

# From Technofeudalism to Technosocialism: FORTH unexpected roles in Decentralised Systems

## Part A: Introduction

We began this presentation with a brief description of a FORTH like engine in Bitcoin (Part B) and how the invisible internet project (Part B), being a crucial alternative to the domain name system, was used as the backbone of Bitcoin but remains relatively obscure.

We then extended the concept of Bitcoin address, as the hash of public key, to Omnihash user identifier and Omnihash JSON, being a generalised representation of digital assets with Decentralised ownerships. (Part C)

We further propose to employ a FORTH like engine in shell libraries for JavaScript and Java applications (Part D: Trojan Horse Super-Shell), in order to consolidate fragmented user base of open source projects and small commercial social media applications to compete with and create one to one replacements for mainstream dominant social media platforms (Goal of Decentralised Systems).

## From Technofeudalism to Technosocialism: FORTH unexpected roles in Decentralised Systems

Technofeudalism is a term coined by former Greek finance minister Yanis Varoufakis as the title of a book which compares top American technology companies to feudal lords, who literally enslaved individual users and free software programmers, exploiting their data and source code without fairly rewarding them.

Technosocialism has been used by many other authors but I use it to describe the very real but underreported progress made by free software programmers and free software, despite the lack of fair mechanisms for rewards so far. Free software programmers or technosocialists have made technofeudalism possible as a matter of fact, but they do not get to enjoy the rewards.

Bitcoin and related projects were supposed to address this problem, and are successful in their own way, so we are now half way towards technosocialism.

Part B:

FORTH-like Engine in Bitcoin  
+ Invisible Internet Project



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406 ✓  bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
407     {
408         static const CScriptNum bnZero(0);
409         static const CScriptNum bnOne(1);
410         // static const CScriptNum bnFalse(0);
411         // static const CScriptNum bnTrue(1);
412         static const valtype vchFalse(0);
413         // static const valtype vchZero(0);
414         static const valtype vchTrue(1, 1);
415
416         // sigversion cannot be TAPROOT here, as it admits no script execution.
417         assert(sigversion == SigVersion::BASE || sigversion == SigVersion::WITNESS_V0 || sigversion == SigV
418
419         CScript::const_iterator pc = script.begin();
420         CScript::const_iterator pend = script.end();
421         CScript::const_iterator pbegincodehash = script.begin();
422         opcodetype opcode;
423         valtype vchPushValue;
424         ConditionStack vfExec;
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
435
436     try
437     {
438         for ( ; pc < pend; ++opcode_pos) {
439             bool fExec = vfExec.all_true();
440
441             //
442             // Read instruction
443             //
444             if (!script.GetOp(pc, opcode, vchPushValue))
445                 return set_error(error, SCRIPT_ERR_BAD_OPCODE);
446             if (vchPushValue.size() > MAX_SCRIPT_ELEMENT_SIZE)
447                 return set_error(error, SCRIPT_ERR_PUSH_SIZE);
448
449             if (sigversion == SigVersion::BASE || sigversion == SigVersion::WITNESS_V0) {
450                 // Note how OP_RESERVED does not count towards the opcode limit.
451                 if (opcode > OP_16 && ++nOpCount > MAX_OPS_PER_SCRIPT) {
452                     return set_error(error, SCRIPT_ERR_OP_COUNT);
453             }
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
```

```
if (opcode == OP_CAT ||
    opcode == OP_SUBSTR ||
    opcode == OP_LEFT ||
    opcode == OP_RIGHT ||
    opcode == OP_INVERT ||
    opcode == OP_AND ||
    opcode == OP_OR ||
    opcode == OP_XOR ||
    opcode == OP_2MUL ||
    opcode == OP_2DIV ||
    opcode == OP_MUL ||
    opcode == OP_DIV ||
    opcode == OP_MOD ||
    opcode == OP_LSHIFT ||
    opcode == OP_RSHIFT)
    return set_error(error, SCRIPT_ERR_DISABLED_OPCODE); // Disabled opcodes (CVE-2010-5137).
```

