

TECHNICAL NOTE

GUIdock-VNC: Using a graphical desktop sharing system to provide a browser-based interface for containerized software

Varun Mittal, Ling-Hong Hung, Jayant Keswani, Daniel Kristiyanto, Sung Bong Lee and Ka Yee Yeung*

*Correspondence: kayee@uw.edu
Institute of Technology, University
of Washington, Tacoma, 1900
Commerce St, 98402 Tacoma,
United States of America
Full list of author information is
available at the end of the article

Abstract

Background: Software container technology such as Docker can be used to package and distribute bioinformatics workflows consisting of multiple software implementations and dependencies. However, Docker is a command line based tool and many bioinformatics pipelines consist of components that require a graphical user interface.

Findings: We present a container tool called GUIdock-VNC that uses a graphical desktop sharing system to provide a browser-based interface for containerized software. GUIdock-VNC uses the Virtual Network Computing protocol to render the graphics within most commonly used browsers. We also present a minimal image builder that can add our proposed graphical desktop sharing system to any Docker packages, with the end result that any Docker packages can be run using a graphical desktop within a browser. In addition, GUIdock-VNC uses the Oauth2 authentication protocols when deployed on the cloud.

Conclusions: As a proof-of-concept, we demonstrated the utility of GUIdock-noVNC in gene network inference. We benchmarked our container implementation on various operating systems and showed that our solution creates minimal overhead.

Keywords: Research Reproducibility; Docker; Containers; Software; Graphical user interface; bioinformatics

1 Findings

2 Background

3 Modern workflows in computational fields such as bioinformatics consist of multiple
4 software implementations each with their own set of dependencies. Software con-
5 tainer technology such as Docker (<http://www.docker.com>), package the depen-
6 dencies along with the software and provide a method to reproduce these complex
7 pipelines on multiple hardware and cloud platforms. For example, BioShaddock [1],

1
2
3
4
5
6 8 BioDocker and and Bioboxes [2] are two frameworks aimed at reproducibly deploy-
7
8 9 ing bioinformatics workflows using Docker containers.
9

10 Many bioinformatics pipelines have a component which requires a Graphical User
11 Interface (GUI) that can potentially limit the portability of the Dockerized work-
12 flows as different platforms use different methodologies to render the GUI. We have
13 previously described and implemented GUIDock-X11 [3], an X11 based methodol-
14 ogy for portably supporting GUI applications in containers on different platforms.
15 While the X11 based display method can be conveniently deployed in the local envi-
16 ronment by exposing a file socket from a container, deploying the image on a cloud
17 and accessing it remotely is non-trivial. In addition, on systems such as Windows
18 where there is no native X11 support, additional client software must be installed
19 by the user to render the X11 graphics locally. Here we describe GUIDock-VNC
20 which implements an improved browser-based solution which does not require the
21 user to map ports, configure firewalls or install any additional specialized software.
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

39 GUIDock-VNC uses the Virtual Network Computing (VNC) protocol [4] to ren-
40 der the graphics. Instead of transferring commands and allowing a local client to
41 render the graphics, VNC transfers a pre-rendered screen. Bandwidth requirements
42 render the graphics, VNC transfers a pre-rendered screen. Bandwidth requirements
43 are minimized by only transferring the differences between the current screen and
44 the last screen. This can actually be less chatty than the X11 methodology which
45 is constantly sending display commands. noVNC is a browser-based VNC client
46 implemented using HTML5 Canvas and WebSockets [5]. All modern browsers can
47 use the HTML5 based noVNC client to display the screen locally. The browser
48 transparently downloads the noVNC client from the container and becomes the
49 terminal, thus eliminating the need for the user to configure and install separate
50 software. This is a major advantage as the users of bioinformatics workflows are not
51 necessarily technically trained in configuring computer systems.
52
53
54
55
56
57
58
59
60
61
62
63
64
65

1
2
3
4
5
6 34 Most importantly, GUIDock-VNC also facilitates the deployment of Docker ap-
7
8 35 plications on the cloud. With a browser based solution, we also have access to
9
10 36 web-based authentication protocols such as Oauth2 [6] which allows for password-
11
12 37 less authentication using an email account. The host service is accessed and au-
13
14 38 thenticated through the HTTP/HTTPS port, greatly simplifying the configuration
15
16 39 necessary to support cloud based platforms.

17 18 19 40 *Our contributions*

20
21 41 We implemented GUIDock-VNC which adds and configures a software layer inside
22
23 42 a Docker container to allow applications to export a GUI using the VNC protocol.
24
25 43 When deployed on the cloud, authentication is provided using Oauth2. In addition,
26
27 44 we provide a set of minimal base images to allow the users to add the host graph-
28
29 45 ical desktop interface to any existing Dockerfiles. No client software installation is
30
31 46 necessary for the users as GUIDock-VNC uses the HTML5 noVNC browser-based
32
33 47 client to display the GUI. All our tools are publicly available on github.

34
35
36 48 We benchmarked the implementation on a real-world bioinformatics pipeline. Our
37
38 49 results showed that noVNC creates minimal overhead and GUIDock-VNC is superior
39
40 50 to our previous work GUIDock-X11 [3] and other virtual machine based deployment
41
42 51 solutions.

43 44 45 52 Related work

46 47 53 *Software containers and Docker*

48
49 54 A software container packages an application with everything it needs to run, in-
50
51 55 cluding supporting libraries and system resources. Containers differ from traditional
52
53 56 virtual machines (VMs) in that the resources of the operating system (OS) and not
54
55 57 the hardware are virtualized. In addition, multiple containers share a single OS
56
57 58 kernel, thus saving considerable resources over multiple VMs.

58
59 59 Linux has supported OS level virtualization for several years. Docker (*https* :
60
61 60 *://www.docker.com/*) is an open source project that provides tools to setup and
62
63
64
65

1
2
3
4
5
6 61 deploy Linux software containers. While Docker can run natively on Linux hosts, a
7
8 62 small Linux VM is necessary to provide the virtualization services on Mac OS and
9
10 63 Windows systems. On non-Linux systems, a single Docker container consists of a
11
12 64 mini-VM, the Docker software layer and the software container. However, multiple
13
14 65 Docker containers can share the same mini-VM, saving considerable resources over
15
16 66 using multiple individual VMs. Recently, support for OS level virtualization has
17
18 67 been added to Windows and the Macintosh operating systems (Mac OS). Beta
19
20 68 versions of Docker for both Windows and Mac OS are now available that allow
21
22 69 Docker to run natively. Subsequently, these beta versions allow native Windows
23
24 70 and Mac OS software to be containerized and deployed in a similar manner [7].
25
26 71 Docker containers therefore provide a convenient and light method for deploying
27
28 72 open-source workflows on multiple platforms.
29
30
31
32
33

34 73 *GUIDock-X11*

35
36
37 74 Although Docker provides a container with the original software environment, the
38
39 75 host system, where the container software is executed, is responsible for rendering
40
41 76 graphics. Our previous work, GUIDock-X11 [3], is one of the solutions in bridging the
42
43 77 graphical information from user and Docker containers by using the X11 common
44
45 78 graphic interface. GUIDock-X11 passes the container X11 commands to a host X11
46
47 79 client which renders the GUI. Security is handled by encrypting the commands
48
49 80 through secure shell (ssh) tunneling. We demonstrated the use of GUIDock-X11 [3]
50
51 81 for systems biology applications, including Bioconductor packages written in R,
52
53 82 C++ and Fortran, and Cytoscape, a stand-alone java-based application with a
54
55 83 graphical user interface. Neither Windows nor Mac OS use X11 natively to render
56
57 84 their graphics. Additional software such as MobaXterm [8] or SoCat [9] are needed
58
59 85 to emulate X11 and locally render the graphics commands exported by the Docker
60
61 86 container. However, a major advantage of the X11 method is that the commands
62
63
64
65

to render the graphics and not the graphics themselves are transmitted, potentially reducing the total bandwidth required.

