]HackingTeam[| gesi RAS

GE.S.I.

Application Assessment

PREVIP

Milano, 30 Marzo 2007

Hacking Team S.r.l.	http://www.hackingteam.it
Via della Moscova, 13 *20121 MILANO (MI) - Italy*	info@hackingteam.it
Tel. +39.02.29060603	*Fax +39.02.63118946*

STORIA DEL DOCUMENTO		
Versione	Data	Modifiche Effettuate
1.0	29 marzo 2007	Prima stesusa
1.1	30 marzo 2007	Revisione ed emissione
//	//	//

INFORMAZIONI	
Data di Emissione	30 marzo 2007
Versione	1.1
Tipologia Documento	Documento di Progetto
Numero di Protocollo	//
Numero Pagine	25
Numero Allegati	//
Descrizione Allegati	//
Redatto da	Massimo Chiodini
Approvato da	Gianluca Vadruccio

Indice

Indice delle figure

Descrizione dell'attività

GE.SI S.p.A. ha richiesto un intervento di *Ethical Hacking* atto a rilevare lo stato di sicurezza del portale denominato UNIFONDI.
L'attività è stata suddivisa in due parti:

1. Analisi black-box: viene analizzato il sistema a partire da tutte le sue componenti pubbliche, cioè tutte quelle raggiungibili senza un controllo degli accessi. In questa fase ci si pone nei panni attaccante generico senza nessun privilegio particolare, che accede tramite Internet all'applicativo.
2. Analisi white-box: il cliente ha messo a disposizione delle credenziali abilitate ad accedere all'applicativo PREVIP. Queste sono state utilizzate per accedere all'applicazione con lo scopo di testare le componenti interne. Questo scenario presuppone un attaccante (cioè un utente reale, o un attaccante che abbia ottenuto in qualche modo delle credenziali) in possesso di credenziali che gli permettano l'accesso all'applicativo secondo il profilo associato.

L'analisi è stata svolta utilizzando sia strumenti automatici di *scanning*, sia agendo manualmente per approfondire gli aspetti più rilevanti e per testare la presenza di vulnerabilità non standardadizzati dell'applicativo.

Per intraprendere l'attività il cliente ha fornito l'ULR del sistema :[http://stageprevip.ras] e i seguenti account utilizzati durante l'analisi White Box:

Fondo	utenza	password	azienda
PREVIP	ALBDAN130967	unifondi	Cassa di risparmio di Alessandria
PREVIP	CUCGIO140654	unifondi	MUNCHENER RUCK ITALIA
PREVIP	LOCANN310866	unifondi	MUNCHENER RUCK ITALIA
PREVIP	MURGIO020747	unifondi	AIG EUROPE

L'intero progetto è stato svolto sull'ambiente di preproduzione presso le infrastrutture del *GE.SI S.p.A*.

Analisi Black-Box

Obiettivo dell'attività è quello di identificare vulnerabilità e lacune su tutte quelle componenti che sono direttamente raggiungibili anche senza un controllo degli accessi.
Questo insieme comprende gli elementi che compongono il sistema sia da un punto di vista applicativo (es. meccanismi di autenticazione, pagine statiche e dinamiche, risorse pubbliche, ecc.), che di tipo prettamente sistemistico (software utilizzato e tecnologia di sviluppo, canali di comunicazione, oggetti installati di default, meccanismi di amministrazione remota, esempi, ecc.)

Test eseguiti

L'attività è stata svolta presso i locali del cliente, il quale ha fornito tutte le informazioni necessarie per eseguire i test. I sistemi analizzati sono quelli di pre-produzione [http://stageprevip.ras, http://securestage.ras].

I test eseguiti possono essere raggruppati nelle seguenti macro-attività:

- Analisi architettura: network analysis, enumeration, channel analysis;
- Autenticazione: brute force, SQL Injection, parameter tampering;
- Input manipulation: SQL injection, cross-site scripting, parameter tampering, fuzzing;
- Information leakage: forceful browsing, resource enumeration, hidden fields.

Gli strumenti utilizzati per lo svolgimento delle attività sono svariati e molteplici e vanno dai sofisticati software di *scan* commerciale, a *sniffer* di rete, *mitm proxy*, *ssl clients*…

Analisi architetturale

L'applicativo è strutturato secondo il classico paradigma 3-tier.

- Il *front-end web* è costituito da un server HTTP.

- La *business logic* è implementata in parte con tecnologia JSP
- Il *data layer* è costituito database relazionale.

La zona pubblica consente di accedere alle informazioni di natura istituzionale ed è stata realizzata tramite elementi *HTML* statici.
La versione del web server e dei moduli utilizzati non sono allineati all'ultima versione, e sono potenzialmente vulnerabili. Se non già completato, si consiglia di installare le hotfix consigliate dal produttore [Security issue: M-01]. Informazioni in merito ad alcuni esempi di vulnerabilità legate a questa versione di software possono essere trovate ai seguiti links:

[http://www.securityfocus.com/bid/19204/]
[http://www.securityfocus.com/bid/11360/]
[http://www.securityfocus.com/bid/10736/]

Il sistema *consente* di utilizzare un canale di comunicazione cifrato per accedere alle proprie risorse.
Di default, il server di frontiera utilizza *SSL TLSv1* per cifrare la connessione, ma accetta anche *handshake* con protocolli di versione minore (sslv2). Il protocollo sslv2 è esposto ad una serie di attacchi (*Ciphersuite rollback attacks, Key-exchange algorithm rollback, Replay attacks*, ecc.) che lo rendono potenzialmente insicuro.
Le *Ciphersuite* disponibili sono:

```
Ciphers common between both SSL endpoints:
RC4-MD5   EXP-RC4-MD5     RC2-CBC-MD5
EXP-RC2-CBC-MD5 DES-CBC-MD5     DES-CBC3-MD5
RC4-64-MD5
---
SSL handshake has read 1112 bytes and written 367 bytes
---
New, SSLv2, Cipher is DES-CBC3-MD5
[..]
Expansion: NONE
SSL-Session:
   Protocol  : SSLv2
   [..]
   Krb5 Principal: None
   Start Time: 1174905489
```

Tra queste, alcune sono da considerarsi con basso grado di sicurezza (es. RC4 40 bit); quindi, se non indispensabili, si consiglia di disabilitarle o di impedire il downgrade del protocollo ssl.

