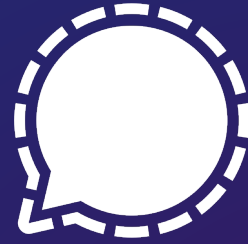


Using Separation Logic for Proving Security Protocol Implementations Secure

Linard Arqint

Joint work with Malte Schwerhoff, Vaibhav Mehta, and Peter Müller

ETH zürich



A GLITCH IN THE MATRIX —

Serious vulnerabilities in Matrix's end-to-end encryption have been patched

Previously overlooked flaws allow malicious homeservers to decrypt and spoof messages.

DAN GOODIN - 9/28/2022, 9:00 AM

A Traceability Attack against e-Passports

Tom Chothia* and Vitaliy Smirnov

School of Computer Science, University of Birmingham, Birmingham, UK

NEWS

Four cryptographic vulnerabilities in Telegram

An international research team of cryptographers completed a detailed security analysis of the popular Telegram messaging platform identifying several weaknesses in its protocol that demonstrate the product falls short of some essential data security guarantees.

16.07.2021 by [Marianne Lucien](#)

2 Comments



Existing Work



Protocol Model



Code Generation



Tamarin

ProVerif

DY*

Our Work



Existing
Implementations

Our Work



Our Work



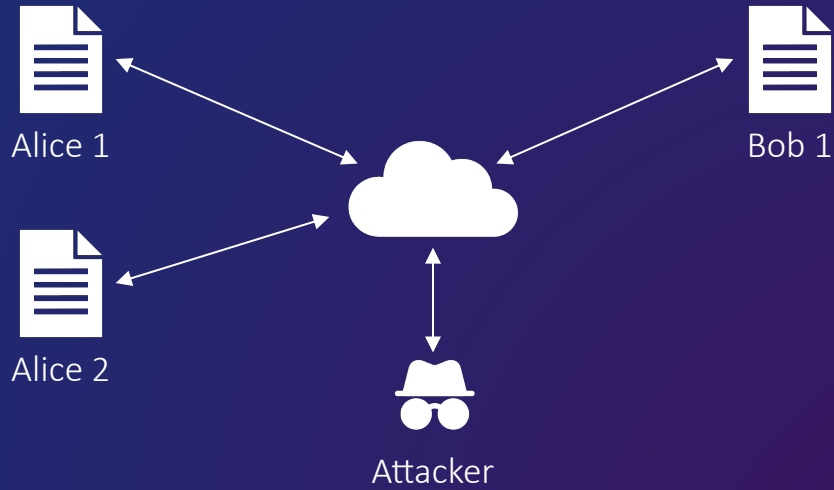
Existing
Implementations

Language agnostic