Table 1 summarizes the differences between GUIDock-VNC and our previous work GUIDock-X11.

Table 1 Comparison between GUIDock-X11 and GUIDock-VNC

Feature	GUIDock-X11	GUIDock-VNC
Can be deployed on phones/tablets?	No	Yes
Security	ssh-tunnel	OAuth2
Bandwidth	Low	Low to Medium
Cloud integration difficulty	Medium	Simple
Dockerfile setup	Manual editing	Automatic conversion of base Docker images

Case study: Inference of gene networks

The inference of gene networks is a fundamental challenge in systems biology. We use gene network inference as a case study to demonstrate that GUIDock-X11 and GUIDock-VNC can be used to yield reproducible results from bioinformatics workflows. We have previously developed inference methods using a regression-based framework, in which we search for candidate regulators (i.e. parent nodes) for each target gene [10, 11, 12]. Our methods are implemented in R, C++, and Fortran, and the implementation is available as a Bioconductor package called *networkBMA* [13] ([http : //bioconductor.org/packages/release/bioc/html/networkBMA.html](http://bioconductor.org/packages/release/bioc/html/networkBMA.html)). In order to visualize the resulting gene networks, we previously developed a Cytoscape app called CyNetworkBMA [14], ([http : //apps.cytoscape.org/apps/cynetworkbma](http://apps.cytoscape.org/apps/cynetworkbma)). Cytoscape is a Java-based stand-alone application with a GUI to analyze and visualize graphs and networks [15, 16, 17]. Our app, CyNetworkBMA [14], integrates our *networkBMA* Bioconductor package into Cytoscape, allowing the user to directly visualize the resulting gene networks inferred from *networkBMA* using the Cytoscape utilities. The integration of multiple pieces of software, each with its own

1
2
3
4
5
6 107 software dependencies, make CyNetworkBMA an ideal proof-of-concept application
7
8 108 for the illustration of the utility of GUIDock-VNC.
9

109 Implementation of GUIDock-VNC

110 Figure 1 shows an overview of GUIDock-VNC.

Figures/Architecture.pdf

Figure 1 Architecture overview of GUIDock-VNC. In the proposed architecture, each container is a self-contained web-server that can be accessed using a single port. When deployed on the cloud, the services can be accessed using the cloud provider's network address translation (NAT) mechanism. Each container is also capable of OAuth2 authentication that can be enabled while deploying the application. Once enabled, the user will be required to sign-in through an identity provider (such as Google in the current prototype). After the authentication, the user browser will be automatically redirected to the application.

111 *Virtual Network Computing (VNC)*

112 VNC is a framebuffer based protocol that was written to view and control remote
113 desktop over the internet [4]. VNC is essentially a server program that attaches to
114 a display server like X11 and creates a proxy between the client and the display
115 server. The proxy server takes in input from the client and relays it to the display
116 server while at the same time the display server sends pre-rendered display images
117 to the client. VNC is thus a network intensive protocol, although the amount of data
118 transferred back and forth can be reduced by using various compression technologies
119 on the transfer layer. In our implementation, we use Xvnc/Xvfb (X virtual frame
120 buffer) [18] to provide a lightweight VNC/X11 display server.

121 *noVNC*

122 noVNC is a browser based VNC client [5]. The name *noVNC* means that the tra-
123 ditional VNC client is not needed, and that a modern browser with HTML5 and
124 websockets support can be used to access and control a remote VNC server. This
125 noVNC technology is particularly interesting as almost all browsers, both desktop

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

1
2
3
4
5 and mobile versions have HTML5 extensions built-in. An additional layer on the
6
7 host is required for the VNC server to communicate through websockets. We use
8
9 nginx (<https://www.nginx.com/>), a fast and light reverse-proxy web server for
10
11 this purpose.
12
13

14 *Authentication methods: Oauth2*

15
16 For containers that are not deployed locally, i.e. on a network or cloud, security is
17
18 a concern as the traffic between the viewer and server can be seen by anyone with
19
20 access to the network. Oauth2 [6] is an authentication method that is commonly
21
22 used by major corporations to validate third party applications. Specifically, Oauth
23
24 allows users to log onto third party websites using their existing Google, Twitter or
25
26 Facebook accounts, thus avoiding the creation of additional accounts for the third
27
28 party websites. In the present era, where it is extremely difficult to host identity
29
30 services and securing communication, public authentication services like Oauth2
31
32 plays an important role. Providers like Google, Facebook, LinkedIn can be used to
33
34 validate any user registered with an email at one of these websites. In our prototype
35
36 we have created a prototype for Google identity server. Each container can be forced
37
38 to login through one of these public providers.
39
40
41
42

43 *Automatic conversion of base Docker images*

44
45 To add the noVNC graphical desktop to a Docker image, the converted image
46
47 requires a web-server, a headless display server running inside the container and
48
49 a VNC server (see Figure 2). The webserver is required to serve javascript based
50
51 noVNC client and the headless display server routes all drawing instructions to the
52
53 VNC server which are then sent to the client running the browser based noVNC
54
55 javascript client. Due to the three active components running inside the container,
56
57 we generate a bash script to work as the entry point for the container. Therefore
58
59 in order to assist users and to let them start application dependent services we
60
61 have created a tool to bootstrap standard images with noVNC capability. The
62
63
64
65

1
2
3
4
5
6 153 tool accepts a json script as input with defined parameters for files to be copied,
7
8 154 additional software to be installed (using apt-get) and the location of the startup
9
10 155 script which is then tied into the entrypoint.

11
12
13 156 The tool is extensible and can be used to define all dockerfile parameters as
14
15 157 standard run commands.

Figures/Infrastructure.pdf

Figure 2 Services running inside container. Apart from user applications, there are two web services running inside the container and a reverse proxy (Nginx) to act as an interface for the container. The first web-service is the noVNC interface connected to the VNC server. The noVNC server is a python plus javascript application framework used for establishing web-sockets for VNC packet interchange. The second web-service is an optional broker service that helps in exchanging data through a datastore interface (Mongodb) and a message passing queue (RabbitMQ).

158 Applications

159 We illustrated the utility of GUIDock-VNC in a proof-of-concept case study of gene
160 network inference. Specifically, we applied GUIDock-VNC to a RNAseq dataset con-
161 sisting of 675 human cancer cell lines [19]. We downloaded the variance stabilized
162 version of the normalized RNAseq data, and extracted a subset of 84 genes that be-
163 long to 21 cancer-related pathways (see Supplementary Table 12 in Klijn *et al.* [19]).
164 We applied the ScanBMA [12] gene network inference algorithm as implemented in
165 the CyNetworkBMA app from within the GUIDock-VNC container.