```
// With SCRIPT_VERIFY_CONST_SCRIPTCODE, OP_CODESEPARATOR in non-segwit script is rejected even in an unexecu
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
483             switch (opcode)
484             {
485                 //
486                 // Push value
487                 //
488                 case OP_1NEGATE:
489                 case OP_1:
490                 case OP_2:
491                 case OP_3:
492                 case OP_4:
493                 case OP_5:
494                 case OP_6:
495                 case OP_7:
496                 case OP_8:
497                 case OP_9:
498                 case OP_10:
499                 case OP_11:
500                 case OP_12:
501                 case OP_13:
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
515             //
516             // Control
517             //
518             case OP_NOP:
519                 break;
520
521             case OP_CHECKLOCKTIMEVERIFY:
522             {
523                 if (!(flags & SCRIPT_VERIFY_CHECKLOCKTIMEVERIFY)) {
524                     // not enabled; treat as a NOP2
525                     break;
526                 }
527
528                 if (stack.size() < 1)
529                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
530
531                 // Note that elsewhere numeric opcodes are limited to
532                 // operands in the range -2**31+1 to 2**31-1, however it is
533             }
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
601
602         case OP_IF:
603         case OP_NOTIF:
604     {
605             // <expression> if [statements] [else [statements]] endif
606             bool fValue = false;
607             if (fExec)
608             {
609                 if (stack.size() < 1)
610                     return set_error(serror, SCRIPT_ERR_UNBALANCED_CONDITIONAL);
611                 valtype& vch = stacktop(-1);
612                 // Tapscript requires minimal IF/NOTIF inputs as a consensus rule.
613                 if (sigversion == SigVersion::TAPSCRIPT) {
614                     // The input argument to the OP_IF and OP_NOTIF opcodes must be either
615                     // exactly 0 (the empty vector) or exactly 1 (the one-byte vector with value
616                     if (vch.size() > 1 || (vch.size() == 1 && vch[0] != 1)) {
617                         return set_error(serror, SCRIPT_ERR_TAPSCRIPT_MINIMALIF);
618                     }
619                 }
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
635
636             case OP_ELSE:
637             {
638                 if (vfExec.empty())
639                     return set_error(error, SCRIPT_ERR_UNBALANCED_CONDITIONAL);
640                 vfExec.toggle_top();
641             }
642             break;
643
644             case OP_ENDIF:
645             {
646                 if (vfExec.empty())
647                     return set_error(error, SCRIPT_ERR_UNBALANCED_CONDITIONAL);
648                 vfExec.pop_back();
649             }
650             break;
651
652             case OP_VERIFY:
653             {
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
673             // 
674             // Stack ops
675             //
676             case OP_TOALTSTACK:
677             {
678                 if (stack.size() < 1)
679                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
680                 altstack.push_back(stacktop(-1));
681                 popstack(stack);
682             }
683             break;
684
685             case OP_FROMALTSTACK:
686             {
687                 if (altstack.size() < 1)
688                     return set_error(error, SCRIPT_ERR_INVALID_ALTSTACK_OPERATION);
689                 stack.push_back(altstacktop(-1));
690                 popstack(altstack);
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
693
694         case OP_2DROP:
695     {
696             // (x1 x2 -- )
697             if (stack.size() < 2)
698                 return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
699             popstack(stack);
700             popstack(stack);
701     }
702     break;
703
704     case OP_2DUP:
705     {
706         // (x1 x2 -- x1 x2 x1 x2)
707         if (stack.size() < 2)
708             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
709         valtype vch1 = stacktop(-2);
710         valtype vch2 = stacktop(-1);
711         stack.push_back(vch1);
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
407
408
409     case OP_2DUP:
410     {
411         // (x1 x2 -- x1 x2 x1 x2)
412         if (stack.size() < 2)
413             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
414
415         valtype vch1 = stacktop(-2);
416         valtype vch2 = stacktop(-1);
417         stack.push_back(vch1);
418         stack.push_back(vch2);
419
420     }
421
422     break;
423
424
425     case OP_3DUP:
426     {
427         // (x1 x2 x3 -- x1 x2 x3 x1 x2 x3)
428         if (stack.size() < 3)
429             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
430
431         valtype vch1 = stacktop(-3);
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
730             case OP_2OVER:
731             {
732                 // (x1 x2 x3 x4 -- x1 x2 x3 x4 x1 x2)
733                 if (stack.size() < 4)
734                     return set_error(serror, SCRIPT_ERR_INVALID_STACK_OPERATION);
735                 valtype vch1 = stacktop(-4);
736                 valtype vch2 = stacktop(-3);
737                 stack.push_back(vch1);
738                 stack.push_back(vch2);
739             }
740             break;
741
742             case OP_2ROT:
743             {
744                 // (x1 x2 x3 x4 x5 x6 -- x3 x4 x5 x6 x1 x2)
745                 if (stack.size() < 6)
746                     return set_error(serror, SCRIPT_ERR_INVALID_STACK_OPERATION);
747                 valtype vch1 = stacktop(-6);
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
755             case OP_2SWAP:
756             {
757                 // (x1 x2 x3 x4 -- x3 x4 x1 x2)
758                 if (stack.size() < 4)
759                     return set_error(serror, SCRIPT_ERR_INVALID_STACK_OPERATION);
760                 swap(stacktop(-4), stacktop(-2));
761                 swap(stacktop(-3), stacktop(-1));
762             }
763             break;
764
765             case OP_IFDUP:
766             {
767                 // (x - 0 | x x)
768                 if (stack.size() < 1)
769                     return set_error(serror, SCRIPT_ERR_INVALID_STACK_OPERATION);
770                 valtype vch = stacktop(-1);
771                 if (CastToBool(vch))
772                     stack.push_back(vch);
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
765             case OP_IFDUP:
766             {
767                 // (x - 0 | x x)
768                 if (stack.size() < 1)
769                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
770                 valtype vch = stacktop(-1);
771                 if (CastToBool(vch))
772                     stack.push_back(vch);
773             }
774             break;
775
776             case OP_DEPTH:
777             {
778                 // -- stacksize
779                 CScriptNum bn(stack.size());
780                 stack.push_back(bn.getvch());
781             }
782             break;
783
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
783
784         case OP_DROP:
785     {
786             // (x -- )
787             if (stack.size() < 1)
788                 return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
789             popstack(stack);
790     }
791     break;
792
793     case OP_DUP:
794     {
795         // (x -- x x)
796         if (stack.size() < 1)
797             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
798         valtype vch = stacktop(-1);
799         stack.push_back(vch);
800     }
801     break;
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
407
408     case OP_NIP:
409     {
410         // (x1 x2 -- x2)
411         if (stack.size() < 2)
412             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
413         stack.erase(stack.end() - 2);
414     }
415     break;
416
417     case OP_OVER:
418     {
419         // (x1 x2 -- x1 x2 x1)
420         if (stack.size() < 2)
421             return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
422         valtype vch = stacktop(-2);
423         stack.push_back(vch);
424     }
425     break;
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
822             case OP_PICK:
823             case OP_ROLL:
824             {
825                 // (xn ... x2 x1 x0 n - xn ... x2 x1 x0 xn)
826                 // (xn ... x2 x1 x0 n - ... x2 x1 x0 xn)
827                 if (stack.size() < 2)
828                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
829                 int n = CScriptNum(stacktop(-1), fRequireMinimal).getint();
830                 popstack(stack);
831                 if (n < 0 || n >= (int)stack.size())
832                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
833                 valtype vch = stacktop(-n-1);
834                 if (opcode == OP_ROLL)
835                     stack.erase(stack.end()-n-1);
836                 stack.push_back(vch);
837             }
838             break;
839
840         case OP_ROT:
```



master ▾

bitcoin / src / script / interpreter.cpp

↑ Top

Code

Blame

Raw



```
406     bool EvalScript(std::vector<std::vector<unsigned char> >& stack, const CScript& script, unsigned int fl
840             case OP_ROT:
841             {
842                 // (x1 x2 x3 -- x2 x3 x1)
843                 // x2 x1 x3 after first swap
844                 // x2 x3 x1 after second swap
845                 if (stack.size() < 3)
846                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
847                 swap(stacktop(-3), stacktop(-2));
848                 swap(stacktop(-2), stacktop(-1));
849             }
850             break;
851
852             case OP_SWAP:
853             {
854                 // (x1 x2 -- x2 x1)
855                 if (stack.size() < 2)
856                     return set_error(error, SCRIPT_ERR_INVALID_STACK_OPERATION);
857                 swap(stacktop(-2), stacktop(-1));
858             }
```

## Part B: Invisible Internet Project

Section page, summary here, highlight words, give addresses

bitcoin/src/i2p.cpp at master · GitHub



← → ⌂ gitcoin.com/bitcoin/bitcoin/blob/master/src/i2p.cpp

Finish update : L



bitcoin / bitcoin



Code

Issues 370

Pull requests 270

Actions

Projects 5

Security



master ▾

bitcoin / src / i2p.cpp



Go to file



achow101 Merge #29833: i2p: fix and improve logs



b27afb7 · 5 months ago



494 lines (414 loc) · 15.4 KB

Code

Blame

Raw



```
1 // Copyright (c) 2020-2022 The Bitcoin Core developers
2 // Distributed under the MIT software license, see the accompanying
3 // file COPYING or http://www.opensource.org/licenses/mit-license.php.
4
5 #include <chainparams.h>
```



master ▾

bitcoin / src / i2p.cpp

↑ Top

Code

Blame

Raw