È bene notare che in ogni caso il sistema consente l'accesso alle componenti applicative anche utilizzando un canale in chiaro.
In ambito di sicurezza è fortemente consigliabile disabilitare questa caratteristica per garantire una maggior riservatezza delle informazioni, e prevenire inconvenienti di ogni sorta, nel caso in cui questi dati transitino su reti o sistemi già compromessi.

Autenticazione

Il front web pubblico consente di accedere alla pagina di login per accedere all'applicativo vero e proprio.
Tale accesso avviene inserendo le credenziali tramite delle normali forms HTML

- *https://securestage.ras/UnifondiRASNP/fondoPrevip/loginPrevip.jsp*
- *https://securestage.ras/UnifondiRASNP*.

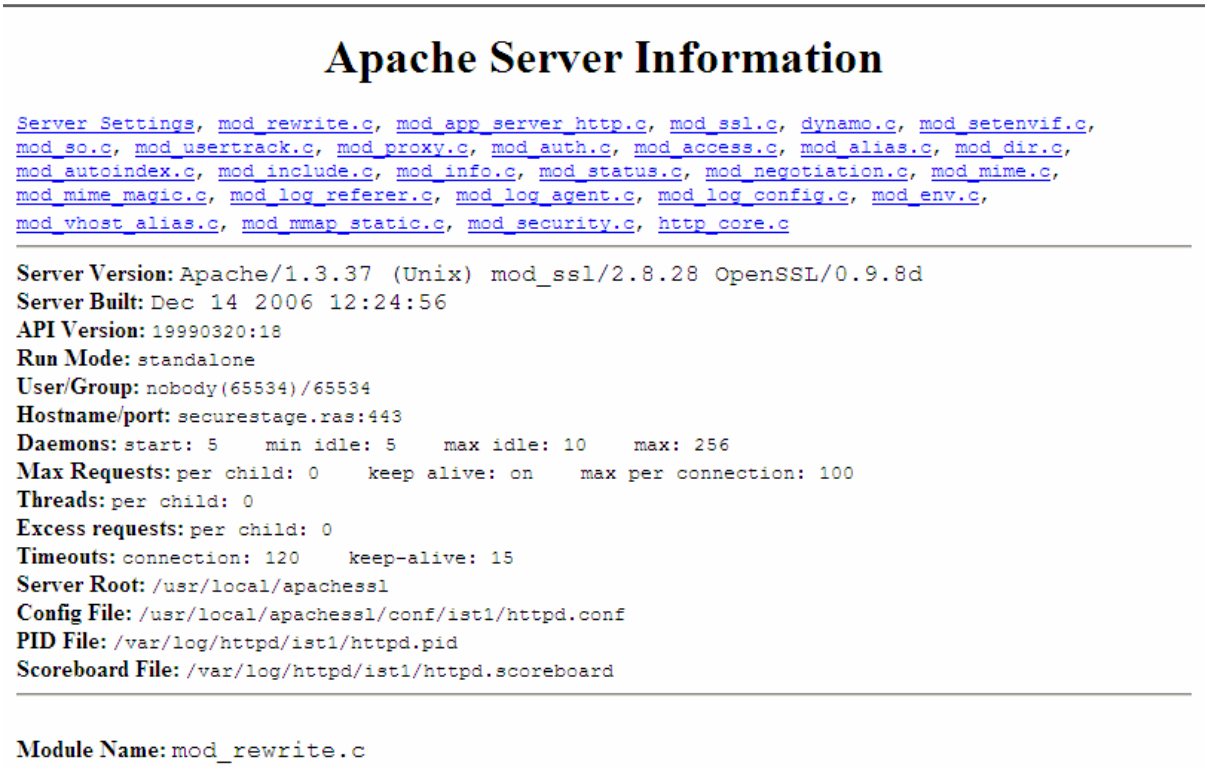
In tutti i casi, la logica effettua una corretta validazione dei dati in input, e non è risultato possibile attaccare il sistema di autenticazione con le comuni tecniche applicative.
Tutte le "eccezioni di codice" sono trattate in modo irreprensibile e non è possibile ottenere informazioni sulla tipologia di dati, o sulle strutture interne effettuando *fuzzing* sui parametri di ingresso (*hidden field, input text*).
Il meccanismo di autenticazione risulta essere ben progettato e resistente ad attacchi diretti e non è stato possibile accedere al sistema senza l'utilizzo di credenziali valide fornite dal cliente.

Input Manipulation

N.A.

Information leakage

Scanning effettuati con tools automatici non hanno evidenziato alcune componenti installate di default (http://securestage.ras/*server-info,* http://securestage.ras*/server-status,* http://securestage.ras/cgi-bin/*test-cgi*, http://securestage.ras/robots.txt) che consentono di ottenere status e configurazioni del front-end.