166 We show that we get identical results after deploying the package across different
167 browsers on different operating systems. Figure 3 shows screenshots of using (a)
168 Internet Explorer on Windows 8.1, (b) Google Chrome on Ubuntu Linux, (c) Safari
169 on MacOS, and (d) Google Chrome on Android. To summarize, we demonstrate
170 reproducibility of analytical results when GUIDock-VNC is deployed on different
171 browsers and different operating systems.

Figures/Screenshots.pdf

Figure 3 CyNetworkBMA on various browsers and operating systems: (a) Internet Explorer on Windows 8.1, (b) Google Chrome on Ubuntu Linux, (c) Safari on MacOS, and (d) Google Chrome on Android. Given that the interaction to the container is made through a browser, any device with a browser supporting HTML5 with consistent user experience and results, even on mobile devices such as Android and iOS devices. In the case of mobile devices, the Docker container runs on remote machines (e.g. on a VM instance on cloud).

172 Benchmarking computational efficiency

173 Since we have added extra services to the container, it is essential to investigate
174 the performance overhead introduced by these additional services and the container.
175 We conducted an extensive empirical study on comparing performance over different
176 platforms with different hypervisors using GUIDock-X11 and GUI-VNC. As a base-
177 line, we compared the performance of running GUIDock-X11 and GUIDock-VNC
178 on Docker containers to running the CyNetworkBMA app natively. In addition, we
179 also compared our results to running the CyNetworkBMA via a virtual machine
180 (VM). We experimented the performance of each of these four options (native,
181 GUIDock-X11, GUIDock-VNC, VM) on the Linux, Macintosh and Windows oper-
182 ating systems.

183 In our benchmarking experiments, we used the time series data of the first net-
184 work from the DREAM 4 crowd sourcing challenge [20, 21]. This simulated dataset
185 consists of 100 genes across 21 time points. In order to account for variability in
186 our empirical experiments, we repeated each configuration, *i.e.* each (OS, option)
187 pair, four times. These replicated experiments are represented by "RUN1", "RUN2",
188 "RUN3", "RUN4" in Table 2. In addition, we added warm-up runs to ensure steady-
189 state execution time.

190 Table 2 shows a consistent minimal overhead of running the proposed container,
191 which is only marginally higher than running application natively. In particular, we
192 computed the ratio of the average execution time over the 4 runs to the "native"

193 baseline execution time. Figure 4 shows the ratio of the average execution time of
 194 each of GUIDock-X11, GUIDock-VNC and VM to the baseline "native" on the Linux,
 195 MacOS and Windows operating systems. On the Linux, MacOS and Windows oper-
 196 ating systems, we observed comparable execution time for both GUIDock-X11 and
 197 GUIDock-VNC.

Table 2 Execution time in empirical study across the Linux, Macintosh and Windows operating systems. "Native" means running the CyNetworkBMA app natively on the corresponding OS. "VM" means running the CyNetworkBMA app from a virtual machine on the corresponding OS. The column "average" is the average execution time over the 4 runs. The column "ratio" is the ratio of the average running time to the "native" baseline.

Platform	Environment	RUN 1	RUN 2	RUN 3	RUN 4	Average	Ratio
Linux	Native	104	100	97	102	100	1
	GUIDock-X11	130	131	131	130	130	1.29
	GUIDock-VNC	134	133	133	134	134	1.39
	VM	187	185	186	185	186	1.92
MacOS	Native	97	97	96	96	96	1
	GUIDock-X11	120	116	118	123	119	1.23
	GUIDock-VNC	123	121	120	124	122	1.26
	VM	148	143	151	150	148	1.54
Windows	Native	125	127	130	128	127	1
	GUIDock-X11	155	157	155	156	155	1.22
	GUIDock-VNC	157	160	162	161	160	1.25
	VM	179	179	182	184	181	1.42

Figures/execRatios2.png

Figure 4 The bar graph shows the ratio of the average execution time to the baseline of running CyNetworkBMA natively for each of Linux, MAC OS and Windows. The first value for each platform is the ratio of the native execution runtime to itself, and therefore is always equal to 1. The remaining three values correspond to the ratio of the average execution time for GUIDock-X11, GUIDock-VNC and VM respectively to the baseline "native".

198 Discussion

199 We present a container tool called GUIDock-VNC that uses a graphical desktop
 200 sharing system to provide a browser-based interface for containerized software. The
 201 merits of GUIDock-VNC are summarized as follows:

202 *No installation on client side.* Our proposed container GUIDock-VNC is a self
 203 contained display server with user application and a HTML5 based VNC client,

1
2
3
4
5
6 204 which can be accessed using the web-browser. Therefore, any HTML5 capable
7
8 205 browser is sufficient on the client side. Almost all modern popular browsers such as
9
10 206 Mozilla Firefox, Webkit (Apple Safari and Google Chrome), Microsoft Edge support
11
12 207 HTML5 extentions.

13
14 208 *Mobile capable.* Since HTML5 extentions that are required to access noVNC are
15
16 209 also supported on mobile versions on modern web-browsers, our GUIDock-VNC
17
18 210 solution is also accessible through a mobile device such as phones and tablets. The
19
20 211 container is a stateful solution, such that it preserves session information even after
21
22 212 a disconnected user session. Therefore, researchers can access the container on the
23
24 213 go and subsequently continue working on the workstation in the labs.

25
26 214 *Cloud integration.* There is a self-contained webserver inside the GUIDock-VNC
27
28 215 container which is required to access the applications. Our GUIDock-VNC container
29
30 216 can be hosted on the cloud by using a reverse proxy or simple NAT (Network Ad-
31
32 217 dress Translation) rule to passon any incoming request to the container. We tested
33
34 218 the solution with major cloud vendors such as Amazon AWS, Google Cloud and
35
36 219 Microsoft Azure and the NAT forwarding works without any network interference.

37
38
39 220 *Security using OAuth2.* To ensure secure access to the container, we are using
40
41 221 OAuth2 from Google (replacable by any other identy provider such as LinkedIn,
42
43 222 Facebook, Twitter etc.) to authenticate users by ensuring the identity registered
44
45 223 with container is owned by the user requesting container access. This can be done
46
47 224 by providing email id, provider user id and secret password as parameters to the
48
49 225 container. If these parameters are provided while initiating the container, the broker
50
51 226 will redirect the user to the identity provider to verify the email address. Once
52
53 227 authenticated the identity provider redirects the user to the container.

54 55 56 57 228 **Availability and requirements**

- 58
59 229 • Project name: GUIDock-VNC
 - 60
61 230 • Project home page: <https://github.com/biodepot>
- 62
63
64
65

- 231 • Contents available for download: Docker Images, Dockerfiles, installation
232 scripts and execution scripts.
- 233 • Operating system(s): Linux, Mac OS X, Microsoft Windows. Specifically, we
234 tested GUIdock-VNC on
 - 235 – Linux: Fedora, Ubuntu 15.04
 - 236 – Mac OS X: 10.9, 10.10
 - 237 – Microsoft Windows: 7, 8.1, 10
 - 238 – Android, IOS
- 239 • Programming language(s): Python, HTML, JavaScript
- 240 • Browsers tested
 - 241 – Google Chrome
 - 242 – Firefox
 - 243 – Safari
- 244 • License: MIT License

245 **Abbreviations**

246 GUI: graphical user interface NAT: network address translation VM: virtual ma-
247 chine VNC: Virtual Network Computing

248 **Declarations**

249 **Acknowledgments**

250 Varun Mittal, Ling-Hong Hung and Ka Yee Yeung are supported by NIH grant
251 U54-HL127624. We would like to thank Microsoft Azure for computing resources.
252 Daniel Kristiyanto is sponsored by the U.S. Department of State and American-
253 Indonesian Exchange Foundation (AMINEF) through Fulbright Scholarship, and
254 gratefully acknowledges funding provided by the University of Washington in the
255 form of full tuition waivers.

