SOURCEFORGE

/ Home / Browse / KingCobra64 Code

KingCobra64

Brought to you by: ka_shrinivaasan

[128ac6]: / KingCobraDesignNotes.txt

Restore

History

Download this file

475 lines (372 with data), 60.2 kB

```
  1 ------------------------------------------------------------------------------------
  2 NEURONRAIN KingCobra - Module for Kernelspace Messaging and Computational Economics
  3
  4 This program is free software: you can redistribute it and/or modify
  5 it under the terms of the GNU General Public License as published by
  6 the Free Software Foundation, either version 3 of the License, or
  7 (at your option) any later version.
  8
  9 This program is distributed in the hope that it will be useful,
 10 but WITHOUT ANY WARRANTY; without even the implied warranty of
 11 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 12 GNU General Public License for more details.
 13
 14 You should have received a copy of the GNU General Public License
 15 along with this program.  If not, see <http://www.gnu.org/licenses/>.
 16
 17 #-------------------------------------------------------------------------------------------
```

```
#K.Srinivasan
#NeuronRain Documentation and Licensing: http://neuronrain-documentation.readthedocs.io/en/latest/
#Personal website(research): https://sites.google.com/site/kuja27/
#-------------------------------------------------------------------------------------------
**************************************************************************************/

Theoretical Interludes, Design Notes and ToDo (long term with no deadline)
--------------------------------------------------------------------------

[This is a major research oriented subsystem of NeuronRain and inspired by COBRA project (done by the author in hi

[KingCobra though a misnomer is expanded as ClOud With ARBiters MimicKING containing the anagram]

1072. (THEORY) There is a cloud of nodes which execute a set of services from randomly created clients.

1073. (THEORY) This cloud could be on NeuronRain (AsFer+USBmd+VIRGO+KingCobra) platform or any other opensource cl

1074. (THEORY) The Clients are publishers of Service requests which are of many types -  miscellaneous types of Se

1075. (THEORY) The Services on the Cloud are Subscribers to these requests of specific type. Thus this is the conv

1076. (THEORY) The requests flow through cloud using a workqueue (which could be a lowlevel Linux workqueue or VIF

1077. (THEORY) The difference is that the Cloud has nodes that "deceive" or "corrupt".

1078. (THEORY) Service requests - are published by the clients in the need of a service which could be defined by

1079. (THEORY) Problem reports - are published by clients which are "dissatisfied" by the quality of service by th

1080. (THEORY) Suggestions - are enhancement requests sent by clients and require manual intervention.

1081. (THEORY) Cloud nodes have a Quality of Service metric calculated by a model.

1082. (THEORY) The cloud has a reporting structure of nodes - either as a graph or tree. The graph is dynamically

1083. (THEORY) The difficult part of the above is using Arbiters to find "faulty" nodes based on problem reports f

1084. (THEORY) Brewer's CAP conjecture proved by [GilbertLynch] as a theorem (still debated) states that only 2 of

1085. (THEORY) CAP theorem does not seem to apply to the above faulty scenario with corrupt nodes under Consistenc
```

```
1086. (THEORY) As "corruption" is more conspicuous with monetary element, if above services are "charged" with a 1

1087. (THEORY) Identifying criminal nodes as in (15) above seems to be beyond the ambit of CAP. Thus CAP with Inte

1088. (THEORY and FEATURE) Analytics on the Problem reports sent to the cloud queue give a pattern of corrupt node

1089. (THEORY) Policing the cloud nodes with arbiters - This seems to be limited by CAP theorem and Integrity as a

1090. (THEORY) Brooks-Iyengar algorithm for sensors in all cloud nodes is an improved Byazantine Fault Tolerant al

1091. (THEORY) BitCoin is a Byzantine Fault Tolerant protocol.