Apache Server Information

Server Settings, mod_rewrite.c, mod_app_server_http.c, mod_ssl.c, dynamo.c, mod_setenvif.c, mod_so.c, mod_usertrack.c, mod_proxy.c, mod_auth.c, mod_access.c, mod_alias.c, mod_dir.c, mod_autoindex.c, mod_include.c, mod_info.c, mod_status.c, mod_negotiation.c, mod_mime.c, mod_mime_magic.c, mod_log_referer.c, mod_log_agent.c, mod_log_config.c, mod_env.c, mod_vhost_alias.c, mod_mmap_static.c, mod_security.c, http_core.c

Server Version: Apache/1.3.37 (Unix) mod_ssl/2.8.28 OpenSSL/0.9.8d
Server Built: Dec 14 2006 12:24:56
API Version: 19990320:18
Run Mode: standalone
User/Group: nobody(65534)/65534
Hostname/port: securestage.ras:443
Daemons: start: 5 min idle: 5 max idle: 10 max: 256
Max Requests: per child: 0 keep alive: on max per connection: 100
Threads: per child: 0
Excess requests: per child: 0
Timeouts: connection: 120 keep-alive: 15
Server Root: /usr/local/apachessl
Config File: /usr/local/apachessl/conf/ist1/httpd.conf
PID File: /var/log/httpd/ist1/httpd.pid
Scoreboard File: /var/log/httpd/ist1/httpd.scoreboard

Module Name: mod_rewrite.c

Figura 1 - Oggetti di installati di default

Le configurazioni mostrano come le *permissions* consentono l'accesso a tutti gli indirizzi IP di classe privata su cui sono attestate le reti interne di RAS *assicurazioni*. Queste lacune sistemistiche non sono state riscontrate sui sistemi in produzione.

Riepilogo vulnerabilità/soluzioni

[Security issue: M01]

Problema: Utilizzo di una versione “obsoleta” dei sistemi web

Descrizione: la versione del web server risulta essere non allineata all’ultima disponibile

Impatti: Il software è afflitto da diversi e noti *bugs* di sicurezza che possono essere sfruttati per ottenere il controllo del sistema o causare malfunzionamenti

Soluzione: aggiornare il sistema all’ultima release disponibile, oppure assicurarsi di avere installato tutte le hotfix rese disponibili dal produttore

[Security issue: L01]

Problema: Canale di comunicazione *clear text*

Descrizione: L’accesso al sistema applicativo può avvenire utilizzando un canale di comunicazione senza cifratura, o con protocolli e standard crittografici deboli

Impatti: I dati e le informazioni veicolati su un canale in chiaro o utilizzando standard non adeguati,che transitano su reti precedentemente compromesse, possono essere visionate o modificate da potenziali attaccanti. .

Soluzione: modificare le configurazioni del sistema web per disabilitare ciphersuite deboli o versioni di protocollo minore di TLSv1/SSLv3.

[Security issue: M02]

Problema: Elementi di default

Descrizione: in fase di installazione non sono stati eliminati alcuni oggetti di default

Impatti: l’accesso a questi oggetti consente di ottenere informazioni dettagliate sulla configurazione e l’installazione del sistema. Informazioni che possono risultare preziose per proseguire un attacco.

Soluzione: rimuovere o modificare i permessi di accesso a tutte quelle componenti che non sono di stretta necessità al buon funzionamento del sistema.

Analisi White-Box

L'obiettivo di questa fase è compiere un'analisi di sicurezza sulle componenti applicative raggiungibili solo dopo aver acceduto al sistema con credenziali valide ed autorizzate.
Le finalità di tale attività sono le seguenti:

- identificare vulnerabilità nella procedura applicative;
- testare i meccanismi di profilatura degli utenti (accesso e *granting* dei diritti);
- verificare le modalità di accesso alle informazioni (sessioni applicative)...

Test eseguiti

Anche questa parte di progetto è stata svolta nei locali del cliente ed i sistemi coinvolti sono i medesimi elencati in precedenza.
I test eseguiti in questa fase sono:

- Analisi architettura: *enumeration*, *channel analysis;*
- Sessions: analisi dei meccanismi di sessione, *cookie tampering;*
- Input manipulation: *SQL injection, cross-site scripting, parameter tampering, fuzzing;*
- Information leakage*: forceful browsing, resource enumeration, hidden fields*.

Strumenti utilizzati per lo svolgimento sono diversi e di comune reperibilità: *sniffer* di rete, *mitm proxy*, *ssl clients*, scanner applicativi...

Analisi architetturale

L'attività non ha messo in luce particolari criticità aggiuntive a quelle già descritte durante l'approccio Black-box.

Autenticazione

L'accesso all'area appllcativa avviene tramite un meccanismo di autenticazione username/password.

La scelta dei nomi utente e delle password è risultata essere di buona complessità e consente un'adeguata protezione contro attacchi di tipo "*account guessing*". Non è stato riscontrato l'utilizzo di meccanismi di *lockout* dei tentativi di accesso errato ma in questo caso, essendo i nomi utenti non facilmente guessabili, non costituisce un problema di sicurezza.

Sessioni

Le sessioni applicative sono implementate utilizzando *session cookies* generati e mantenuti dal server e dall'applicativo.

I meccanismi di validazione delle credenziali e il ciclo di vita delle *sessioni* sono gestiti correttamente, e durante i test non si è riusciti a forzarli o riutilizzare sessioni per accedere in maniera illecita al sistema.

Il corretto utilizzo dei meccanismi di *sessione* permette di prevenire furti di identità o di informazioni: tutti i dati relativi all'utente loggato vengono mantenuti server-side e risulta impossibile accedere a risorse di proprietà di un altro profilo utente.

Input manipulation

Il sistema gestisce con puntualità e perizia tutto l'input inviato *client side*: *forms*, *hidden fields*, parametri ecc. vengono controllati dalla logica lato *server* e puntualmente puliti per mezzo di *sanity check*.