```
87     * @throw std::runtime_error if conversion fails
88     */
89     static CNetAddr DestBinToAddr(const Binary& dest)
90     {
91         CSHA256 hasher;
92         hasher.Write(dest.data(), dest.size());
93         unsigned char hash[CSHA256::OUTPUT_SIZE];
94         hasher.Finalize(hash);
95
96         CNetAddr addr;
97         const std::string addr_str = EncodeBase32(hash, false) + ".b32.i2p";
98         if (!addr.SetSpecial(addr_str)) {
99             throw std::runtime_error(strprintf("Cannot parse I2P address: \"%s\"", addr_str));
100        }
101
102        return addr;
```



Omni\*Chat -- Liang Ng -- 2024-04-27

AnyDesk 768086256

Free license (non-professional use). Start trial license.

Activities BiglyBT Sab 13 Apr, 11:29

Anon - 224455

DHT=0, Nodes=1/1/0, Requests=0.2/0.2

[01:16] EFAA53  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"13332.0"]  
[09:34] EE6A59  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"13332.0"]  
[01:53] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:7777 2222+d2s:dbl put:je:  
[01:53] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"9999.0"]  
[09:34] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"9999.0"]  
[04:12] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:7777 2266+d2s:dbl put:je:  
[04:12] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"10043.0"]  
[09:34] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"10043.0"]  
[09:19] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:1199 2266+d2s:dbl put:je:  
[09:21] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"3465.0"]  
[09:34] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"3465.0"]  
[09:22] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:1199 220066+d2s:dbl put:je:  
[09:22] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"221265.0"]  
[09:34] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"221265.0"]  
[09:34] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:1199 2200+d2s:dbl put:je:  
[09:34] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"3399.0"]  
[11:27] BDF12C  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:1126 2200+d2s:dbl put:je:  
[11:27] EEA559  
ECHO ["a":"b","k":"j","d":"c","n":"m","dbl":"3326.0"]  
OMNI:jo:gc:dup:hm:b:a put:c:d put:j:k put:m:n put:1129 2200+d2s:dbl put:je:

File View Community Tools Window Help

Open Torrents Find Torrents...

Dashboard Decentralized Chat Friends

Use independent windows for pop-out. Force pop-out windows to the side bar

My Torrents Library New Active Inactive Paused

Notifications

Play a sound when a message is received. Use the IP Filter to filter messages and notifications

Private Chat

Enable private chat Only accept chats from pinned participants

Integration

Make my friend key automatically available Consider setting up your public profile (see here)

Content Discovery Swarm Discover... Subscriptions Chat Overview

Devices In Progress Disk Internet

Plugins & Extras Friends

Advanced Share anonymous torrent and chat destination Unfortunately using separate destinations

Test

Public beta chat Open

Anonymous beta chat (I2P) Open

Create/join custom channel: key 224455

Advanced Share anonymous torrent and chat destination Unfortunately using separate destinations is currently unreliable

Test

SSR-2024-04-13-11.29.04.mkv 4:03 / 24:32 • Introduction >

Please visit here for details

Anon - 1

Liang PC

## Using PC

ests=0.2/0.2, refs=3: '224455'

```
d put: j k put: m n put: 7777 5555 + d2s: dbl put: je:  
{"m": "dbl": "13332.0"}  
{"m": "dbl": "13332.0"}
```

```
d put:j k put:m n put:7777 2222 + d2s:dbl put:je:  
:"m","dbl":"9999.0"}  
,"m","dbl":"9999.0"}]
```

```
d put: j k put: m n put: 7777 2266 + d2s: dbl put: je:  
{"m": "dbl": "10043.0"}  
--> null
```

```
d put: j k put: m n put: 1199 2266 + d2s: dbl put: je:  
:"m","dbl":"3465.0"}]
```

```
d put:j k put:m n put:1199 220066 + d2s: dbl put:je  
:"m","dbl":"221265.0"}  
}
```

www.gutenberg.org/cache/epub/22126/pg22126.html

09-341 EE6459

`CHO["a","b","c","d","e","f","m","dbl":"3399.0"]`

DMN | jo: gc: dup: hm: b a put: c d put: j k put: m n put: 1126 2200 + d2s: dbl put: je:

11

-0

## i2p.i2p / core / java / src / net / i2p / data / Hash.java Top

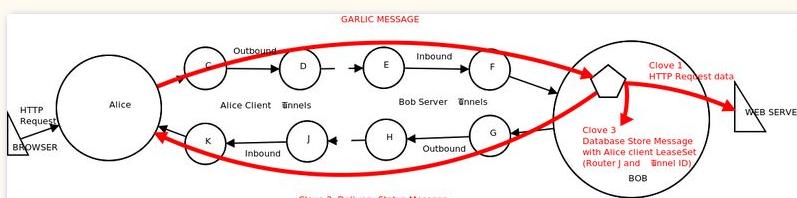
Code Blame

