Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

Analisis Vulnerability Management Pada Container Docker Menggunakan *Opensource Scanner* Berdasarkan Standar *Cyber Resilience Review* (CRR)

Milenia Oktaviana*, Adityas Widjajarto, Ahmad Almaarif

Fakultas Rekayasa Industri, Sistem Informasi, Universitas Telkom, Bandung, Indonesia
Email: [1,*]mileniaari@student.telkomuniversity.ac.id, [2]adtwjrt@telkomuniversity.co.id,
[3]ahmadalmaarif@telkomuniversity.co.id
Email Penulis Korespondensi: mileniaari@student.telkomuniversity.ac.id
Submitted: **02/09/2022**; Accepted: **26/09/2022**; Published: **30/09/2022**

Abstrak—Teknologi container yang banyak digunakan untuk menyediakan layanan IT salah satunya adalah Docker. Kerentanan pada teknologi container yaitu Docker membutuhkan pengelolaan secara khusus. Pengelolaan kerentanan ini dapat dilakukan secara teknis dengan software vulnerability scanner dan panduan standar Cyber Resilience Review (CRR). Eksperimen dilakukan dengan scanner Aquasec dan Anchore yang melakukan vulnerability scanning pada dua sistem Docker Images. Dua sistem rentan tersebut memiliki perbedaan versi yaitu versi – 1 dan versi – 2. Elemen software pada versi – 2 memiliki tingkat versioning lebih tinggi daripada versi – 1. Data percobaan berupa vulnerability report dianalisis berdasarkan Cyber Resilience Review (CRR) yang berfokus pada empat tahap yaitu Define a Strategy, Develop a Plan, Implement the Capability, Assess and Improve the Capability. Sehingga didapatkan hasil Category Vulnerability yaitu 30 Closed Vulnerability, 10 Open Vulnerability, dan 13 Newly Vulnerability. Kelanjutan penelitian ini dapat menggunakan aspek Patch Management dengan tools software yang lebih bervariasi.

Kata Kunci: Docker; Vulnerability; Scanner; Cyber Resilience Review (CRR)

Abstract–One of the most widely used container technologies to provide IT services is Docker. The vulnerability in container technology, namely Docker, requires special management. Management of this vulnerability can be done technically with a software vulnerability scanner and standard Cyber Resilience Review (CRR) guidelines. Experiments were carried out with Aquasec and Anchore scanners that performed vulnerability scanning on two Docker Images systems. The two vulnerable systems have different versions, namely version – 1 and version – 2. The software elements in version – 2 have a higher versioning level than version – 1. Experimental data in the form of vulnerability reports are analyzed based on Cyber Resilience Review (CRR) which focuses on four stages namely Define a Strategy, Develop a Plan, Implement the Capability, Assess and Improve the Capability. So that the results of Category Vulnerability are obtained, namely 30 Closed Vulnerability, 10 Open Vulnerability, and 13 Newly Vulnerability. Continuation of this research can use aspects of Patch Management with more varied software tools.

Keywords: Docker; Vulnerability; Scanner; Cyber Resilience Review (CRR)

1. PENDAHULUAN

Aplikasi berbasis *website* berkembang sangat pesat seiring dengan perkembangan komputer dan internet saat ini. Pengguna yang memiliki layanan aplikasi *server-client* atau aplikasi berbasis *website* yang bersifat *public* banyak menggunakan layanan *cloud computing* untuk *hosting* aplikasinya. Virtualisasi dan *container* merupakan bagian yang sangat penting dalam penerapan *cloud computing*, karena sangat berpengaruh pada efisiensi pengelolaan sumber daya infrastruktur *cloud computing*. Docker adalah salah satu teknologi berbasis *container* yang paling banyak digunakan [1]. Docker mendistribusikan aplikasi (misalnya *Apache*, MySQL, Joomla) dalam bentuk *images*. Menurut Rui Shu dkk, pada tahun 2017 [2] terdapat hampir 100 repositori resmi dan mengidentifikasi sekitar 100.000 repositori komunitas *public*. Dari hasil penelitian tersebut menghasilkan hipotesis untuk penelitian ini dimana akan memberikan proses analisis kerentanan pada Docker *Images* yang tersedia secara umum di Docker Hub dengan sampel data Docker *Images* yang digunakan adalah Docker dan Joomla. Sehingga aset IT lebih terfokuskan pada *images* pilihan. Menurut T. Astriani dkk tahun 2021 [3] terdapat kerentanan pada Docker dimana kerentanan tersebut akan dikelola berdasarkan *Cybersecurity Framework* dan *vulnerability report* yang dihasilkan berasal dari *Vulnerability Scanner* yang digunakan. Penelitian tersebut membuktikan bahwa implementasi dua *scanner* yang berbeda akan menghasilkan *vulnerability report* yang berbeda dan data tersebut akan diolah berdasarkan standar NIST 800-115 sebagai standar proses untuk menemukan kerentanan pada Docker. Penelitian tersebut juga menganalisis resiko *vulnerability* yang dihasilkan. Pada pembahasannya penelitian tersebut memiliki kompleksitas terlalu jauh sehingga proses *vulnerability management* dengan menggunakan standar NIST 800-115 tidak diimplementasikan dengan detail. Namun, untuk penelitian kali ini akan berbeda dengan penelitian terdahulu yang sudah dijelaskan. Pada penelitian ini akan membangun kerangka kerja yang secara otomatis dapat menjelaskan proses mulai dari menemukan, mengunduh, dan menganalisis *vulnerability* pada Docker *Images*. Sehingga diberikan kontribusi membangun sistem *Vulnerability Management* pada *Container* Docker berdasarkan standar *Cyber Resilience Review* (CRR). Menurut Ghugar tahun 2017 [4] menjelaskan dan membuktikan bahwa upaya pendekatan *Vulnerability Management* yang efisien & mampu menangani resiko kerentanan dihasilkan dari suatu parameter *Vulnerability Management*. ehingga penelitian tersebut menggunakan parameter sebagai proses. Dengan didasarkan penelitian tersebut yang menyatakan bahwa *Vulnerability Management* yang efisien