1092. (THEORY) Byzantine Fault Tolerance in Clouds is described in http://www.computer.org/csdl/proceedings/cloud/

1093. (THEORY) Related to point 18 - The problem of fact finding or fault finding using a cloud police has the sam

1094. (THEORY) Reference article on cloud BFT for Byzantine, Corrupt brokers - Byzantine Fault-Tolerant Publish/Su


------------------------------------------------------------
1095. KingCobra messaging request-response design - options
------------------------------------------------------------

1095a. Implementing a message subscription model in kernelspace where clients publish the message that is queued-i

1095b. (DONE-minimal implementation) At present a minimum kernelspace messaging system that queues remote request

------------------------------------------------------------------------------------------------
1095c.(DONE) KingCobra - VIRGO queue - VIRGO cpupooling , mempooling and queue service drivers interaction schemat
------------------------------------------------------------------------------------------------

        KingCobraClient ==========><REQUEST:id==================> VIRGO cpupooling service =====> VIRGO Queue =
               ||
               ||
                <================= VIRGO Queue <====== VIRGO cpupooling service ====<REPLY:id==================

        KingCobraClient ==========><REQUEST:id==================> VIRGO mempooling service =====> VIRGO Queue =
               ||
               ||
```

```
                <================= VIRGO Queue <====== VIRGO mempooling service ====<REPLY:id>==================

        KingCobraClient ==========>=<REQUEST:id>==================> VIRGO Queue service ======================
               ||
               ||
                <================= VIRGO Queue service ============================<REPLY:id>==================



1095d. (ONGOING) kingcobra_servicerequest_kernelspace() distinguishes the "REQUEST" and "REPLY" and optionally per

1095e.Above option 24b implements a simple p2p queue messaging in kernel. To get a Topic-like behaviour in VIRGO c

1096. (ONGOING) For the timestamp service, EventNet described in http://sourceforge.net/p/asfer/code/HEAD/tree/Asf

784. (THEORY - Neuro Cryptocurrency Implemented in AstroInfer - this section is an extended draft on respective to
        m1=MAC_alloc(denomination)
        m2=m1 (---- this is disallowed)
Linux kernel allocation functions - kmalloc() - have a krefs functionality for reference counting within kernel. F

784.1 Schematic Diagram for Cloud Perfect Forwarding with AsFer+VIRGOQueue+KingCobraUserspace:
-------------------------------------------------------------------------------------

    Telnet or other client ===========> VIRGO Queue Service Listener ======> VIRGO Workqueue Handler
                                                                                |
                                                                                |  [KernelSpace]
                                                                               V
        AsFer Cloud Perfect Forwarding Client <===== KingCobra Userspace shell script(call_usermodehelper)
                        |                                                       |
                        |                                                       |
  Virtual Currency      |                                                       |  [UserSpace]
                       V                                                       V
        AsFer Cloud Perfect Forwarding Server <=================================/


References:
----------
784.2 An example distributed transactional memory implementation in cloud - http://infinispan.org/tutorials/simple

1097. (THEORY) SIMULATING A VIRTUAL ECONOMY with above MAC protocol (Message-as-currency): If each message sent is
```

```
785. (THEORY - this section is an extended draft on respective topics in NeuronRain AstroInfer Design - https://gi

                                ---------------------
                                |Unique MAC id MINT|
                                ---------------------
                                        ||
        -----money trail-----------------
        |
        V
        ....