256 Competing interests

257 The authors declare that they have no competing interests.

258 Author's contributions

259 VM is the primary developer for GUIDOCK-VNC. KYY coordinated the manuscript
260 preparation. VM, LHH and KYY drafted the manuscript. LHH designed the bench-
261 marking experiments. VM, LHH, DK, JK and SBL performed the benchmarking
262 experiments. DK and SBL contributed to the comparison of GUIDOCK-VNC and
263 GUIDOCK-X11. VM, JK and SBL contributed to the writing of the user manual.
264 VM and DK made the videos in additional data files. All authors tested GUIDOCK-
265 VNC, read and approved the final manuscript.

266 References

- 267 1. Moreews, F., Sallou, O., Ménager, H., Le Bras, Y., Monjeaud, C., Blanchet, C., Collin, O.: Bioshadock: a
268 community driven bioinformatics shared docker-based tools registry. *F1000Research* **4**, 1443 (2015)
- 269 2. Bio Docker: Docker for Bioinformatics. <http://biodocker.org/>
- 270 3. Hung, L.-H., Kristiyanto, D., Lee, S.B., Yeung, K.Y.: Guidock: Using docker containers with a common
271 graphics user interface to address the reproducibility of research. *PLOS One* **11**(4), 0152686 (2016)
- 272 4. Richardson, T., Stafford-Fraser, Q., Wood, K.R., Hopper, A.: Virtual network computing. *IEEE Internet*
273 *Computing* **2**(1), 33 (1998)
- 274 5. noVNC by kanaka. <http://kanaka.github.io/noVNC/>
- 275 6. OAuth. <http://oauth.net/>
- 276 7. Docker for Mac and Windows Beta: the simplest way to use Docker on your laptop.
277 <https://blog.docker.com/2016/03/docker-for-mac-windows-beta/>
- 278 8. MobaXterm: Enhanced terminal for Windows with X11 server, tabbed SSH client, network tools and much
279 more. <http://mobaxterm.mobatek.net/>
- 280 9. socat: Multipurpose relay (SOcket CAT). <http://www.dest-unreach.org/socat/doc/socat.html>
- 281 10. Yeung, K.Y., Dombek, K.M., Lo, K., Mittler, J.E., Zhu, J., Schadt, E.E., Bumgarner, R.E., Raftery, A.E.:
282 Construction of regulatory networks using expression time-series data of a genotyped population. *Proceedings*
283 *of the National Academy of Sciences* **108**(48), 19436–19441 (2011)
- 284 11. Lo, K., Raftery, A., Dombek, K., Zhu, J., Schadt, E., Bumgarner, R., Yeung, K.Y.: Integrating external
285 biological knowledge in the construction of regulatory networks from time-series expression data. *BMC Systems*
286 *Biology* **6**(1), 101 (2012)
- 287 12. Young, W.C., Raftery, A.E., Yeung, K.Y.: Fast Bayesian inference for gene regulatory networks using
288 ScanBMA. *BMC Systems Biology* **8**(1), 47 (2014)
- 289 13. Yeung, K.Y., Fraley, C., Young, W.C., Bumgarner, R., Raftery, A.E.: Bayesian model averaging methods and R
290 package for gene network construction. In: *Big Data Analytic Technology For Bioinformatics and Health*
291 *Informatics (KDDBHI), Workshop at the 20th ACM SIGKDD Conference on Knowledge Discovery and Data*
292 *Mining (KDD)* (2014)

- 1
2
3
4
5
6 293 14. Fronczuk, M., Raftery, A.E., Yeung, K.Y.: Cynetworkbma: a cytoscape app for inferring gene regulatory
7 294 networks. Under revision
- 8
9 295 15. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker,
10 296 T.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome*
11 297 *Research* **13**(11), 2498–2504 (2003)
- 12
13 298 16. Christmas, R., Avila-Campillo, I., Bolouri, H., Schwikowski, B., Anderson, M., Kelley, R., Landys, N.,
14 299 Workman, C., Ideker, T., Cerami, E., Sheridan, R., Bader, G.D., Sander, C.: Cytoscape: A Software
15 300 Environment for Integrated Models of Biomolecular Interaction Networks, pp. 12–16 (2005).
16 301 <http://educationbook.aacrjournals.org/cgi/content/full/2005/1/12>
- 17
18 302 17. Cline, M.S., Smoot, M., Cerami, E., Kuchinsky, A., Landys, N., Workman, C., Christmas, R., Avila-Campillo, I.,
19 303 Creech, M., Gross, B., Hanspers, K., Isserlin, R., Kelley, R., Killcoyne, S., Lotia, S., Maere, S., Morris, J., Ono,
20 304 K., Pavlovic, V., Pico, A.R., Vailaya, A., Wang, P.-L., Adler, A., Conklin, B.R., Hood, L., Kuiper, M., Sander,
21 305 C., Schmulevich, I., Schwikowski, B., Warner, G.J., Ideker, T., Bader, G.D.: Integration of biological networks
22 306 and gene expression data using cytoscape. *Nature Protocols*, 2366–2382 (2007)
- 23
24 307 18. Xvfb: virtual framebuffer X server for X Version 11.
25 308 <https://www.x.org/releases/X11R7.6/doc/man/man1/Xvfb.1.xhtml>
- 26
27 309 19. Klijn, C., Durinck, S., Stawiski, E.W., Haverty, P.M., Jiang, Z., Liu, H., Degenhardt, J., Mayba, O., Gnad, F.,
28 310 Liu, J., Pau, G., Reeder, J., Cao, Y., Mukhyala, K., Selvaraj, S.K., Yu, M., Zynda, G.J., Brauer, M.J., Wu,
29 311 T.D., Gentleman, R.C., Manning, G., Yauch, R.L., Bourgon, R., Stokoe, D., Modrusan, Z., Neve, R.M., de
30 312 Sauvage, F.J., Settleman, J., Seshagiri, S., Zhang, Z.: A comprehensive transcriptional portrait of human
31 313 cancer cell lines. *Nature Biotechnology* **33**, 306–312 (2015)
- 32
33 314 20. Marbach, D., Schaffter, T., Mattiussi, C., Floreano, D.: Generating Realistic In Silico Gene Networks for
34 315 Performance Assessment of Reverse Engineering Methods. *Journal of Computational Biology* **16**(2), 229–239
35 316 (2009)
- 36
37 317 21. Marbach, D., Prill, R.J., Schaffter, T., Mattiussi, C., Floreano, D., Stolovitzky, G.: Revealing strengths and
38 318 weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences* **107**(14),
39 319 6286–6291 (2010)

320 Additional Files

- 321 Additional file 1 — User manual for GUIDock-VNC.
- 322 Additional file 2 — Video of GUIDock-VNC demo
- 323 Available on Youtube <https://youtu.be/iaVPnLhOLg0>.
- 324 Additional file 3 — Video of deploying GUIDock-VNC on the cloud
- 325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365

Figure Captions

Figure 1

Architecture overview of GUIdock-VNC. In the proposed architecture, each container is a self-contained web-server that can be accessed using a single port. When deployed on the cloud, the services can be accessed using the cloud provider's network address translation (NAT) mechanism. Each container is also capable of OAuth2 authentication that can be enabled while deploying the application. Once enabled, the user will be required to sign-in through an identity provider (such as Google in the current prototype). After the authentication, the user browser will be automatically redirected to the application.

