PENGUJIAN CELAH KEAMANAN APLIKASI BERBASIS WEB MENGGUNAKAN TEKNIK *PENETRATION TESTING* DAN DAST (*DYNAMIC APPLICATION SECURITY TESTING*)

Bagus Wicaksono[1], Rr. Yuliana Rachmawati Kusumaningsih[2], Catur Iswahyudi[3]
[1, 2, 3] Jurusan Informatika, FTI, IST AKPRIND
[1]bagusw96@hotmail.com, [2]yuliana@akprind.ac.id, [3]catur@akprind.ac.id

ABSTRACT

Globalization is marked by advances in technology, information and communication. A transparent information management system that has high security features is a necessity for companies and organizations. Cybercrime is increasing along with globalization in the field of technology which is also increasing. Cybercrime is found in many cases of website attacks to get important data on the website. The threat of cybercrime comes from malware, supply chain attacts, to ransomware.

This study aims to determine the vulnerability on the website, done with the Penetration Test and Dynamic Application Security Testing (DAST) methods. The security holes tested were particularly Broken Access Control, Cross Site Scripting (XSS), also Sql Injection, and then efforts were made to improve the website.

Test results show that Broken Access Control vulnerabilities can be prevented by making id which is difficult to guess, XSS can be prevented when users enter javascript syntax input by converting data to character entities, and Sql Injection can be prevented by using enscape () when querying databases.

Keywords: *Broken Access Control*, XSS, *Sql Injection*, DAST, *Pentest*

INTISARI

Globalisasi ditandai dengan kemajuan di bidang teknologi, informasi dan komunikasi. Sistem pengelolaan informasi yang transparan dan memiliki fitur keamanan tinggi menjadi suatu kebutuhan bagi perusahaan dan organisasi. Kejahatan siber semakin berkembang seiring dengan arus globalisasi di bidang teknologi yang juga kian meningkat. Kejahatan siber banyak ditemukan pada kasus penyerangan situs web untuk mendapatkan data penting pada situs web. Ancaman kejahatan siber berasal dari *malware*, *supply chain attact*, hingga *ransomware*.

Penelitian ini bertujuan untuk mengetahui celah keamanan pada situs web, dengan metode *Penetration Test* dan *Dynamic Application Security Testing* (DAST). Celah keamanan yang diuji khususnya Broken Access Control, Cross Site Scripting (XSS), dan Sql Injection, dan kemudian dilakukan upaya perbaikan pada situs web.

Hasil pegujian menunjukkan celah keamanan Broken Access Control dapat dicegah dengan membuat kode (id) yang sulit untuk ditebak, XSS dapat dicegah ketika user memasukkan inputan syntax javascript dengan mengkonversi data ke entitas karakter, dan Sql Injection dapat dicegah dengan menggunakan enscape() pada saat pencarian basisdata.

Kata Kunci : Broken Access Control, XSS, Sql Injection, DAST, Pentest

PENDAHULUAN

Perkembangan Teknologi Informasi (TI) menyebabkan perubahan dan cara pandang manusia dalam kehidupan sehari-hari. Perkembangan TI telah memasuki era dimana teknologi lebih cepat dari yang pernah dibayangkan sebelumnya. Komputer tidak hanya berfungsi sebagai pengolah data, namun telah menjadi senjata utama dalam persaingan perusahaan dalam berkompetisi untuk menjadi yang terbaik (Stiawan, 2005).

Adanya kemudahan yang ditawarkan oleh TI, jarang ditemukan sisi kehidupan yang tidak menggunakan TI sebagai sarana untuk membantu dalam menyelesaikan pekerjaannya, mulai dari hal sederhana sampai dengan yang kompleks. Hal tersebut memunculkan kebutuhan akan keamanan untuk sebuah sistem komputer. Kebutuhan keamanan komputer berbeda-beda sesuai dengan aplikasi yang dijalankannya. Contohnya dalam sebuah sistem akademik tentunya keamanan sistemnya berbeda dengan sistem yang ada di perbankan.

