JuTech
Jurnal Ilmiah Teknologi & Informasi

Website: http://ojs.itb-ad.ac.id/index.php/JUTECH
E-ISSN: 2797 - 4111

SISTEM KEAMANAN BERLAPIS MENGGUNAKAN METODE ACCESS CONTROL LIST DAN ENKRIPSI SOURCE CODE PADA WEB LOGIN

Kamarudin[1], Mukhaimy Gazali[2] (*)

[1]*Universitas Muhammadiyah Banjarmasin, Banjarmasin*
[2]*Universitas Muhammadiyah Banjarmasin, Banjarmasin*

Abstract

Web documents are the most important assets for organizations to represent reports managed by servers. Meanwhile, the form is the entrance for the input provided by the user to be able to take advantage of server resources, be it a database, or other features that are owned by the server. This research focuses on the security of the web login form which is the entry point for various users. The level of authority (authorization) of users is set based on the Access Control List (ACL) which is arranged in such a way by the administrator. Multi-layered security goes through several stages that must be passed by the user, from validation, verification, authentication to authorizing according to the rights given to him by the administrator. The results found provide more value for web developers to set a standard security system for the web login form they build. However, various other security opportunities may be found in subsequent research, as developments

Kata Kunci: *Webform, Weblogin, Security, Acl, Encryption*

Juli – Desember 2021, Vol 2 (2) : hlm 1-13

(*) Korespondensi: kamarudin@umbjm.ac.id *(Kamarudin)*, mukhaimy.gazali@umbjm.ac.id *(Mukhaimy Gazali)*

PENDAHULUAN

Keamanan web merupakan isu terpenting dalam setiap implementasi aplikasi berbasis online ini. Bebagai peristiwa berkenaan keamanan web di berita-berita IT atau umum menjadi perhatian yang cukup mengkhawatirkan bagi para developer web dan praktisi web. Kerentanan pada platform web dapat memungkinkan serangan keamanan dengan konsekuensi mulai dari kerugian ekonomi hingga pelanggaran privasi, misalnya dalam kasus yang tidak pantas diungkapkan berkenaan dengan catatan kesehatan elektronik. Masalah keamanan ini meningkatkan kebutuhan akan pertahanan yang lebih efektif untuk platform web.

Meskipun demikian, sebagaimana diketahui bahwa melindungi layanan aplikasi web tidaklah mudah, mengingat kompleksitas dasar dari Web. Platform web beraneka ragam dan termasuk sejumlah besar komponen dan teknologi yang berbeda, hal ini membuka peluang akan serangan besar yang terjadi pada aplikasi web. Kelemahan aplikasi web yang dapat dimanfaatkan melalui browser yang dapat dengan mudah mengekspos data penting dan sensitif, melalui protocol web dapat merusak dan mempengaruhi kerahasiaan data yang dikirimkan, demikian juga kesalahan penulisan kode pada web dapat dimanfaatkan untuk masuknya konten-konten berbahaya pada halaman web yang dipercaya (Bugliesi et al., 2016).

Tidak bisa dipungkiri bahwa peretas yang berbahaya terus berupaya mencari kerentanan software yang tidak diketahui dan mencoba memperoleh keuntungan (uang) dengan mengekspolitasi kerentanan tersebut untuk mecuri data dan merusak layanan yang ada, atau bahkan dengan menjual informasi tentang kerentanan tersebut di pasar gelap (J. Radianti, 2010; S. Frei et al., 2010). Untuk mengurangi jumlah kerentanan dalam produk mereka, penyedia perangkat lunak memanfaatkan berbagai pendekatan pengujian dan audit. Namun, upaya semacam itu tidak dapat menghilangkan semua kerentanan karena, misalnya, kendala ekonomi dan kompleksitas teknis (Zhao et al., 2014).

Perancang aplikasi bersama dengan administrator dari sebuah system harus menentukan keamanan untuk aplikasi. Pilihan yang dibuat oleh pengembang dan administrator mungkin akan berdampak pada kinerja system. Meskipun otentikasi dan pembuatan pengguna biasanya tugas administrative, karena tugas-tugas ini mungkin berdampak pada pendefinisian keamanan untuk berbagai jenis klien dan pengguna. Selain itu juga harus mengintegrasikan dengan atau bahkan lebih mengembangkan pendaftaran pengguna untuk pengguna aplikasi web (Tommi Tulisalo et al., 2002).