I tentativi di *fuzzing, Sql injection, cross-site scripting*, ecc. non hanno dato nessun risultato. Le poche *code exception* trovate non danno informazioni sulla tipologia di dati, o sulle strutture interne utilizzate dalla logica.

POST https://securestage.ras/UnifondiRASNP/is/actionIsDettaglioContributiInit.do HTTP/1.1
[..]

Cookie:JSESSIONID=0001PgBZ80bKLWT6Q0I0YcHoHp_:10rhnar6r;
JSESSIONID_NSERVA1=0001ximt4-eHI6o4Xa2vh_apXmU:10pmum2sr

method=aggiorna&tipoContributo=TUTTI&dataIni=&dataFin=&raggruppamentoData=NESSUNO&raggruppamento=' and 1=1 ;--//

Response: HTTP 200 OK

<td align="left" class="ErrorText">

[..]
**
**
Errore in fase di esecuzione della query
[..]
</table>
[..]
Host:securestage.ras

Uno dei pochi tentativi di *parameter manipulation* che ha avuto successo è quello relativo al dettaglio di richiesta trasferimento fondi [Security issue: H01]: questo usa un *hidden field* con codice numerico per indicare la pratica. Il *tampering* del codice consente ad un utente valido di accedere a informazioni di altri utenti.

POST https://securestage.ras/UnifondiRASNP/is/actionIsTrasferimentiAltriFondi.do HTTP/1.1
[..]
Cookie: JSESSIONID_NSERVA1=0001-CJvPCVJFoMJIYFF0JmRLYn:10pmum2sr

method=to&methodParams=606938

Il parametro “MethodParams” può essere manipolato per visualizzare informazioni inserite da altri utenti.
L’esempio che segue mostra come i dettagli di un trasferimento fondi dell’utente locann310654, sia accessibile anche da una sessione creata con l’utente ALBDAN130967 (MethodParams = 606938)