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

menggunakan parameter, maka penelitian ini memiliki fokus pada implementasi kerangka kerja *Vulnerability Management* berdasarkan standar *Cyber Resilience Review* (CRR) untuk menganalisis hasil remediasi *vulnerability* yang menghasilkan suatu *Vulnerability Patch* yang berguna untuk mengatasi dan mengurangi tingkat kerentanan [5]. Hal ini akan menjadi perbandingan hasil proses *Vulnerability Management* dari parameter yang berbeda. Selain itu, penelitian ini juga menganalisis penilaian proses mitigasi dari aset IT yang memiliki perbedaan versi. Dengan memiliki data kenaikan versi suatu *Software Container*, maka proses mitigasi kerentanan dapat dianalisis. Selanjutnya membandingkan *tools container scanner* yang akan digunakan yaitu Aquasec [6] dan Anchore sebagai *tools* yang bersifat *opensource* untuk *Container Scanner* khususnya Docker. Sehingga dapat mengetahui seberapa jauh Aquasec dan Anchore dalam mendeteksi kerentanan pada Docker *Images* [7]. Hal ini juga akan menjadi perbandingan evaluasi untuk mengakomodasikan *Vulnerability Management*.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Penelitian ini membuat suatu sistematika penyelesaian masalah yang didasarkan pada metrik dan metode yang digunakan yaitu *Vulnerability Management Metrics* dan metode *Cyber Resilience Review* (CRR). Berikut sistematika penelitian yang dilakukan.

2.1.1 Tahap Awal

Pada tahap awal penelitian ini dimulai dengan mengidentifikasi masalah berdasarkan latar belakang yaitu mengenai bagaimana mengelola kerenantanan pada *container* menggunakan *vulnerability management.* Tahap ini juga menjelaskan metode mana yang akan digunakan untuk menganalisis *vulnerability management*.

2.1.2 Tahap Hipotesis

Tahap awal dari *Cyber Resilience Review* (CRR) **[8]** yaitu menentukan strategi untuk menganalisis kerentanan mulai dari menentukan ruang lingkup strategi dengan cara mendokumentasikan aset dan layanan yang dinilai dan dipantau. Strategi yang digunakan adalah *Self-Assessment* [**9]** dimana melakukan penilaian secara mandiri dengan panduan yang ada pada standar *Cyber Resilience Review* (CRR). Setelah itu, dilakukan *Vulnerability Mitigation* dengan menganalisis strategi untuk menaikkan versi *software container*.

2.1.3 Tahap *Design Platform* Pengujian

Pada tahap ini mulai menyusun strategi perencanaan *vulnerability management* berdasarkan standar *Cyber Resilience Review* (CRR) yaitu *Preparation for Vulnerability Analysis and Resolution*. Tahapan ini sebagai persiapan strategi resolusi yang digunakan dalam penelitian ini dengan tahapan pertama yaitu mengidentifikasi standar remediasi sebagai panduan untuk melakukan proses remediasi kerentanan. Selain itu juga mengidentifikasi *periodic activity* dan *tools*, selanjutnya mengidentifikasikan *source of vulnerability information*.

2.1.4 Tahap Implementasi

Pada tahapan implementasi, dilakukan proses *Conduct Vulnerability assessment activities* berdasarkan standar *Cyber Resilience Review* (CRR) yaitu *Process for identifying & analyzing vulnerabilities* untuk mengidentifikasi dan menganalisis kerentanan pada Docker dengan mendeteksi aset *vulnerability* menggunakan dua versi dan kategori yang berbeda di dalamnya yaitu docker versi baru dan versi lama selanjutnya ada joomla versi baru dan versi lama.