Untuk memenuhi aturan-aturan penentuan keamanan pada aplikasi web, maka dibutuhkan daftar akses kontrol (access control list, ACL) kepada tiap-tiap pengguna berdasarkan wewenang akses masing-masing dengan tipe pengguna yang berbeda. Dibutuhkannya ACL karena memiliki implementasi umum di semua platform (Davies et al., 2012). Alasan-alasan keamanan menjadikan sebuah aturan akses menjadi hal penting yang ditujukkan kepada pengguna yang memiliki wewenang akses yang berbeda. Sejumlah penelitian telah mengidentifikasi hubungan

aturan dalam ACL yang dapat menyebabkan aturan berlebihan atau bertentangan (Davies et al., 2012). Penggunaan ACL setidaknya mampu untuk mempertegas aturan keamanan yang ditetapkan kepada pengguna tentang hak dan wewenang atas penggunaan sumber daya aplikasi web yang digunakan.

Meskipun demikian, masih adanya pengguna tidak sah yang berupaya menerobos ke dalam system dengan mengabaikan aturan keamanan yang dibuat berdasarkan ACL. User ini memanfaatkan kelemahan pada source code of static web (biasanya hasil generate dari dynamic web) yang dapat terlihat hampi di semua browser web. Selain itu, pengguna sah pun kadang kala memiliki keingintahuan 'berlebihan' atas system aplikasi web yang dibuat, yang kemudian berupaya memanfaatkan source code of static web untuk menerobos system secara tidak sah ke domain yang berbeda berdasarkan aturan ACL yang telah dibuat untuk dirinya. Penggunaan peran memungkinkan untuk mendefinisikan aplikasi tanggung jawab dan menentukan lebih lanjut akses ke elemen basis data (Tommi Tulisalo et al., 2002).

Untuk itu, data (source code) yang terenkripsi, key management dan pendekatan lainnya mutlak digunakan (Kaczmarczyk et al., 2016). Hal ini tidak lain demi untuk menjaga kerahasian dari source code of static web, baik itu berkenaan dengan laporan-laporan, terlebih lagi pada form input yang biasa digunakan oleh pengguna untuk menakses data internal yang ada di server.

Pada paper ini akan dibuat sebuah metode sistem keamanan berlapis dengan mengkombinasikan antara metode access control list (ACL) dan enkripsi data dan source code pada login web form. Pendekatan ini memungkinkan pemrosesan data sensitif yang hanya orang yang berwenang dengan kunci enkripsi dapat memiliki akses yang diberikan. Data terinkripsi tidak pernah terlihat oleh server. Selain itu, server tidak memproses kunci enkripsi, karena data yang terinkripsi sulit untuk mendekripsi mereka tanpa kunci. Kunci enkripsi umumnya selalu untuk satu klien. Selain itu, setiap orang yang menggunakan layanan memiliki nama pengguna dan kata sandi yang unik untuk mengakses aplikasi.

.