```
21     public class Hash extends SimpleDataStructure {  
106         }  
107  
108         /**  
109             * For convenience.  
110             * @return "{52 chars}.b32.i2p" or null if data not set.  
111             * @since 0.9.25  
112             */  
113     public String toBase32() {  
114         if (_data == null)  
115             return null;  
116         return Base32.encode(_data) + ".b32.i2p";  
117     }  
118  
119         /**  
120             * @since 0.9.17
```

## End-to-End Message Bundling

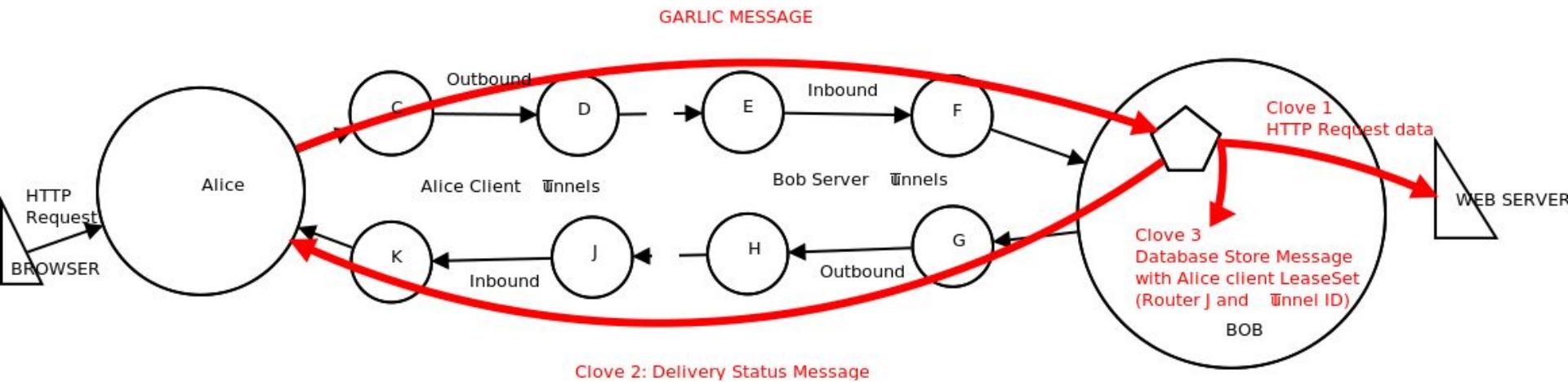
At the layer above tunnels, I2P delivers end-to-end messages between Destinations. Just as within a single tunnel, we use ElGamal/AES+SessionTag for the encryption. Each client message as delivered to the router through the I2CP interface becomes a single Garlic Clove with its own Delivery Instructions, inside a Garlic Message. Delivery Instructions may specify a Destination, Router, or Tunnel.

Generally, a Garlic Message will contain only one clove. However, the router will periodically bundle two additional cloves in the Garlic Message:



1. A Delivery Status Message, with Delivery Instructions specifying that it be sent back to the originating router as an acknowledgment. This is similar to the "reply block" or "reply onion" described in the references. It is used for determining the success or failure of end to end message delivery. The originating router may, upon failure to receive the Delivery Status Message within the

# <https://geti2p.net/en/docs/how/garlic-routing>



- Email
- Torrents
- Configuration**
  - Address Book Help
  - Hidden Services Manager
  - Settings
  - Setup
- Diagnostics**
  - Graphs Logs NetDB Peers
  - Profiles Tunnels
- Help & FAQ**

- Changelog
- FAQ
- Licenses
- Network
- Sidebar
- Troubleshoot

**Peers**

Active:	21/119
Fast:	30
High Capacity:	20
Floodfill:	93
Known:	172

**Tunnels**

Exploratory:	4
Client:	21
Participating:	0
Share Ratio:	0.00

I2P Hidden Services						
Name	Type	Points at	Preview	Status	Control	
<b>A01 Omni*Web HTTPS</b>	Standard server	127.0.0.1:443	No Preview		<b>Stop</b>	
		<b>Destination:</b> yo6sgmfq7pfvvp2e4kcuhjtfg7wfllt63igwcukhbmuqm6lu3a3a.b32.i2p				
		<b>Description:</b> A01 Omni*Web HTTPS				
<b>HTTP Phos tunnel</b>	HTTP server	127.0.0.1:80	<b>Preview</b>		<b>Stop</b>	
		<b>Destination:</b> weor7pxsuxpkkbvh4sgta7hvoom5jrzemvy253y5norcamadujaq.b32.i2p				
		<b>Description:</b> HTTP Phos tunnel				
<b>I2P HTTPS Tunnel</b>	HTTP server	127.0.0.1:7668	<b>Preview</b>		<b>Stop</b>	
		<b>Destination:</b> 6fjdbkxk7z3ykh5fauag3gopui7sx4sdzererl7jvsrdiwsboqra.b32.i2p				
		<b>Description:</b> I2P HTTPS Tunnel				
<b>I2P webserver</b>	HTTP server	127.0.0.1:7658	No Preview		<b>Start</b>	
		<b>Hostname:</b> mysite.i2p				
		<b>Description:</b> My eepsite				

## Part C: Omnihash & Omni\*Shell

- Hash of Everything



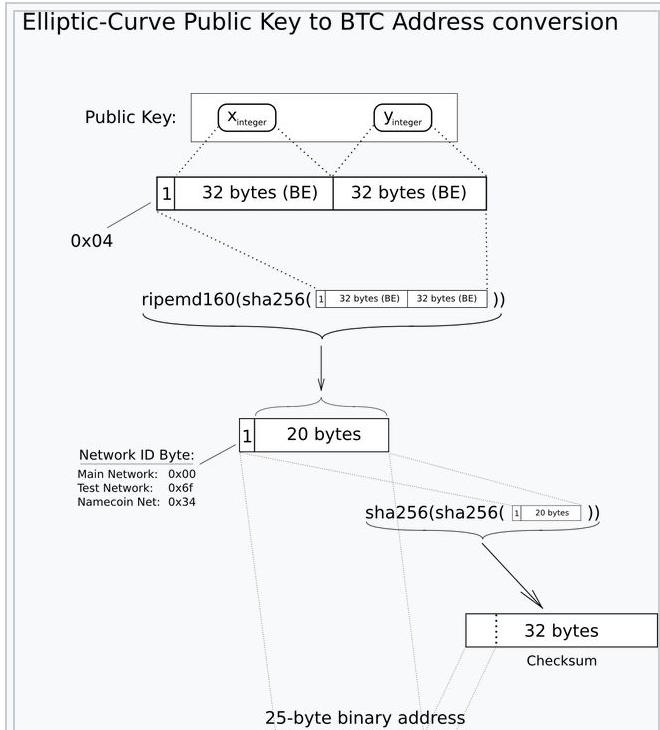
# [https://en.bitcoin.it/wiki/Technical\\_background\\_of\\_version\\_1\\_Bitcoin\\_addresses](https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses)

## Technical background of version 1 Bitcoin addresses

This article may be too technical for some users. The more basic article on [Bitcoin Addresses](#) may be more appropriate.

A Bitcoin address is a 160-bit hash of the public portion of a public/private ECDSA keypair. Using public-key cryptography, you can "sign" data with your private key and anyone who knows your public key can verify that the signature is valid.

A new keypair is generated for each receiving address (with newer [HD wallets](#), this is done deterministically). The public key and their associated private keys (or the seed needed to generate them) are stored in the [wallet](#) data file. This is the only file users should need to [backup](#). A "send" transaction to a specific Bitcoin address



# Omni\*Web: Def'n Ownership of Digital Assets w' Ownership JSON, bs'd on decentral...

https://8a75-2001-d08-da-5986-f093-ef4e-d34-4828.ngrok-free.app//oxw/oxw.php?nn=adam

150%



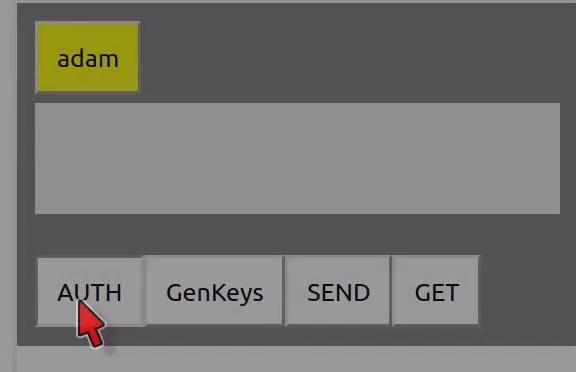
<https://www.youtube.com/watch?v=FofGUYJS7js>

2. Click GenKeys to generate private-public keypair. Then click AUTH again to initiate Authentication.  
Server will return Public Key PBK and browser will display it.

3. Enter "myhash e:" and click on the Nickname button labeled "adams"  
The server will return the hash of your public key

4. Press F12 to open the Developer Tools, click  
Enter the following command to adjust the position

```
document.querySelectorAll('.chat-popup')[1].style.height = '400px'  
document.querySelectorAll('.chat-popup')[1].style.display = "none"  
document.querySelectorAll('.chat-popup')[1].style.display = "block"
```