Figure 2

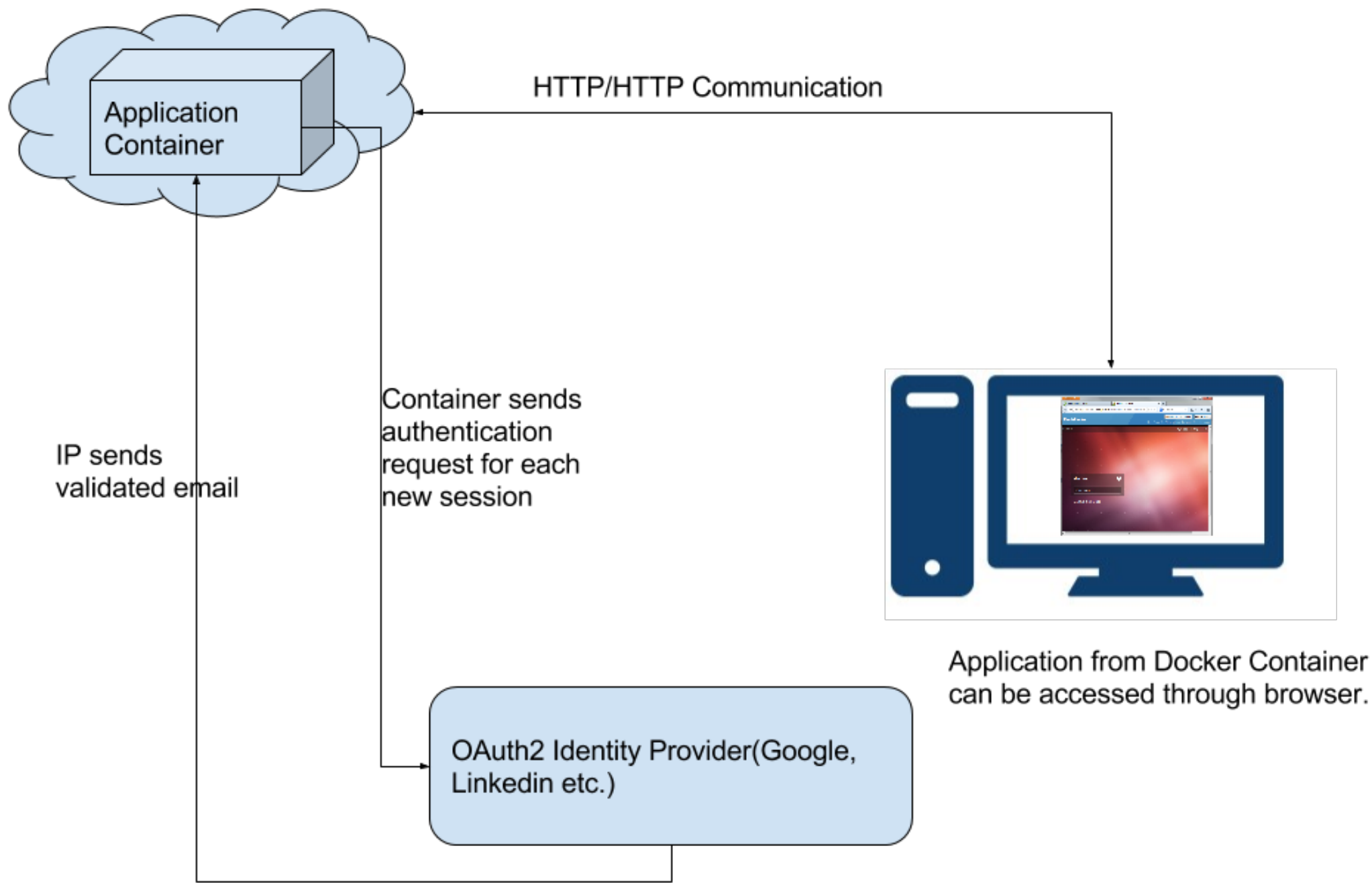
Services running inside container. Apart from user applications, there are two web services running inside the container and a reverse proxy (Nginx) to act as an interface for the container. The first web-service is the noVNC interface connected to the VNC server. The noVNC server is a python plus javascript application framework used for establishing web-sockets for VNC packet interchange. The second web-service is an optional broker service that helps in exchanging data through a datastore interface (Mongodb) and a message passing queue (RabbitMQ).