METODE

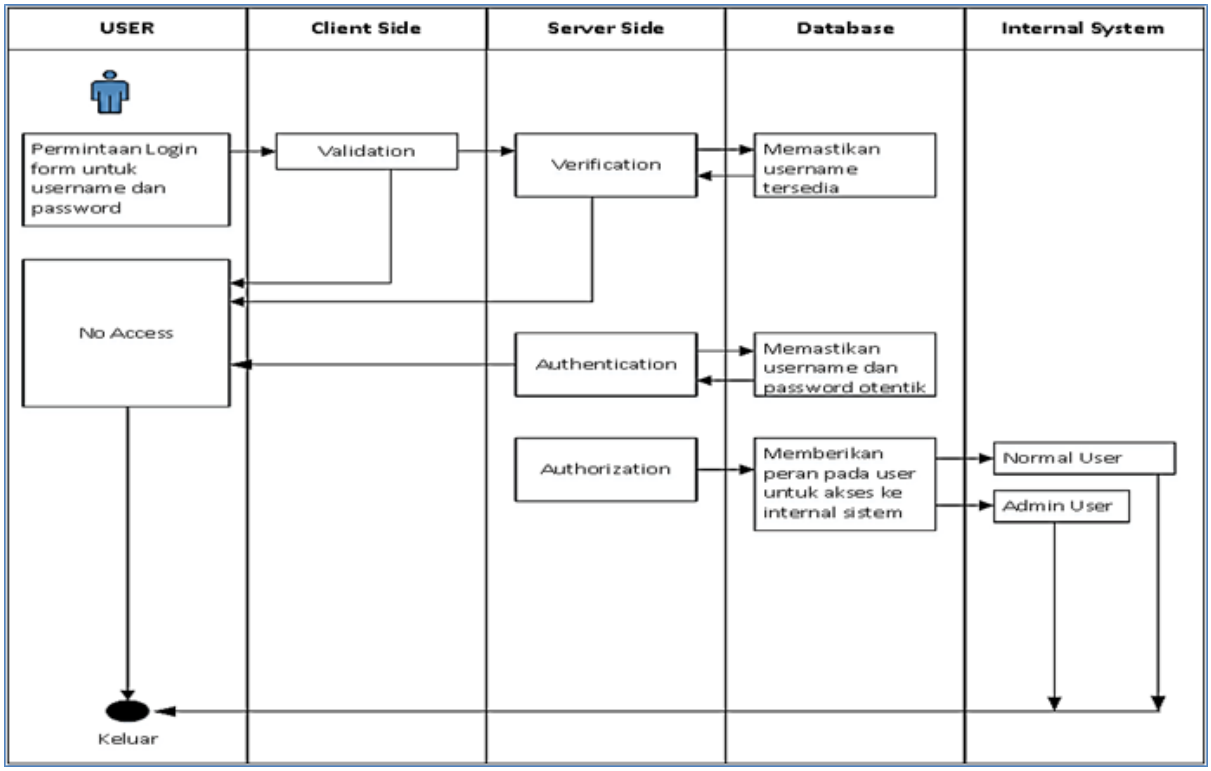
Metode ACL pada penlitian ini menggunakan tahapan-tahapan berlapis untuk memberikan system keamanan pada web login form sesuai dengan peran ACL yang dimiliki oleh user. Berikut ini adalah daftar peran ACL yang diberikan kepada pengguna :

1. No Access, tidak memiliki akses ke dalam internal system.
2. Normal User, memiliki akses ke dalam internal system pengguna sesuai dengan hak yang diberikan oleh Admin User.
3. Admin User, memiliki hak akses ke area admin dan semua area kontrol pengguna. Admin User biasa disebut pula sebagai super user.

Ada pun tahapan lapisan keamanan yang diberikan meliputi :

- Validation, digunakan untuk melakukan validasi terhadap isian form yang digunakan oleh user.
- Verification, memastikan bahwa username yang digunakan untuk login benar-benar terdapat di dalam database.
- Authentication, memastikan dengan dilakukan komparasi bahwa antara username dan password yang digunakan untuk login adalah benar-benar berpasangan.
- Authorization, memberikan otoritas atau kewenangan kepada user bersangkutan sesuai dengan peran yang dimilikinya.

Ada pun desain model untuk metode yang digunakan adalah sebagai berikut:



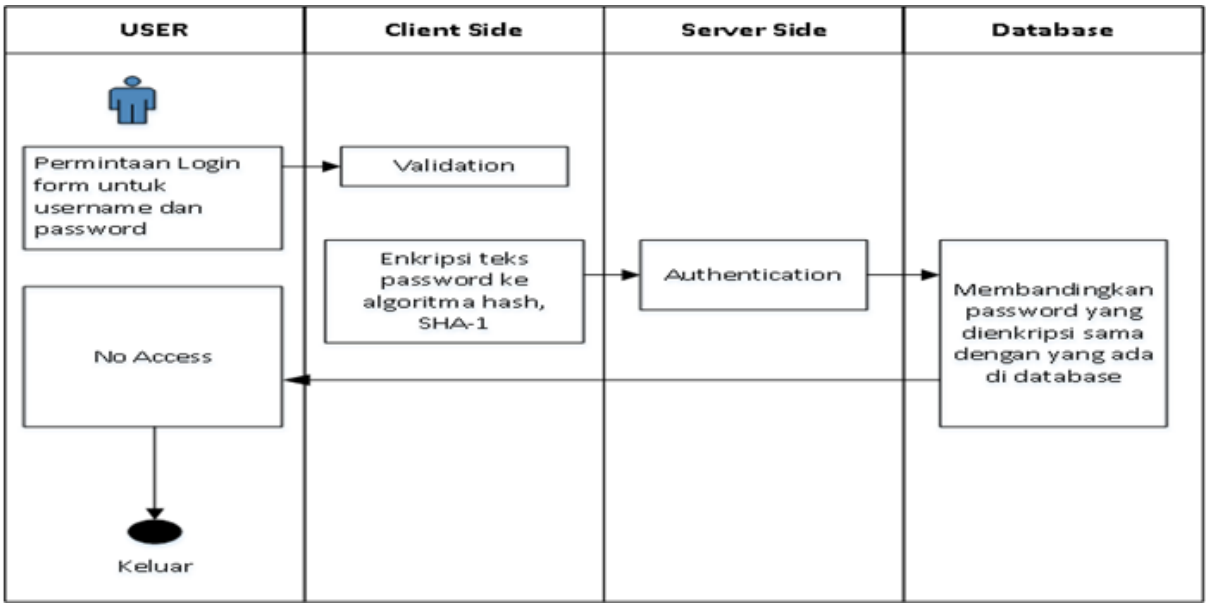
Gambar 1. Model Access Control List Login Form

Proses diatas merupakan tahapan keamanan berlapis yang diberikan kepada pengguna untuk bisa masuk ke dalam internal system dengan menggunakan metode Acces Control List (ACL). Adapun penjelasan dari model di atas adalah berikut ini:

1. Pengguna melakukan permintaan untuk bisa masuk ke internal system dengan memasukkan username dan password. Peran saat ini yang dimiliki pengguna adalah sebagai No Access.
2. Setelah memasukkan data login, system akan melakukan validasi data melalui client-side (dalam hal ini browser web) untuk memastikan bahwa data yang

diinputkan sudah benar-benar sesuai ketentuan login dan aturan yang telah ditetapkan. Jika proses validasi client-side berhasil, maka selanjutnya akan dilakukan proses enkripsi data, dalam hal ini field password menggunakan algoritma hash (sha-1). Lebih rincinya mengenai enkripsi ini dapat dilihat pada penjelasan model di bawah ini.

3. Selanjutnya dilakukan proses verifikasi, yaitu dengan membandingkan username yang dimasukkan oleh pengguna dengan username yang telah terdaftar atau tersimpan di dalam database. Jika tidak sama username yang dimaksud, maka peran pengguna akan dikembalikan ke posisi No Access.
4. Jika proses verifikasi username berhasil, selanjutnya dilakukan proses otentikasi, yaitu proses untuk membandingkan password yang diinputkan pengguna (dalam hal ini password sudah dienkripsi) identik dengan password yang dimiliki username yang sudah tersimpan di database (password yang tersimpan di database pun sudah dienkripsi, jadi admin server tidak pernah tahu isi password yang asli). Jika proses otentikasi gagal, dalam hal ini password yang diinput pengguna tidak otentik dengan password yang dimiliki username yang ada di database, maka proses otentikasi gagal dan peran pengguna dikembalikan ke peran No Access.
5. Selanjutnya akan lakukan proses authorization (otoritas, kewanangan). Proses ini penting sekali untuk bisa menempatkan pengguna sesuai dengan peran yang telah ditatapkan oleh organisasi, hak dan wewenangnya, berperan penting karena akan menentukan apa-apa saja yang boleh dilakukannya demi untuk menunjang perannya di dalam organisasi. Jika pengguna hanya sebagai user biasa, maka diberikan peran Normal User. Dan jika pengguna sebagai admin/superuser, maka diberikan peran Admin User.



Gambar 2. Model Metode Otentikasi Enkripsi Data

Model di atas menunjukkan bagaimana proses (activity) yang terjadi atas data password yang dinputkan pengguna. Tahapan proses yang terjadi adalah :