        Buyer   ======= sends MAC message (REQUEST id) =======> Seller (stores the MAC in local cash reserve and p
        ||                                                      ||
        <============ sends the goods and services (REPLY id) ===

In the above schematic, money with unique id in cloud reaches a buyer after many buyer-seller transitions called '

References:
-----------
785.1 Price fixing for items in Buyer-Seller-Trader networks - Trading Networks - Market Equilibrium and Walrasian
785.2 Algoithmic Game Theory - Market Equilibrium for Price - Equilibrium is a strategic standoff - both players c
785.3 Price-setting in Trading networks - Chapter 11 - https://www.cs.cornell.edu/home/kleinber/networks-book/netw

1098. (THEORY) VALUE FOR ELECTRONIC MONEY: How is above MAC money earned - This again requires linking value to mo

786. (THEORY and IMPLEMENTATION - this section is an extended draft on respective topics in NeuronRain AstroInfer
        ------------------------------------------------------------------------------------------------------
        |value(i) = summation(value(ingredients of i)) + cost(integrating the ingredients to create item i) |
        ------------------------------------------------------------------------------------------------------
Obviously the above recursion combinatorially explodes into exponential number of nodes in the recursion tree. Ide

-------------------------------------------------------------------------
787. (THEORY - this section is an extended draft on respective topics in NeuronRain AstroInfer Design - https://gi
-------------------------------------------------------------------------

        Buyer            A-------<id><refcnt:0>----------------------> Seller <id><refcnt:1> (increments refcnt)
        (<id><refcnt:1>  |
         <id><refcnt:0>  |
        after decrement  |
        refcnt           |
```

```
        )---------------->
Above has to be transactional (i.e atomic across cloud nodes)

1099. (THEORY) MAC protocol reaper
-------------------------------
Reaper thread in each cloud node harvests the zero refcounted allocations and invokes destructors on them. Same  M

1100. (THEORY) Cloud Policing With Arbiters - Revisited:
-----------------------------------------------------
When a suspect node is analyzed when a complaint problem is filed on it, (1) it is of foremost importance on how t

------------------------------------------------------------------------------------------------------------
788. (THEORY) MAC Money Flow as MaxFlow problem - this section is an extended draft on respective topics in Neuror
------------------------------------------------------------------------------------------------------------
Transactions happening in a cloud are edges between the nodes involved (buyer and seller). Thus it creates a huge

1101. (THEORY) Cycles and components in above MAC Money Flow Graph:
-----------------------------------------------------------
Above graph of money transactions could be cyclic which implies a supply chain. Strongly connected components of t

1102. (THEORY) STOCK TRADING:
---------------------------
One of the component in above MAC Money Flow Graph of cloud could be a virtual Stock Exchange. Based on the financ

------------------------------------------------------------------------------------------------------------
789. (THEORY) Analysis of Poverty and Alleviation through above money flow graph - this section is an extended dra
------------------------------------------------------------------------------------------------------------
Weights of the edges of money flow graph are the denominations of the transaction. Thus high value edges and low v

Previous Linear Program could be weighted to : Sum(w(i) * RichEdges) - Sum(w(k) * PoorEdges). Money Flow Graph (or
        R(i) = Rich Edges
        P(i) = Poor Edges
minimize:
        Poverty = Sum_1_to_x(w(i) * R(i)) - Sum_1_to_y(w(k) * P(i))
subject to constraint:
        x >= epsilon * y, 0 < epsilon < 1

References:
----------
789.1 J-PAL case study of social programs fund flow reforms - https://www.povertyactionlab.org/case-study/fund-flo
```

```
218
219  ------------------------------------------------------------------------------------------------------------
220  790.(THEORY) Demand and Supply and Value() function - Quantitative Majority Circuit - this section is an extended
221  ------------------------------------------------------------------------------------------------------------
222  Alternative to the recursive definition of value() function above can be done through Demand and Supply - more the
223
224  References:
225  ----------
226  790.1 Barrington Theorem - Majority can be computed by Bounded Width Branching Program (BWBP) of width 5 and lengt
227  790.2 Szemeredi Regularity Lemma - Section 1.3 - http://www.math.ucsd.edu/~fan/teach/262/read/reg.pdf - "...the Re
228
229  ------------------------------------------------------------------------------------------------------------
230  791.(THEORY) Hidden or Colored Money - this section is an extended draft on respective topics in NeuronRain Astro]
231  ------------------------------------------------------------------------------------------------------------