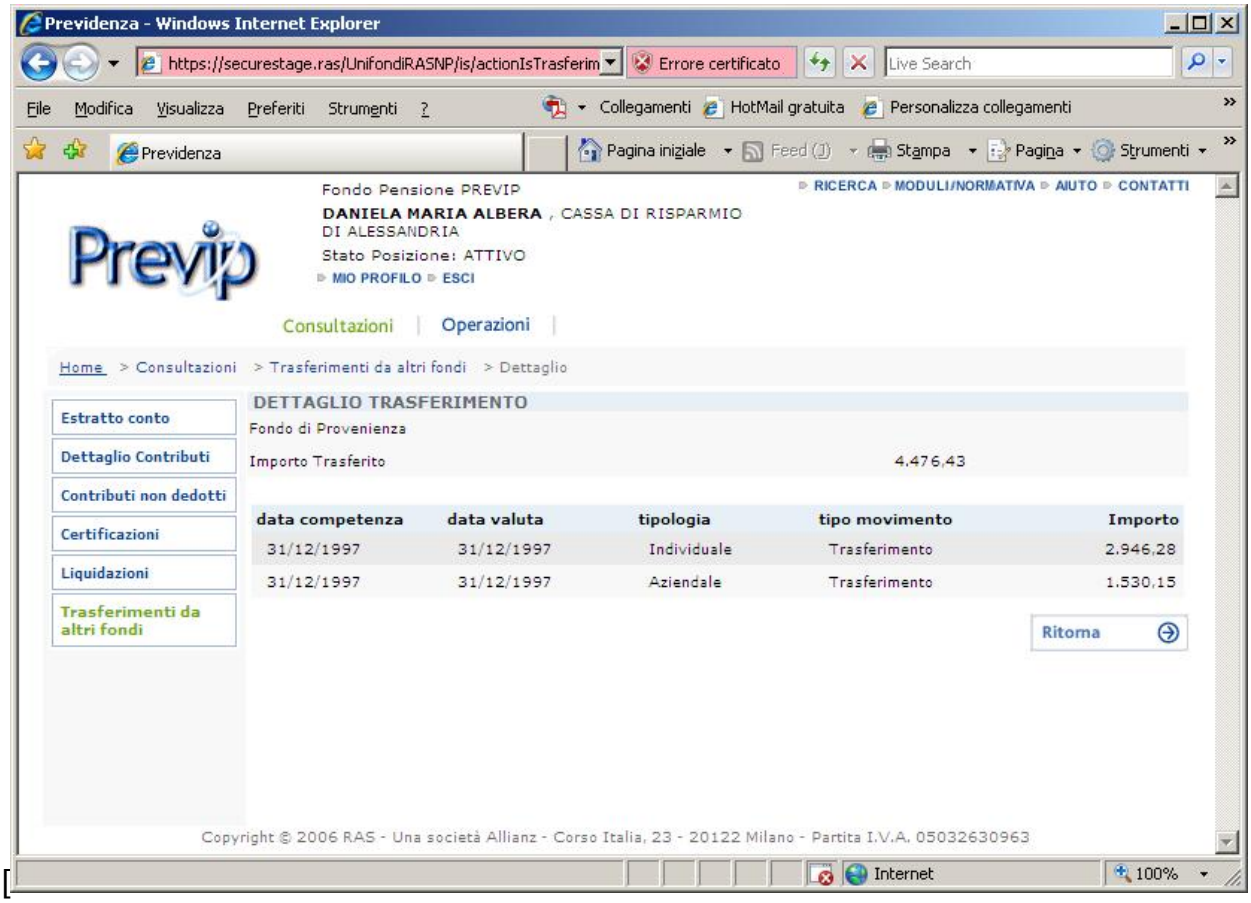


Figura 2 - Dettaglio trasferimento fondi A

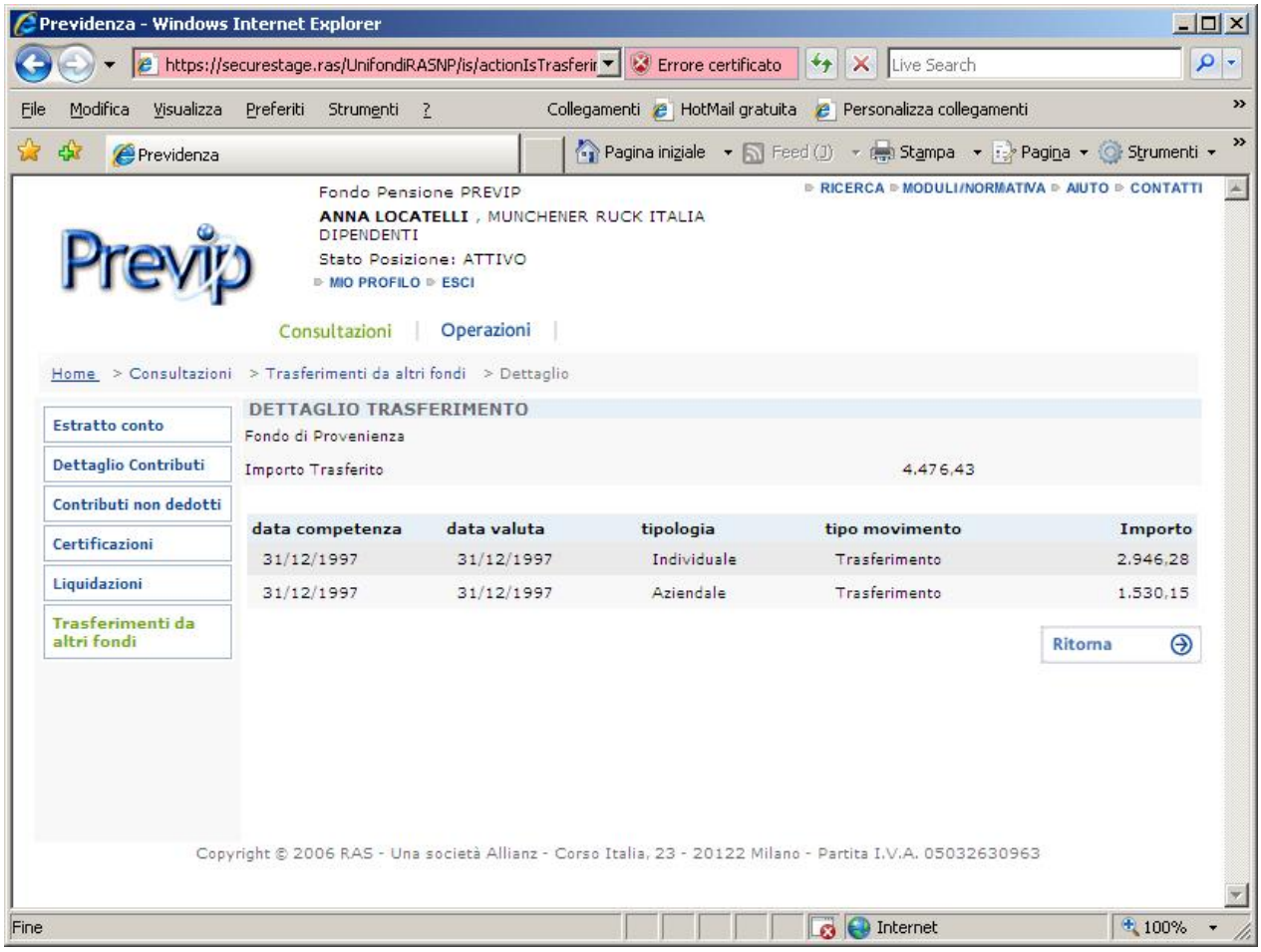


Figura 3 - Dettaglio trasferimento fondi B

Information leakage

Utilizzando le componenti per la visualizzazione delle FAQ è possibile enumerare file e directory del server applicativo [Security issue: M03]: tramite URL [https://securestage.ras/UnifondiRASNP/priv/actionFAQInit.do?method=dettaglio&webcURL=/../minisiti/it] un utente è in grado di reperire le risposte alle domande frequenti. La logica applicativa usa come parametro il nome del file da visualizzare (webcURL), e modificando tale nome è possibile enumerare e visualizzare cartelle e file del server web.