```
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH": ["SESSION:", "AUTH", "i:", "s:", ":"], 1 => 'jd:', 2 => 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH', ), 1 => NULL, )
```

8a75-2001-d08-da-5986-f X +

https://8a75-2001-d08-da-5986-f093-ef4e-d34-4828.ngrok-free.app//oxw/oxw.php?nn=adam

150%

1. Click AUTH button to initiate Authentication.

Server will return Public Key PBK = null if private-public keypair is not initialised.

2. Click GenKeys to generate private-public keypair. Then click AUTH again to initiate Authentication.

Server will return Public Key PBK and browser will display it.

3. Enter "myhash e:" and click on the Nickname button labeled "adam".

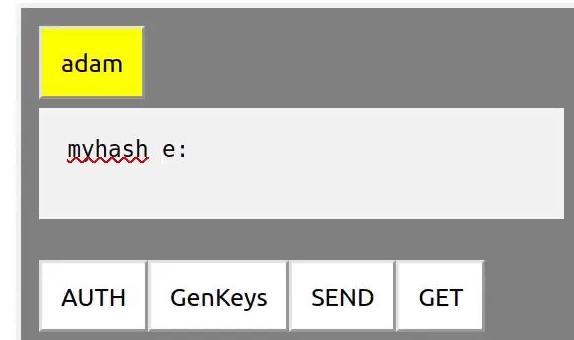
The server will return the hash of your public key (HPBK).

4. Press F12 to open the Developer Tools, click Console tab to open browser console.

Enter the following command to adjust the position and visibility of the dialog box:

```
document.querySelectorAll('.chat-popup')[1].style.bottom="200pt"
document.querySelectorAll('.chat-popup')[1].style.right="400pt"
document.querySelectorAll('.chat-popup')[1].style.display="none"
document.querySelectorAll('.chat-popup')[1].style.display="block"
```

```
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH":["SESSION:","AUTH","i:","s:","",""]}', 1 => 'jd:', 2 => 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH', ), 1 => NULL, )
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH":["SESSION:","AUTH","i:","s:","",""]}', 1 => 'jd:', 2 => 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH', ), 1 => array ( 'SP' => '6c1c1397', 'PASS' => '20240929_201140', 'PBK' =>
'LS0tLS1CRUdJTiBQVUJMSUMgS0VZLS0tLS0KTUIJQ0lqQU5CZ2txaGtpRzl3MEJBUVGQUFPQ0FnOEFNSUIDQ2dLQ0FnRUF3WkVQamF
'NN' => 'adam', ), )
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH1":["SESSION:","AUTH","i:","PBK","i:","o_hash.json","l_cdwjs","h_b64","s:","",""]}', 1 => 'jd:', 2
=> 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH1', ), 1 => 'Graph/dmeta/HnzRI9nOdQ==', )
```



8a75-2001-d08-da-5986-f × +  
https://8a75-2001-d08-da-5986-f093-ef4e-d34-4828.ngrok-free.app//oxw/oxw.php?nn=adam 150% ☆  
Home Page

1. Click AUTH button to initiate Authentication.

Server will return Public Key PBK = null if private-public keypair is not initialised.

2. Click GenKeys to generate private-public keypair. Then click AUTH again to initiate Authentication.

Server will return Public Key PBK and browser will display it.

3. Enter "myhash e:" and click on the Nickname button labeled "adam".

The server will return the hash of your public key (HPBK).

4. Press F12 to open the Developer Tools, click Console tab to open browser console.

Enter the following command to adjust the position and visibility of the dialog box:

```
document.querySelectorAll('.chat-popup')[1].style.bottom="200pt"
document.querySelectorAll('.chat-popup')[1].style.right="400pt"
document.querySelectorAll('.chat-popup')[1].style.display="none"
document.querySelectorAll('.chat-popup')[1].style.display="block"
```

```
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH":["SESSION:","AUTH","i:","s:","",""]}', 1 => 'jd:', 2 => 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH', ), 1 => NULL, )
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH":["SESSION:","AUTH","i:","s:","",""]}', 1 => 'jd:', 2 => 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH', ), 1 => array ( 'SP' => '6c1c1397', 'PASS' => '20240929_201140', 'PBK' =>
'LS0tLS1CRUdJTiBQVUJMSUMgS0VZLS0tLS0KTUIJQ0lqQU5CZ2txaGtpRzl3MEJBUVGQUFPQ0FnOEFNSUIDQ2dLQ0FnRUF3WkVQamF
'NN' => 'adam', ), )
fgl_s 333 < 2 > array ( 0 => array ( 0 => '{"B_AUTH1":["SESSION:","AUTH","i:","PBK","i:","o_hash.json","l_cdwjs","h_b64","s:","",""]}', 1 => 'jd:', 2
=> 'cdw:', 3 => 'am:', 4 => 'lcdw:', 5 => 'B_AUTH1', ), 1 => 'Graph/dmeta/HnzRI9nOdQ==', )
```



Omni\*Web: Def'n Ownership of Digital Assets w/ Ownership JSON, bs'd on decentral...

```
5 // commands to create Ownership JSON
6 jb={owner: "HnzRl9n0dQ=="}
7 jb.nick='adam'
8 jb.jstr='{"phone":123,"email":"bac@abc","name":"beta"}'
9 jb.hjs=bnToB64(cyrb53(jb.jstr))
10
11 // Ownership JSON object in console
12 jb
13 Object {
14   owner: "HnzRl9n0dQ==",
15   nick: "adam",
16   jstr: '{"phone":123,"email":"bac@abc","name":"beta"}' ,    [
17   hjs: "GMpBTscsMw=="
18 }
19
20 // JSON string ownership JSON
21 JSON.stringify(jb)
22 '{"owner": "HnzRl9n0dQ==", "jstr": {"\\\"phone\\\":123, \\\"email\\\":\\\"bac@abc\\\", \\\"name\\\":\\\"beta\\\""}, "hjs": "GMpBTscsMw=="}'
23
24 // JSON string ownership JSON, prettified
25 [
26   "owner": "HnzRl9n0dQ==",
27   "jstr": {"\\\"phone\\\":123, \\\"email\\\":\\\"bac@abc\\\", \\\"name\\\":\\\"beta\\\""}, "hjs": "GMpBTscsMw=="
28 ]
```



1:26 / 9:36



Console Network Style Editor Performance »

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
» jd.jstr="yo6sgm7q/pfvvp2e4kcuhjtf7wfltt63igwcukhbmum6lu3a3a.b32.i2p"
← "yo6sgmfq7pfvvp2e4kcuhjtf7wfltt63igwcukhbmum6lu3a3a.b32.i2p"

» jd
← ▶ Object { owner: "LlwHci_t", nick: "donald", jstr:
  "yo6sgmfq7pfvvp2e4kcuhjtf7wfltt63igwcukhbmum6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:
  "Bw7BA+t48Q==" }

» hjs=bnToB64(cyrb53(jd.jstr))
← "A0t0bR270w=="

» jd
← ▶ Object { owner: "LlwHci_t", nick: "donald", jstr:
  "yo6sgmfq7pfvvp2e4kcuhjtf7wfltt63igwcukhbmum6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:
  "Bw7BA+t48Q==" }
  date: "2024-10-18T15:18:49.260Z"
  hjs: "Bw7BA+t48Q=="
  jstr: "yo6sgmfq7pfvvp2e4kcuhjtf7wfltt63igwcukhbmum6lu3a3a.b32.i2p"
  nick: "donald"
  owner: "LlwHci_t"
  ▶ <prototype>: Object { ... }
```