232  In an ideal Cloud with only MAC currencies, colored money can co-exist if (not limited to) some money trails are n
233
234  Total storage of money in Flow Market Graph = | Incoming money flow at Source - Incoming money flow at Sink | i.e
235
236  There is a special vertex in Money Flow Market designated as Direct Taxation Hub which has incoming direct tax mor
237
238  Colored Money = | Total storage money * Direct Taxation rate - Incoming Flow at Direct Taxation Hub Node|
239  Colored Money = | Incoming money flow at Source * Direct Taxation rate - Incoming money flow at Sink * Direct Taxa
240
241  Previous is an approximate naive zero-knowledge estimation of Colored money in Money Flow Market. This assumes tha
242
243  Above estimate is for direct taxation and resultant evasion. For indirect taxes (on Goods and Services etc.,), the
244
245  Colored Money = | Total Goods and Services * Taxation rate - Incoming Flow at Indirect Taxation Hub Node|
246  Colored Money = | GDP^2 * Tax-to-GDP ratio - Incoming Flow at Indirect Taxation Hub Node| where GDP is assumed to
247
248
249  References:
250  -----------
251  791.1 Network Flows over time over Storage Area Networks (SAN) - https://hal.inria.fr/inria-00071643/document
252  791.2 Network Flow - [GoldbergTardosTarjan] - http://www.cs.cornell.edu/~eva/network.flow.algorithms.pdf
253  791.3 Algorithmic Game Theory - Flow Markets - [TimRoughGarden] - http://theory.stanford.edu/~tim/books.html
254  791.4 RFID tagged currencies - $100 bill - http://www.businessinsider.in/New-Smart-Paper-Could-Put-An-End-To-Dark-
255  791.5 Cons of RFID currencies - http://www.prisonplanet.com/022904rfidtagsexplode.html
256  791.6 Mechanism Design and Machine Learning - https://www.cs.cmu.edu/~mblum/search/AGTML35.pdf - Design of algorit
257  791.7 Financial and Economic Networks - https://supernet.isenberg.umass.edu/bookser/innov-ch1.pdf
```

```
258
259   ------------------------------------------------
260   1103. Commits as on 1 March 2014
261   ------------------------------------------------
262   Example java Publisher and Listeners that use ActiveMQ as the messaging middleware have been committed to reposit
263   ------------------------------------------------
264   1104. Commits as on 17 March 2014
265   ------------------------------------------------
266   KingCobra userspace library and kernelspace driver module have been implemented that are invoked 1) either in use
267   2) or through intermodule invocation through exported symbols in KingCobra kernel module, by the workqueue handle
268
269   ------------------------------------------------
270   1105. Commits as on 22 March 2014
271   ------------------------------------------------
272   Minimalistic Kernelspace messaging server framework with kernel workqueue,handler and remote cloud client has beer
273
274   ------------------------------------------------
275   1106. Commits as on 29 March 2014
276   ------------------------------------------------
277   Initial commits for KingCobra Request Response done by adding 2 new functions parse_ip_address() and reply_to_publ
278
279   ------------------------------------------------
280   1107. Commits as on 30 March 2014
281   ------------------------------------------------
282   Both VIRGO cpupooling and mempooling drivers have been modified with use_as_kingcobra_service boolean flag for ser
283
284   ------------------------------------------------
285   1108. Commits as on 6 April 2014
286   ------------------------------------------------
287   Fixes for REQUEST and REPLY headers for KingCobra has been made in virgo_cloudexec_mempool recvfrom() if clause ar
288
289   ------------------------------------------------
290   1109. Commits as on 7 April 2014
291   ------------------------------------------------
292   New function parse_timestamp() has been added to retrieve the timestamp set by the VIRGO mempool driver before pus
293
294   ------------------------------------------------
295   1110. Commits as on 29 April 2014
296   ------------------------------------------------
297   Intial commits for disk persistence of KingCobra request-reply queue messages have been done with addition of new
```

```
298
299   ----------------------------------------
300   1111. Commits as on 26 August 2014
301   ----------------------------------------
302   KingCobra driver has been ported to 3.15.5 kernel and bugs related to a kernel_recvmsg() crash, timestamp  parsing
303
304   ------------------------------------------------
305   Version 14.9.9 release tagged on 9 September 2014
306   ------------------------------------------------
307   ------------------------------------------------
308   Version 15.1.8 release tagged on 8 January 2015
309   ------------------------------------------------
310
311   ----------------------------------------
312   1112. Commits as on 17 August 2015
313   ----------------------------------------
314   KingCobra + VIRGO Queuing port of Linux Kernel 4.1.5 :
315    - changed the REQUEST_REPLY.queue disk persisted queue path to /var/log/kingcobra/REQUEST_REPLY.queue
316    - kernel built sources, object files
317    - kern.log with logs for telnet request sent to VIRGO queue driver, queued in kernel work queue and handler invoc