1. Pengguna melakukan permintaan pada login form untuk username dan password. Pengguna diharuskan memasukkan username dan password yang sebelumnya sudah diberikan oleh organisasi. Peran pengguna saat ini adalah sebagai No Access.
2. Setelah dilakukan proses validasi, dari client-side melalui browser web akan melakukan proses enkripsi terhadap teks password yang dberikan oleh pengguna, yang selanjutnya akan dikirim ke server bersamaa dengan username yang diberikan. Jadi, selama pengiriman data dari client-side ke server-side data password sudah dalam keadaan terenkripsi dengan menggunakan algoritma hash (sha-1).
3. Pada server-side, proses dilanjutkan dengan melakukan otentikasi yaitu membandingkan antara password yang dikirim dengan password yang ada di database, dengan penyeranta kecocokan username. Masing-masing password, baik dari sisi server atau dari sisi klien (yang diinputkan pengguna login), sudah benar-benar dienkrip dengan metode algoritma yang sama. Jika tidak benar-benar otentik, maka peran user akan dikembalikan ke posisi No Access.

HASIL DAN PEMBAHASAN

Persiapan Pengujian

Sebelum dilakukan proses pengujian maka akan dilakukan persiapan-persiapan berkaitan dengan pengujian yang meliputi :

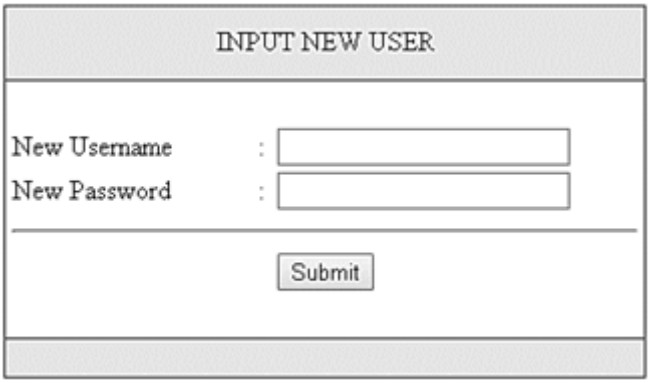
1. Struktur database
2. From input data untuk menyimpan data username dan password pengguna ke database.

Adapun struktur database yang digunakan adalah sebagai berikut :.

Name	Type	NULL
Keys (1)		
Primary Key	fusername	
Fields (3)		
fusername	varchar(8)	No
fpasswd	varchar(255)	Yes
fperan	varchar(15)	Yes

Gambar 3. Struktur Database

Kemudian dibuat sebuah form input menggunakan html untuk digunakan memasukkan data username dan password, yang selanjutnya dikirim ke server untuk disimpan ke dalam database. Berikut ini adalah form untuk membuat dan menyimpan data baru untuk username dan password :



Gambar 4. Form Input New User

Form di atas disimpan dalam file html dengan nama inputForm.html, dimana untuk source code dapat dilihat pada bagian lampiran. Tidak jelaskan bagaimana proses keamanannya pada bagian ini, karena tidak merupakan bagian dari pokok bahasan utama dari penelitian ini.

Berikut ini adalah data yang sudah tersimpan ke dalam database, yang nantinya akan digunakan untuk bahan percobaan dari penelitian ini.

Record	fusername	fpasswd
1	udin1234	0FF6F2C78C3F785FD15525E78E1FE9A223479ED1
2	yuliati8	3C17C905712AE547A70C08EBCF22700B2DF7D74C

Pengujian I : Input Validation

Pengunjian validasi ini akan dibuat menjadi dua bagian, yaitu validasi dari sisi klien dan dari sisi server. Hal ini untuk memberikan hasil yang berbeda dari asumsi prespektif yang berbeda.

- *Pengujian sisi klien*

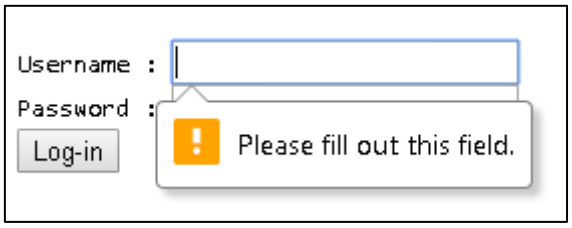
Pada pengujian ini menggunakan sebuah web login form dengan penggalan source code seperti berikut ini:

```
<form method="post" action="loginAction.cfm">
Username : <input type="text" name="vuser" size="20"
                              maxlength="8" required>
Password : <input type="password" name="vpwd" size="20"
                              required>
<input type="submit" value="Log-in">
</form>
```

Form ini disimpan pada file loginForm.html dengan format HTML5, dikarenakan agar bisa menggunakan validation input “required” yang mewajibkan pengguna mengisi form yang diberikan. Dalam hal ini, form tidak boleh kosong. Kode lengkap dari file ini dapat dilihat pada bagian lampiran.