Inspector Debugger Console Network Style Editor Performance »

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
» cyrb53.toString()
← "function(str, seed = 0) {
    let h1 = 0xdeadbeef ^ seed, h2 = 0x41c6ce57 ^ seed;
    for (let i = 0, ch; i < str.length; i++) {
        ch = str.charCodeAt(i);
        h1 = Math.imul(h1 ^ ch, 2654435761);
        h2 = Math.imul(h2 ^ ch, 1597334677);
    }
    h1 = Math.imul(h1 ^ (h1>>>16), 2246822507) ^ Math.imul(h2 ^ (h2>>>13), 3266489909);
    h2 = Math.imul(h2 ^ (h2>>>16), 2246822507) ^ Math.imul(h1 ^ (h1>>>13), 3266489909);
    return 4294967296 * (2097151 & h2) + (h1>>>0);
}"
```

```
» bnToB64.toString()
← "function bnToB64(bn) {
    var hex = BigInt(bn).toString(16);
    if (hex.length % 2) { hex = '0' + hex; }

    var bin = [];
    var i = 0;
```

Inspector Debugger Console Network Style Editor Performance »

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
    return 4294967296 * (2097151 & h2) + (h1>>>0);  
}"
```

```
» bnToB64.toString()
```

```
← "function bnToB64.bn) {  
  var hex = BigInt.bn).toString(16);  
  if (hex.length % 2) { hex = '0' + hex; }  
  
  var bin = [];  
  var i = 0;  
  var d;  
  var b;  
  while (i < hex.length) {  
    d = parseInt(hex.slice(i, i + 2), 16);  
    b = String.fromCharCode(d);  
    bin.push(b);  
    i += 2;  
  }  
  
  return btoa(bin.join(''));  
}"
```

Inspector Debugger Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests ⚙

"Bw7BA+t48Q==" }

» jd

← ► Object { owner: "LlwHci\_t", nick: "donald", jstr:  
"yo6sgmfq7pfvvp2e4kcuhjtfq7wfllt63igwcukhbmumq6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:  
"Bw7BA+t48Q==" }

» f\_add.toString()

← "function(){s.push(parseInt(s.pop())+parseInt(s.pop()))}"

» f\_h5364=function(){s.push(bnToB64(cyrb53(JSON.stringify(s.pop()))))}

← ► function f\_h5364()

» s

← ► Array(10) [ "a", "b", "c", "g", "h", "i", "e", "f", 7, 99 ]

» jd

← ► Object { owner: "LlwHci\_t", nick: "donald", jstr:  
"yo6sgmfq7pfvvp2e4kcuhjtfq7wfllt63igwcukhbmumq6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:  
"Bw7BA+t48Q==" }

» s.push(id)

»

Top ⬆️ 🔍

Inspector Debugger Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
← ▶ function f_h5364()
» s
← ▶ Array(10) [ "a", "b", "c", "g", "h", "i", "e", "f", 7, 99 ]
» jd
← ▶ Object { owner: "LlwHci_t", nick: "donald", jstr:
  "yo6sgmfq7pfvvp2e4kcuhjtfq7wfltt63igwcukhbmumq6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:
  "Bw7BA+t48Q==" }
» s.push(jd)
← 11
» f('h5364:')
h5364: omni.js:129:17
  0 h5364: omni.js:75:19
← undefined
» s[10]
← "HnqGtQrVYg=="
»
```

Top ↕

Inspector Debugger Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
← ▶ Array(10) [ "a", "b", "c", "g", "h", "i", "e", "f", 7, 99 ]
» jd
← ▶ Object { owner: "LlwHci_t", nick: "donald", jstr:
  "yo6sgmfq7pfvvp2e4kcuhjtfq7wfltt63igwcukhbmuqm6lu3a3a.b32.i2p", date: "2024-10-18T15:18:49.260Z", hjs:
  "Bw7BA+t48Q==" }
» s.push(jd)
← 11
» f('h5364:')
h5364: omni.js:129:17
  0 h5364: omni.js:75:19
← undefined
» s[10]
← "HnqGtQrVYg=="
» bnToB64(cyrb53(JSON.stringify(jd)))
← "HnqGtQrVYg=="

Top
```

Inspector Debugger Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests ⚙

```
» f_b64=function(){s.push(bnToB64(s.pop()))}
← ► function f_b64()

» f_e=function(){s.push(eval(s.pop()))}
← ► function f_e()

» f('jd e: je: h53: b64:')
 jd e: je: h53: b64:                                         omni.js:129:17
 0 jd                                                 omni.js:75:19
 1 e:                                                 omni.js:75:19
 2 je:                                                 omni.js:75:19
 3 h53:                                                 omni.js:75:19
 4 b64:                                                 omni.js:75:19
← undefined

» s[12]
← "HnqGtQrVYg=="
```

Inspector Debugger Console Network Style Editor Performance » 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests

```
» f_je=function(){s.push(JSON.stringify(s.pop()))}
← ► function f_je()
» f_h53=function(){s.push(cyrb53(s.pop()))}
← ► function f_h53()
» f_b64=function(){s.push(bnToB64(s.pop()))}
← ► function f_b64()
» f_e=function(){s.push(eval(s.pop()))}
← ► function f_e()
» f('jd e: je: h53: b64:')
jd e: je: h53: b64: 0 jd 1 e: 2 je: 3 h53: 4 b64:
```

omni.js:129:17  
omni.js:75:19  
omni.js:75:19  
omni.js:75:19  
omni.js:75:19

Top



main ▾

omnixtar.github.io / oxshell / js / omni.js

↑ Top

Code

Blame

Raw



```
58  ▼  phosinit=function(){
59  ▼  Phos=function(){
60      // var S=[] // this is local, not accessible outside
61      $ = this // macro
62      this.S=[]
63      var S=this.S // still need var S for local code access
64      S[0]=[]
65      var S0=S[0]
66      var $CDW = {}
67      S[0].$CDW = $CDW;
68      S0.skip = 0;
69      S0.CDW = [];
70      S0.dlb = {};
71  ▼  var FGLA = function($WA) {
72      // arguments[0].split(' ').map(e=>{
73      var c_cdw=false; var i=0, ic, w=$WA;
```



main

omnixtar.github.io / oxshell / js / omni.js

↑ Top

Code

Blame

Raw



```
58     phosinit=function(){
59         S0.CDW = [];
60         S0.dlb = {};
61     var FGLA = function($WA) {
62         // arguments[0].split(' ').map(e=>{
63         var c_cdw=false; var i=0, ic, W=$WA;
64         $WA.map(e=>{
65             console.log(i, e);
66             var $v=e, $vk=i; $l=$v.length;
67             if (!c_cdw && $v==':') {
68                 c_cdw=true;
69                 console.log(' CDW start ', W[i+1])
70                 ic = i+2; // start index of CDW
71                 CDN=W[i+1]
72                 $CDW[CDN]=[]
73             }
74             if (c_cdw) {
75                 $CDW[CDN].push($v)
76             }
77         })
78     }
79 }
```



main ▾

omnixtar.github.io / oxshell / js / omni.js

↑ Top

Code

Blame

Raw



```
58     phosinit=function(){
71         var FGLA = function($WA) {
110             S0.CDW.pop();
111             console.log(1183, 'after FGLA', JSON.stringify(S0.CDW));
112             S0.cda = end(S0.CDW);
113         }
114         else if ($v[$l - 1] == ':') { // colon suffix word after symbol else : will fail
115             var $fn = $v.substr(0, $l-1);
116             if (typeof eval("f_"+$fn)!=="undefined")
117                 // console.log('is func', $v, typeof eval("f_"+$fn))
118                 eval("f_"+$fn+"()")
119             } else s.push(e)
120         }
121         i++
122     }
123 }
```