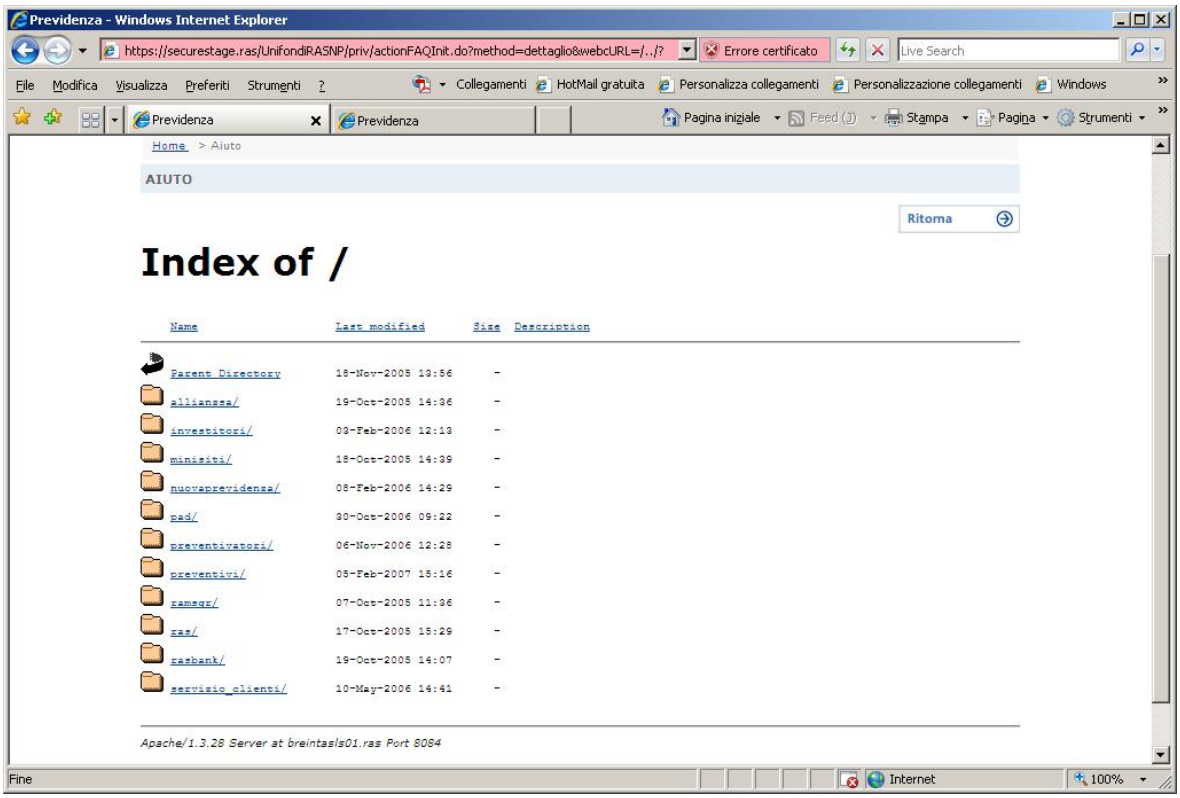


Figura 4 - Directory Enumeration

Riepilogo vulnerabilità/soluzioni

[Security issue: M03]

Problema: È possibile utilizzare il sistema di visualizzazione delle FAQ per enumerare risorse di sistema.

Descrizione: La logica per visualizzare le risposte alle domande frequenti, usa come parametro il *relative pathname* della pagina che contiene la risposta. Modificandolo risulta possibile visualizzare risorse come directory, pagine HTML, ecc. non direttamente accessibili seguendo il normale flusso operativo.

Impatti: l'accesso a questi elementi può essere utile per ottenere informazioni preziose per proseguire un attacco**.**

Soluzione: revisione della logica: effettuare *sanity check* sul parametro, restringendo la tipologia e le *locations* dei files e a cui può avere accesso la componente.

[Security issue: H01]

Problema: Un utente è in grado di visualizzare dati relativi ai dettagli trasferimento fondi inseriti da un altro utente applicativo.

Descrizione: Eseguendo del parameter tampering sul codice relativo al dettaglio di trasferimento fondi, un utente può enumerare tutti i codici validi e visionare i dati inseriti da altri utenti.

Impatti: utenti appartenenti ad altre organizzazioni (società, privati, ecc.) possono visionare dati e informazioni che dovrebbero essere riservati solo a coloro che ne hanno pieno diritto.

Soluzione: modificare la struttura di come vengono reperite le informazioni aggiungendo un meccanismo di controllo di accesso ai dati.

Considerazioni conclusive

Nella sua globalità il sistema UNIFONDI mostra un buon livello di attenzione a problematiche di sicurezza.

Le poche lacune riscontrate, non sono risultate utili a produrre scenari di attacco plausibili, e non abbassano il buon grado di sicurezza del sistema. Questa considerazione può essere valida sia in presenza di un attaccante non in possesso di credenziali valide (approccio *Black Box*), sia nel caso in cui si sia ottenuto un account per accedere all'applicativo vero e proprio (approccio *White Box*).

I meccanismi preposti a garantire la discrezionalità degli accessi (sistemi di autenticazione) è risultata essere robusta e ben implementata, e protegge il sistema contro i più comuni attacchi di tipo applicativo.

L' ingegnerizzazione della logica applicativa è da considerarsi di buon livello e costituisce una solida base per il trattamento e l'integrità delle informazioni.

La riservatezza dei dati è mantenuta da un ottimo sistema di gestione delle *sessioni* (*application sessions*) che non permette nessun tipo di "*escamotage*" atto ad aggirare la logica di controllo.

Unica segnalazione degna di nota riguarda la gestione dei dettagli relativa alla richiesta di trasferimento fondi in cui all'interno di una sessione applicativa valida ed autenticata un utente (con *skills* tecnici medio-bassi) può enumerare e visionare pratiche riservate di altri utenti.

Un intervento di tipo sistemistico può aumentare maggiormente il grado di sicurezza del sistema, forzando l'utilizzo esclusivo di standard crittografici forti (SSLV3/TLS1), veicolando tutto il flusso informativo su canali sicuri e riservati.

N.B.

Le debolezze evidenziate nel presente documento sono state prontamente segnalate al cliente durante lo svolgimento dei test, il quale ha già in corso d'opera gli interventi di manutenzione correttiva necessari.

Sintesi finale e raccomandazioni

L'attività di analisi ha messo in luce alcuni problemi di differente gravità (sintetizzati nella tabella seguente) che necessitano di interventi correttivi in grado di eliminare del tutto la possibilità che possano essere sfruttati per scopi fraudolenti.