Menurut data yang dirilis oleh *International Data Corporation* (Tribunnews, 2018), mayoritas perusahaan yang ada di ASEAN masih fokus pada keamanan operasional dasar, belum masuk pada level pengelolaan yang baik dan optimal. Sekitar 69,4% perusahaan ASEAN terutama Indonesia masih tahap adhoc, dan 0,2% perusahaan sudah mencapai tahap optimized. Padahal serangan terhadap keamanan Sistem Informasi (SI) semakin berkembang dan meluas secara cepat. Ancaman yang terjadi sepanjang 2018 berasal dari empat hal, mulai dari *Malware*, *Supply chain attack*, hingga *Ransomware*. Hampir 40% dari perusahaan global menilai teknik deteksi lanjutan (*advanced detection technique*) sebagai cara paling efektif untuk mendeteksi ancaman keamanan siber (Haryadi, 2018).

Salah satu contoh kejahatan siber yang paling populer di tahun 2019 adalah kasus pembobolan perusahaan Cryptocurrency *Binance* yang merupakan perusahaan penukaran mata uang kripto terbesar di dunia. *Binance* diretas oleh kelompok hacker yang membuat perusahaan tersebut mengalami kerugian 7.000 *Bitcoin* dengan total senilai USD 41 juta (Rp 588 miliar). Menurut *Binance*, peretas menggunakan berbagai jenis teknik serangan untuk melakukan aksinya. Misalnya, menyebarkan virus dan menggunakan serangan phishing dalam mendapatkan informasi keamanan yang dibutuhkannya. Dengan cara tersebut ternyata hacker bisa mengakses “hot wallet” milik Binance.

Penelitian ini bertujuan untuk mengetahui celah keamanan pada situs web, dengan metode *Penetration Test* dan *Dynamic Application Security Testing* (DAST). Celah keamanan yang diuji khususnya Broken Access Control, Cross Site Scripting (XSS), dan Sql Injection, dan kemudian dilakukan upaya perbaikan pada situs web tersebut.

TINJAUAN PUSTAKA

Penelitian tentang bagaimana mengamankan sistem informasi dari berbagai kemungkinan serangan siber telah dilakukan oleh beberapa peneliti. Pranata dkk. (2015) telah mengukur bagaimana Secure Socket Layer (SSL) mampu mengamankan data dalam jaringan, menggunakan teknik sniffing. Sniffing adalah teknik pemantauan setiap paket yang melintas dalam sebuah jaringan. Sniffing mampu menangkap semua paket yang masuk dan keluar melalui jaringan, termasuk password, username, dan masalah sensitif lainnya. Pengujian celah keamanan dengan metode OWASP pernah dilakukan oleh Muhsin & Fajaryanto (2015). Pengujian tersebut bertujuan untuk mengetahui kerentanan sistem informasi dari serangan dari pihak yang tidak bertanggung jawab. Pengujian dilakukan pada aplikasi Si Ujo menggunakan metode OWASP (Open Web Application Security Project) versi 4. Hasil pengujian penetrasi menggunakan OWASP pada aplikasi tersebut menunjukkan bahwa manajemen otentifikasi, otorisasi dan manajemen sesi belum diimplementasikan dengan baik sehingga perlu perbaikan lebih lanjut oleh pengelola aplikasi.

Aini dkk. (2018) telah melakukan penelitian untuk mengamankan sistem informasi berbasis web dengan cara membatasi hak akses setiap user. Hasil penelitian menunjukkan bahwa pembatasan hak akses mampu menjaga keamanan data dari user yang tidak berhak. Dengan pengelolaan hak akses, aplikasi memiliki integritas dan keamanan yang lebih baik. Sedangkan Widyantoro (2018) melakukan penilaian dan pengujian keamanan sistem informasi menggunakan metode VAPT (Vunerability Assessment & Penetration Testing).