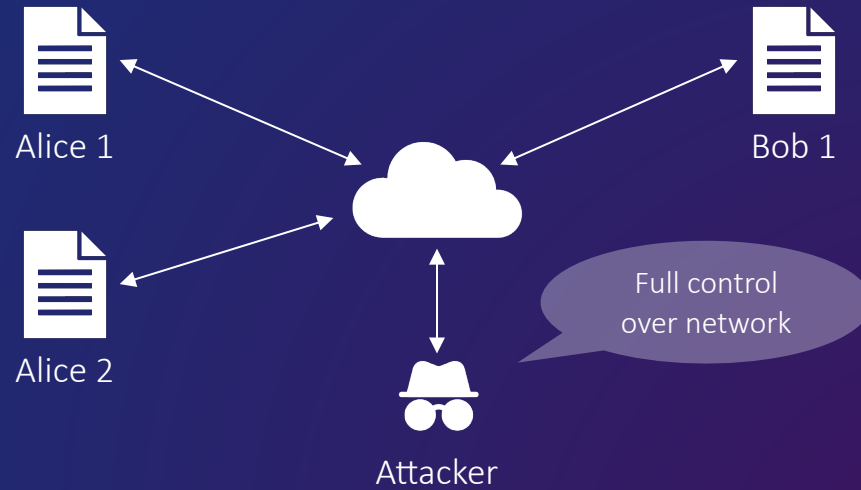
Verifier agnostic

Protocol agnostic

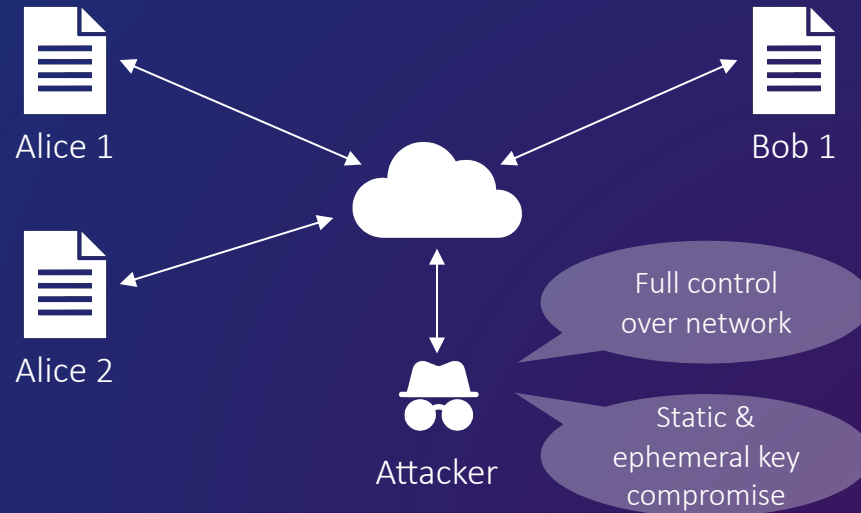
Approach



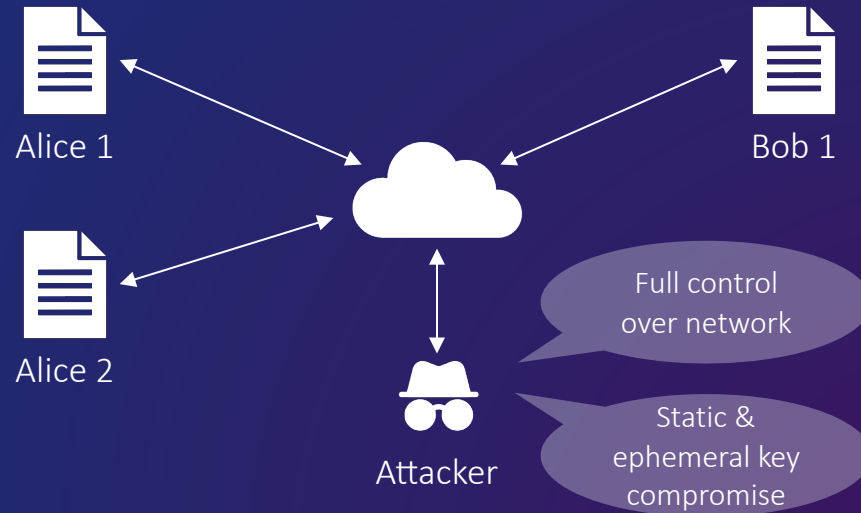
Approach



Approach

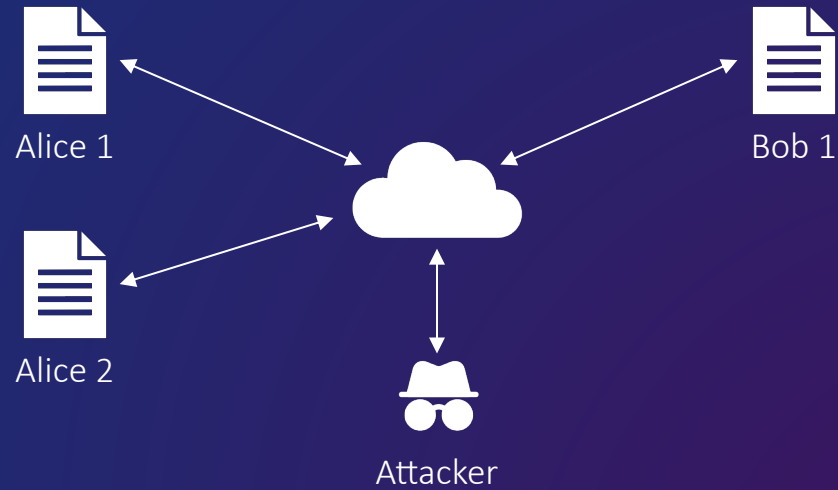


Approach



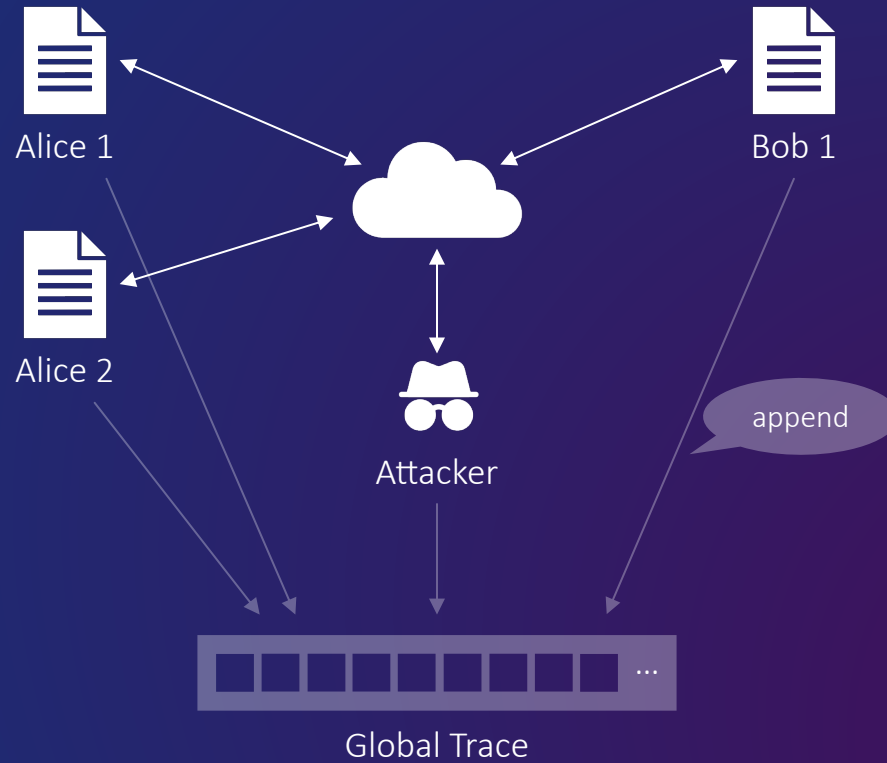
→ Model as a program with threads

Approach

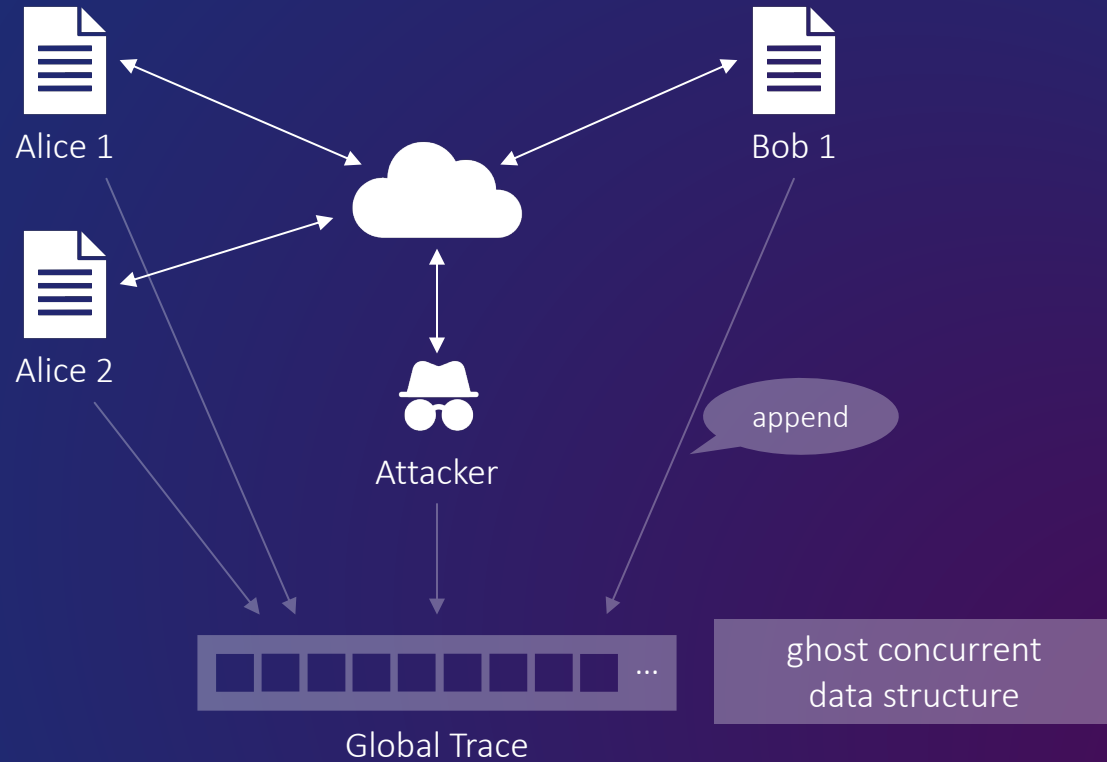


Global Trace

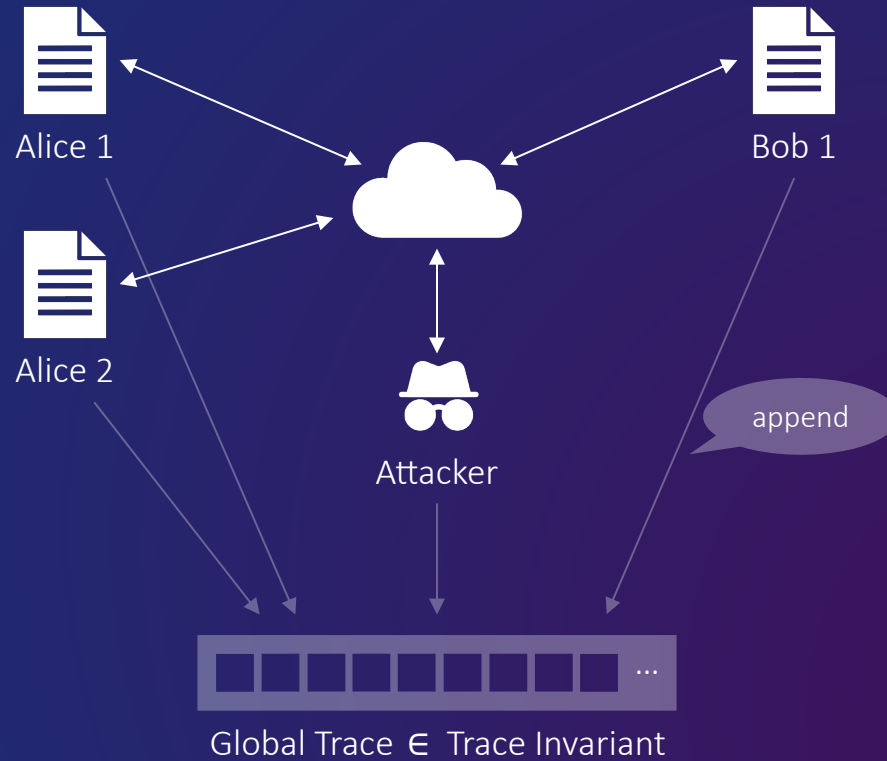
Approach



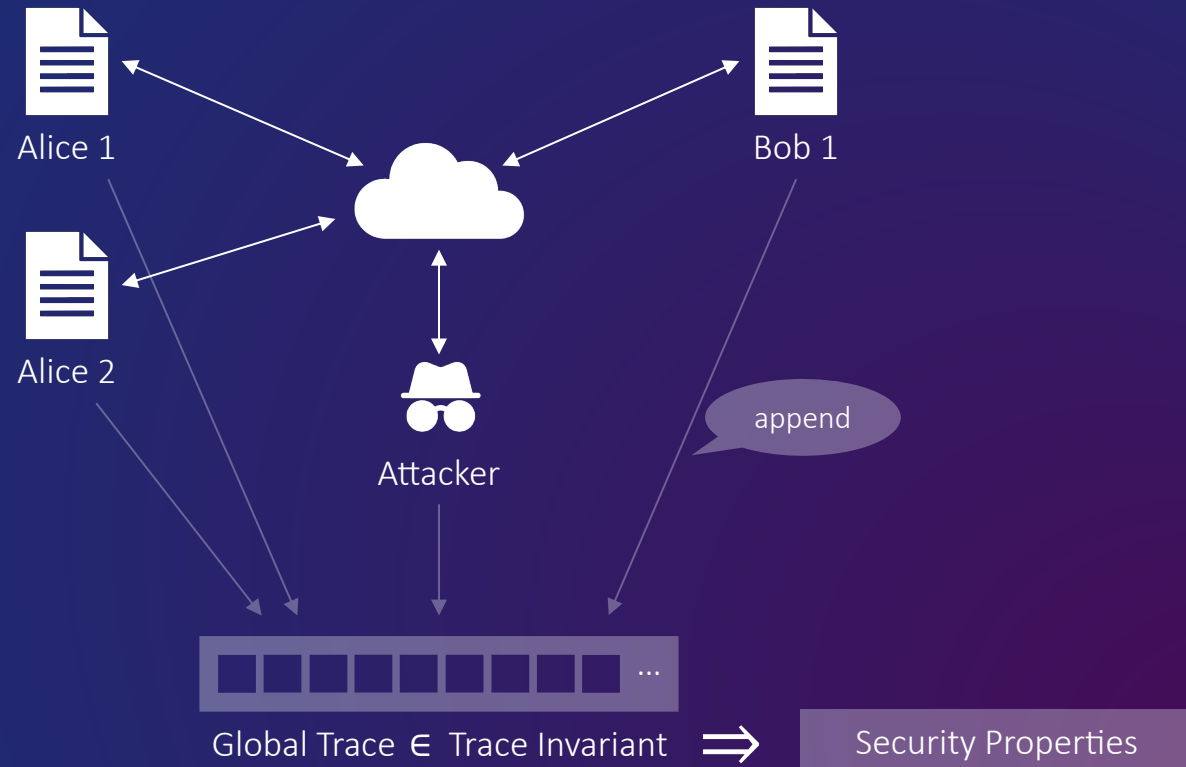
Approach



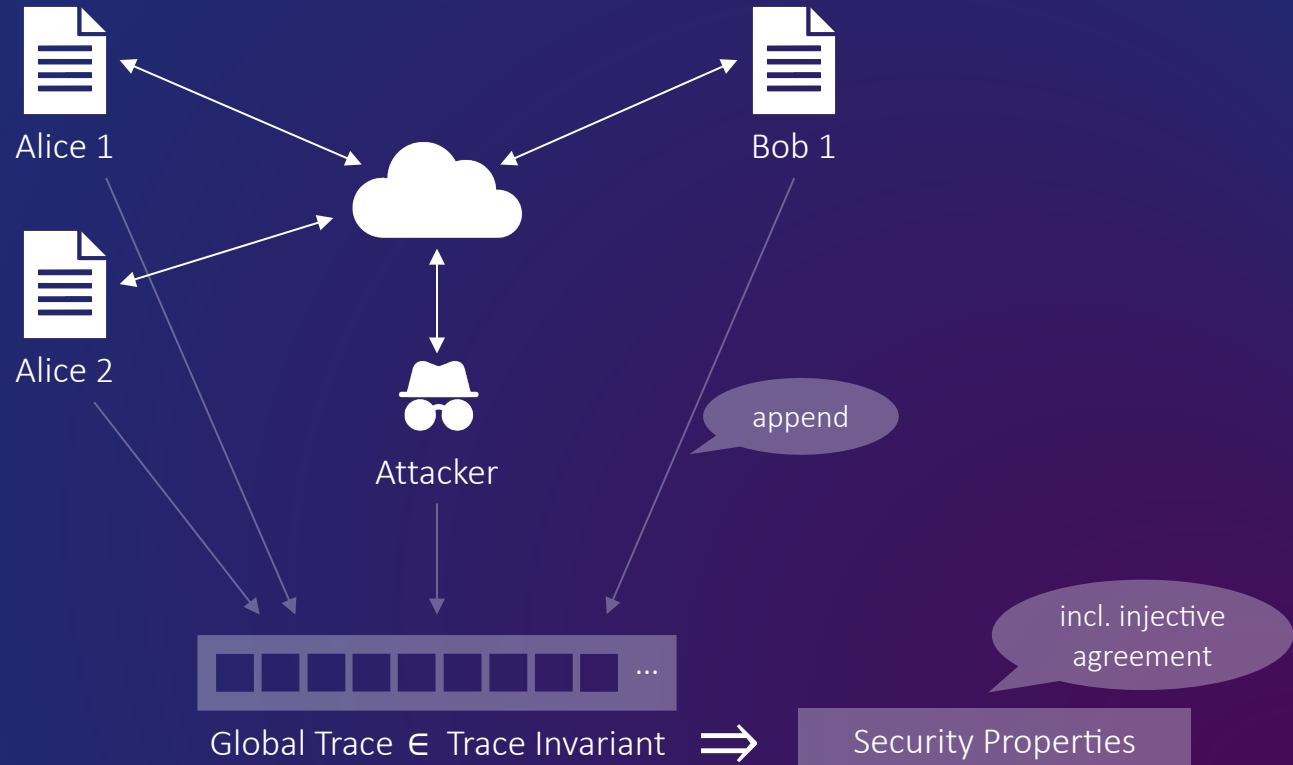
Approach

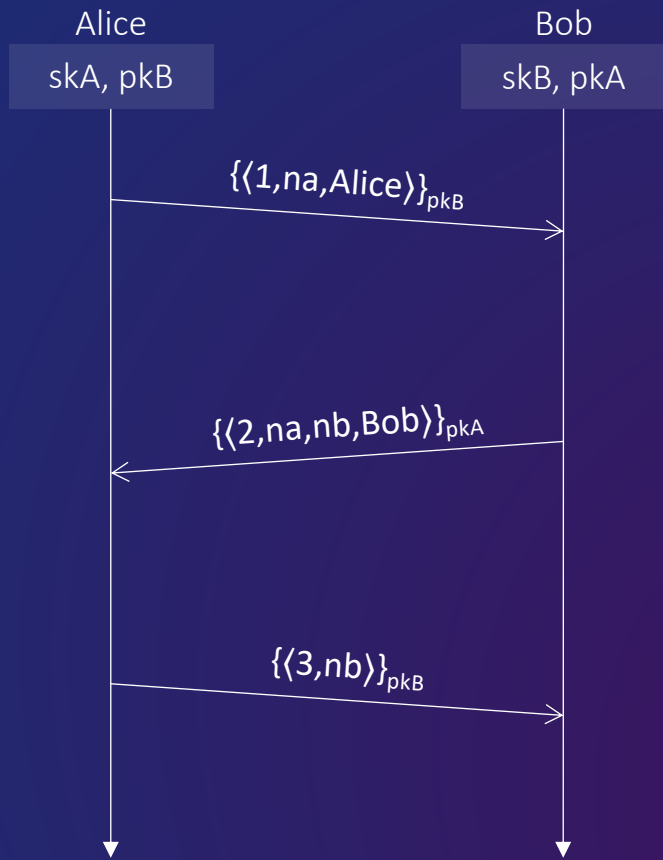