2.1.5 Tahap Analisis

Setelah kerentanan ditemukan, tahap selanjutnya yang dilakukan adalah tahap analisis. Sebelum tahap ini dimulai perlu untuk mencatat semua hasil kerentanan yang ditemukan. Tahapan analisis ini merupakan tahapan mengidentifikasi dan mengukur data-data ditemukan pada tahapan implementasi dimana terdapat delapan data. Proses yang dilakukan pada tahapan ini masih sama dengan tahapan sebelumnya berdasarkan satndar *Cyber Resilience Review* (CRR) yaitu *Process for identifying & analyzing vulnerabilities.* Tahapan ini menghasilkan *vulnerability report* yang akan dinilai dan dikategorikan.

2.1.6 Tahap Akhir

Pada tahap akhir, penelitian ini akan menghasilkan laporan hasil penelitian dengan langkah akhir yaitu mengambil kesimpulan dari setiap proses penelitian yang dilakukan. Kesimpulan yang didapatkan adalah penilaian hasil proses mitigasi yang dilakukan berdasarkan dua data dan scanner yang berbeda. Selanjutnya penelitian ini memiliki hasil analisis *Vulnerability Evaluation* yang menghasilkan *Category Vulnerability* dan *Prioritize Vulnerability* **[10]**. Selanjutnya penelitian ini juga memberikan saran yang dapat bermanfaat bagi yang ingin melakukan penelitian yang serupa. Selanjutnya akan dilampirkan dokumentasi berupa tahap-tahap selama proses penelitian berlangsung.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

3. HASIL DAN PEMBAHASAN

3.1 *Vulnerability Evaluation* Berdasarkan Total *Vulnerability* Bulan Pertama dan Bulan Kedua

Pada tahapan ini, hasil *Vulnerability* dari setiap bulan akan dihitung perbandingan frekuensi kerentanan yang dihasilkan menggunakan analisis statistik deskriptif **[11]** dengan rumus penentuan *persentase* dari perubahan akurasi ketepatan dua data sebagai berikut.

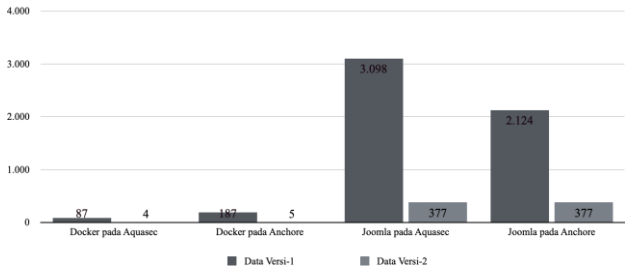
Rumus % Frekuensi *Vulnerability* (P) : $\frac{Total\ Vulnerability\ Bulan\ Kedua - Total\ Vulnerability\ Bulan\ Pertama}{Total\ Vulnerability\ Bulan\ Pertama}$ X 100 (1)

Sehingga dalam perhitungan ini menggunakan hasil total *vulnerabilities* diatas, dapat disimpulkan bahwa:

1. Docker Versi - 1 berdasarkan Aquasec pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 0% *vulnerability* tidak mengalami penurunan atau peningkatan.
2. Docker Versi - 1 berdasarkan Anchore pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 35,92% sehingga frekuensi pada *vulnerability* mengalami penurunan.
3. Docker Versi - 2 berdasarkan Aquasec pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 50% sehingga *vulnerability* mengalami penurunan.
4. Docker Versi - 2 berdasarkan Anchore pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 50% sehingga *vulnerability* mengalami penurunan.
5. Joomla Versi - 1 berdasarkan Aquasec pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 0,74% sehingga dapat disimpulkan bahwa frekuensi pada *vulnerability* mengalami penurunan *Vulnerability* sebesar 0,74%.
6. Joomla Versi - 1 berdasarkan Anchore pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 0,42% sehingga *vulnerability* mengalami penurunan *vulnerability* sebesar 0,42%.
7. Joomla Versi - 2 berdasarkan Aquasec pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 5,99% sehingga *vulnerability* mengalami penurunan *vulnerability* sebesar 5,99%.
8. Joomla Versi - 2 berdasarkan Anchore pada bulan pertama dan kedua memiliki presentase frekuensi total *Vulnerability* sebesar 8,94% sehingga dapat disimpulkan bahwa frekuensi pada *vulnerability* mengalami penurunan *vulnerability* sebesar 8,94%.

3.2 *Vulnerability Evaluation* Berdasarkan Total *Vulnerability* Versi Lama dan Versi Baru

Berikut merupakan diagram perubahan total *vulnerabilities* pada versi – 1 dan versi – 2.