Pada penelitian ini dilakukan pengujian celah keamanan aplikasi berbasis web dengan metode *Penetration Testing* dan DAST (*Dynamic Application Security Testing*),

terutama pada celah keamanan *Broken Access Control*, *Cross Site Scripting*, dan *Sql Injection* pada website yang diuji. *Penetration Testing* adalah upaya yang sah dan resmi untuk menemukan dan berhasil mengeksploitasi sistem komputer untuk tujuan membuat sistem lebih aman (Engebretson, 2011). Sedangkan DAST (*Dynamic Application Security Testing*) merupakan program yang berkomunikasi dengan aplikasi website melalui untuk mengidentifikasi potensi kerentanan keamanan dalam aplikasi website dan kelemahan arsitekturnya (Shura, 2009).

PEMBAHASAN

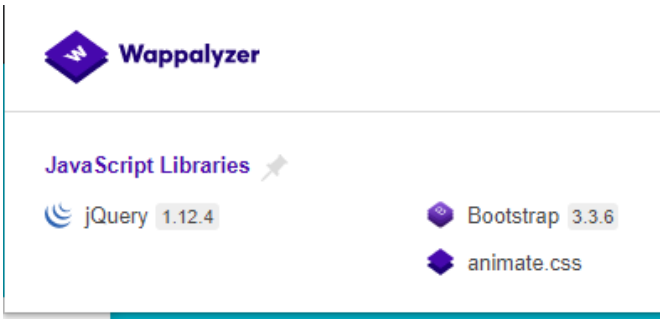
Tahap-tahap yang dilakukan untuk menemukan celah keamanan pada aplikasi berbasis web meliputi *scope*, *reconnaissance*, *vulnerability detection*, *information analysis & planning*, dan *penetration testing*. Pada proses *vulnerability detection* di dalamnya terdapat metode DAST (*Dynamic Application Security Testing*) dalam menemukan celah keamanan pada *website* dengan bantuan aplikasi, misalnya *Acunetic*. Pengujian dilakukan pada situs web http://bagusw.win. Langkah–langkah dalam melakukan penetrasi pada situs web bagusw.win dijelaskan dalam sub bab berikut.

a. *Scope*

Dalam melakukan *penetration test*, pertama kali adalah menetapkan *scope. Scope* yang dipilih menetapkan *website* bagusw. sebagai objek pengujian penetration test. Situs web bagusw.win adalah sebuah website yang menyediakan layanan untuk mahasiswa atau siswa SMK yang ingin mengajukan pendaftaran magang di Perusahaan. Data diri yang diinputkan dan terkirim akan diproses oleh admin, yang merupakan seorang staf HRD dalam perusahaan.

b. Reconnaissance

Setelah menentukan scope untuk menentukan penetration test, langkah selanjutnya adalah dilakukan *reconnaissance*, yang bertujuan untuk mengumpulkan informasi sebanyak mungkin dari website bagusw.win. Untuk mengumpulkan informasi pada website digunakan tools Wappalyzer.



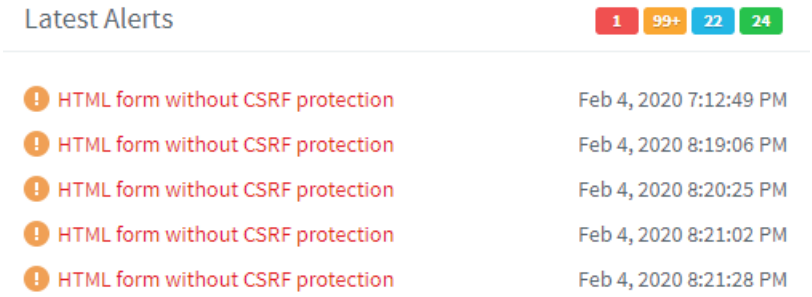
Gambar 1. Hasil deteksi situs web *bagusw.win* dengan Wappalyzer