318   KingCobra service request kernel function for the popped request; disk persisted /var/log/kingcobra/REQUEST_REPLY.
319
320   ----------------------------------------
321   1113. Commits as on 14 October 2015
322   ----------------------------------------
323   AsFer Cloud Perfect Forwarding binaries are invoked through call_usermodehelper() in VIRGO queue. KingCobra commar
324
325   ----------------------------------------
326   1114. Commits as on 15 October 2015
327   ----------------------------------------
328   - Updated KingCobra module binaries and build generated sources
329   - kingcobra_usermode_log.txt with "not found" error from output redirection (kingcobra_commands.c). This error is
330   - kingcobra_commands.c has been changed to invoke absolute path executable. With uncommenting of fd_install and se
331
332   ---------------------------------------------------------------------------
333   1115. Commits as on 10 January 2016
334   ---------------------------------------------------------------------------
335   NeuronRain KingCobra research version 2016.1.10 released.
336
337   ---------------------------------------------------------------------------------------------------
```

```
1116. NEURONRAIN VIRGO Commits for virgo_clone()/telnet -> VIRGO cpupooling -> VIRGO Queue -> KingCobra
- as on 12 February 2016
-------------------------------------------------------------------------------------------
VIRGO commit:
https://github.com/shrinivaasanka/virgo-linux-github-code/commit/72d9cfc90855719542cdb62ce40b798cc7431b3d

Commit comments:
-------------------------------------------------------------------------------------------
Commits for Telnet/System Call Interface to VIRGO CPUPooling -> VIRGO Queue -> KingCobra
-------------------------------------------------------------------------------------------
*) This was commented earlier for the past few years due to a serious kernel panic in previous kernel versions - <
*) In 4.1.5 a deadlock between VIRGO CPUPooling and VIRGO queue driver init was causing following error in "use_as
  - "gave up waiting for virgo_queue init, unknown symbol push_request()"
*) To address this a new boolean flag to selectively enable and disable VIRGO Queue kernel service mode "virgo_que
*) With this flag VIRGO Queue is both a kernel service driver and a standalone exporter of function symbols - push
*) Incoming request data from telnet/virgo_clone() system call into cpupooling kernel service reactor pattern (vir
*) This resolves a long standing deadlock above between VIRGO cpupooling "use_as_kingcobra_service" clause and VIR
*) This makes virgo_clone() systemcall/telnet both synchronous and asynchronous - requests from telnet client/virg
either synchronous RPC functions executed on a remote cloud node in kernelspace (or) an asynchronous invocation th
 clause path to VIRGO Queue driver which enqueues the data in kernel workqueue and subsequently popped by KingCobr
*) Above saves an additional code implementation for virgo_queue syscall paths - virgo_clone() handles, based on c
data passed to it either as a remote procedure call or as a data that is pushed to VIRGO Queue/KingCobra pub-sub k
*) Kernel Logs and REQUEST_REPLY.queue for above commits have been added to kingcobra c-src/testlogs/

-------------------------------------------------------------------------------------------------------------
1117. Commits - KingCobra 64 bit and VIRGO Queue + KingCobra telnet requests - 17 April 2017
-------------------------------------------------------------------------------------------------------------
*) Rebuilt KingCobra 64bit kernel module
*) telnet requests to VIRGO64 Queueing module listener driver are serviced by KingCobra servicerequest
*) Request_Reply queue persisted for this VIRGO Queue + KingCobra routing has been committed to c-src/testlogs.