Inspector Debugger Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests ⚙

```
11 d omni.js:75:19
1176 before FGLA [[ "d", 11, {} ]]
0 e omni.js:75:19
1 f omni.js:75:19
1183 after FGLA []
← undefined
» f('3 4 +')
3 4 + omni.js:129:17
0 3 omni.js:75:19
1 4 omni.js:75:19
2 +
← undefined
» s
← ▶ Array(9) [ "a", "b", "c", "g", "h", "i", "e", "f", 7 ]
» omnistart()
```

Console Network Style Editor Performance » ⓘ 1

Filter Output Errors Warnings Logs Info Debug CSS XHR Requests ⚙

```
2 +
← undefined
» s
← ▶ Array(9) [ "a", "b", "c", "g", "h", "i", "e", "f", 7 ]
» f_add.toString()
← "function(){s.push(parseInt(s.pop())+parseInt(s.pop()))}"
» f('44 55 add:')
44 55 add:                                         omni.js:129:17
0 44                                         omni.js:75:19
1 55                                         omni.js:75:19
2 add:                                         omni.js:75:19
← undefined
» s
← ▶ Array(10) [ "a", "b", "c", "g", "h", "i", "e", "f", 7, 99 ]
»
```

```
gunzipinit=function(){gunzip=e=>{const r=Uint8Array.from(atob(e),(e=>e.charCodeAt(0))),n=new DecompressionStream("gzip"),t=n.writable.getWriter();return t.write(r),t.close(),new Response(n.readable).arrayBuffer().then((function(e){return(new TextDecoder).decode(e)})),getgzfn_js=async function(s){var s1=await gunzip(s),j3=JSON.parse(s1),j4=JSON.parse(j3.s);return eval(j4.n+="+"+j4.s),j3}}}  
omnigz="H4sIAAAAAAAA8UZa1PbSPIz/hUdFxdrYqFY5o1PbBEI2aQg2Ttyy255vdRYGtksuSSxjzC8t+ve2YkzbkNp/0H0Dq6e7pd/eM4lkaKpmlINMwmUXCmeUJg8cGwC3PoYAAoiycTUSqvDAXXIn3iaA3p1mEuZyqJushbuEVQh0plcvhTAmnqR6mouICU4l79faa3/LvIRd5iLi0L61W2w2z6MHj06lIo+0xTCKnwPWhRuPtW5jciFueXCV0M5ukcsyLcZMR/GTy8WP3g5/c3QVBo1wKYquiwx4b4UM+3N6cgwqVu1AIeGiHUZqAyRCwdgnwH0keDQUIoY/NZIL466Gb/nhTi2dy28p+niLAeHiCXhuBC0e/j4T8A9vESkI0Xv7baxrvmFY0Sl9XDM8+MsEkfKkaxXrWsxzrkae3IySxx8/RNpx0jubG9tbW7v7vh1504ictci+9v7u5ubWzu7uxb5qfESb/x7eHjo7zDcoru1s9ftbnd2GS4sscW/hLeJeJvdnZ2tvf39zr7l/pwYFv+7fGv7P8s3F2qWp7DV3d/a39nt7u/AG3C6nf1df9uH17gtg7Zl0NGRMky/Zu92tipngwU4w5SVAT4W9yjt0zn6iCGNcE9lFxiY6chBYwlGSNPcW8dCP8A30bRkrU6LdwSn3vapMRvKFNC6A969l2HQvkSlQ9DerjDoBbgUITMNyidg+JxyvNCkFy0XCQyFI50kwMbZUCv7li7DBHVy0zFeTY5tpHkROW6TL3prBg7QwtADgF06ZmkTMYdqow7hHqdydRptZg2YZlo84Q5/cweAVPtzVUCE8HTApIsHVHsT2Ao8J+AiUx1LEVklb32g8f04PQzbXbte0VALB18qlubmUUVOkm4gx0sMw6j5Y8XX2oYml03eCw0Pl18+ewVekXGD8iNue0D0r8myZ1VHaMzmGWFlkivCQb0UTEp5/bzaDZjkERfw6fGhztgmWI6s0SbdFhugw1P5zsHclf99NoFARNWkxyo1n6TU7RRMrUrJ9/u3t7cx7df0PcxnyxXrUMNBDB4SNJriAP/iNTtXeU5/xB093hKhs6grk0Iol6NUGRmJsG2qoizCbTXBQFskWrCj5BcZBzk7kqSL27XCo+RL0MhLrEZ5E7rGfVV3pV0DmiemGSFegfl5j+WxRT1Ek4qYcMI6JnHifB3s3imFh4aixSx6n0EezRMNWu/oodggSLRM68SD8gxhNjTy6KMfoWp1fxRcCLhzSEedVmj7o7+QG/41KBMRCC3evNQLtd545TYFhcb9Uh15vo11IpE4hbXmpcu+VpBk/LPQa7VajusSfNjMfeD8/3Pv32aaT+Fwb3FMPZaCRyzQxI/IPNzQPfb8yp6q4ksYwb9m4zGUEnCII7mUbZnVe8fu2Uj0F/wHqEqKrVxtUVeRS9v8WHM1kokYr8pxdXnCYGCzVgLbk7d0DGhnikcmzhv2ez1q0ATBFsgCqDDJstwCogMmi2M3S3RLa/tZu1t9+ZK4Jqa2Fmg1OMxF+wfcinInJrBMqUIUhX0EXUKUqggpgK5Ci/DSSD/+ev5GXPrYdksp2mmmyKFrY+HHTRU86AZBGm/hG34g5/OvVMnZQfCk2lqGAVILXkY7mSTExHzWaIcDCz3lc805kZTiodjbTccJtIXLPb3DYapZH2m9/VMpP3Kk5kIXvkU1o3GdJwVyyne+AwBCwCq1BhsFAcXGBH0rMayAEkVN+SJC2mmqIch5TNmHaAJCxkJTbi0PUFjUyDzMM80lCAeMdNvhr0BEV6hZJJASr0QX"
```



Files master

[omnixchat / core / src / com / biglybt / plugin / net / buddy / BuddyPluginBeta.java](https://github.com/omnixtar/omnixchat/blob/master/core/src/com/biglybt/plugin/net/buddy/BuddyPluginBeta.java)

