

ö-ö.net: einfach sicher chatten
(crypto-secured HTML5 p2p-live-chat client)

Andreas Briese

Eine Unternehmung der eduToolbox@Bri-C GmbH, Sarstedt

25. März 2014

Index

Connecting to ö-ö.net

Alice connect

Bob connect

Diffie-Hellman-Handshake

prime, public keys & sessKey/cipherKey

code Sample: dhKey

code Sample: dhExchange

Textmessages

CryptoJS.rabbit, l(ese)key & s(end)Key

message format

Filetransmissions

Salsa20/20

Salsa20/20

message format

Server ping



einfach

sicher

chatten

[http\(s\)://ö-ö.net](http(s)://ö-ö.net)

first client connect (Alice)

Alice GET https://xn- - -0gab.net/chat ¹

ö-ö → ö-ö webApp: ≈150kb html/css/js/img/audio

Alice Math.random() = seedrandom() ← Salsa20/20 PRNG
seeded with 256 bytes from (browser)crypto.random()
upgrade http(s) → websocket ws(s)

ö-ö js to eval: send sha256(webApp script-parts at Alice)

Alice sha256(script-parts at Alice) →

ö-ö ? sha256(delivered scripts) === sha256(script-parts at Alice)

NO → ws(s) close / del from register

YES → enable ws(s)-receive function

YES → "server connect – waiting 4 partner"

YES → URL to connect with Alice

¹(punicode for ö-ö.net: xn- - -0gab.net)

Alice websocket frames

(opera 20 developer tools)

Name	Data	Length	Time
chat	...	622	10:48:21
data:audio/ogg;base...	...	622	10:48:21
data:image/png;base...	1256#NrD#CEYGYA5wjgCwFYBsAGANA65mQOzj7QbjgKLFZgaiTholjqQpj006Vpd1EJlrGf8CKAjrRcNCXrQok/JLj5j/dBQq8i8HTlxJKCckmZQcBqm8UpeEjn...	1261	10:48:21
data:image/png;base...	1252#NrDsE48ZwDIUAMCA0BaAjJgCvhtgZigFzI1tpKDjJIN0dNRILszwF108jTvisQYXppMBYjHQwi6ZMQkFQ3LmnmkTQkQZAjoYzAumLpDyXFwJjo7UJnC2CG...	1257	10:48:21
chatSocket	...	622	10:48:20
data:image/png;base...	...	622	10:48:20
data:image/png;base...	&&&	3	10:48:20
data:image/jpeg;base...	&&&	3	10:48:20
	???https://A?-A?-net/7361kuk	28	10:47:46
	server connect -- waiting 4 partner	36	10:47:46
	websocket.onmessage=onMessage;	31	10:47:46
	b10a5a9ef97fed1e6041c37ee434f13d096dd26fe343de7624cb52e772554528	65	10:47:46
	websocket.send(function(scripts){return sha1recv+btolscripts.join("").replace(/\\/g,"\\")})(function(){var i=0,t=Stag("script"),ttl=[];for(i<t.length;Xttl,push...	209	10:47:46

connect: last five lines in grafik
white: incoming ; green: outgoing

connect of Alice partner (Bob)

Bob GET https://xn- - -0gab.net/?Alice

ö-ö ? (Alice in websocket register)

NO → "ws is not registered"

YES (Alice ↔ Bob) ↓ open-register: rm Alice

YES ö-ö webApp → Bob

Bob Math.random() = seedrandom() ← Salsa20/20 PRNG

seeded with 256 bytes from (browser)crypto.random()

upgrade http(s) → websocket ws(s)

ö-ö js to eval: send sha256(webApp script-parts at Bob)

Bob sha256(script-parts at Bob) →

ö-ö ? sha256(delivered scripts) === sha256(script-parts at Bob)

NO → ws(s) close / del from register

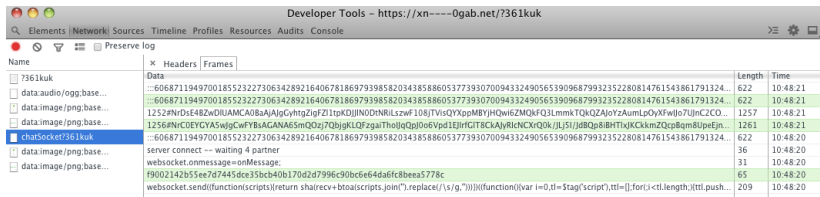
YES → enable ws(s)-receive function

YES → "server connect – waiting 4 partner"

Bob

websocket frames

(opera 20 developer tools)



connect: last four lines in grafik
white: incoming ; green: outgoing

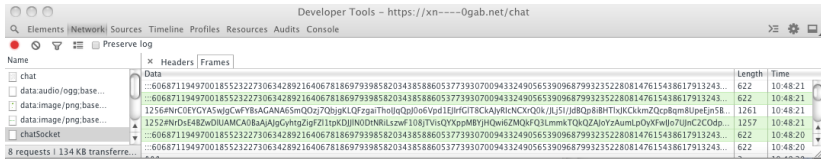
... Alice & Bob connected and approved ...