Gambar 1 menunjukkan hasil deteksi situs web bagusw.win menggunakan aplikasi *Wappalyzer*, yang menginformasikan bahwa teknologi yang dipakai oleh web bagusw.win adalah *jQuery* versi 1.12.4 dan *Bootstrap* versi 3.3.6. Setelah mendapatkan informasi tersebut, selanjutnya dilakukan pencarian informasi apakah pada teknologi ini terdapat celah keamanan *Cross Site Scripting* (XSS). Berdasarkan referensi pencarian yang dilakukan dari website https://snyk.io/test/npm/jquery/ 1.12.4 diperoleh informasi bahwa jQuery versi 1.12.4 memiliki celah keamanan *XSS* dengan tingkat medium. Hal tersebut mengindikasikan adanya kemungkinan untuk dilakukan *penetration test* dari serangan XSS. Selanjutnya mencari informasi tentang *Bootstrap* yang dipakai pada website bagusw.win. Informasi yang didapatkan dari https://snyk.io/test/npm/bootstrap/3.3.6

dinyatakan bahwa *Bootstrap* versi 3.3.6 juga terdapat celah keamanan *XSS* dengan status medium.

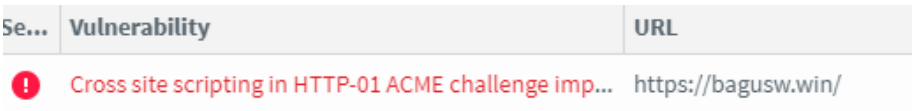
c. Vulnerability Detection

Pada tahap identifikasi celah keamanan web bagusw.win digunakan aplikasi *Acunetix*, yang dapat memeriksa kerentanan seperti *SQL Injection*, *Cross Site Scripting*, dan kerentanan lainnya. Pada tahap identifikasi celah keamanan pada bagusw.win digunakan *Acunetix trial* versi 12. Untuk menjalankan aplikasi ini, dapat menuliskan https://localhost: 13443/#/ pada browser.



Gambar 2. Hasil deteksi dengan Acunetix 12

Gambar 2 menunjukkan hasil dari proses *scanning* yang dilakukan oleh *Acunetix* dengan mendapatkan peringatan (*alert)* warna merah yang menandakan bahwa statusnya adalah tinggi (*high).* Apabila *alert* warna merah dibuka diperoleh informasi seperti pada Gambar 3 yang menginformasikan bahwa terdapat celah *Cross Site Scripting* dengan status *high.*



Gambar 3. Hasil penemuan XSS oleh Acunetix

d. Information Analysis & Planning

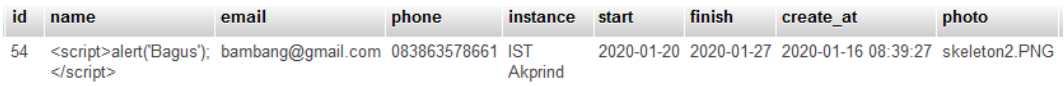
Hasil dari *vulnerability detection* manampilkan celah keamanan pada web bagusw.win. Pada proses *Information Analysis & Planning* dipilih celah keamanan yang digunakan sebagai bahan *penetration testing* adalah XSS, *Sql Injection*, dan *Broken Access Control*.

e. Penetration Testing

Penetration testing yang dilakukan pada web bagusw.win memanfaatkan celah keamanan berikut:

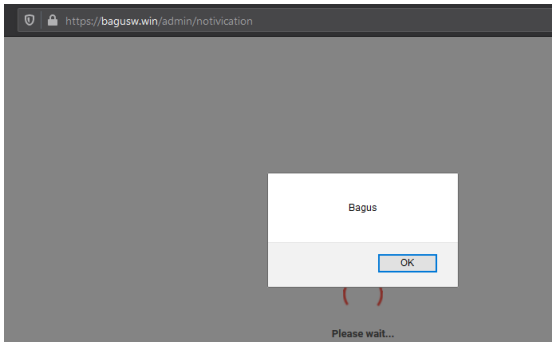
1) XSS

Cross Site Scripting (XSS) memanfaatkan celah keamanan dengan menginputkan *script javascript.* Pengujian penetration testing dilakukan dengan memasukkan informasi pada formulir yang disediakan oleh *website* bagusw.win. Formulir yang dipilih adalah pendaftaran peserta magang, dengan alamat https://bagusw.win/pendaftran/. Halaman tersebut berisikan *form* untuk pengguna yang akan mendaftar magang di Perusahaan. Pengujian *penetration test* dilakukan dengan memasukkan *source code javascript*, yaitu '*<script>alert('Bagus'); </script>'* pada salah satu *form* yang sudah disediakan.