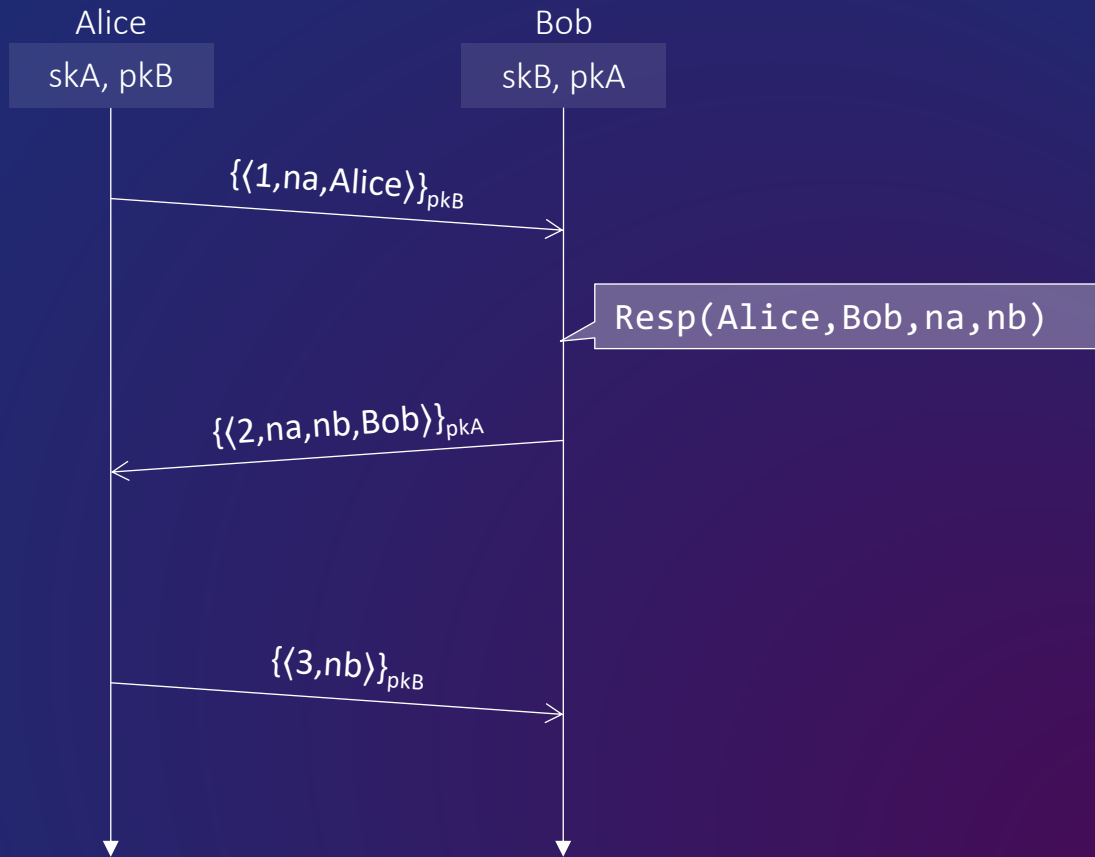
Approach

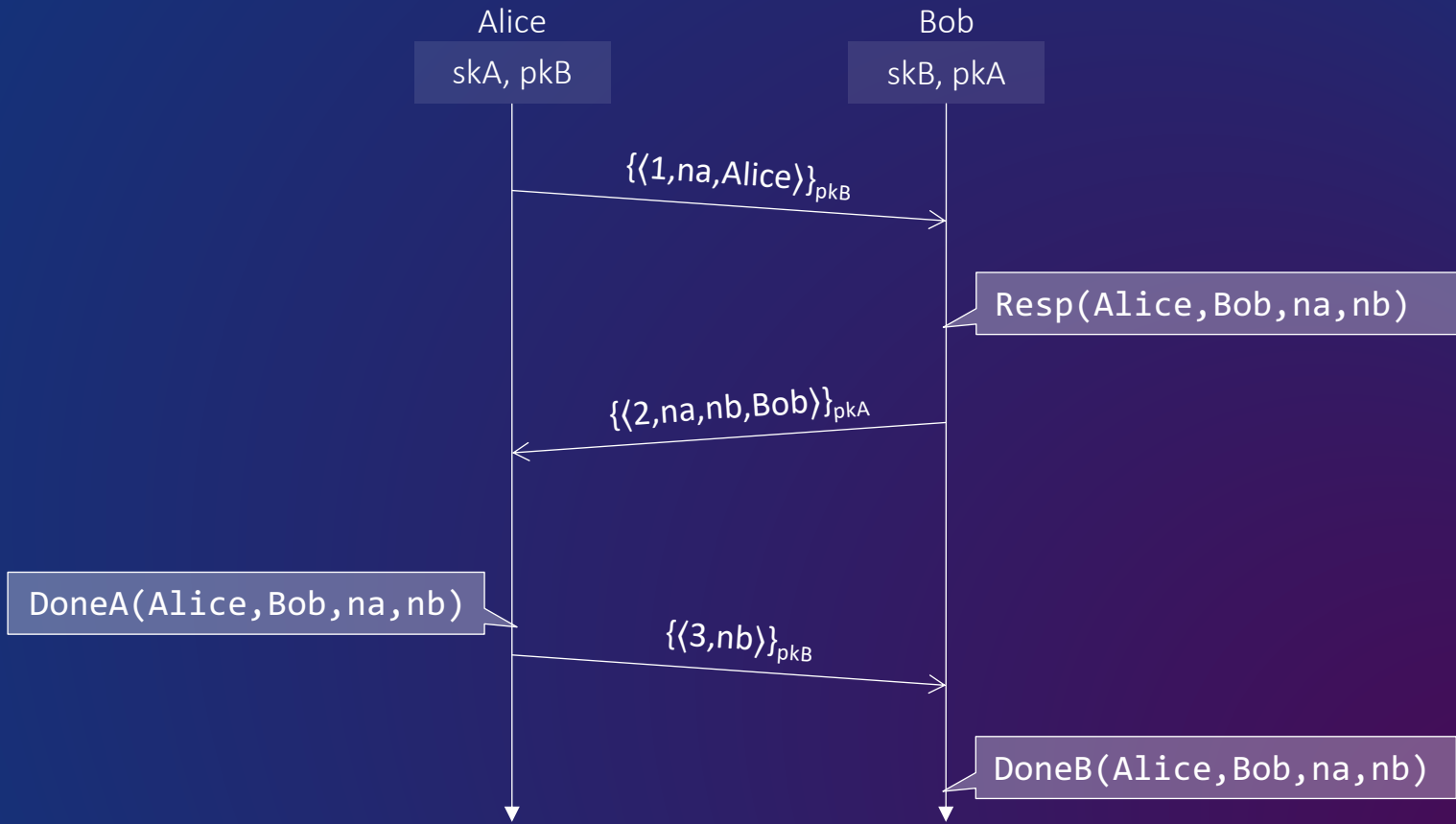


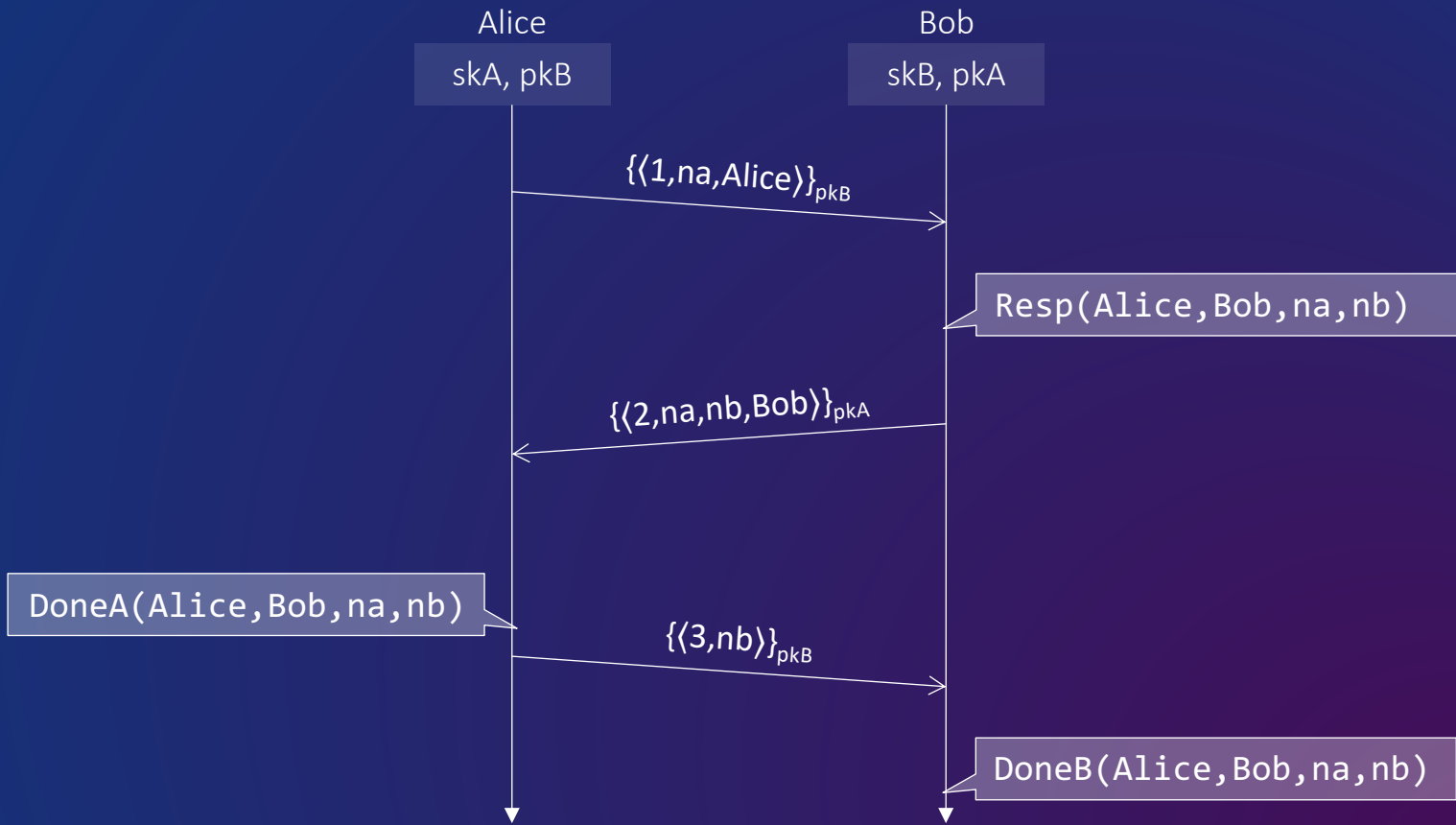
Approach



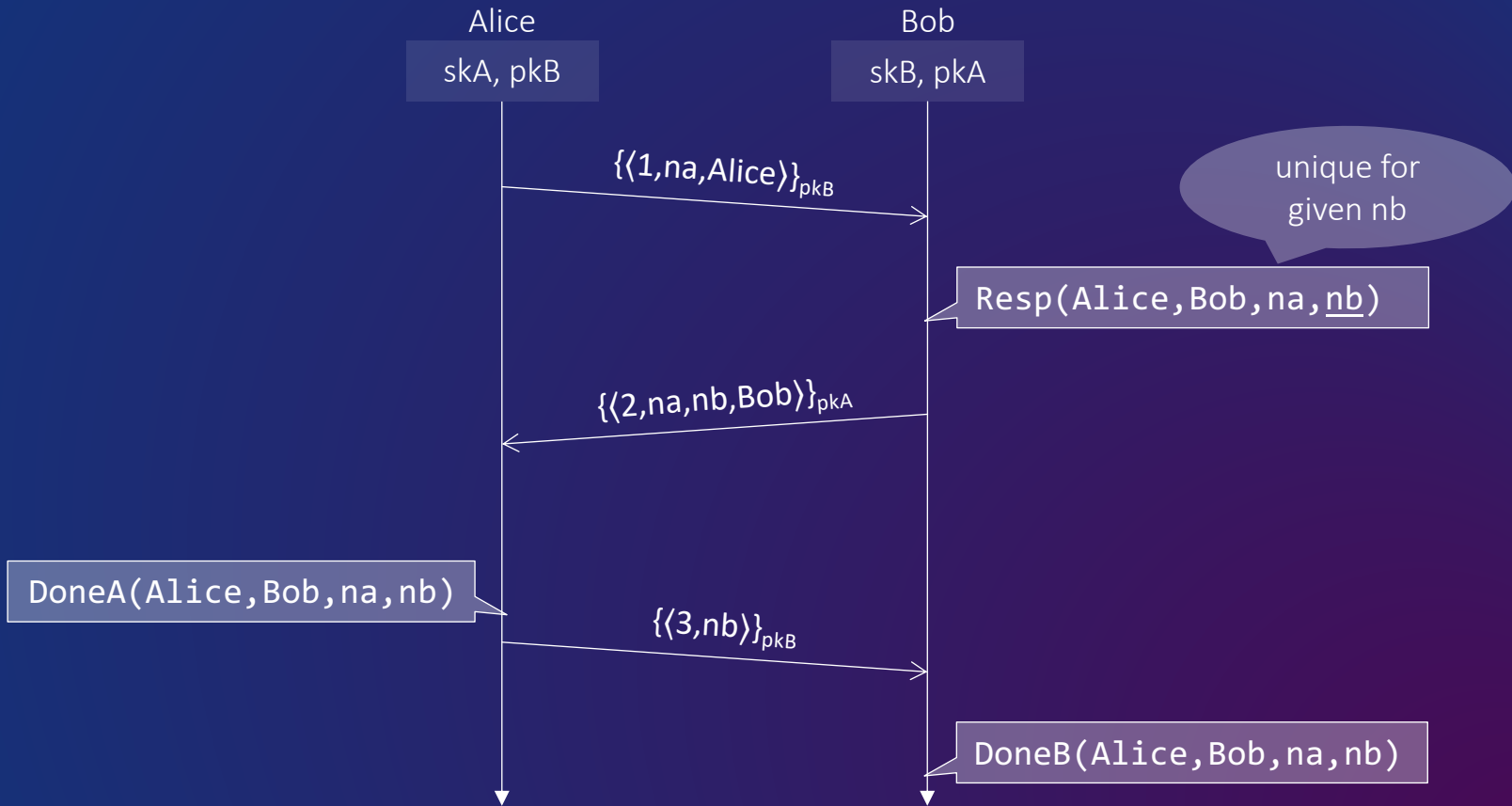


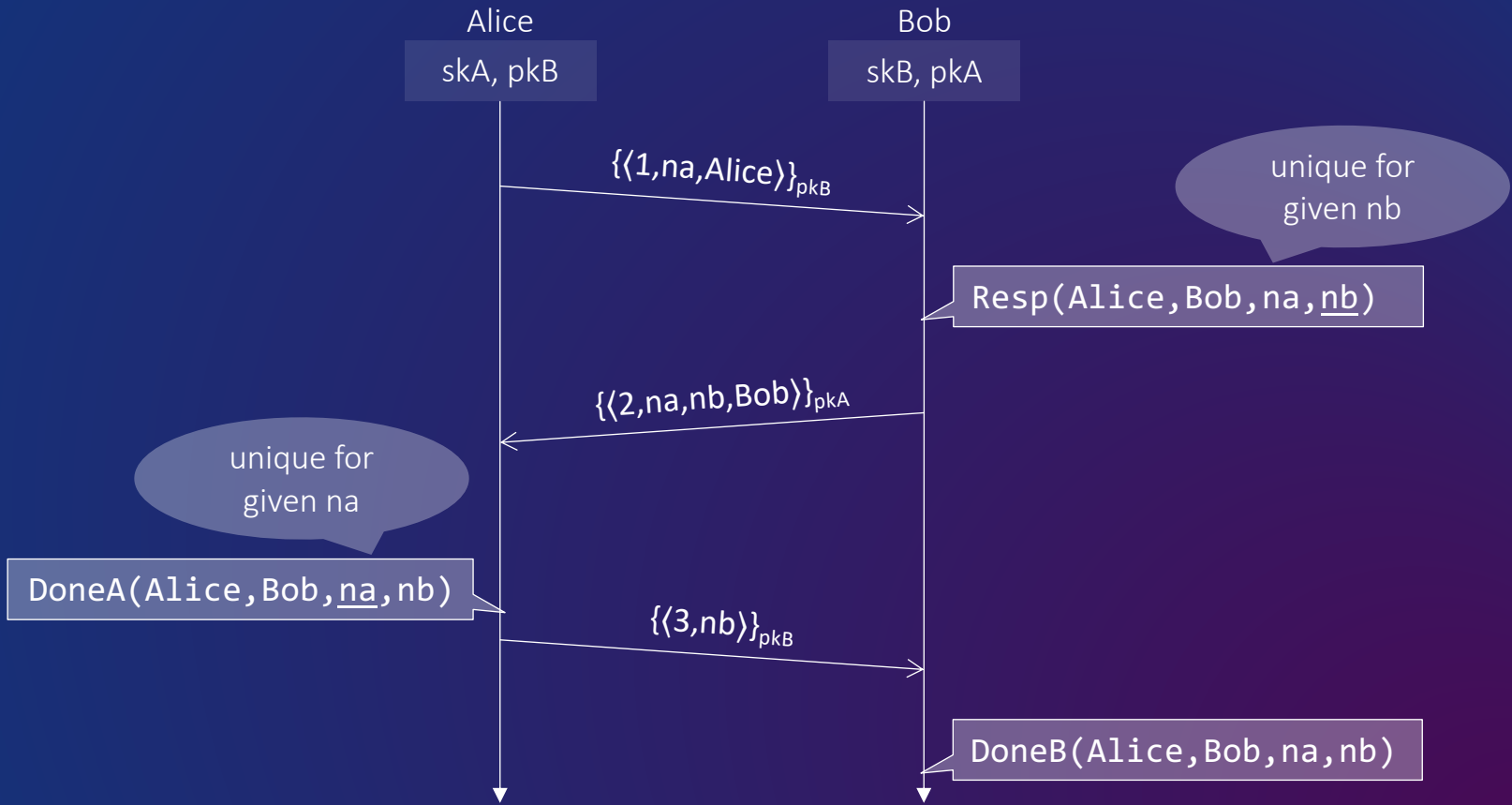


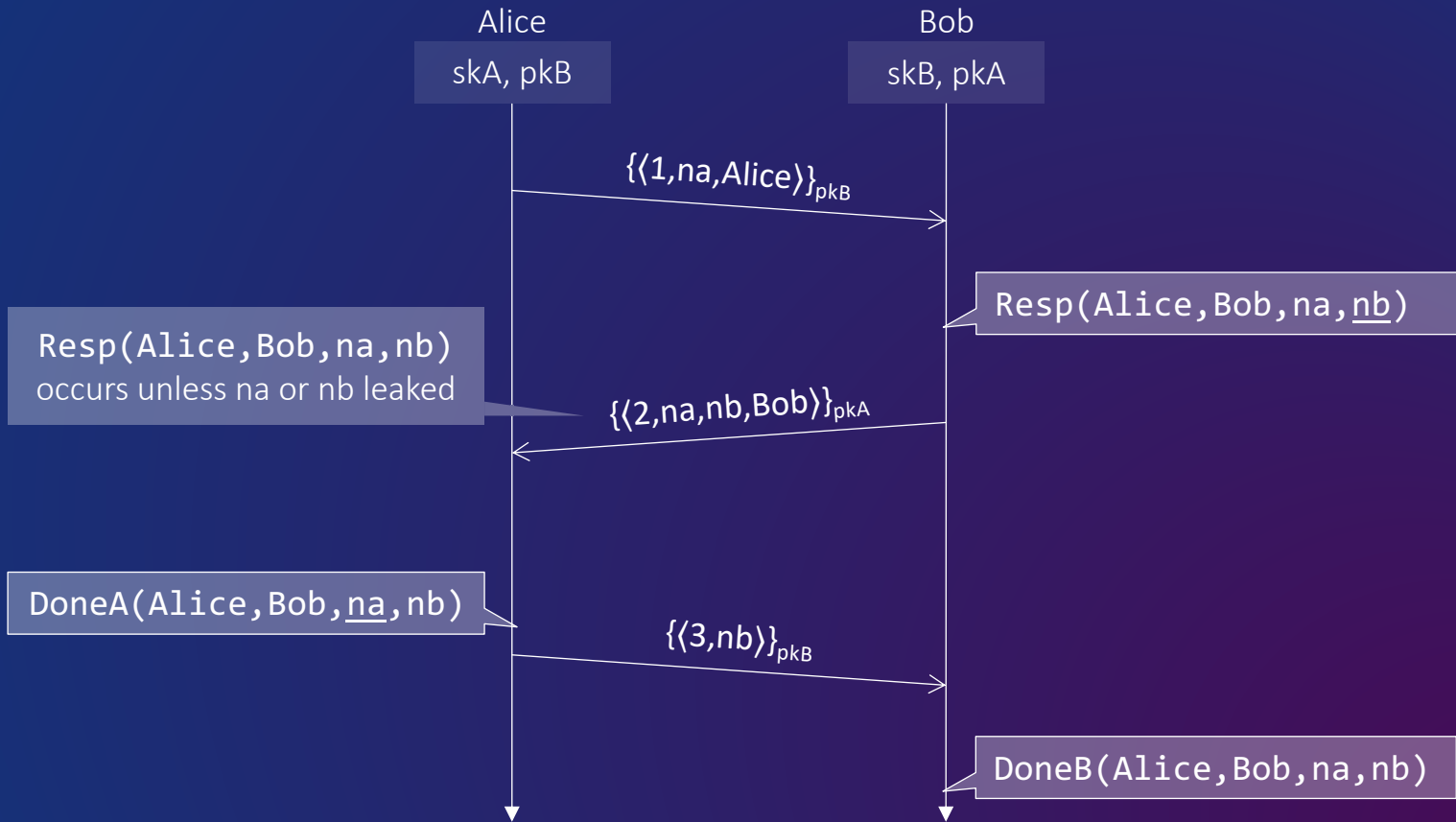


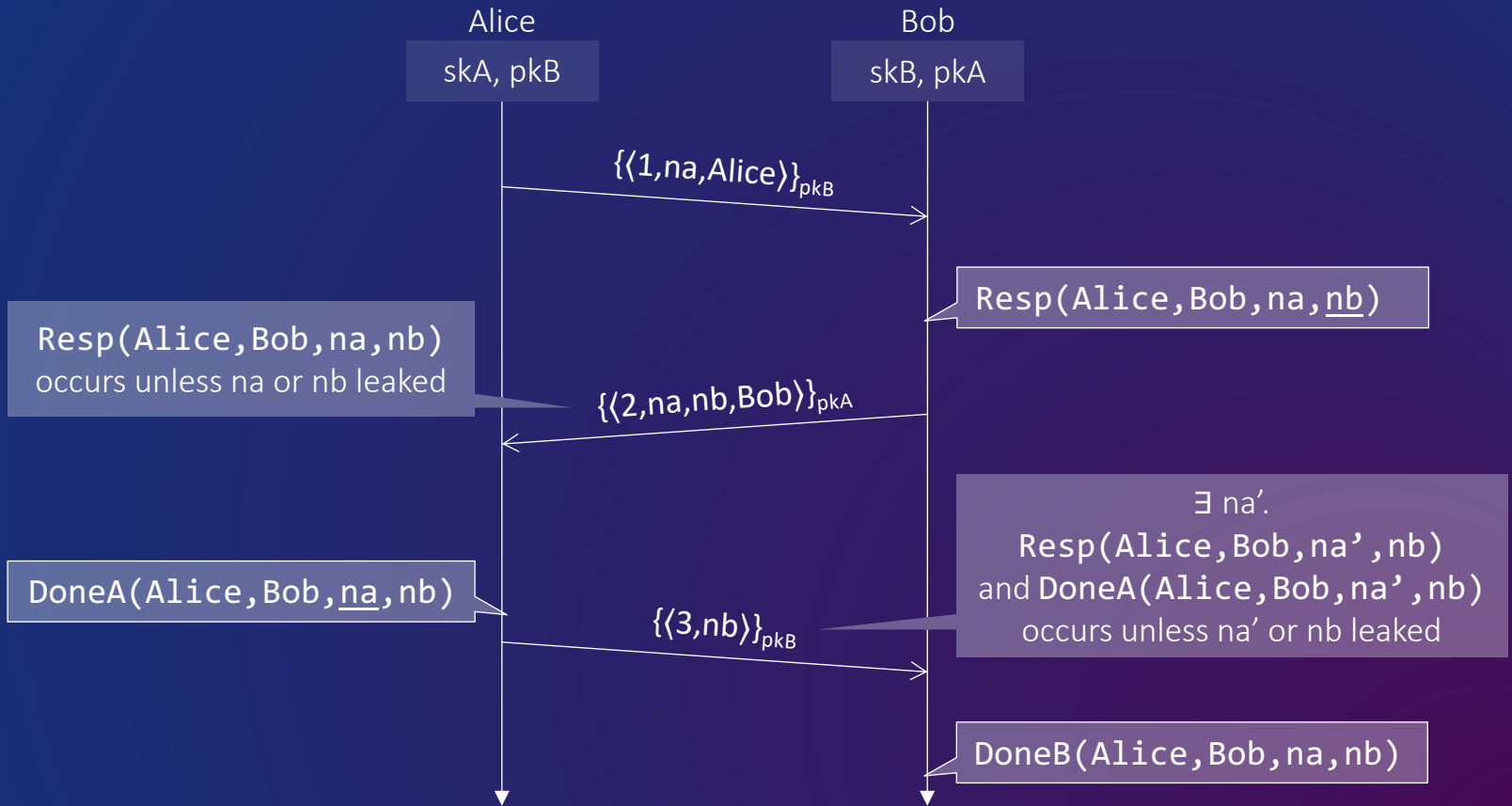


DoneB(Alice, Bob, na, nb) \Rightarrow Exactly one DoneA(Alice, Bob, na, nb)









```

func main(pkB []byte) {
  n := random()
  msg := enc(n, pkB)
  send(msg)
}

```

Alice

```

func send(msg) {

  io.send(msg)
}

```

Library



Global Trace \in Trace Invariant