- ö-ö $\text{rand}_{1..1000}$ 2056bit save prime ($g=2$) \rightarrow Alice
 Alice 2056bit save prime \rightarrow Bob
 Bob ?(prime bitsize===2056) \rightarrow calc Bob pubKey
 Alice ?(prime bitsize===2056) \rightarrow calc Alice pubKey
 Bob $\text{RABBIT}_{\text{sha}(\text{siteKey})+\text{sha}(\text{prime}+\text{recv})}(\text{pubKey}) \rightarrow$ Alice
 Alice $\text{RABBIT}_{\text{sha}(\text{siteKey})+\text{sha}(\text{prime}+\text{recv})}(\text{pubKey}) \rightarrow$ Bob
 Bob ? (pubKey bitsize $>$ 370 && pubKey $<$ prime-2)
 NO \rightarrow kill session
 YES \rightarrow calc sessKey/cipherKey
 \Rightarrow prime \rightarrow Alice
 Alice ? (pubKey bitsize $>$ 370 && pubKey $<$ prime-2)
 NO \rightarrow kill session
 YES \rightarrow calc sessKey/cipherKey
 \Rightarrow prime \rightarrow Bob

Alice (top) & Bob

Diffie-Hellman-Handshake websocket frames

(opera 20 developer tools)



Developer Tools - https://xn----0gab.net/chat

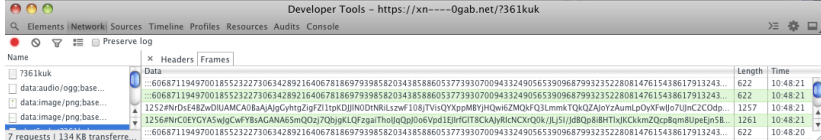
Elements | Network | Sources | Timeline | Profiles | Resources | Audits | Console

Preserve log

Name: chat

Headers: Data

Name	Data	Length	Time
chat	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:21
data.audio/ogg;base...	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:21
data.image/png;base...	1256#NrD#COEYCYASwjgCwFYBSAGANA6SmQOzj70bjgKLQFzgaiThollqQpJ0o6Vpd1EJlRfGIT8CkAjrRlcNCXrQ0k/JLjSi/jdBQp8iBHtIkJCkkmZQcpBqm8UpeEjn5B...	1261	10:48:21
data.image/png;base...	1252#NrDsE48ZwDIUAMCA0BaAJgCyhtgZigfZi1tpkDJJIN0DtNRILszwf108jTVisQYXppMBYjHQw6ZMQkFQ3LmmkTQkQZAJozAumLpOyXfwljo7UjnC2COdp...	1257	10:48:21
chatSocket	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:20
8 requests 134 KB transferred	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:20



Developer Tools - https://xn----0gab.net/7361kuk

Elements | Network | Sources | Timeline | Profiles | Resources | Audits | Console

Preserve log

Name: 7361kuk

Headers: Data

Name	Data	Length	Time
7361kuk	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:21
data.audio/ogg;base...	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:21
data.image/png;base...	1252#NrDsE48ZwDIUAMCA0BaAJgCyhtgZigfZi1tpkDJJIN0DtNRILszwf108jTVisQYXppMBYjHQw6ZMQkFQ3LmmkTQkQZAJozAumLpOyXfwljo7UjnC2COdp...	1257	10:48:21
data.image/png;base...	1256#NrD#COEYCYASwjgCwFYBSAGANA6SmQOzj70bjgKLQFzgaiThollqQpJ0o6Vpd1EJlRfGIT8CkAjrRlcNCXrQ0k/JLjSi/jdBQp8iBHtIkJCkkmZQcpBqm8UpeEjn5B...	1261	10:48:21
7 requests 134 KB transferred	:::6068711949700185523227306342892164067818697939858203438588605377393070094332490565390968799323522808147615438617913243...	622	10:48:20

white: incoming; green: outgoing

function dhKeys(sessPrime)² ³

(github.com/AndreasBriese/oe-oe.net/oeoe.js)

```
672. function dhKeys(sPrime){
673.     if(!warten()){
674.         try{
675.             sessPrime = str2bigInt(sPrime, 10, 200);
676.             if(bitSize(sessPrime)<2056){
677.                 if(websocket.readyState===websocket.OPEN){
678.                     websocket.send('!!!DISCONNECT!');
679.                     websocket.close();
680.                     alert('illegal DH param')
681.                 }
682.             }
683.             sessSecret = int2bigInt(0, 1);
684.             for(;(bitSize(sessSecret)<370&&(greater(sessPrime, sub(sessSecret,int2bigInt(2, 1))))));{
685.                 sessSecret = randBigInt(primeBits, 0);
686.             }
687.             var sessGen = powMod(int2bigInt(2, 1), sessSecret, sessPrime);
688.             publicKey = "---" + bigInt2str(sessGen, 10);
689.             // console.log(sPrime, publicKey);
690.             return true;
691.         }catch(e){
692.             throw e;
693.             return false;
694.         }
695.     }
696. }
```