Gambar 4. Hasil isian form pendaftaran pada basisdata

Gambar 4 memperlihatkan hasil dari informasi pendaftaran magang dan isian *script* berhasil tersimpan dalam basisdata. Langkah selanjutnya adalah masuk ke dalam sistem informasi untuk melihat hasil isian pendaftaran magang dan mengarahkan ke url https://bagusw.win/admin/ notivication/. Gambar 5 merupakan contoh dari sistem yang berhasil diserang pada celah keamanan *Cross Side Scripting* (XSS).



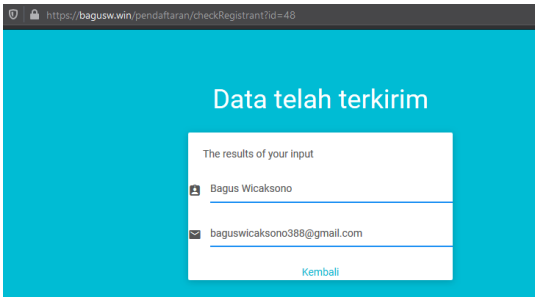
Gambar 5. Tampilan situs web terkena serangan XSS

2) Broken Access Control

Pengujian Broken Access Control pada web bagusw.win menggunakan tiga parameter, yaitu:

a) *Insecure Id*

Pengujian yang pertama kali diperhatikan adalah melihat *url* ketika menjalankan fungsi yang disediakan oleh situs web bagusw.win. Penyerangan dilakukan dengan menuliskan *url* https://bagusw.win/pendaftaran/check Registrant?id=48.

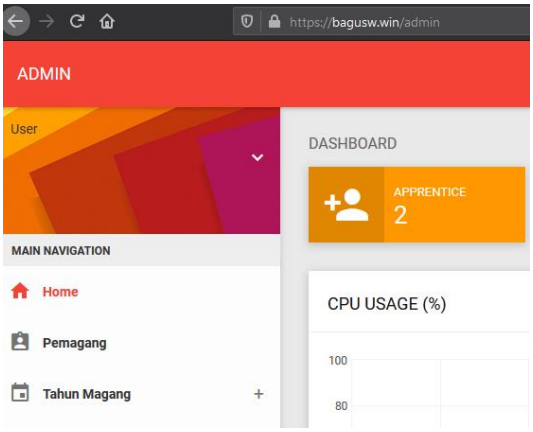


Gambar 6. Tampilan penyerangan pada parameter Insecure Id

Gambar 6 menunjukkan hasil penyerangan yang dilakukan dan berhasil mendapatkan informasi dari *user* dengan id 48. Informasi tersebut dapat dilihat oleh *user* yang sudah mengirimkan informasi pada *form* pendaftaran magang dan *user* lain tidak berhak melihatnya. Maka pada *website* bagusw.win untuk *Insecure Id* dapat ditebak dengan mudah.

b) Forced Browsing Past Access Control Checks

Pengujian dilakukan dengan melakukan penyerangan pada *url* https://bagusw.win/admin, menggunakan aplikasi peramban (*browser)*. Hasil yang diperoleh ditunjukkan pada Gambar 7, yang memperlihatkan penyerang dapat masuk ke halaman *dashboard admin*. Padahal seharusnya halaman *dashboard admin* hanya dapat diakses dengan login terlebih dahulu pada *url* https://bagusw.win/ login. Jadi, pada parameter *Forced Browsing Past Access Control Checks* pada *website* bagusw.win belum aman atau terdapat celah di dalamnya.