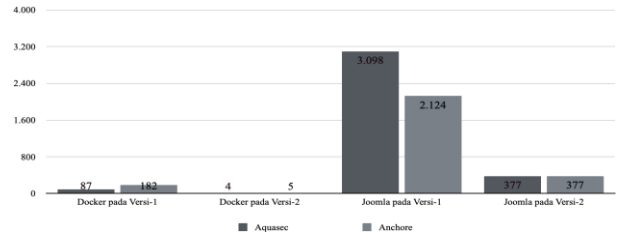


Gambar 1. Diagram Perbandingan Total Vulnerabilities versi lama dan versi baru

Dari diagram tersebut dapat diambil kesimpulan bahwa strategi menaikkan versi atau proses mitigasi *Vulnerability Container* docker dan joomla dapat dikatakan berhasil karena total *Vulnerability* versi – 2 mengalami jumlah penurunan dari total *Vulnerability* versi – 1.

3.3 *Vulnerability Evaluation* Berdasarkan Total *Vulnerability* Aquasec dan Anchore

Berikut merupakan diagram yang menunjukkan perbandingan frekuensi total *Vulnerability* yang dihasilkan oleh Aqusec dan Anchore:



Gambar 2. Diagram Perbadingan Total Vulnerabilities Aquasec dan Anchore

Dalam penelitian ini Anchore menghasilkan frekuensi total *Vulnerability* lebih besar pada setiap versi yaitu pada Docker Versi – 1, Docker Versi – 2, dan jumlah yang sama pada Joomla Versi – 2. Sedangkan Aquasec

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

menghasilkan jumlah paling banyak pada Joomla Versi – 1 dan memiliki jumlah yang sama dengan Anchore pada Joomla Versi-2.

3.4 *Vulnerability Evaluation* Berdasarkan Time for Detection Aquasec dan Anchore

Analisis yang digunakan berdasarkan analisis statistik deskriptif menggunakan rumus *persentase* perbandingan *Data Time for Detection* (TD) oleh masing-masing *tools scanner*. Berikut rumus *persentase* yang digunakan.

Rumus % *Frekuensi Vulnerability* (P) : $\frac{Total\ TD\ Aquasec}{Total\ TD\ Anchore}$ X 100 (2)

Berikut perbandingan data berdasarkan pengujian *Time For Detection*:
1. Aquasec 33% lebih cepat menghasilkan frekuensi data kerentanan
2. Aquasec 2% lebih cepat menghasilkan frekuensi data kerentanan
3. Aquasec 76% lebih cepat menghasilkan frekuensi data kerentanan
4. Anchore 15% lebih cepat menghasilkan frekuensi data kerentanan

3.5 *Categorize and Prioritize Vulnerabilities*

Dalam tahapan ini bertujuan untuk mengkategorikan dan memprioritaskan *Vulnerability* yang ditemukan **[12]**. Hasil analisis hubungan versi - 1 dan versi - 2 menyatakan bahwa terdapat 3 kategori data yang digunakan. Kategori tersebut adalah *Closed Vulnerability*, *Open Vulnerability* dan *Newly Vulnerability*. Selanjutnya memprioritaskan *Vulnerability* yang kritis berdasarkan *severity level* **[13]** yang dihasilkan. Berikut hasil analisis kategori dan prioritas data.

3.5.1 Data Docker

1. Pengujian Aquasec

Tahapan ini adalah proses mengidentifikasi tiga kategori hubungan data *vulnerbility* versi - 1 dan versi - 2. Dari tiga kategori tersebut, pengujian *Container* Docker berdasarkan Aquasec memiliki dua kategori yang terpenuhi yaitu *Closed Vulnerability* dan *Newly Vulnerability*. Untuk *Open Vulnerability* tidak digunakan karena data yang dihasilkan tidak memenuhi kriteria yang artinya tidak ada data versi - 1 yang muncul kembali pada data versi – 2.

2. Pengujian Anchore

Setelah melakukan pengujian berdasarkan Aquasec, tahapan selanjutnya yaitu melakukan identifikasi kategori *vulnerability* data berdasarkan hasil pengujian Anchore. Kategori yang dihasilkan terdapat *Closed Vulnerability* dan *Newly Vulnerability*.

