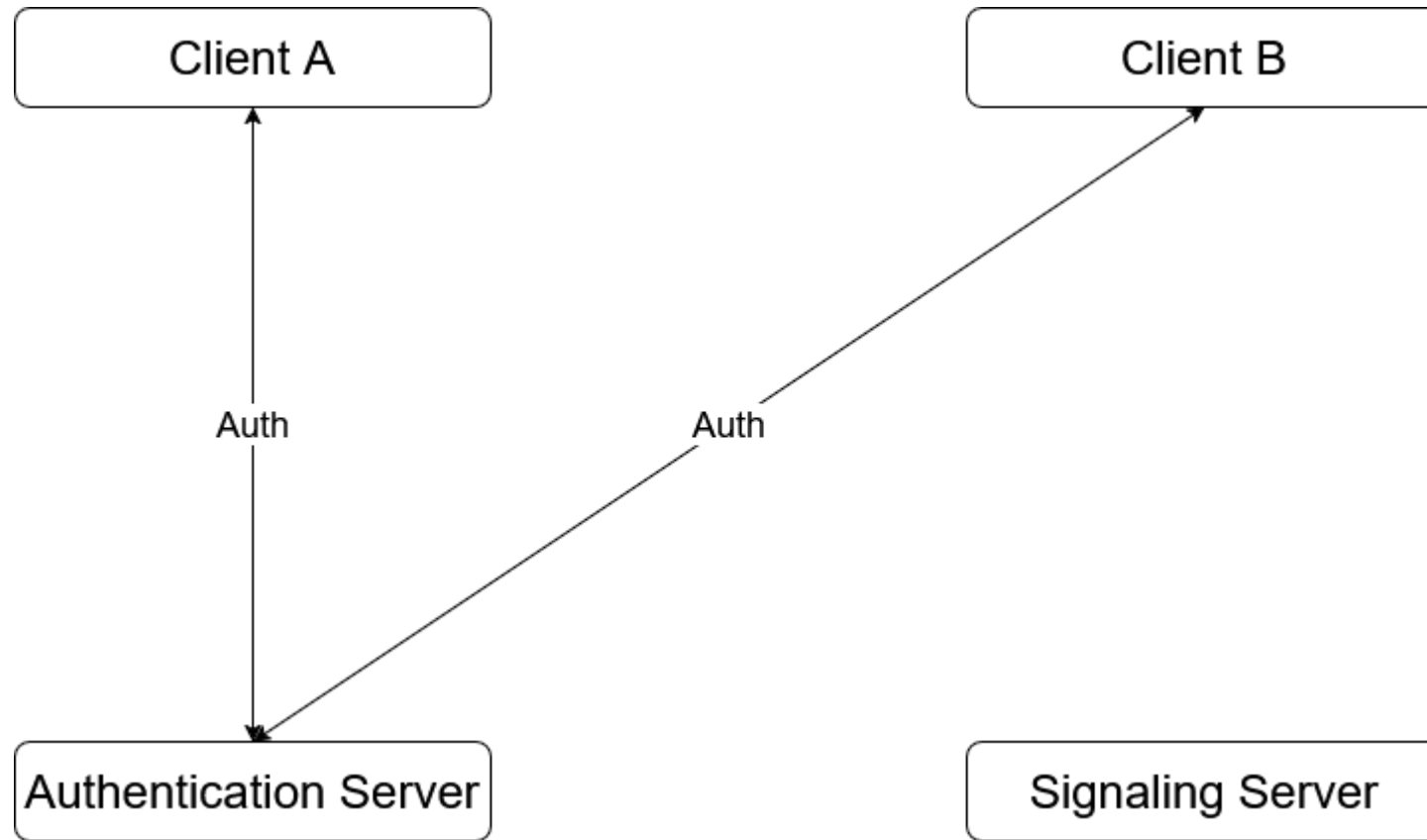




P2P E2EE messaging

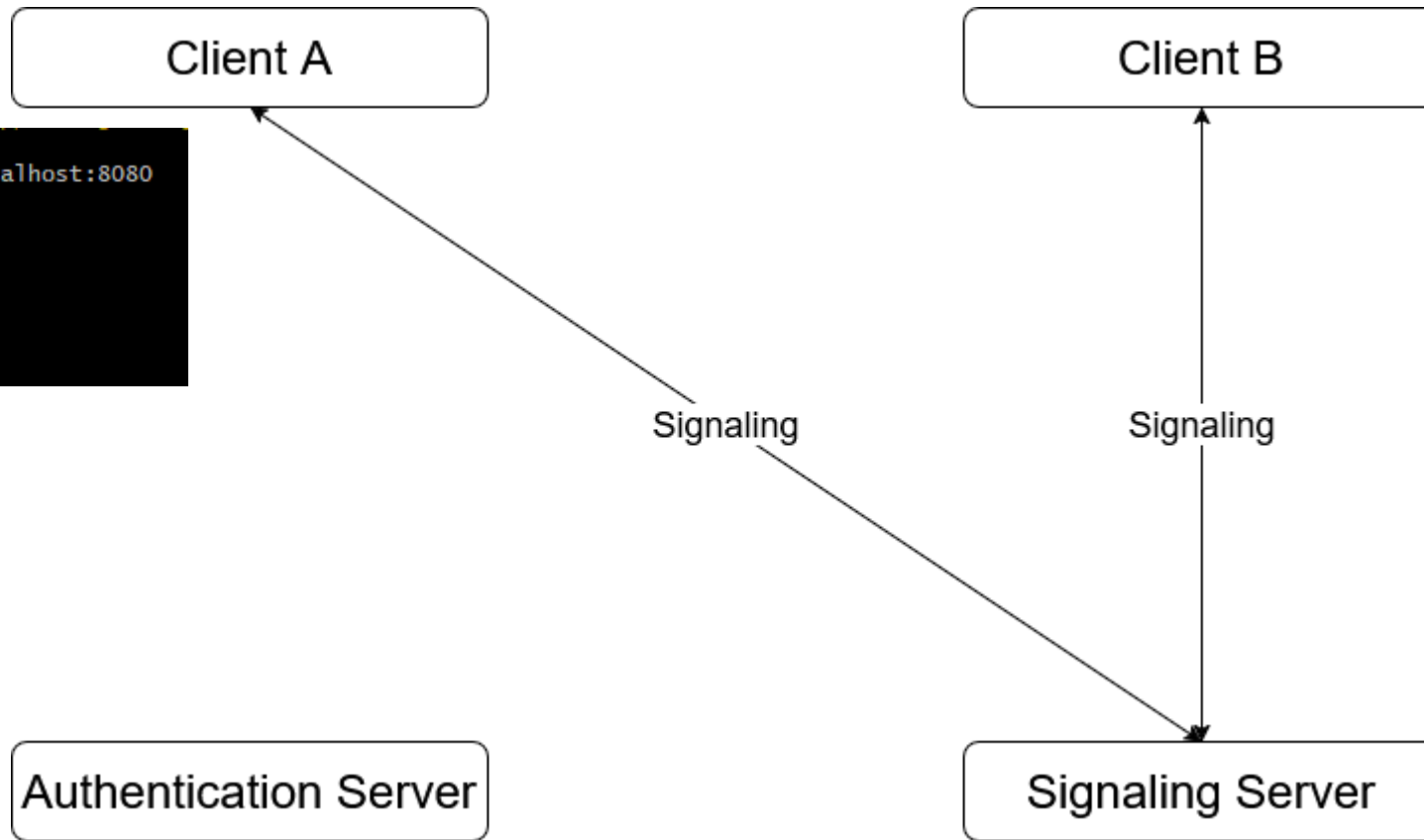
Joona Korkeamäki

Workflow



Token received: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkpvcyVYiIsImhhdCI6MTc0NjYxMzYwNywiZXBwIjozNzQ2NjE3MjA3fQ.0W8xC3nNqvtsHc0MoDvCuDooEmg9T5Xk5hgDmfIgCwk [script.js:37:13](#)
Connected to signaling server! [script.js:41:31](#)