*) kern.log for this routing has been committed in VIRGO64 queueing directory
*) Similar to other drivers struct socket* reinterpret cast to int has been removed and has been made const in que

-------------------------------------------------------------------------------------------------------------
779. (FEATURE-DONE) Commits - CVXPY implementation for Eisenberg-Gale Convex Program - 18 August 2017 -  - this se
-------------------------------------------------------------------------------------------------------------
(*) First commits for Convex Optimized Market Equilibrium Prices
(*) Imports CVXPY Convex Program solver
(*) Objective function is a logistic variant of Eisenberg-Gale Convex Program i.e uses money * log(1+e^utility) ir
 money * log(utility) because of curvature error (log is error flagged as concave and logistic is convex per:
```

```
 http://www.cvxpy.org/en/latest/tutorial/functions/index.html#vector-matrix-functions)
(*) Formulates constraints and objective functions based on http://www.cs.cmu.edu/~sandholm/cs15-892F13/algorithmi
(*) But, For all installed solvers ECOS, ECOS_BB, SCS, LS solved convex program prints value as None despite all c
(*) Obviously it should have worked. Therefore this is only a partial implementation commit.
(*) This implementation uses numpy randomly initialized arrays for Money each buyer has and per-good utility(happi
(*) Replacing money with perceived merit values translates this Market Equilibrium - Intrinsic Value versus Market
 Equilibrium - Intrinsic Merit versus Perceived Merit. This has been already described in NeuronRain AsFer Design

        - http://sourceforge.net/p/asfer/code/HEAD/tree/asfer-docs/AstroInferDesign.txt and
        - https://github.com/shrinivaasanka/asfer-github-code/blob/master/asfer-docs/AstroInferDesign.txt

------------------------------------------------------------------------------------------------------------
780. (FEATURE-DONE - this section is an extended draft on respective topics in NeuronRain AstroInfer Design - http
------------------------------------------------------------------------------------------------------------
(*) import dccp has been added
(*) DCCP is the recent advancement and generalization of DCP for convex-concave programs
(*) method='dccp' has been added as parameter to solve()
(*) Objective function has been changed to log() from logistic() - curvature is concave which is in conflict with
eisenberg-gale convex program in textbooks. Reason for this contradiction is unknown.
(*) But DCCP overcomes the DCP limitation and solve() prints converged solutions for objective functions
(*) logs have been committed to testlogs/
(*) CVXOPT solver has been installed but it does not solve the Eisenberg-Gale objective function. Only SCS solver
KKT conditions indirectly.

------------------------------------------------------------------------------------------------------------
1118. (FEATURE-DONE) Commits - Convex Optimization - DCCP - 22 August 2017
------------------------------------------------------------------------------------------------------------
(*) Verbose set to True for printing Splitting Conic Solver progress information
(*) logs committed to testlogs/

------------------------------------------------------------------------------------------------------------
1119. (FEATURE-DONE) Commits - Convex Optimization update - 29 August 2017
------------------------------------------------------------------------------------------------------------
(*) Removed hardcoded variable values in objective and constraints
(*) In the context of pricing, ECOS Error Metrics print the matrices of market clearing prices for goods
(Reference - pages 3072 and 3073 of https://web.stanford.edu/~boyd/papers/pdf/ecos_ecc.pdf - KKT conditions in ECO

------------------------------------------------------------------------------------------------------------
778. (FEATURE-DONE) Convex Optimization - Pricing Computation - 30 August 2017 - this section is an extended draft
------------------------------------------------------------------------------------------------------------
(*) Prices of Goods/Services have been computed explicitly from Karush-Kuhn-Tucker Conditions (1,2,3 and especiall
```

```
418  (*) References:
419          - Pages 106-108 of http://www.cs.cmu.edu/~sandholm/cs15-892F13/algorithmic-game-theory.pdf)
420          - KKT conditions and Conic Optimization- https://arxiv.org/pdf/1312.3039.pdf
421  (*) logs committed to testlogs/
422
423  ------------------------------------------------------------------------------------------------------------
424  1120. (FEATURE-DONE) KingCobra Kernelspace Messaging Driver for 4.13.3 64-bit kernel - 24 September 2017
425  ------------------------------------------------------------------------------------------------------------
426  (*) KingCobra driver in GitHub and SourceForge at present are 32-bit based on mainline 4.1.5 kernel
427  (*) Both USB-md and KingCobra kernel modules are subsidiaries of VIRGO kernel
428  (*) There is a necessity for 64-bit version of KingCobra for interoperability to VIRGO64 64-bit kernel on mainline
429  (*) This requires separate repository for KingCobra because of significant kernel function changes between 4.1.5 a
430  idiosyncrasies of 64-bit
431  (*) KingCobra driver has been rebuilt on 4.13.3 64-bit kernel after some changes to function prototypes and new ki
432  initialized with these commits
433  (*) KingCobra kernel sockets have been TLS-ed by kernel_setsockopt(TX_TLS) newly introduced in 4.13 kernel.