Hasil Pengujian

Form diuji melalui browser dan didapatkan hasil seperti berikut ini:



Gambar 5. Form Field Username Required

Dari gambar di atas dapat diuraikan bahwa:

1. Jika form field tidak diisi, maka akan diberikan peringatan bahwa form harus diisi dan tidak boleh kosong. Demikikan juga untuk field password juga harus diisi.
2. Pada bagian field username, diberikan maxlength=8, hal ini dimaksudkan agar isian field username tidak bisa lebih dari 8 karakter. Hanya saja kelemahannya adalah field ini masih bisa digunakan jika kurang dari 8 karakter.

Meskipun demikian, form harus bisa mencegah user melakukan login jika field username kurang dari 8 karakter, dan user hanya bisa login jika karakter yang diinputkan benar-benar tepat berjumlah 8 karakter. Untuk mengatasinya, akan gunakan validasi dari sisi server yang akan bahas di bawah ini.

- *Pengujian sisi server*

Pada pengujian sisi server, digunakan source code sisi server, coldfusion, dengan nama file inputAction.cfm, yang merupakan file yang digunakan untuk meng-handle pengiriman data form dari file inputForm.html. Berikut ini adalah source code validasi yang digunakan untuk mencegah user menginputkan data kurang dari 8 karakter dari form field username.

```
<cfset vuser = "#form.vuser#">

<!--- validasi sisi server terhadap form field username --->
<cfif len(vuser) lt 8>
    <script>
        alert('Username tidak boleh kurang dari 8 karakter');
        history.back();
    </script>
</cfif>
```

Hasil Pengujian

Saat form field username diisi kurang dari 8, maka akan muncul penolakan atas isian tersebut. Pemberitahuan penolakan dari sisi server atas kurangnya jumlah form field yang ditetapkan. Saat form di isi sesuai dengan aturan validasi, maka proses dilanjutkan ke lapiran kedua keamanan yaitu proses verifikasi.

Pengujian II : Verification

Pada pengujian kali ini melibatkan proses di web server dan juga perbandingan data dengan database. Pengujian bertujuan untuk memastikan bahwa username yang digunakan untuk login benar-benar terdapat di dalam database. File source code yang

digunakan untuk proses verifikasi masih mengunakan file loginAction.cfm, berikut ini adalah source code yang digunakan untuk melakukan verifikasi:

```
<!--- verifikasi untuk memastikan username terdaftar di dalam database --->
<!--- cek keberadaan username dari field input dengan di database --->
<cfquery datasource="danMandiri" name="cekUser">
select fusername from tb_users
where fusername="#vuser#"
</cfquery>

<!--- jika tidak ditemukan, maka verifikasi dibatalkan --->
<cfif cekUser.RecordCount eq 0>
    <script>
    alert('Verifikasi gagal. Username tidak ditemukan!');
    history.back();
    </script>
</cfif>
```

Pengujian dilakukan dengan menginputkan username yang tidak terdaftar di database, dengan asumsi agar bisa mendapatkan hasil respon penolakan (gagal) dari proses verifikasi username pengguna.

Hasil Pengujian

1. Saat isian form field ini dijalankan, maka akan dilakukan pengecekan ke database atas field username. Database memberikan respon bahwa usename tidak tersimpan di dalam database, kemudian web server side menerima informasi tersebut dan menampilkannya di browser. Berikut ini adalah tampilan pemberitahuan dari hasil proses verifikasi atas ketidaksediaan username di dalam database.
2. Saat dimasukkan username yang sesuai dengan isi database, maka proses keamanan akan berlanjut ke lapisan berikutnya.

Pengujian III - Authentication

Pengujian ini akan melibatkan pengujian terhadap password yang sudah dienkripsi dengan algoritma hash (sha-1). Hal ini dikarenakan password yang dibuat -bersamaan pembuatan username- menggunakan enkripsi data dan tersimpan di database. Agar password yang digunakan untuk login dapat di-otentikasi, maka password tersebut harus di enkripsi juga dengan metode yang sama. Berikut ini adalah source code perubahan pada file inputLogin.html yang mempengaruhi proses enkripsi pada password.