Gambar 7. Tampilan hasil pengujian pada parameter *Forced Browsing Past Access Control Checks*

c) Client Side Caching

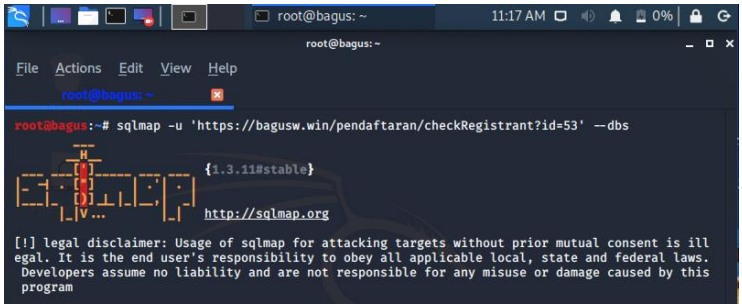
Pengujian pada parameter *client side caching* dilakukan dengan melakukan login pada *website* bagusw.win dan masuk ke dalam *dashboard admin.* Ketika sudah berhasil melakukan *login*, selanjutnya melakukan *logout*. Tetapi, ketika melakukan *login* diketahui *url admin* adalah https://bagusw.win/ admin dan masih dapat digunakan untuk masuk ke halaman *dashboard admin* padahal sudah melakukan *logout*. Gambar 8 menunjukkan adanya celah keamanan *client side caching* yang terdapat infomasi *email* pada cookies *browser* sehingga dapat masuk ke halaman *admin*.

Gambar 8. Tampilan dashboard admin

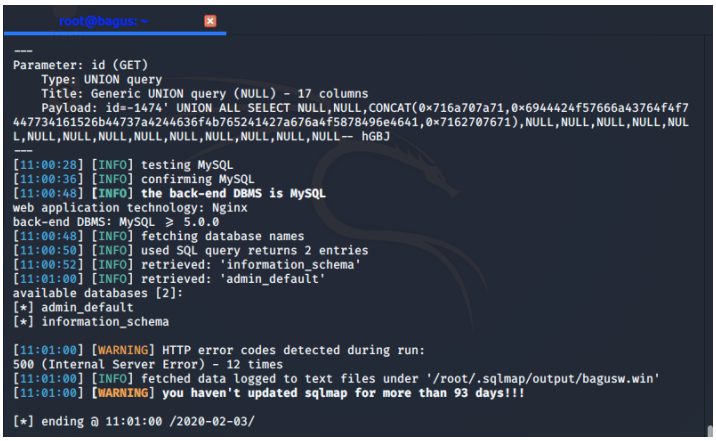
3) Sql Injection

Sql Injection adalah jenis serangan yang dilakukan oleh penyerang yang dapat menimbulkan banyak kerugian karena rusaknya database dari situs web. Teknik *sql injection* dapat mencuri informasi penting seperti *username* dan *password*, merubah database, dan memasukkan konten berbahaya.

Gambar 9 menunjukkan langkah-langkah *penetration testing* ke dalam *database* bagusw.win menggunakan *tools sqlmap* dengan memasukan perintah "*sqlmap u 'https:// bagusw.win/pendaftaran/checkRegistrant?id=53' –dbs "*. Perintah tersebut untuk mengidentifikasi alamat *url* yang akan diserang. Perintah –dbs adalah perintah yang digunakan untuk melihat *database* yang tersimpan di server *website* bagusw.win. Ketika perintah tersebut dijalankan, *sqlmap* melakukan *scanning* untuk menemukan celah pada *website* bagusw.win. Sedangkan Gambar 10 merupakan hasil dari *penetration testing* dengan menggunakan *sqlmap* dan berhasil mendapatkan informasi *database* yang terdapat di *server* bagusw.win; yaitu *admin_default* dan *information_schema*.