434  (*) After this complete request-reply traffic from VIRGO64 system calls to VIRGO64 queueing and KingCobra is encry
435
436  ------------------------------------------------------------------------------------------------------------
437  1121. (FEATURE-DONE) Commits - telnet - VIRGO64Queue - KingCobra64 -  25 September 2017
438  ------------------------------------------------------------------------------------------------------------
439  (*) Disk persisted KingCobra64 REQUEST-REPLY Queue written by VIRGO64 Queue to KingCobra64 telnet invocation after
440  has been committed
441
442  ------------------------------------------------------------------------------------------------------------
443  1122. (FEATURE-DONE) VIRGO64 Queueing Kernel Module Listener - KingCobra64 - 4.13.3 - 6 October 2017
444  ------------------------------------------------------------------------------------------------------------
445  (*) telnet client connection to VIRGO64 Queue and a subsequent workqueue routing (pub/sub) to KingCobra64 has been
446  (*) TX_TLS socket option has not been disabled and is a no-op because it has no effect on the socket.
447  (*) REQUEST_REPLY.queue for this routing from VIRGO64 queue and persisted by KingCobra64 has been committed to Kin
448
449  ------------------------------------------------------------------------------------------------------------
450  777. (FEATURE-DONE) KingCobra64 Neuro Electronic Currency transactional cloud move - Perfect Forward - 17 January
451  ------------------------------------------------------------------------------------------------------------
452  (#) Neuro Currency cloud perfect forward has been made transactional by wrapping it by Python Transaction Manager
453  Python Application Server)
454  (#) imports transaction python package and invokes begin() and commit() on subprocess call to neuro cloud move cli
455
456  ------------------------------------------------------------------------------------------------------------
457  776. (FEATURE) Concurrent Managed Workqueue(CMWQ), VIRGO64 Queueing and KingCobra64 messaging - 12 June 2019 - thi
```

```
458 ----------------------------------------------------------------------------------------------------
459 1. Existing workqueue underneath VIRGO64 queueing and requests routed by it to KingCobra64 messaging are old legac
460 have been revamped to Concurrent Managed Workqueue which supports concurrent messaging and lot of other options ir
461 2. create_workqueue() in VIRGO64 Queueing has been changed to alloc_workqueue() of Concurrent Managed Workqueue.
462 3. VIRGO64 Queueing request routing to KingCobra64 messaging has been tested with CMWQ and queueing log and kingcc
463 Queue have been committed to respective testlogs of the drivers
464 4. reading from stream has been disabled in virgo_kernel_analytics.h
465 5. Reference - CMWQ documentation - https://www.kernel.org/doc/html/v4.11/core-api/workqueue.html
466 6. Byzantine Fault Tolerance in KingCobra64 persisted queue can be made available by performant CMWQ and routing t
467 REQUEST_REPLY.queue by any of the practical BFT protocols available.
468 7. Most important application of CMWQ based VIRGO64-KingCobra64 is in the context of kernelspace hardware messagir
469 analytics driven embedded systems.
470 8. An example usecase which is a mix of sync and async I/O in kernelspace:
471         (*) Analytics Variables computed by userspace machine learning are read over socket stream by kernel_analy
472 exported kernelwide
473         (*) Some interested Drone driver in kernel (example PXRC) reads the analytics variables synchronously and
474         (*) VIRGO Queuing routes the queued messages to KingCobra64 driver
```

© 2020 Slashdot Media. All Rights Reserved.

Terms Privacy Opt Out Advertise