Hasil Pengujian

1. Saat form ini dijalankan di web browser dan data username dan password diisikan, saat diklik tomobol Log-in, maka secara otomatis, nilai dari password yang diinput pengguna akan dienkripsi dalam algoritma hash (sha-1). Untuk source code pengkodean algoritma hash ini menggunakan sumber dari pihak ketiga -detailnya dapat dilihat pada source code file sha1.js. Berikut ini adalah gambar form field password yang dienkripsi secara otomatis.

```
<!doctype html>
<html>
<head>
<script src="sha1.js" type="text/javascript"></script>
</head>

<body>
<pre>
<form method="post" action="loginAction.cfm"
            onSubmit="vpwd.value = hex_sha1(vpwd.value);">
Username : <input type="text" name="vuser" size="20"
                  maxlength="8" required>
Password : <input type="password" name="vpwd" size="20"
                  required>
<input type="submit" value="Log-in">
</form>
</pre>
</body>
</html>
```

2. Proses login akan dihandle oleh file inputAction.cfm yang sebelumnya telah dilakukan modifikasi source code-nya, agar mampu untuk melakukan proses otentikasi.
3. Saat proses pada sisi server, inputAction.cfm melakukan pengecekan kesesuaian data password yang dienkripsi dengan yang terdapat di dalam database. Saat perbandingan password ternyata tidak otentik berdasarkan username yang digunakan, maka akan diberikan pemberitahuan bahwa proses otentikasi gagal.
4. Saat proses otentikasi berhasil, maka akan diberikan pesan bahwa proses otentikasi berhasil. Berikut ini adalah tampilan keberhasilan proses otentikasi dari password yang dienkripsi.

```
Otentikasi berhasil.

Data form login    : udin1234 - 3DE1AA7CBB8564363A6AA591B6984020EA515290

Data pada database :udin1234 - 3DE1AA7CBB8564363A6AA591B6984020EA515290
```

5. Setelah proses otentikasi berhasil. Dilanjutkan dengan proses authorization yaitu pemberian kewanangan dan hak sesuai peran yang diberikan oleh organisasi.

Pengujian IV - Authorization

Pengujian pada lapisan terakhir yaitu pemberian peran sesuai dengan peran yang ditetapkan oleh organisasi. Berikut ini adalah list data peran yang tersimpan di dalam database:

Record	fusername	fpasswd	fperan
1	udin1234	3DE1AA7CBB8564363A6AA591B6984020EA515290	normal
2	yuliati8	3C17C905712AE547A70C0BEBCF22700B2DF7D74C	admin

Pada pengujian ini, dibutuhkan 2 buah file tambahan yang digunakan sebagai destinasi dari pengguna setelah login, yang disesuai dengan peran masing-masing pengguna. File tersebut adalah pageNormal.cfm dan pageAdmin.cfm.

Berikut ini source code dari loginAction.cfm yang telah dimodifikasi agar mampu menangani proses keamanan penyerahan kewenangan kepada pengguna.

```
<!--- cek kesesuian password berdasarkan username,
    dari form input dengan yang tersimpan di database --->
<cfquery datasource="dsnMandiri" name="cekPwdByUsr">
select * from tb_users <!--- select semua field yang ada --->
where fpasswd="#vpwd#" and fusername="#vuser#"
</cfquery>

<!--- jika tidak singkron, maka proses otentikasi dibatalkan --->
<cfif cekPwdByUsr.RecordCount eq 0>
    <script>
    alert('Otentikasi gagal. Password salah.')
    history.back()
    </script>
<cfelse>
    <cfoutput>
        <!--- proses keamanan authorization --->
        <cfif cekPwdByUsr.fperan is "normal">
            <cflocation url="pageNormal.cfm">
        <cfelse>
            <cflocation url="pageAdmin.cfm">
        </cfif>
    </cfoutput>
</cfif>
```

Hasil Pengujian

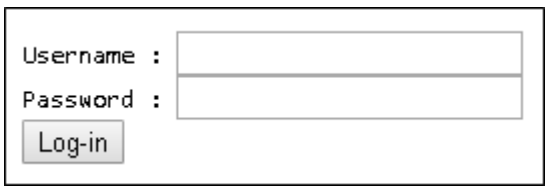
1. Saat dijalankan, application web server di atas akan memproses pengecekan kewenangan atas peran yang dimiliki oleh pengguna yang sudah login.
2. Hasil tampilan yang ditujukkan untuk pengguna dengan akun udin1234 adalah berperan sebagai “user Normal”.