Gambar 9. Perintah Sqlmap

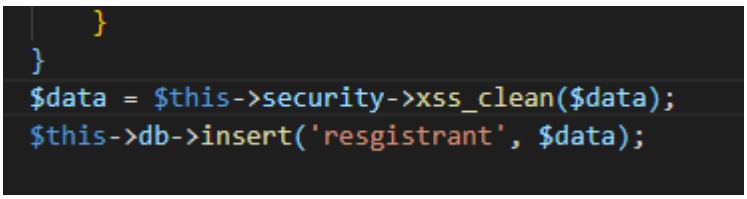


Gambar 10. Hasil penetration testing dengan Sqlmap

Hasil pengujian keamanan yang dilakukan pada *website* bagusw.win berhasil menemukan celah keamanan XSS, *Sql Injection*, dan *Broken Access Control.* Sehingga tahap berikutnya adalah menutup celah tersebut, yang bertujuan agar dengan penyerangan yang sama bisa ditutup. Celah yang ditutup meliputi :

a. XSS

Dalam mengatasi serangan XSS dapat dilakukan dengan mengubah *script* ke dalam karakter, sehingga kode php menangkapnya bukan sebuah *script javascript*. Yaitu ketika user memasukan inputan *script*, sebelum masuk ke database diubah ke dalam karakter terlebih dahulu, seperti ditunjukkan pada Gambar 11.



Gambar 11. Source Code untuk mengatasi XSS

b. *Broken Access Control*

Dalam mengatasi celah Broken Access Control, akan dibagi menjadi tiga parameter. Karena dalam melakukan penyerangan terhadap serangan Broken Access Control meliputi tiga parameter, diantaranya adalah:

a) *Insecure Id*

Pengamanan yang dilakukan adalah dengan membuat *id* yang sebelumnya memungkinkan dapat ditebak dengan mudah oleh penyerangan seperti yang ditunjukan pada Gambar 12. Penanganannya adalah dengan membentuk *id* yang susah untuk ditebak. Pertama kali yang dilakukan adalah membuat sebuah token yang isinya berbagai karakter acak yang susah dan disimpan ke dalam *database*.

```
public function coba()
{
    $token = base64_encode(random_bytes(64));
    $start = $this->input->post('start');
    $d = strtotime($start);
    $d2 = date('Y', $d);
```

Gambar 12. Implementasi pengacakan karakter didalam Php

b) *Forced Browsing Past Access Control Checks*
Untuk mengatasi pada parameter *Forced Browsing Past Access Control Checks* dapat dilakukan dengan menambahkan sebuah fungsi yang berfungsi untuk mengecek apakah user yang menggunakan *website* bagusw.win sudah berhasil melakukan *login* atau belum sebelum ke url https://bagusw.win/admin, yang menunjukkan halaman *dashboard admin*. Untuk implementasinya dapat dilihat pada Gambar 13, yang merupakan isi dari *construction* salah satunya adalah mencari *session id admin* yang sudah disimpan ketika berhasil ketika melakukan *login* pada halaman *login*.

```
public function __construct()
{
    parent::__construct();
    if (!$this->db->get_where('admin', ['id' => $this->session->userdata('id')])->row_array()) {
        header('Location: /login');
    }
    $this->load->model('Admin_model');
}
```

Gambar 13. Fungsi construction dalam mengecek admin

c) *Client Side Caching*
Dalam mengatasi celah *client side caching* dapat dilakukan dengan cara menghapus informasi pada *session* saat *user* melakukan *logout*. Karena dalam *website* bagusw.win ini, ketika *user* berhasil melakukan *login* maka *website* akan menyimpan informasi *session* yang meliputi *id* dan *email*. Implementasinya dapat dilihat pada Gambar 14, yaitu dalam menghapus informasi *sessions* yang disimpan, dapat menggunakan fungsi *unset_userdata* yang disediakan oleh *codeigniter*.

```
public function logout()
{
    $this->session->unset_userdata('email');
    $this->session->unset_userdata('id');
```