Vulnerabilità	Impatto	Soluzione
Utilizzo di software non allineati all'ultima versione. (Web server e moduli ssl)	Il sistema usa versioni datate di web server e dei moduli utilizzati da quest'ultimo. Tali versioni soffrono di diverse vulnerabilità che possono compromettere il buon funzionamento del sistema e l'integrità delle informazioni trattate.	Aggiornare il software con l'ultima versione disponibile oppure effettuare una puntale revisione delle hotfix fornite dal produttore.
Canale di comunicazione clear text (http) o con standard di basso profilo di sicurezza (sslv2).	È possibile raggiungere il sistema utilizzando un canale di comunicazione in chiaro o con standard di cifratura non adeguati. Le informazioni applicative che transitano su reti o sistemi compromessi possono essere esposte alla visione da parte di persone non autorizzate.	Modificare le configurazioni del sistema web per disabilitare ciphersuite deboli o versioni di protocollo minori di TLSv1/SSLv3.
Oggetti o componenti installati di default.	In fase di installazione non sono stati eliminati alcuni oggetti di default. L'accesso a questi oggetti consente di ottenere informazioni dettagliate sulla configurazione e l'installazione del sistema. Informazioni che possono risultare preziose per proseguire un attacco	Rimuovere o modificare i permessi di accesso agli oggetti o alle componenti menzionate nel documento.
È possibile utilizzare il sistema di visualizzazione delle FAQ per enumerare e visionare risorse di sistema.	La logica per visualizzare le risposte alle domande frequenti, usa come parametro il relativo pathname della pagina che contiene la risposta. Modificandolo risulta possibile visualizzare risorse come directory, pagine HTML, ecc. non direttamente accessibili seguendo il normale flusso operativo. Impatti: l'accesso a questi elementi può consentire di accedere a che possono risultare preziose per proseguire un attacco	Effettuare sanity check sul parametri delle logica applicativa, restringendo la tipologia e le locations dei files e a cui può avere accesso la componente.
Un utente è in grado di visualizzare dati relativi ai dettagli trasferimento fondi inseriti da un altro utente applicativo.	Eseguendo del parameter tampering sul codice relativo al dettaglio di trasferimento fondi, un utente può enumerare tutti i codici validi e visionare i dati inseriti da altri utenti. Utenti appartenenti ad altre organizzazioni (società, privati, ecc.) possono visionare dati e informazioni che dovrebbero essere riservati solo a coloro che ne hanno pieno diritto.	Modificare la struttura di come vengono reperite le informazioni aggiungendo un meccanismo di controllo di accesso ai dati.

Appendice: terminologia utilizzata

Gli attacchi di livello applicativo sfruttano vulnerabilità (sia di natura logico-architetturale, sia di natura implementativa) per indurre nelle applicazioni comportamenti anomali, le cui conseguenze possono essere le più disparate: crash dell'applicazione, furto di dati, accesso ai sistemi su cui le applicazioni sono eseguite, ecc.
Le tecniche di attacco che sono state utilizzate sono descritte nei seguenti sottoparagrafi.

Cross site scripting

- **Obiettivo**: furto d'identità ai danni di utenti di applicazioni web che fanno uso di cookie per la gestione delle sessioni.
- **Attaccanti**: chiunque sia interessato al furto dell'identità di un utente autorizzato (che abbia stabilito una sessione con l'applicazione web).
- **Descrizione**: si tratta di una tecnica che, mediante l'inserimento di elementi di scripting nei parametri inviati all'applicazione, provoca l'esecuzione degli stessi da parte del browser della vittima. Gli elementi di scripting causano l'invio dei cookie settati dall'applicazione target sul browser della vittima verso server un HTTP sotto il controllo dell'attaccante. Solitamente l'obiettivo dell'attacco è il cookie di sessione della vittima. La conoscenza di questo cookie permette infatti di sottoporre richieste all'applicazione utilizzando l'identità della vittima. Gli attacchi di cross site scripting sono possibili quando l'applicazione web restituisce al browser (per normale logica di funzionamento o a causa di una condizione di errore) parametri sottoposti dall'utente in una precedente richiesta.
- **Condizioni necessarie per l'attacco**: assenza di controlli sull'input ricevuto ed errori relativi all'escaping di metacaratteri nell'HTML ritornato al browser.
- **Probabilità di successo**: questo attacco richiede l'uso di tecniche di social engineering per indurre la vittima a stabilire una sessione con l'applicazione target e sottoporre ad essa una richiesta contenente il codice malizioso. Frequentemente questo viene fatto per mezzo di email che invitano a seguire un link verso l'applicazione target. La probabilità di successo di tali attacchi è solitamente bassa.

- **Aspetti facilitanti**: in alcuni casi è possibile inserire elementi di scripting in parametri che vengono salvati su database e restituiti all'utente ad ogni successiva richiesta (database XSS). Questo determina l'invio di cookie verso il server HTTP sotto il controllo dell'attaccante ogni volta che la vittima accede all'applicazione ed aumenta in modo considerevole la probabilità che l'attaccante riesca ad utilizzarlo con successo.
- **Contromisure**: gli attacchi di tipo XSS possono essere neutralizzati mediante le seguenti tecniche:
  - filtraggio dei parametri in input, mediante filtri che eliminano dall'input i metacaratteri utilizzati in HTML e linguaggi di scripting client-side (<, >, apici, ecc.);
  - escaping dei metacaratteri contenuti nei parametri in input che devono essere inseriti in pagine HTML restituite al browser degli utenti.