Code

Blame

Raw

```

2292
2293 logMessage(
2294     Stack<Object> stackAny = new Stack<>();
2295
2296     Stack<HashMap<String, String>> stackHM = new Stack<>();
2297
2298     // Stack<> stackAll = new Stack<>();
2299
2300     String expr = msg;
2301
2302
2303     System.out.println(expr);
2304     System.out.println("Input\tOperation\tStack after");
2305
2306
2307     // Phoscript (Phos) Engine
2308     for (String token : expr.split("\\s+")) {
2309         System.out.print(token + "\t");
2310         switch (token) {
2311             case "+":
2312                 System.out.print("Operate\t\t");
2313                 stack.push(stack.pop() + stack.pop());
2314                 break;
2315             case "-":
2316                 System.out.print("Operate\t\t");
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3198
3199
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3298
3299
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3398
3399
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3498
3499
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3598
3599
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3698
3699
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3798
3799
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3898
3899
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3998
3999
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4098
4099
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4198
4199
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4298
4299
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4398
4399
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469

```

[https://github.com/omnixtar/omnixchat/blob/master/core/src/com/biglybt/plugin/net/  
buddy/BuddyPluginBeta.java](https://github.com/omnixtar/omnixchat/blob/master/core/src/com/biglybt/plugin/net/buddy/BuddyPluginBeta.java)



master ▾

[omnixchat](#) / [core](#) / [src](#) / [com](#) / [biglybt](#) / [plugin](#) / [net](#) / [buddy](#) / **BuddyPluginBeta.java**

↑ Top

[Code](#)[Blame](#)[Raw](#)

```
2292         logMessage(  
2333             Stack<Double> stack      = new Stack<>();  
2334             Stack<String> stackStr = new Stack<>();  
2335             Stack<Object> stackAny = new Stack<>();  
2336             Stack<HashMap<String, String>> stackHM = new Stack<>();  
2337             // Stack<> stackAll = new Stack<>();  
2338             String expr = msg;  
2339  
2340             System.out.println(expr);  
2341             System.out.println("Input\tOperation\tStack after");  
2342  
2343             // Phoscript (Phos) Engine  
2344             for (String token : expr.split("\\s+")) {  
2345                 System.out.print(token + "\t");  
2346                 switch (token) {  
2347                     case "+":  
2348                         System.out.print("Operate\t\t");  
2349                         stack.push(stack.pop() + stack.pop());  
2350                     break;
```

## Super-program or Super-Shell?

Understanding of program – change after using  
“super shell” – metaprogramming in Phoscript.

Interaction between programs, via “super-shell” –  
become “super-program”.

UNIX pipe is limited to one computer.

Omnihash data + I2P = global data sharing.

## Part D: “Trojan Horse Super-Shell?”

- Consolidating Users across Open Source Projects & Commercial Social Media by adding Omni\*Shell to generate Omnihash User ID as *universal decentralised ID*.

## From Technofeudalism to Technosocialism: FORTH unexpected roles in Decentralised Systems

Technofeudalism is a term coined by former Greek finance minister Yanis Varoufakis as the title of a book which compares top American technology companies to feudal lords, who literally enslaved individual users and free software programmers, exploiting their data and source code without fairly rewarding them.

Technosocialism has been used by many other authors but I use it to describe the very real but underreported progress made by free software programmers and free software, despite the lack of fair mechanisms for rewards so far. Free software programmers or technosocialists have made technofeudalism possible as a matter of fact, but they do not get to enjoy the rewards.

Bitcoin and related projects were supposed to address this problem, and are successful in their own way, so we are now half way towards technosocialism.

At section interval, recap past sections, preview this section

Trojan horse: small size of Omni\*Shell, 3kb minified gzipped JavaScript.

Football Analogy: I2P (Invisible Internet Project) and Bitcoin have proven the technical viability of fundamental Decentralised algorithms, produced many millionaires and billionaires – carrying football past half field. (25 years, beneficiaries are like Bankers)

Omni\*Web plan to make more people rich – like McDonald's vs. Hongkong & Shanghai Bank – second half of football field – next 25 years.

Existing web ecosystem – *dominated by a few huge social media applications* – difficult for small applications to grow – many reasons – *User Fragmentation*.

Solution: Omnihas User ID

Strategy: *Trojan Horse Super-Shell*, not the computer virus type – *security guaranteed by hash & public key cryptography*.

Omni\*Shell in *JavaScript & Java* – cover most client applications in web & Android – *easily ported to other programming languages* – Apple – *server side – PHP, Python, C++* – (show screenshots)

Probability of success?

Cloning: GNU & Linux began by cloning UNIX.

Now most commercial software companies use unspecified amount of free software.

Modify free software licence – separation of disclosure & royalties.

Track usage (execution) of free software using hashcodes – claim payments.

BitTorrent used to be carrying >30% of all Internet traffic.

Heavy censorship on popular platforms such as Reddit & TikTok – decentralised social media – impossible to censor – *FILTER model – users may choose various types of filters to filter unwanted text, videos, photos or user IDs* – data stored on devices owned by users – or access control belongs to users, not platform.

**Omni\*Web  
2030 Targets!!**

MMAGA Revenues 2023 (USD billions)

Total: USD 1.611 Trillion (+7.26%)

$0.1\% \times 1.6T$   
 $= \text{USD } 1.6B$



**A literal trillion dollar question: Do you know that the combined revenues of the top 5 technology companies in 2023 already exceeded USD 1.6 trillion and are growing strongly above 5% annually?**

## Part E: FORTH roles in future

Bitcoin & cryptocurrencies actually made FORTH (variant) *the most valuable and popular* (by number of devices, nodes, wallets) in the world – (need decentralised systems to gather statistics!!)

Omni\*Web “trojan horse” strategy will (hopefully) attract more programmers to use FORTH (Phoscript or dialects) as interface programming language, if Omni\*Web plan of making *Omnihash User ID as the universal decentralised ID* works.

Metaprogramming & Mathematics are important – SymForth example.

Decentralised AI – ideas threatening commercial interests are filtered – *hash based indices allowed AI training data to be distributed in users computers & mobile devices* – much bigger (?) than commercially owned assets?

Same dilemma in history – *land grab from farmers* – now land owned by nation state or corporation – do we want this to happen again – *now we know we can prevent it?*

Although we have used ideological terms such as techno feudalism and techno socialism to express our views of reducing global rich poor gaps through Decentralised systems, the Chinese or Asian principle of Taichi views conflicts not as binary black versus white, but as forces in opposition feeding on each other, as quoted by Danish Physicist Niels Bohr in Latin as Contraria sunt complementa or in English “opposites are complementary”.

This philosophy can be seen in the Chinese Weiqi or Go chess, where on a board with 19 by 19 lines namely 361 spots, the goal of the game is to simply outdo the opponent by occupying at least one spot more than the opponent with the chosen black or white pieces, instead of killing the King of the opponent in the European chess.

Technosocialism = “better pay for some programmers”, not threatening Washington & Wall Street.

Metanarchy = Metaverse + (Mon)-archy = self governance through metaverse