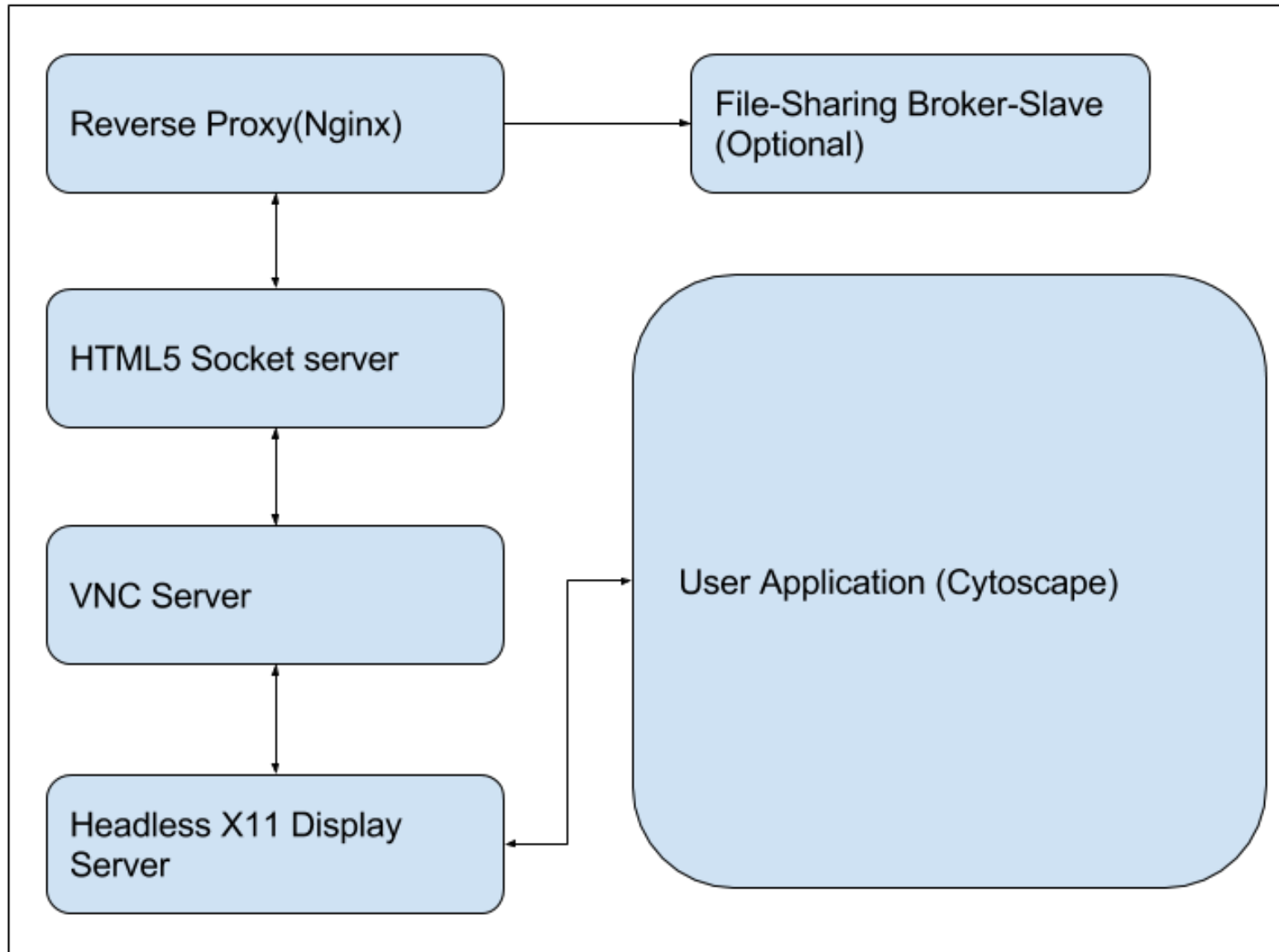
Figure 3

CyNetworkBMA on various browsers and operating systems: (a) Internet Explorer on Windows 8.1, (b) Google Chrome on Ubuntu Linux, (c) Safari on MacOS, and (d) Google Chrome on Android. Given that the interaction to the container is made through a browser, any device with a browser supporting HTML5 with consistent user experience and results, even on mobile devices such as Android and iOS devices. In the case of mobile devices, Docker run on other machines (e.g. on a VM instance on cloud).

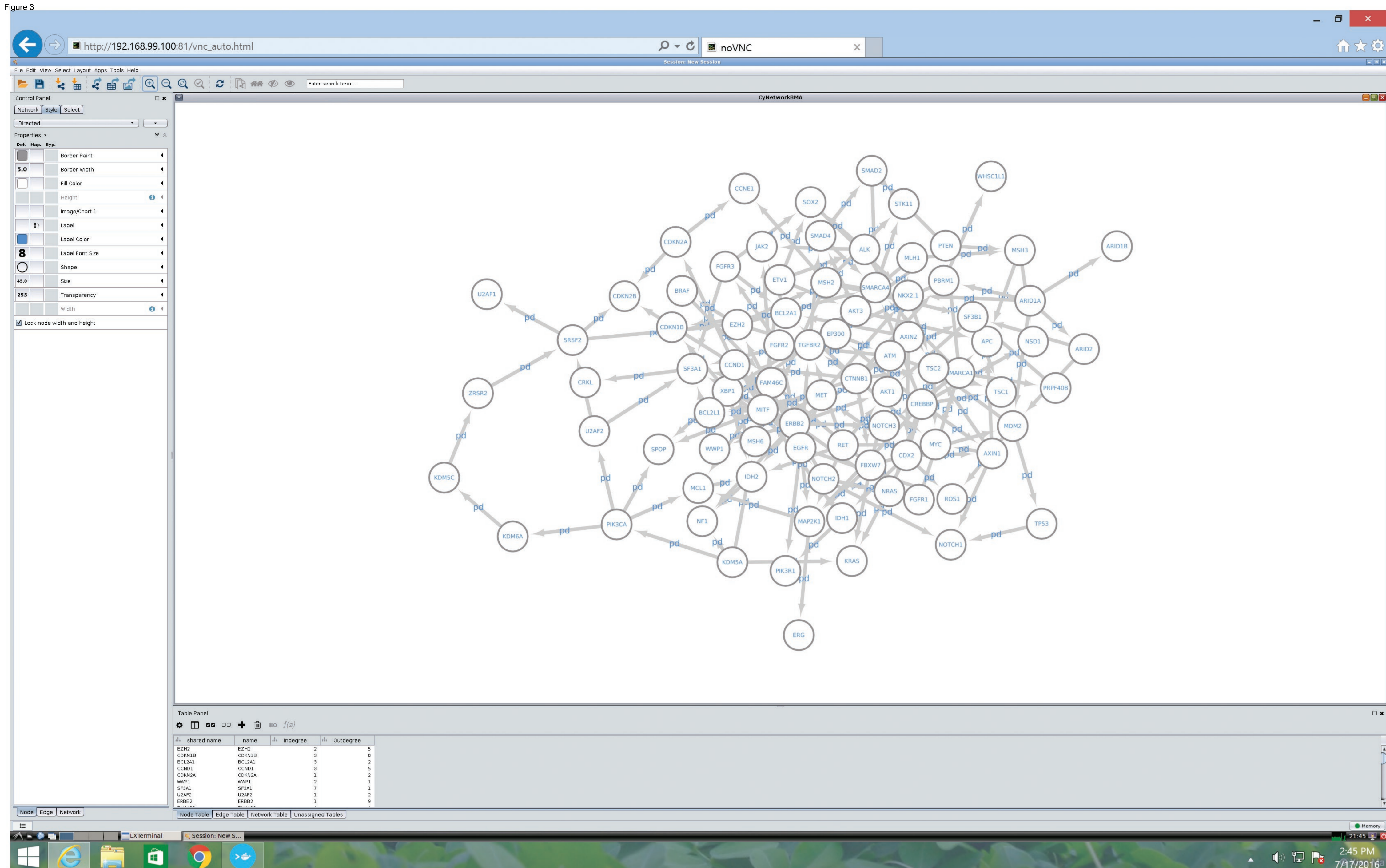
1
2
3
4
5
6 351 **Figure 4**

7
8 352 The bar graph shows the ratio of the average execution time to the baseline of
9
10 353 running CyNetworkBMA natively for each of Linux, MAC OS and Windows. The
11
12 354 first value for each platform is the ratio of the native execution runtime to itself, and
13
14 355 therefore is always equal to 1. The remaining three values correspond to the ratio of
15
16 356 the average execution time for GUIDock-X11, GUIDock-VNC and VM respectively
17
18 357 to the baseline "native".
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