3.5.2 Data Joomla

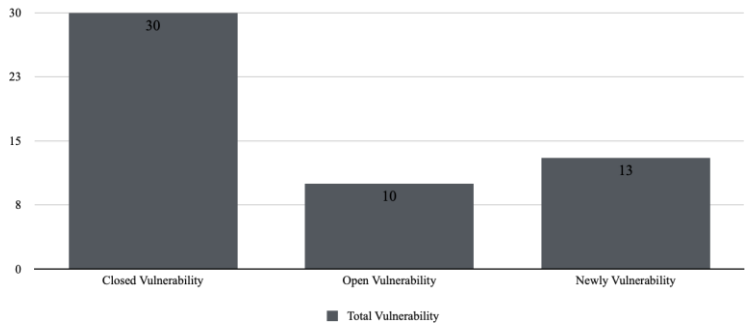
1. Pengujian Aquasec

Setelah melakukan analisis dan identifikasi *Container* Docker, tahapan analisis selanjutnya yaitu *Container* Joomla. *Container* Joomla versi – 1 dan versi – 2 dianalisis berdasarkan pengujian Aquasec sehingga menghasilkan kategori data yaitu *Closed Vulnerability*, *Open Vulnerability* dan *Newly Vulnerability*.

2. Pengujian Anchore

Setelah pengujian *Container* Joomla menggunakan pengujian Aquasec, tahapan analisis selanjutnya yaitu analisis berdasarkan pengujian Anchore. Pada pengujian ini dihasilkan kategori data hubungan *Container* Joomla versi – 1 dan versi – 2 dengan kategori data yang dihasilkan yaitu *Closed Vulnerability*, *Open Vulnerability*, dan *Newly Vulnerability*.

Dari data total *Category vulnerability* yaitu sebanyak 53 *Data* ditemukan sebanyak 30 *Closed Vulnerability*, 10 *Open Vulnerability* dan 13 *Newly Vulnerability*.



Gambar 3. Diagram Perbadingan Category Vulnerabilities

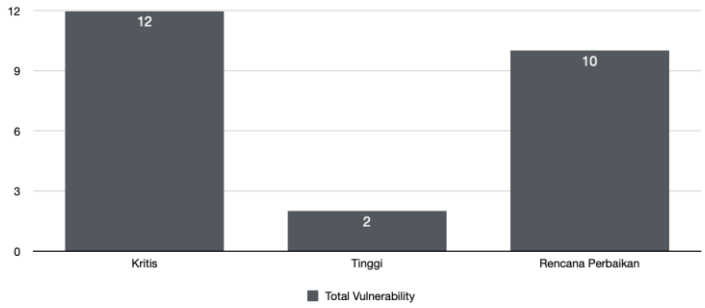
Berikut adalah persentase yang menunjukkan data *Closed Vulnerability* sebanyak 36%, *Open Vulnerability* sebanyak 32% dan *Newly Vulnerability* sebanyak 32%. Dari data tersebut persentase terbanyak adalah *Closed Vulnerability* sehingga dapat disimpulkan bahwa pada penelitian ini proses mitigasi berhasil dilakukan karena

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

kerentanan yang tertutup atau kerentanan yang ditemukan di versi -1 tidak ditemukan kembali di versi - 2 memiliki persentase paling banyak.

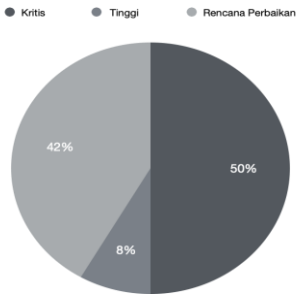
3.5.3 Identifikasi *Vulnerability Patch* dan *Time for Remediation*

Pada penelitian ini data *Patching Rate* yang dihasilkan berjumlah 24 *patch* yang masing-masing memiliki informasi seperti *Vulnerability Library*, *Publicly Disclosed*, *Patch Released*, *Time for Remediation*, dan *Patch by Severity* [14]. Berikut merupakan diagram yang memberikan informasi jumlah *patching rate* yang dihasilkan dari tabel diatas.



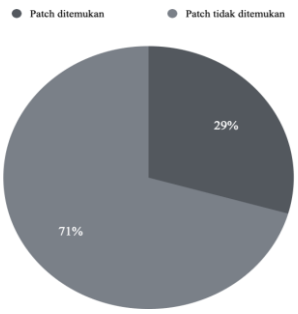
Gambar 4. Diagram Total Patching Rate

Dari diagram diatas *Patch* by *Severity* dengan status kritis sebanyak 12 data dan tinggi sebanyak 2 data serta rencana perbaikan sebanyak 10 data . *Persentase Patching Rate* yang dihasilkan adalah sebagai berikut:



Gambar 5. Presentase Patching Rate

Dengan data yang didapatkan adalah *Patching Rate* dengan *level* kritis sebanyak 58%, dan status rencana perbaikan sebanyak 43%. Dari proses identifikasi *patch vulnerability* yang dilakukan terdapat 24 *patch* yang ditemukan dari total 53 *vulnerability*. Sehingga dapat disimpulkan bahwa persentase *patch vulnerability* sebagai berikut:



Gambar 6. Persentase Patch Vulnerability

Sebanyak 24 *patch* ditemukan artinya 71% kerentanan yang ditemukan sudah dapat diperbaiki dan 29 *patch* tidak ditemukan dan memiliki persentase sebanyak 29% kerentanan belum dapat diperbaiki. Berikut adalah *vulnerability list* yang belum dapat diperbaiki dilihat dari *Category* data *Newly Vulnerability* dan *Open Vulnerability*.