+

I/O

```
func main(pkB []byte) {  
  n := random()  
  msg := enc(n, pkB)  
  send(msg)  
}
```

Alice

```
func send(msg) {  
  //@ trace.lock()  
  //@ trace.append(Send(msg))  
  //@ trace.unlock()  
  
  io.send(msg)  
}
```

Library



Global Trace \in Trace Invariant

+

I/O

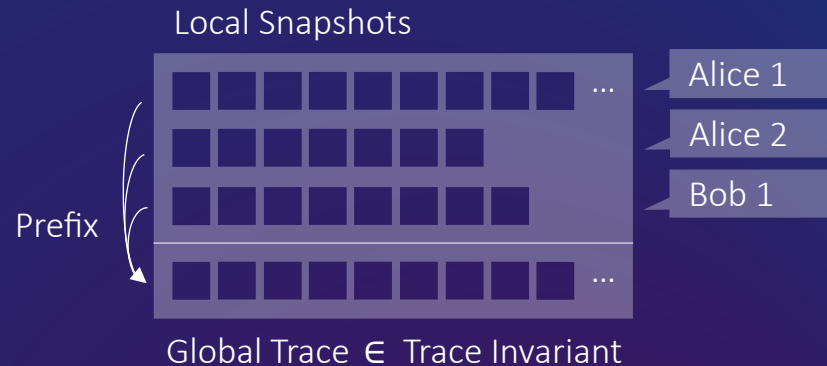
```
func main(pkB []byte) {
  n := random()
  msg := enc(n, pkB)
  send(msg)
}
```

Alice

```
func send(msg) {
  //@ trace.lock()
  //@ trace.append(Send(msg))
  //@ trace.unlock()

  io.send(msg)
}
```

Library



+

I/O

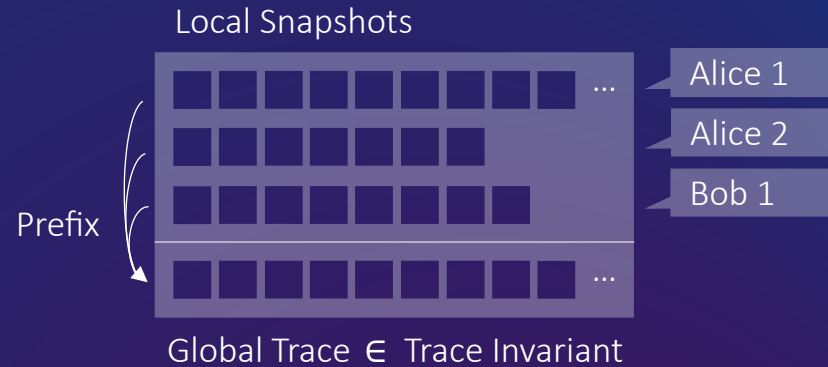