Pengujian V – Source Code Encryption

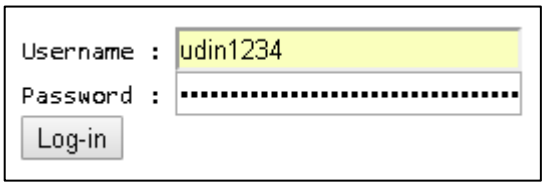
Untuk pengujian enkripsi source code ini menggunakan tool dari pihak ketiga (SmartGb, 2018), tanpa menghilangkan esensi untuk pengamanan data source code asli. Enkripsi ini hanya berlaku untuk sisi klien, sedangkan source code dari sisi server tidak dapat digunakan.

Hasil Pengujian.

1. Saat login dijalankan, file dapat berjalan sempurna tanpa memberikan pesan error.



2. Enkripsi terhadap field password form login berfungsi dengan baik, walaupun sudah disisipi dengan algoritma hash (sha-1) yang digunakan untuk mengenkripsi value dari field password. Tidak ada pesan error yang ditampilkan.



Proses login berhasil berjalan baik hingga menampilkan welcome home setelah login.

KESIMPULAN

Secara umum kesimpulan penelitian ini adalah sebagai berikut:

1. Web login form dapat menjadi pintu masuk bagi hacker untuk melakukan penetrasi terhadap internal system jika tidak menggunakan pengamanan yang lebih baik.
2. Metode ACL dapat digunakan untuk system keamanan berlapis pada web login form.
3. Penanganan system keamanan metode ACL dapat dilakukan melalui server side terhadap web login form.
4. Kelemanan yang muncul dari keamanan web login form menggunakan metode ACL dapat dieksplorasi seminimal mungkin. Hal ini dikarenakan keterbatasan hak akses yang diberikan.
5. Keamanan enkripsi source code dapat mengatasi kelemahan dari keamanan web login form metode ACL dikarenakan sulitnya hacker untuk mengetahui variable yang digunakan pada field form.

DAFTAR PUSTAKA

Bugliesi, M., Calzavara, S., Focardi, R., 2016. ***Formal methods for Web security***. J. Log. Algebr. Methods Program. 87. https://doi.org/10.1016/j.jlamp.2016.08.006

Davies, J.N., Comerford, P., Grout, V., Davies, J.N., Comerford, P., Grout, V., 2012. ***Principles of Eliminating Access Control Lists within a Domain***. Future Internet 4, 413–429. https://doi.org/10.3390/fi4020413

J. Radianti, 2010. In Proceedings of the 4th International Conference on Emerging Security Information Systems and Technologies (SECURWARE), ***Eliciting information on the vulnerability black market from interviews***. Vanice, Italy.

Kaczmarczyk, V., Bradáč, Z., Fiedler, P., Arm, J., 2016. ***Client side data encryption/decryption for web application***. IFAC-Pap., 14th IFAC

Conference on Programmable Devices and Embedded Systems PDES 2016 49, 241–246. https://doi.org/10.1016/j.ifacol.2016.12.041

S. Frei, D. Schatzmann, B. Plattner, B. Trammell, 2010. ***Economics of Information Security and Privacy, Modeling the security ecosystem - The dynamics of (In)Security***. Springer Verlag.

SmartGb, S., 2018. ***Encrypt HTML***. SmartGb.

Tommi Tulisalo, Rune Carlsen, Andre Guirard, Pekka Hartikainen, Grant McCarthy, Gustavo Pecly, 2002. D***omino Designer 6: A Developer's Handbook*** [Book] [WWW Document]. URL https://www.safaribooksonline.com/library/view/domino-designer-6/073842658X/ (accessed 9.5.18).

Zhao, M., Grossklags, J., Chen, K., 2014. ***An Exploratory Study of White Hat Behaviors in a Web Vulnerability Disclosure Program***, in: Proceedings of the 2014 ACM Workshop on Security Information Workers, SIW '14. ACM, New York, NY, USA, pp. 51–58. https://doi.org/10.1145/2663887.2663906