Gambar 14. Evaluasi pada function logout

c. Sql Injection
Sebelum mengatasi *sql injection* yang terdapat pada *website* bagusw.win, terlebih dahulu mencari penyebab *sql injection* dapat menyerang *database* baguw.win. Menurut informasi forum *stackoverflow* (https://stackoverflow.com/questions/1615792/does-codeigniter-automatical_ly-prevent-sql-injection); *Codeigniter* tidak melakukan proses *Escape* saat

menggunakan " $this→db→query ". Untuk mengatasi celah keamanan *Sql Injection* dilakukan dengan cara mengubah *query database* menjadi query " $this→ db→get_where" seperti Gambar 15.

```
$registrantAdmin['registrant'] = $this->db->get_where('resgistrant', ['id' => $token])->row_array();
```

Gambar 15. Query untuk mengatasi Sql Injection

KESIMPULAN

Berdasarkan penelitian yang sudah dilakukan, dapat diperoleh kesimpulan sebagai berikut:

1. Pada pengujian celah keamanan dengan metode DAST (*Dynamic Application Security Testing*) pada *website* bagusw.win yang meliputi serangan *Cross Site Scripting*, *Broken Access Control*, dan *Sql Injection*, pada *website* dapat dibuktikan. Sehingga dapat dilakukan proses evaluasi untuk memperbaiki ketiga celah keamanan tersebut.
2. *Penetration testing* pada celah keamanan *Cross Site Scripting* didapatkan bahwa *website* bagusw.win setelah memasukkan inputan *<sript>alert ('bagus')</sript>* tampilan *website* menampilkan *alert* yang berisikan bagus. Evaluasi yang dilakukan adalah sebelum ke database berupa *script* dirubah ke karakter biasa, sehingga *source code* pada *website* tidak membaca inputan *script*.
3. Penetration testing pada celah keamanan *Broken Access Control* didapatkan bahwa penyerang dengan mudah dapat menebak id user lain dikarenakan menggunakan angka. Evaluasi yang dilakukan adalah merubah id angka menjadi id yang berisikan karakter acak.
4. Penetration testing pada celah keamanan *Sql Injection* dengan memasukan karakter (') pada akhir *url* yang ber *id* dan didapatkan *error* pada *query database* yang dapat dilihat di *browser*. Evaluasi yang dilakukan adalah dengan menambahkan method *enscape ()* pada *query database*.

DAFTAR PUSTAKA

Aini, Q., Rahardja, U., Madiistriyatno, H., & Martianda, Y. D. (2018). Pengamanan Pengelolaan Hak Akses Web Berbasis Yii Framework. SYNTAX Jurnal Informatika, 52-63.

Ciampa, M. (2012). Security + Guide to Network Security Fundamentals. Boston: Course Technology.

Engebretson, P. (2011). The Basics of hacking and penetration Testing. Waltham: Elsevier.

Haryadi, M. (2018, September 21). Tribuntechno. Retrieved from Tribunnews: https://www.tribunnews.com/techno/2018/09/21/transformasi-digital-wajib-didukung-peningkatan-standar-kualitas-keamanan-cyber.

Muhsin, M., & Fajaryanto, A. (2015). Penerapan Pengujian Keamanan Web Server Menggunakan Metode OWASP versi 4 (Studi Kasus Web Server Ujian Online). Multitek Indonesia, 31-42.

Pranata, H., Abdillah, L. A., & Ependi, U. (2015). Analisis Keamanan Protokol Secure Socket Layer (SSL) Terhadap Proses Sniffing di Jaringan. Student Colloquium Sistem Informasi & Teknik Informatika (SC-SITI), 1-6.

Shura, B. (2009). Web Application Security Scanner Evaluation Criteria. Stanford: Creative Commons Attribution License.

Stiawan, D. (2005). Sistem Keamanan Komputer. Jakarta: PT Elex Media Komputindo.

Widyantoro, F. (2018). Security Assessment menggunakan tool nessus untuk mencari celah keamanan web Aplikasi MoU perguruan Tinggi XYZ. Skripsi. Jurusan Teknik Informatika. Fakultas Teknik. Universitas Muhammadiyah Yogyakarta.