```
func main(pkB []byte) {
  n := random()
  msg := enc(n, pkB)
  send(msg)
}
```

Alice

```
func send(msg) {
  //@ trace.lock()
  //@ trace.append(Send(msg))
  //@ setSnap(trace)
  //@ trace.unlock()

  io.send(msg)
}
```

Library



+

I/O

```

func main(pkB []byte) {
  n := random()
  msg := enc(n, pkB)
  send(msg)
}

```

Alice

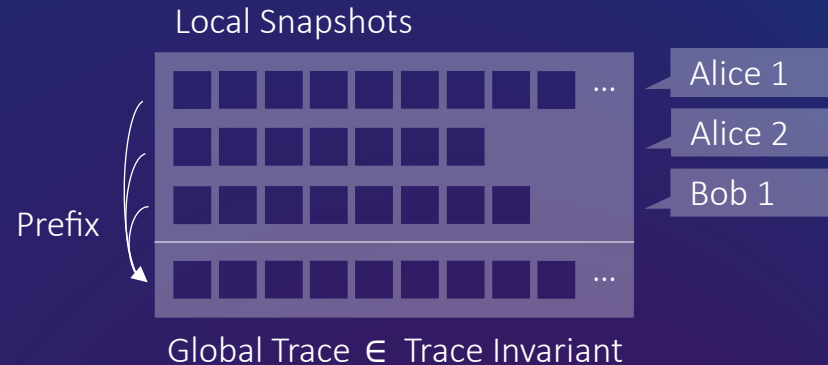
```

/*@ req msgInv(getSnap(), msg)
func send(msg) {
  //@ trace.lock()
  //@ trace.append(Send(msg))
  //@ setSnap(trace)
  //@ trace.unlock()

  io.send(msg)
}

```

Library