SQL Injection

- **Obiettivo**: gli attacchi basati su SQL injection possono avere diversi obiettivi:
  - accesso ad informazioni riservate memorizzate sui database server che costituiscono il data layer dell'architettura applicativa attaccata;
  - accesso non autorizzato all'applicazione, aggirando il meccanismo di autenticazione;
  - esecuzione di comandi sui server del data layer.
- **Attaccanti**: utenti autorizzati che mirano ad accedere ad informazioni per le quali non possiedono diritti di accesso; utenti non autorizzati.
- **Descrizione**: si tratta di tecniche di manipolazione dei parametri in input utilizzati dall'applicazione per eseguire query SQL sul database. Lo scopo è sovvertire la logica della query in modo da ottenere:
  - messaggi di errore contenenti informazioni sulla struttura del database utilizzato;
  - informazioni differenti da quelle che la query dovrebbe estrarre;
  - recordset vuoti o tali da produrre un malfunzionamento dei meccanismi di autenticazione, allo scopo di accedere all'applicazione senza essere in possesso di credenziali valide.

- **Condizioni necessarie per l'attacco**: mancanza di filtri di validazione dell'input, che eliminano dai parametri inviati dall'utente token pericolosi, come parole chiave riservate del linguaggio SQL (ad esempio, SELECT, OR, ecc.).
- **Probabilità di successo**: fortemente dipendenti dalla logica applicativa.
- **Aspetti facilitanti**: la visualizzazione, sul lato client, dei messaggi di errore relativi all'accesso al database permette all'attaccante di raccogliere informazioni sulla sua struttura, aumentando significativamente le probabilità di successo.
- **Contromisure**: gli attacchi di tipo SQL injection possono essere neutralizzati mediante le seguenti tecniche:
  - filtraggio dei parametri in input, mediante filtri che eliminano dall'input token riservati e metacaratteri del linguaggio SQL;
  - gestione degli errori di accesso al layer di accesso ai dati, allo scopo di intercettare e bloccare la visualizzazione lato client dei messaggi di errore.

Path traversal

- **Obiettivo**: browsing di directory presenti sul web server ma non appartenenti alle applicazioni web pubblicate su di esso, per le quali non è previsto l'accesso da parte degli utenti.
- **Attaccanti**: questo tipo di attacco può essere portato da chiunque possa stabilire una connessione HTTP verso i server su cui è pubblicata l'applicazione.
- **Descrizione**: un attacco di path traversal consiste nella sottomissione di richieste verso il web server per risorse il cui URL contiene path non appartenenti alle applicazioni web pubblicate su di esso. Poiché in generale tali path non sono noti all'attaccante, essi vengono specificati in forma relativa, partendo dalla posizione di risorse note ed utilizzando sintassi del tipo "../../" per navigare a ritroso il file system. Si noti che questo attacco non è in alcun modo correlato alla logica applicativa, ma sfrutta eventuali vulnerabilità del server HTTP.
- **Condizioni necessarie**: queste tecniche possono essere utilizzate in presenza di web server sui quali non sono installate le security patch opportune; in ogni caso, la loro applicabilità non dipende dalla particolare applicazione pubblicata sul web server.

- **Probabilità di successo**: dipende dall'accuratezza della manutenzione del web server.
- **Aspetti facilitanti**: nessuno
- **Contromisure**: aggiornamento dei web server mediante applicazione delle opportune patches.

Cookie poisoning

- **Obiettivo**: gli obiettivi possono essere molteplici; essi dipendono dalla logica dell'applicazione attaccata. In generale, le tecniche di cookie poisoning mirano a provocare comportamenti non previsti nell'applicazione attaccata in modo da poter interagire con essa in modi non previsti dal programmatore.
- **Attaccanti**: gli attacchi di cookie poisoning possono provenire da qualsiasi utente sul cui browser l'applicazione setta cookie.
- **Descrizione**: le tecniche di cookie poisoning consistono nella modifica dei dati contenuti nei cookie inviati dall'applicazione all'utente, allo scopo di produrre errori e/o portare l'applicazione in stati non correttamente gestiti quando i cookie sono restituiti al server. Per essere in grado di apportare le modifiche opportune, l'attaccante deve conoscere la logica con cui i dati contenuti nei cookie sono processati dall'applicazione.
- **Probabilità di successo**: dipendenti dal livello di conoscenza da parte dell'attaccante della logica di elaborazione dei dati contenuti nei cookie.
- **Aspetti facilitanti**: la memorizzazione nei cookie di parametri critici dal punto di vista della sicurezza è un errore comune, che rende pericolosi gli attacchi di cookie poisoning.
- **Contromisure**: limitare l'uso dei cookie alla memorizzazione di informazioni non critiche; nel caso questo non sia possibile, devono essere utilizzate tecniche (ad esempio crittografia) per impedire la modifica dei cookie.

Forceful browsing

- **Obiettivo**: accesso non autorizzato a pagine e/o funzionalità dell'applicazione.

- **Attaccanti**: chiunque non sia in possesso di credenziali per accedere a tali pagine/funzionalità.
- **Descrizione**: gli attacchi di forceful browsing consistono semplicemente nella sottomissione di richieste HTTP per URL corrispondenti a pagine protette, senza seguire il percorso di navigazione previsto dal programmatore e, in particolare, aggirando le pagine di autenticazione.
- **Probabilità di successo**: dipendenti dal livello di conoscenza da parte dell'attaccante della struttura dell'applicativo. Tale livello può essere molto elevato per ex utenti che sono stati disabilitati.
- **Aspetti facilitanti**: messaggi di errore non propriamente gestiti dall'applicazione possono contenere informazioni sulla struttura delle directory dell'applicazione sul web server e semplificare la costruzione degli URL da utilizzare per compiere l'attacco.
- **Contromisure**: implementare una logica di controllo dello stato della sessione che impedisca l'accesso ad ogni parte dell'applicazione ad utenti non associati a sessioni autenticate.