3.5.4 Identifikasi *Root Cuases*

Management perbaikan memiliki *work products* lainnya yaitu hasil analisis *Root Causes*. Menurut *Cyber resilience review* (CRR) **[15]**, metrik dari proses *vulnerability management* mencangkup data kerentanan yang memerlukan analisis *Root Causes*. Sehingga tahapan analisis ini didapatkan dari hasil pengukuran proses perbaikan yang dilakukan sebelumnya yaitu *Vulnerability Patch*. *Patch* yang dihasilkan paling banyak adalah *level* kritis yaitu

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

memiliki *persentase* sebanyak 53%. Oleh karena itu, tahapan ini adalah memprioritaskan untuk menganalisis *Root Causes* dari *Critical Vulnerability*. Berikut merupakan analisis kerentanan dengan status kritis berdasarkan perhitungan *Vulnerability Patch* dan *Time for Remediation*.

1. zlib1g

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla versi – 2 dengan pengujian Aquasec. Dengan *Vulnerability* ID CVE-2018-25032. zlib adalah *library* yang mengimplementasikan metode kompresi *deflate* yang ditemukan di gzip dan PKZIP. Zlib1g pada versi 1.2.12 memiliki kemungkinan kerusakan memori pada saat kompresi *Data*. Dimana data yang terkompresi dapat menimpa jarak tabel sehingga menghasilkan outout yang rusak kerana jarak yang tidak valid, dan dapat mengakibatkan akses diluar batas dan dapat merusak aplikasi.

2. binutils-*common*

Vulnerability ini ditemukan pada *Newly Vulnerability List Container* Joomla berdasarkan pengujian menggunakan Aquasec. Dengan *Vulnerability* ID CVE-2017-13716. *Vulnerability* ini mengenai rutinitas simbol pada C++ di cplus-dem.c. dimana rutinitas ini dapat memungkinkan penyerang menyebabkan penolakan layanan (alokasi memori yang berlebihan dan *crash* aplikasi) melalui *file* yang dibuat.

3. libblkid1

Vulnerability ini ditemukan pada *Open Vulnerability List Container* Joomla berdasarkan pengujian menggunakan Aquasec. Dengan *Vulnerability* ID CVE-2022-0563. *Vulnerability* ini merupakan sebuah kecacatan yang ditemukan di utilitas util-linux chfn dan chsh ketika dikompilasi dengan dukungan Read*line*. *Library* pada *Readline* menggunakan variabel lingkungan "INPUTRC" untuk mendapatkan jalur ke *file* konfigurasi. Ketika *library* tidak dapat menguraikan *file* yang ditentukan, hal ini akan menghasilkan suatu pesan kesalahan yang berisi *Data* dari *file* tersebut. Kesalahan ini memungkinkan pengguna yang tidak memiliki hak untuk membaca *file* pada *root* sehingga berpotensi menyebabkan peningkatan *privilege*.

4. Dmsetup

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla berdasarkan pengujian Aquasec dan Anchore. Dengan *Vulnerability* ID CVE-2020-8991. *Vulnerability* ini adalah vg_lookup di daemons/lvmetad/lvmetad-*core*.c di LVM2 2.02 yang salah mengelola memori sehingga menyebabkan kebocoran memori lvmetad. Namun, RedHat membantah CVE-2020-8991 ini bukan kerentanan karena tidak ada rute yang jelas ke eskalasi *privilege* atau penolakan layanan melalui bug.

5. libbz2-1.0

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla. Dengan *Vulnerability* ID CVE-2019-12900 dan berdasarkan pengujian Aquasec dimana BZ2_decompress di decompress.c dan bzip2 hingga versi 1.0.6 memiliki penulisan di luar batas ketika ada banyak penyeleksi. Kerentanan ini ditemukan di dalam komponen *Core* RDBMS (bzip2) dari Oracle *Database* Server. Versi yang didukung yang terpengaruh adalah 11.2.0.4, 12.1.0.2, 12.2.0.1, 18c dan 19c. Kerentanan yang mudah dieksploitasi memungkinkan penyerang dengan *privilege level* DBA dengan akses jaringan melalui Oracle Net untuk mengkompromikan *Core* RDBMS (bzip2). Serangan yang berhasil dari kerentanan ini dapat mengakibatkan pengambilalihan *Core* RDBMS (bzip2).