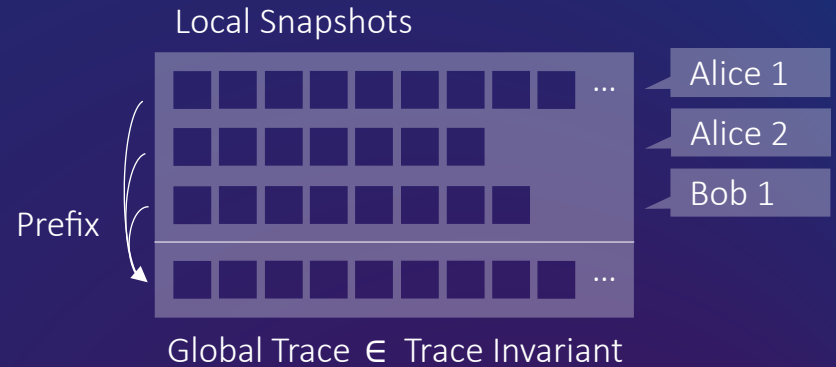
```
func main(pkB []byte) {
  n := random()
  msg := enc(n, pkB)
  send(msg)
}
```

Alice

```
//@ req msgInv(getSnap(), msg)
func send(msg) {
  //@ trace.lock()
  //@ trace.append(Send(msg))
  //@ setSnap(trace)
  //@ trace.unlock()

  io.send(msg)
}
```

Library



Parameterized

+

I/O

Evaluation

Verification libraries for C & Go

Evaluation

Verification libraries for C & Go

Library	LOC	LOS	[s]
Go / Gobra	83	6,932	126.1
C / VeriFast	343	3,837	0.8

Evaluation

Verification libraries for C & Go

NSL (in C & Go) and
signed DH key exchange (in Go)

