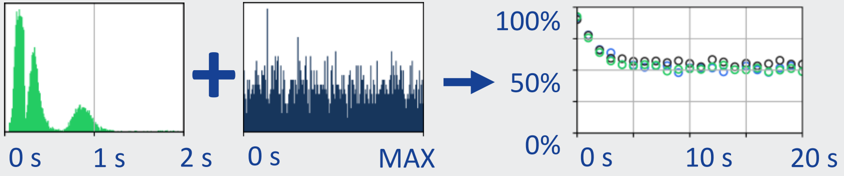
Convergence Analysis

Varying sample sizes – number of messages



Countermeasures Simulation

Adding random maximum delays to data



RESULTS OVERVIEW

Receiver Country (Round 1)

	DE	GR	NL
DE	0.88	0.06	0.06
GR	0.07	0.63	0.29
NL	0.06	0.22	0.72

74%

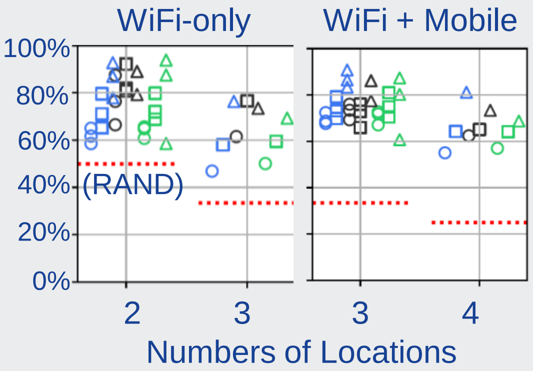
	DE	GR	NL
DE	0.90	0.02	0.08
GR	0.02	0.85	0.13
NL	0.09	0.13	0.77

84%

	AE	DE	GR	NL
AE	0.86	0.01	0.05	0.08
DE	0.04	0.81	0.06	0.09
GR	0.05	0.06	0.63	0.26
NL	0.09	0.06	0.18	0.67

74%

Device-at-Location (R2)



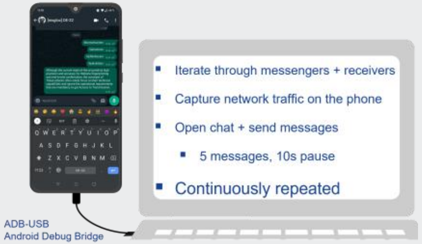
Network Connection (R2)

DE-22	92%	90%	92%
DE-23	90%	73%	89%
DE-24	94%	94%	92%

AE-22	56%	91%
AE-23	63%	82%
AE-24	76%	89%

DISCUSSION

Setup to collect message timing data



Overview of messenger infrastructures

Messenger PCAPs

- ☑ **Analyze Messenger Server Locations**
- ☑ **Identify Messages and Confirmations in Network Traffic**
- ☑ **Analyze Timings to Predict Location of Message Receivers**

Theodor Schnitzler, Katharina Kohls, Evangelos Bitsikas, Christina Pöpper
Hope of Delivery: Extracting User Locations from Mobile Instant Messengers
NDSS 2023 | The Network and Distributed System Security Symposium

MEASURING MESSENGERS: ANALYZING INFRASTRUCTURES AND MESSAGE TIMINGS TO EXTRACT USER LOCATIONS IN INSTANT MESSENGERS

LASER 2023 | Learning from Authoritative Security Experiment Results Workshop
San Diego, CA, USA | March 03, 2023

Theodor Schnitzler
theodor.schnitzler@tu-dortmund.de
@the0retisch

Research Center Trustworthy Data Science and Security, Germany
@rctrustworthy

CENTER FOR TRUSTWORTHY DATA SCIENCE AND SECURITY
UA RUHR | RESEARCH ALLIANCE