

# "Risk" in an untrusted setting

Jude Southworth

February 18, 2023

# Risk

- ▶ *Risk* is a popular strategy board game.
- ▶ It is played on a single board, depicting a world map, partitioned into regions.
- ▶ A player owns a region of the map by stationing troops within the region.
- ▶ Players fight for regions by gambling some of their troops against the troops in the other player's region.



# Risk

- ▶ *Risk* has a variant called "fog of war".
- ▶ In this variant, players can only see the number of troops stationed within regions they neighbour.
- ▶ This variant is therefore only played online, in a **trusted setup**.



# Proposition

- ▶ Play fog-of-war Risk in an untrusted setup.
- ▶ In the untrusted setup, the same guarantees should be made as the trusted setup, but on a peer-to-peer network.

# Rationale

## ▶ **Decentralised**

- ▶ Longer lifespans than centralised platforms.
- ▶ More resistant to censorship and can help promote anonymity and privacy.
- ▶ Encourages user freedom.

## ▶ **Security**

- ▶ Constantly looking for ways to secure against threats specific to decentralised infrastructures.
- ▶ Security issues can be devastating even to decentralised infrastructures.

# State of the field

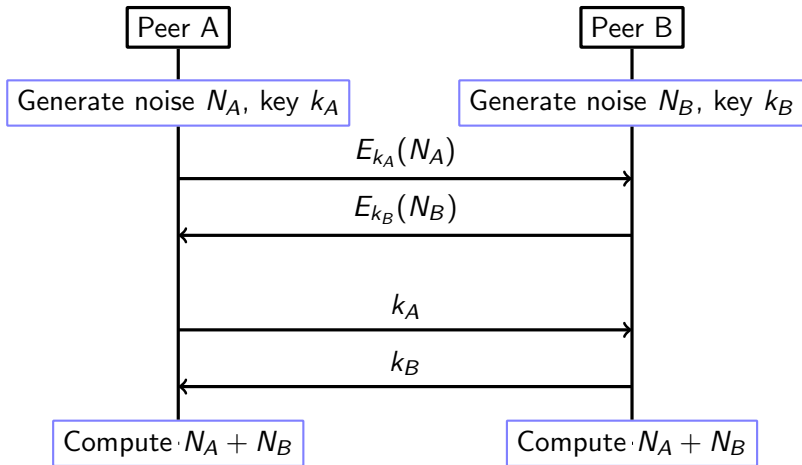
- ▶ Private key encryption and signatures.
- ▶ Additive homomorphic encryption.
- ▶ **Monero, Zcash.** Decentralised ledgers respectively using the *Bulletproof* and *ZK-SNARK* zero-knowledge proof systems.
  - ▶ ZKP protocols exist for proving properties of Paillier cyphertexts (Ivan Damgård, Mads Jurik,, Jesper Nielsen).
- ▶ **Web platform.** ECMAScript 2019+
  - ▶ Web Locks API.
  - ▶ BigInt API.

# Results

Emulated P2P environment using WebSockets.

# Results

Produce shared random values without beacons using commitment schemes.





# Results

Generating large primes using ECMAScript BigInt and Rabin-Miller.

```
function random2048() {  
  const byteArray = new BigUint64Array(32);  
  window.crypto.getRandomValues(byteArray);  
  let intRepr = 0n;  
  for (let int of byteArray) {  
    intRepr <<= 64n;  
    intRepr += int;  
  }  
  
  return intRepr;  
}
```

```
function generate_bigint() {  
  let intRepr = random2048();  
  
  // Drop the MSB to force into range from above  
  intRepr >>= 1n;  
  
  // Add  $2^{127}$  to force into range from below  
  intRepr += 2n ** 127n;  
  
  return intRepr;  
}
```

```
function generate_prime() {  
  while (true) {  
    let n = generate_bigint();  
    if (small_prime_test(n) && miller_rabin(n, 40)) {  
      return n;  
    }  
  }  
}
```

# Results

Implementation of the Paillier additive homomorphic cryptosystem.

```
> privKey
< PrivKey {n: 13479248814608379617357412543513248758050026676304...30369471323494566374238737650709040
4217397061389n, lambda: 13479248814608379617357412543513248758050026676304...46331745286785188426010
▶ 2326652688978669549016256n, mu: 57977246335538708756607021053336381801257879512114...874415729094208
169931129593876728854911101029285n}

> pubKey
< PubKey {n: 13479248814608379617357412543513248758050026676304...303694713234945663742387376507090404
▶ 217397061389n, g: 13479248814608379617357412543513248758050026676304...30369471323494566374238737650
7090404217397061390n}

> pubKey.encrypt(200n)
< 18042341624400104783941567284395799279527918352431...122879729467724499339573444479501353522496008180n

> pubKey.encrypt(200n)
< 52187291153685565605320097415042154992314375060706...408666894904348939306399591190681370117864191728n

> let c1 = pubKey.encrypt(100n)
< undefined

> let c2 = pubKey.encrypt(900n)
< undefined

> privKey.decrypt(c1)
< 100n

> privKey.decrypt(c2)
< 900n

> privKey.decrypt(c1 * c2)
< 1000n
```

# Results

Implementation of Risk.

## Next steps

- ▶ Zero-knowledge system for fog-of-war.
- ▶ Reduce issues of players halting play.
  - ▶ Handle disconnects.
- ▶ Check for modular biases.
- ▶ General analysis.

# Citations

*Image* Risk game board by CMG Lee, the asterisk denoting the missing link in the 40th Anniversary Collector's Edition, based on shapes from  
[http://commons.wikimedia.org/wiki/File:Risk\\_board.svg](http://commons.wikimedia.org/wiki/File:Risk_board.svg). 11  
November 2008. CC-BY-SA 4.0