6. Libbinutils

Vulnerability ini ditemukan pada *Open Vulnerability List Container* Joomla berdasarkan pengujian Aquasec dan *Newly Vulnerability List Container* Joomla berdasarkan pengujian menggunakan Anchore. Terdapat cacat di *library* BFD binutils. Penyerang dapat memasukkan *file* yang dibuat ke aplikasi yang ditautkan dengan BFD, dan menggunakan fungsionalitas DWARF, sehingga dapat menyebabkan dampak pada ketersediaan sistem melalui konsumsi memori yang berlebihan. Menurut Red Hat, read_section() di dwarf2.c dari BFD dapat menyebabkan konsumsi memori yang berlebihan saat menangani bagian debug DWARF yang rusak. Hal ini dapat berdampak pada ketersediaan sistem, penolakan layanan, atau *crash* pada aplikasi yang terhubung dengan fungsi DWARF *library* BFD jika mereka mem-parsing *file* dari sumber yang tidak tepercaya.

7. apt

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla berdasarkan pengujian menggunakan Anchore dengan *Vulnerability* ID CVE-2019-3462. Pada *Vulnerability* ini ditemukan bahwa apt-key di apt semua versi tidak memvalidasi gpg keys dengan benar dengan *keyring master* yang mengarah ke potensi serangan *man-in-the-middle*. Untuk mengatasi kerentanan ini, Debian *Security* Advisory merilis perbaruan keamanan pada tanggal 22 Januari 2019 dimana kerentanan pada apt dapat diperbaiki dengan menonaktifkan pengalihan untuk mencegah eksploitasi.

8. Gpgv

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla berdasarkan pengujian Aquasec dengan *Vulnerability* ID CVE-2018-12020. *Vulnerability* ini ditemukan pada mainproc.c di GnuPG sebelum 2.2.8 dimana salah menangani nama *file* asli selama tindakan dekripsi dan verifikasi, yang memungkinkan penyerang jarak jauh untuk memalsukan output yang dikirim GnuPG pada deskriptor *file* 2 ke program lain yang menggunakan opsi "--status-fd 2". Analisis perbaikan kerentanan melakukan *commit file* pada https://dev.gnupg.org/T4012 dan kerentanan ini sudah tidak ditemukan kembali.

9. Libapr1

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla berdasarkan pengujian Aquasec dengan *Vulnerability* ID CVE-2021-35940. Dimana Array berada di luar batas yang dibaca di fungsi

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

apr_*time*_exp*() diperbaiki di rilis Apache Portable *Runtime* 1.6.3 (CVE-2017-12613). Perbaikan untuk masalah ini tidak diteruskan ke cabang APR 1.7.x. Oleh karena itu, versi 1.7.0 mengalami kemunduran dibandingkan dengan 1.6.3 dan rentan terhadap masalah yang sama. Namun kerentanan ini sudah diperbaiki di versi 1.7.0 dengan melakukan *update* versi atau merubah indeks waktu.

10. libcryptsetup4

Vulnerability ini ditemukan pada *Closed Vulnerability List Container* Joomla berdasarkan pengujian Anchore dengan *Vulnerability* ID CVE-2016-4484. dimana merupakan kerentanan pada Skrip initrd Debian untuk paket cryptsetup 2:1.7.3-2 dan ini memungkinkan penyerang terdekat secara fisik untuk mendapatkan akses *shell* melalui banyak upaya *login* dengan kata sandi yang tidak valid. Analisis perbaikan kerentanan ini dijelaskan bahwa dapat menggunakan beberapa *scripts* ke paket terdistribusi pada distro yang digunakan.

11. github.com/open*containers*/runc

Vulnerability ini ditemukan pada *Newly Vulnerability List Container* Docker berdasarkan pengujian Aquasec dan Anchore dengan *Vulnerability* ID CVE-2022-29162 yang merupakan kerentanan alat CLI untuk memunculkan dan menjalankan *Container* di Linux sesuai dengan spesifikasi OCI. Sebuah *bug* ditemukan di runc sebelum versi 1.1.2 di mana `runc exec --cap` membuat proses dengan kemampuan proses yang menciptakan proses yang tidak biasa. Bug ini telah diperbaiki di runc 1.1.2. Perbaikan ini mengubah `runc exec --cap` sedemikian rupa seperti menghapus 40 *lines* dari 5 *files* pada github berikut: https://github.com/opencontainers/runc/commit/d04de3a9b72d7a2455c1885fc75eb36d02cd17b5