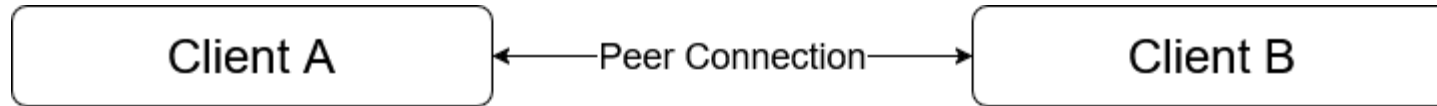
Workflow



```
$ node signaling.js
signaling server running on wss://localhost:8080
New connection attempt
Bob connected
New connection attempt
Alice connected
New connection attempt
Invalid token, closing connection.
Bob disconnected
```

```
PeerConnection created
Received answer
Remote answer set successfully.
Received ice-candidate
Successfully added ICE candidate.
Received ice-candidate
Successfully added ICE candidate.
Received ice-candidate
Successfully added ICE candidate.
Data channel open!
```

Workflow



```
origmsg: Moi
msg:   ▶ Uint8Array(3) [ 77, 111, 105 ]
symkey:
Local derived key (length: 32): ▶ Array(32) [ 161, 175, 132, 245, 21, 116, 123, 244, 152, 114, ... ]
data:   ▶ ArrayBuffer { byteLength: 19 }
encrypted: ▼ Object { iv: (12) [...], data: (19) [...] }
  ▶ data: Array(19) [ 102, 17, 94, ... ]
  ▶ iv: Array(12) [ 158, 182, 45, ... ]
  ▶ <prototype>: Object { ... }
```

Authentication Server

```
▼ Object { type: "encryptedMessage", iv: (12) [...], data: (19) [...] }
  ▶ data: Array(19) [ 102, 17, 94, ... ]
  ▶ iv: Array(12) [ 158, 182, 45, ... ]
  type: "encryptedMessage"
  ▶ <prototype>: Object { ... }
data:   ▶ ArrayBuffer { byteLength: 19 }
symkey:
Local derived key (length: 32): ▶ Array(32) [ 161, 175, 132, 245, 21, 116, 123, 244, 152, 114, ... ]
Decryption:
  ▶ ArrayBuffer { byteLength: 3 }
Received message: Moi
```

Signaling Server

Technologies and frameworks

Node.js

Express.js

WebRTC

Web Crypto Api

Jenkins

Cryptography

- ECDH
- AES-GCM
- Session forward secrecy
- JSON WebToken
- Self signed ssl

```
async function encryptMessage(message) {
  console.log("origmsg: ", message)
  const encoder = new TextEncoder();
  const encodedMessage = encoder.encode(message);
  const iv = window.crypto.getRandomValues(new Uint8Array(12)); // Initialization vector
  console.log("msg: ", encodedMessage)

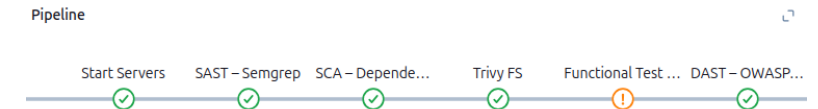
  console.log("symkey: ")
  await logSymmetricKey(mySymmetricKey, "Local derived key");

  const encrypted = await window.crypto.subtle.encrypt(
    { name: "AES-GCM", iv },
    mySymmetricKey,
    encodedMessage
  );
  console.log("data: ", encrypted)

  return { iv: Array.from(iv), data: Array.from(new Uint8Array(encrypted)) };
}
```

Testing





- DevsecOps pipeline



✓ #16 (Apr 23, 2025, 3:27:57 PM)



Build Artifacts

 npm-audit.json	361 B	view
 semgrep-output.json	597.11 KiB	view
 trivy-fs.json	567 B	view
 zap-report.html	71.45 KiB	view

Future Improvements

- Message-level forward secrecy
- Functional tests to the devsecops pipeline
- Offline message queueing.
- Public key identities.
- Ssl from let's encrypt/CA

AI Acknowledgements

- Debugging
- Jenkins pipeline setup
- Html