Evaluation

Verification libraries for C & Go

WireGuard VPN protocol

NSL (in C & Go) and
signed DH key exchange (in Go)

Evaluation

Verification libraries for C & Go

WireGuard VPN protocol

NSL (in C & Go) and
signed DH key exchange (in Go)

WireGuard	LOC	LOS	[s]
Go / Gobra	557	5,815	268.1

Evaluation

Verification libraries for C & Go

WireGuard VPN protocol

NSL (in C & Go) and
signed DH key exchange (in Go)

Inj. agreement and forward secrecy
proven for WireGuard

Evaluation

Verification libraries for C & Go

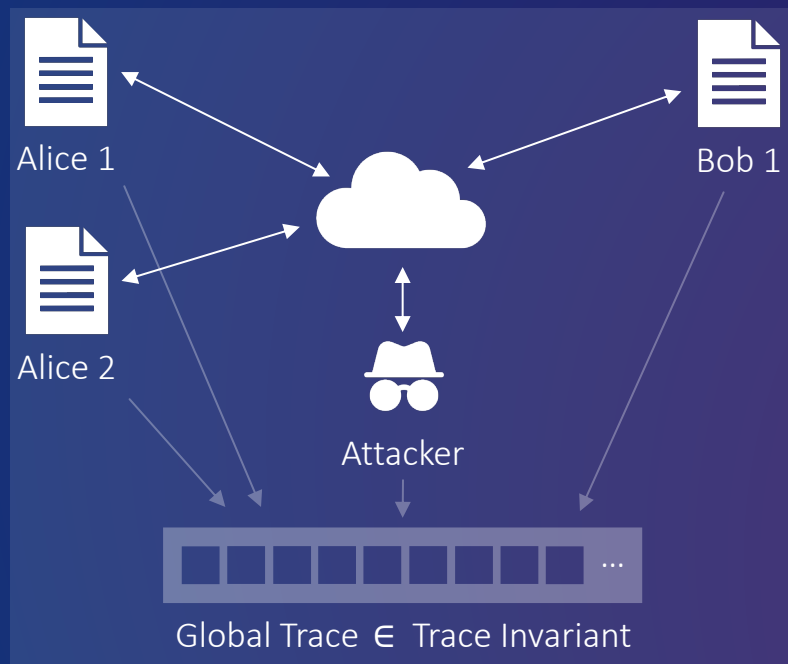
WireGuard VPN protocol

NSL (in C & Go) and
signed DH key exchange (in Go)

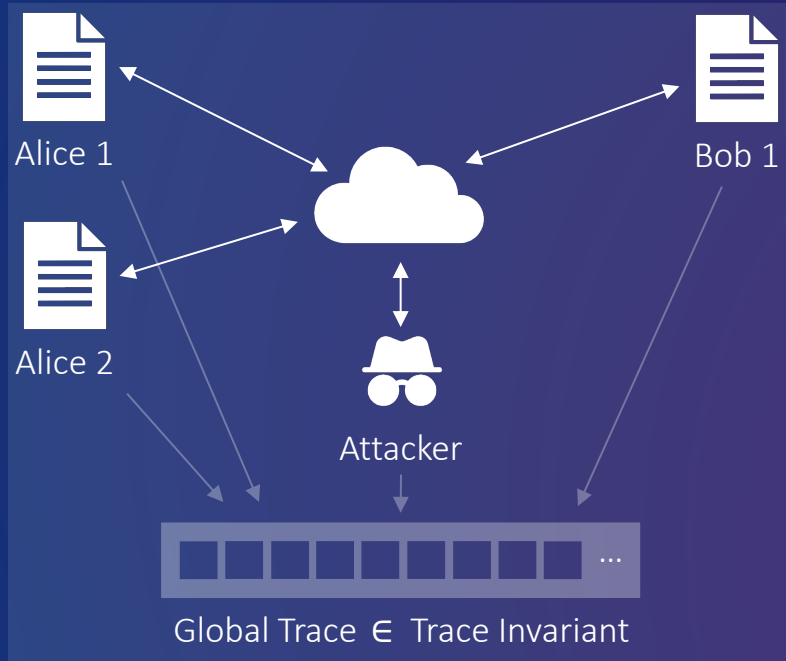
Inj. agreement and forward secrecy
proven for WireGuard

GitHub [viperproject/SecurityProtocolImplementations](https://github.com/viperproject/SecurityProtocolImplementations)

Conclusions

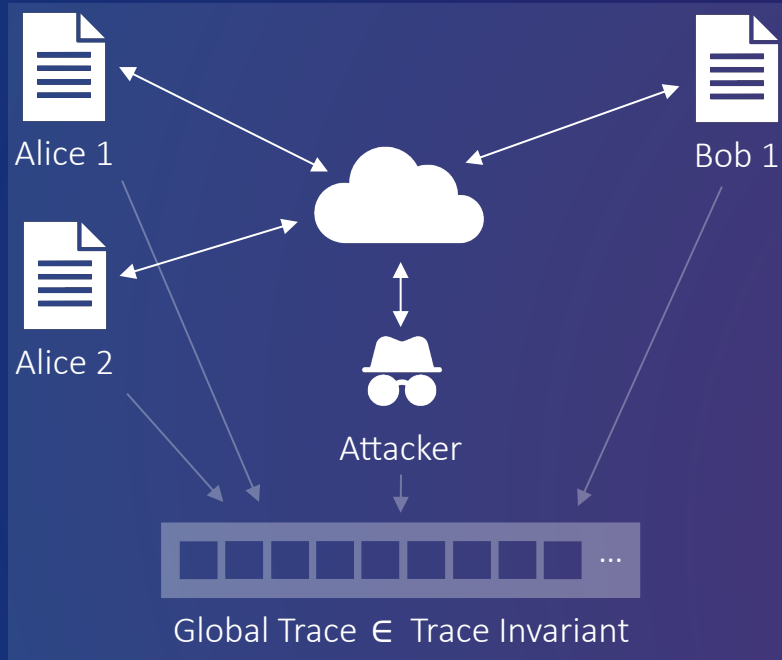


Conclusions



WireGuard

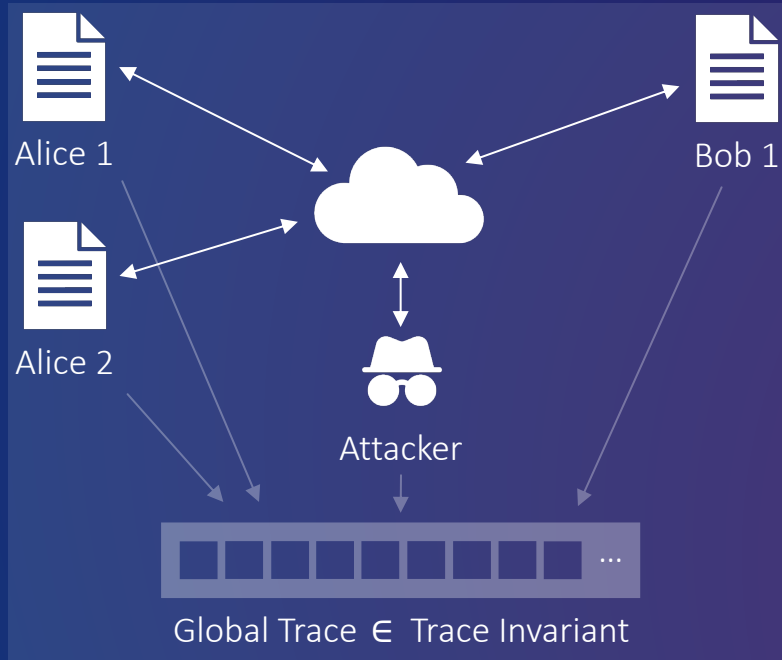
Conclusions



WireGuard

CCS '23: A Generic Methodology for the
Modular Verification of Security
Protocol Implementations

Conclusions



WireGuard

CCS '23: A Generic Methodology for the Modular Verification of Security Protocol Implementations

Soundness proof