² algorithm from <http://tools.ietf.org/html/rfc4253section-8>

³ www.bsi.bund.de "Kryptographische Verfahren: Empfehlungen und Schlüssellaengen" BSI TR-02102-1

function dhExchange(publKey)⁴ ⁵

(github.com/AndreasBriese/oe-oe.net/oeoe.js)

```
698 · function dhExchge(pKey){
699 ·     var partnerKey = str2bigInt(pKey, 10, 200);
700 ·     if(greater(int2bigInt(2, 1), partnerKey) || greater(partnerKey, sub(sessPrime, int2bigInt(2, 1)))){
701 ·         if(websocket.readyState===websocket.OPEN){
702 ·             websocket.send('!!!DISCONNECT!');
703 ·             websocket.close()
704 ·         }
705 ·         alert("Unsichere Schluesselsequenz! Sitzung wird abgebrochen!");
706 ·     }
707 ·     cipherKey = powMod(partnerKey, sessSecret, sessPrime);
708 ·     sessKey = sha(bigInt2str(cipherKey, 10));
709 ·     //console.log("ourSecret:", cipherKey);
710 ·     var l = cipherKey.length-2;
711 ·     var ba = new ArrayBuffer(2*l);
712 ·     var ab = new Uint16Array(ba);
713 ·     for(;l;ab[--l]=cipherKey[l]);
714 ·     cipherKey = new Uint8Array(ba);
715 ·     //console.log("compare:", cipherKey, ab);
716 ·     sessSecret = null;
717 ·     websocket.send(":::"+bigInt2str(sessPrime,10));
718 ·     if(!cllr) nickPopUP();
719 ·     return true;
720 · }
721 ·
```

⁴ algorithm from <http://tools.ietf.org/html/rfc4253section-8>

⁵ www.bsi.bund.de "Kryptographische Verfahren: Empfehlungen und Schlüssellaengen" BSI TR-02102-1

text-encryption using CryptoJS.rabbit

- * CryptoJS v3.1.2
- * code.google.com/p/crypto-js
- * (c) 2009-2013 by Jeff Mott. All rights reserved.
- * code.google.com/p/crypto-js/wiki/License

usage text-en/decryption with encryption-secret \leftarrow sKey/lKey

sKey sKey = sha256(last incoming message number + salt from last incoming textmessage) + sessKey

lKey lKey = sha256(outgoing message number + salt from outgoing textmessage) + sesskey

textmessage message format

length	separator	encrypted payload (compressed & base64 encoded string)
1260	#	NrCME4CYGYBYFZKQGywDSoOy..

Salsa20/20 stream-encryption

- * Salsa 20
- * unified from <http://code.ohloh.net/file?fid=v-0xBPS2xdNtbDPzjFZXCY89n0Acid=vPv6llgR6co>
- * Contribution to Cryptocat by Dmitry Chestnykh
21-01-2013
- * Snippet had been analysed and tested against D.J.Bernsteins original C-Implementation
- * at <http://cr.yip.to/snuffle.html>
(<http://cr.yip.to/snuffle/salsa20/ref/salsa20.c>)
- * ABriese 2014/20/02

Salsa20/20 stream-encryption (ff.)

send-protokoll:

1. calc cipherKey start index: $\text{rand}_{0\dots256}$
 2. seed salsa20 \leftarrow 32 bytes / 8 bytes (256 bits) from the cipherKey (30 bytes of the cipherKey reserved and never used)
 3. file \rightarrow blob
 4. XOR each byte against 8bit from salsa20 PRNG
 5. compose sendfile
 6. sendfile \rightarrow POST <https://ö-ö.net/upload>
- ö-ö incoming bytes \Rightarrow receiver-websocket

file transmission

sendfile message format

→ POST <https://ö-ö.net/upload>

65 bytes plain text	1 byte num	120 bytes ____ Salsa20/20 encrypted payload ____	70 bytes	file blob data
addressee	start index	file name	file format	blob
vh7jcv	212	encrypted payload _____		

server checks

messenger websocket responding (ping)

The server checks responsiveness of open (but unconnected) websockets by sending: "&&&" (3 Bytes)