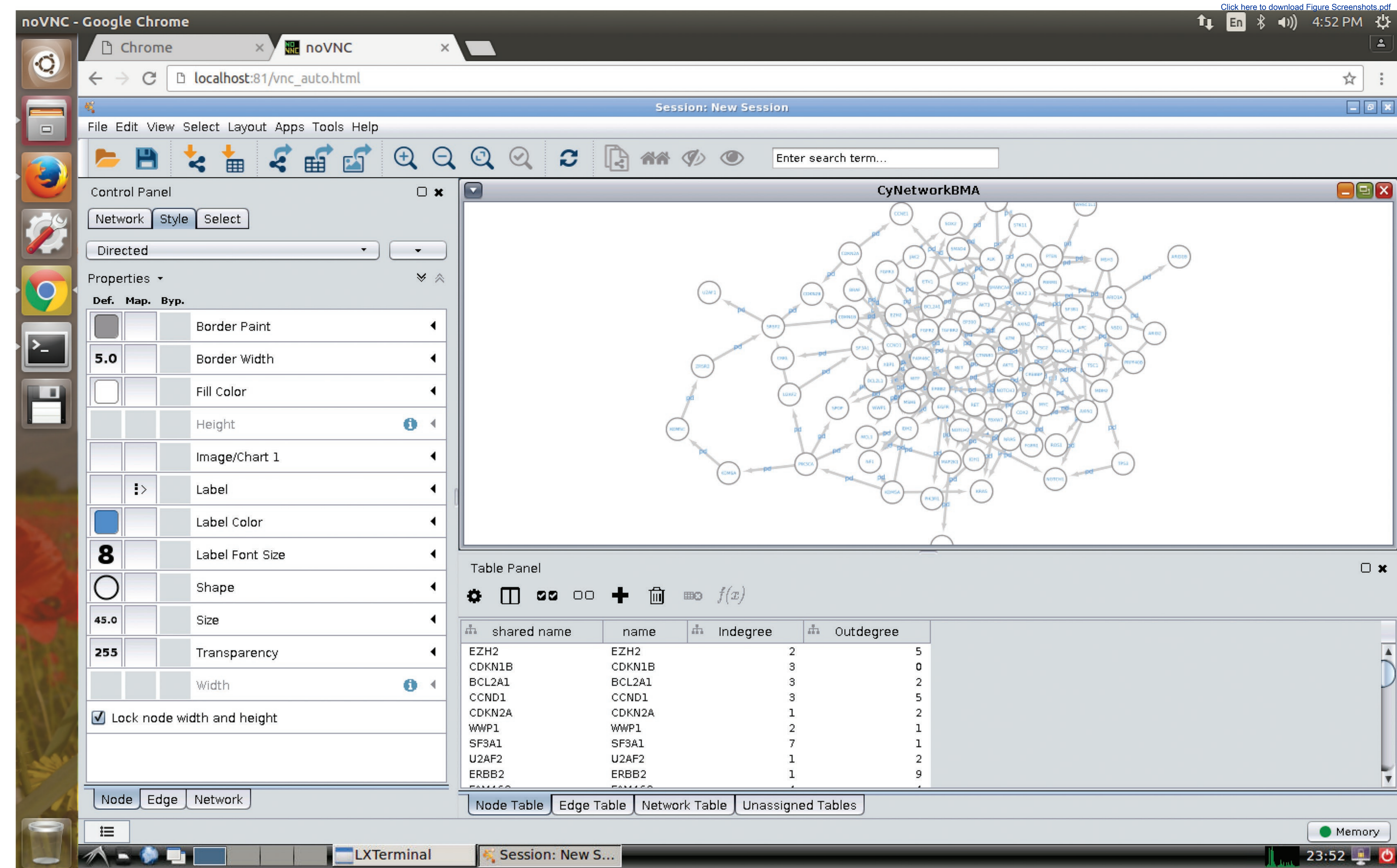




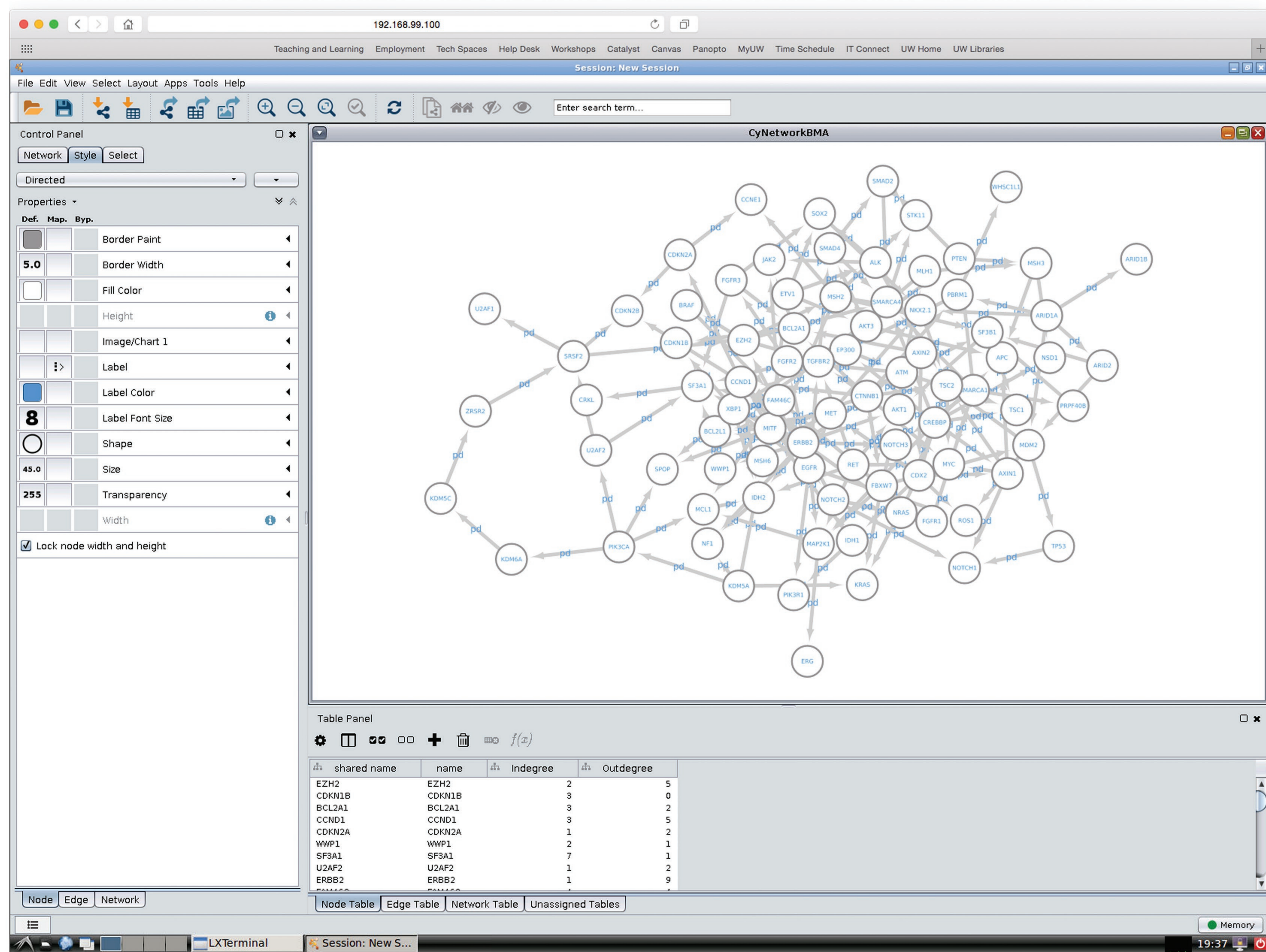
Docker Application Image



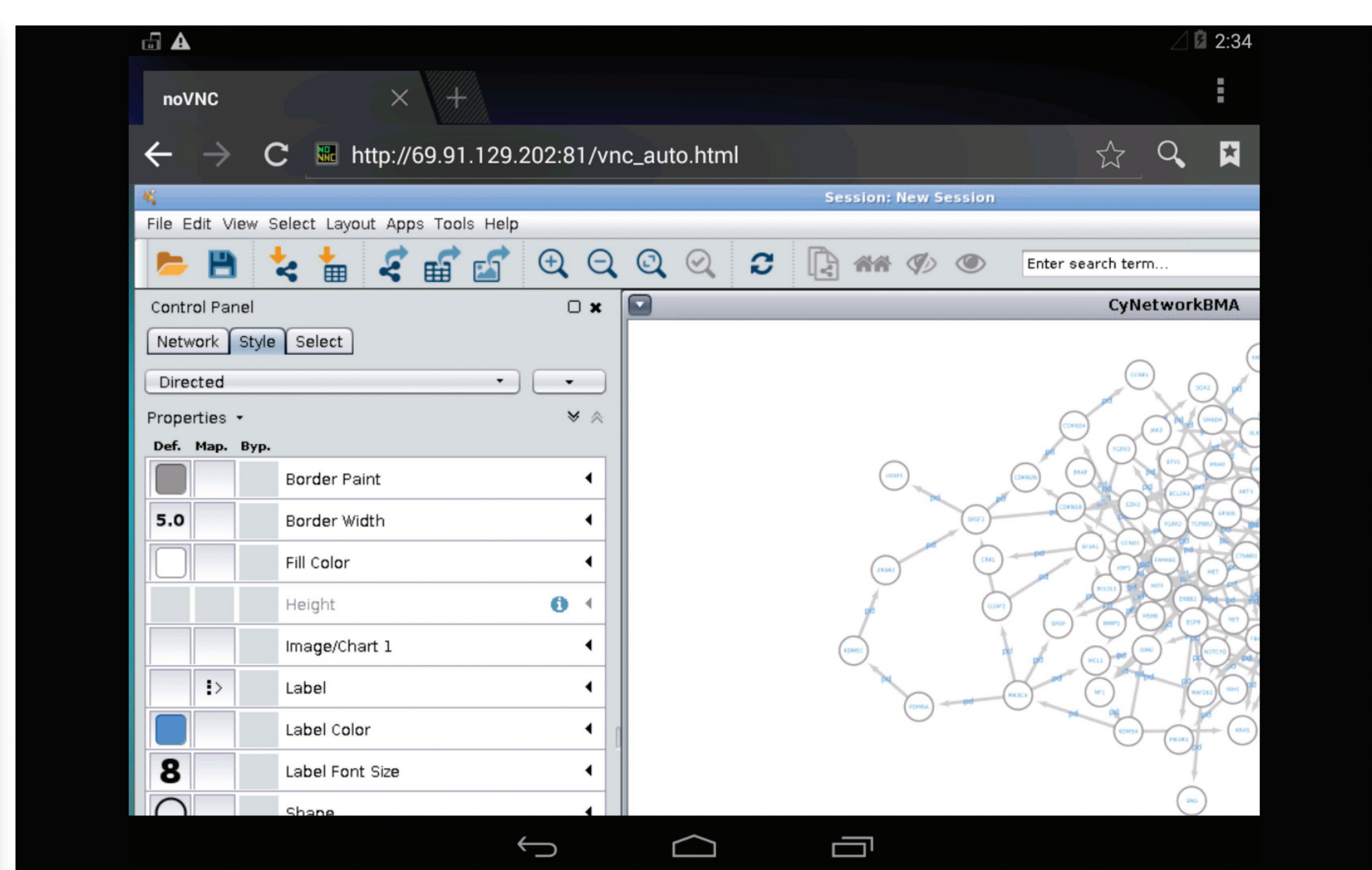
(a)



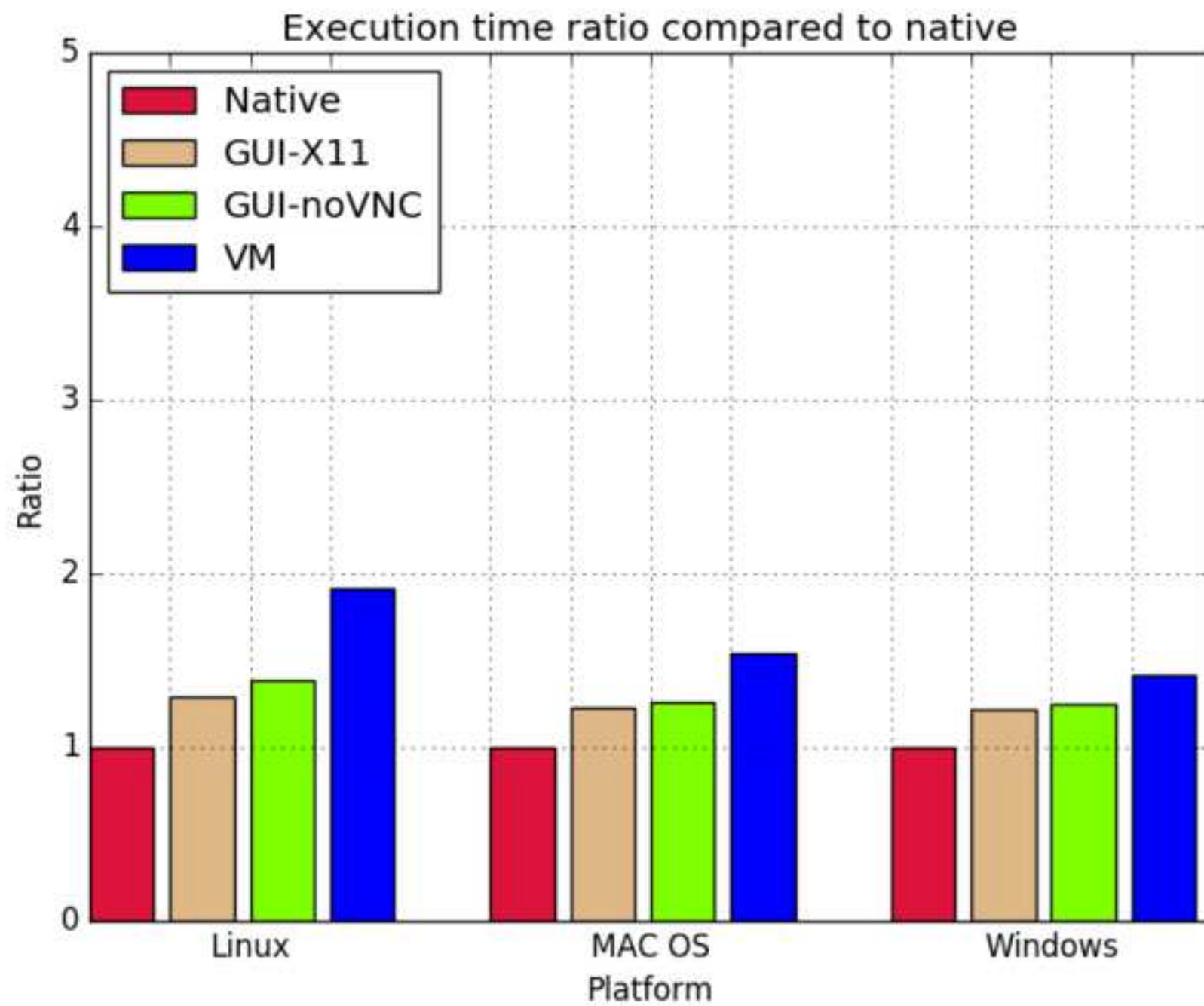
(b)



(c)



(d)





Click here to access/download
Supplementary Material
UserManual.pdf





Click here to access/download
Supplementary Material
GUIDock-VNC Demo.mov





Click here to access/download
Supplementary Material
noVNC cloud.mp4





Click here to access/download
Supplementary Material
bmc_article.pdf