4. KESIMPULAN

Dilihat dari Total *Vulnerability* yang dihasilkan dari masing-masing pengujian menggunakan Aquasec dan Anchore, versi – 2 memiliki hasil lebih sedikit daripada versi – 1. Yang artinya, Total Vulnerabulity *Container* Docker dan *Container* Joomla pada versi – 2 atau versi terbaru mengalami penurunan. Sehingga proses mitigasi yang dilakukan dapat dikatakan berhasil dengan strategi menaikkan versi *Sofware Container* Docker dan *Container* Joomla yang dianalisis apakah *Vulnerability* yang dihasilkan dapat mengalami penurunan. Pengujian berdasarkan Aquasec maupun Anchore yang jelas sudah berhasil menurunkan jumlah *vulnerabilities* dengan strategi membuat sistem yang baru. Sehingga proses mitigasi suatu sistem berhasil. Berdasarkan perbandingan *tools scanner* yang dilakukan, yaitu Aquasec dan Anchore. Anchore memiliki frekuensi menghasilkan total *Vulnerability* lebih besar dibandingkan Aquasec. Sedangkan hasil perbandingan Aquasec dan Anchore berdasarkan *Time for Detection*, Aquasec merupakan *Container Scanner* paling cepat menghasilkan data *Vulnerability* dibandingkan dengan Anchore. Hal ini dapat dianalisis karena Aquasec memiliki *Vulnerability sources* lebih banyak daripada Anchore. Selanjutnya hubungan data aset dalam penelitian ini yaitu *Container* Docker dan *Container* Joomla pada versi – 1 dan versi – 2 menghasilkan kategori data standar *Cyber resilience review* (CRR) yaitu *Closed Vulnerability*, *Open Vulnerability*, dan *Newly Vulnerability*. Dari semua proses yang dilakukan standar *Cyber Resilience Review* (CRR) sebagai periodisitas kegiatan *Vulnerability Management* dengan membuat kegiatan *periodic* untuk memperhitungkan *management* perubahan dan kerangka waktu.

REFERENCES

[1] M. Y. Darus, Alat Penilaian Kerentanan Web untuk Sistem Manajemen Konten, 2020.
[2] Shu, R., Gu, X., & Enck, W. (2017). A study of security vulnerabilities on docker hub. CODASPY 2017 - Proceedings of the 7th ACM Conference on Data and Application Security and Privacy, 269–280.
[3] T. Astriani, "Analisa Kerentanan Pada Vulnerable Docker Menggunakan Scanner Openvas Dan Docker Scan Dengan Acuan Standar NIST 800-115," Jurnal Teknik Informatika dan Sistem Informasi, vol. 8, pp. 2041-2050 , 2021.
[4] Ghugar, U. (n.d.). Approach to an Efficient Vulnerability Management Program. Retrieved September 7, 2017, from https://www.academia.edu/33788012/Approach_to_an_Efficient_Vulnerability_Management_Program
[5] M. Ninovic, "PATCHING, AND WHY IT STILL MATTERS," 2021. [Online]. Available: https://paraflare.com/patching-and-why-it-still-matters/.
[6] LinuxSec, "Menentukan Severity Kerentanan Menggunakan CVSS," 2021. [Online]. Available: https://www.linuxsec.org/2021/09/cara-menghitung-cvss.html.
[7] M. P. Dazzi, "Docker: Why multi-arch images matters?," 2020. [Online]. Available: https://medium.com/gft-engineering/docker-why-multi-arch-images-matters-927397a5be2e.
[8] U.S. Department of Homeland Security Cybersecurity and Infrastructure Security Agency, Cyber Resilience Review (CRR): Method Description and Self-Assessment User Guide, 2020.
[9] U.S. Department of Homeland Security Cybersecurity and Infrastructure Security Agency, Cyber Resilience Review (CRR): Method Description and Self-Assessment User Guide, 2020.
[10] wikiHow, "Cara Menghitung Persentase Perubahan," [Online]. Available: https://id.wikihow.com/Menghitung-Persentase-Perubahan.
[11] C. M. University, Cyber Resilience Review (CRR) Method Description and Self-Assessment User Guide, 2014.
[12] C. Sree, "What Vulnerability Management Metrics Could Make or Break Your Security Program," [Online]. Available: https://www.secpod.com/blog/what-vulnerability-management-metrics-could-make-or-break-your-security-program/.
[13] D. Bisson, "High-Severity Vulnerabilities Now Take Nearly 250 Days to Remediate, Survey Finds," 2021. [Online]. Available: https://securityintelligence.com/news/news-vulnerabilities-25-days-remediate/.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 4, Nomor 1, September 2022
e-ISSN 2685-998X
DOI 10.30865/json.v4i1.4787

[14] E. Office, "Bandingkan Dua Kolom Untuk Kecocokan Dan Perbedaan Di Excel," [Online]. Available: https://id.extendoffice.com/documents/excel/6392-excel-compare-two-columns.html#percentagechange.
[15] CISA.GOV, "Cybersecurity Framework," [Online]. Available: https://www.cisa.gov/uscert/resources/cybersecurity-framework.