THE CUSTOMER SUCCESS PLATFORM
SALES SERVICE MARKETING COMMUNITY ANALYTICS APPS

Secure Coding
SSL, SOAP and REST

Astha Singhal
Product Security Engineer
salesforce.com

dreamforce

Safe Harbor

Safe harbor statement under the Private Securities Litigation Reform Act of 1995:

This presentation may contain forward-looking statements that involve risks, uncertainties, and assumptions. If any such uncertainties materialize or if any of the assumptions proves incorrect, the results of salesforce.com, inc. could differ materially from the results expressed or implied by the forward-looking statements we make. All statements other than statements of historical fact could be deemed forward-looking, including any projections of product or service availability, subscriber growth, earnings, revenues, or other financial items and any statements regarding strategies or plans of management for future operations, statements of belief, any statements concerning new, planned, or upgraded services or technology developments and customer contracts or use of our services.

The risks and uncertainties referred to above include – but are not limited to – risks associated with developing and delivering new functionality for our service, new products and services, our new business model, our past operating losses, possible fluctuations in our operating results and rate of growth, interruptions or delays in our Web hosting, breach of our security measures, the outcome of any litigation, risks associated with completed and any possible mergers and acquisitions, the immature market in which we operate, our relatively limited operating history, our ability to expand, retain, and motivate our employees and manage our growth, new releases of our service and successful customer deployment, our limited history reselling non-salesforce.com products, and utilization and selling to larger enterprise customers. Further information on potential factors that could affect the financial results of salesforce.com, inc. is included in our annual report on Form 10-K for the most recent fiscal year and in our quarterly report on Form 10-Q for the most recent fiscal quarter. These documents and others containing important disclosures are available on the SEC Filings section of the Investor Information section of our Web site.

Any unreleased services or features referenced in this or other presentations, press releases or public statements are not currently available and may not be delivered on time or at all. Customers who purchase our services should make the purchase decisions based upon features that are currently available. Salesforce.com, inc. assumes no obligation and does not intend to update these forward-looking statements.

Astha Singhal

Product Security Engineer

Astha Singhal

- Working with product teams from design to implementation to help them build secure applications for our customers.
- Conduct penetration tests on salesforce applications.
- Facilitating the security process via better security training and enabling self-service for product teams.
- Helping them understand security bugs and guiding through remediation of security issues.

API first

- Defining the behaviors of an application in terms of its operations, their inputs and outputs and underlying types.
- Exposing these API interfaces to build custom integrations.
- Custom integrations can build wider functionality based on the API
- Makes your app easier to integrate with - Facilitates broader usage
- Don’t have to worry about the presentation layer
- Building more API centric applications

Building secure APIs

- No presentation layer -> no UI layer security issues.
- Specific issues relating to the API format
- Issues like secure storage, authentication, authorization still exist
- Business logic flaws

SOAP

- **S**imple **O**bject **A**ccess **P**rotocol is a protocol specification for exchanging structured information in the implementation of web services.
- Uses the XML message format and relies on another transfer protocol (like HTTP) for transmission

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPriceResponse>
    <m:Price>34.5</m:Price>
  </m:GetStockPriceResponse>
</soap:Body>

</soap:Envelope>
```

REST

- **RE**presentational **S**tate **T**ransfer is a means of expressing specific entities in a system by URL path elements.
- Allows interaction with a web-based system via simplified URL's
- No POST body required

```
GET http://www.example.org/inStock/getStockPrice/
IBM
```

Building secure APIs

- Make sure to do proper input validation
  - Make sure that the user input adheres to the expected format.
  - Make sure to build secure whitelists for limiting the inputs.
  - Don’t rely on blacklist filtering.
- Make sure to use secure parsers for parsing XML in SOAP requests
  - Don’t build your own parsers!
- Validate incoming content types
  - The server should make sure that the request content matches the specified content type header
  - Reject unexpected Content type headers.
- Set Content-type correctly based on the response type.
  - Don’t assume that the user supplied Accept header is a valid response content type.
  - Set `X-Content-Type-Options: nosniff`

Authentication

- Since there is no UI tied to it, the API endpoints make assumptions about how authentication information is provided.
- It is important to understand these assumptions and build secure implementations when building custom integrations.
- A few ways that can be used to provide authentication information are:
  - Custom headers
  - Session cookies
  - API keys
  - Username and Password

Authentication issues

- Don’t use BASIC Auth
  - Sending your long term secret in each and every request is not a good security practice.
  - Make sure to establish a session based short term authentication credential.
- Use session based authentication using custom headers
  - Instead of using the Cookie header, make sure to use custom Auth header to send session token values.
- Using API keys for Authentication
  - Make sure to not send your API keys as a part of the URL. URLs may be getting logged in places like proxy servers and third party tracking sites.
- OAuth tokens to verify identity via another provider
  - Make sure that the Oauth tokens are least privilege.

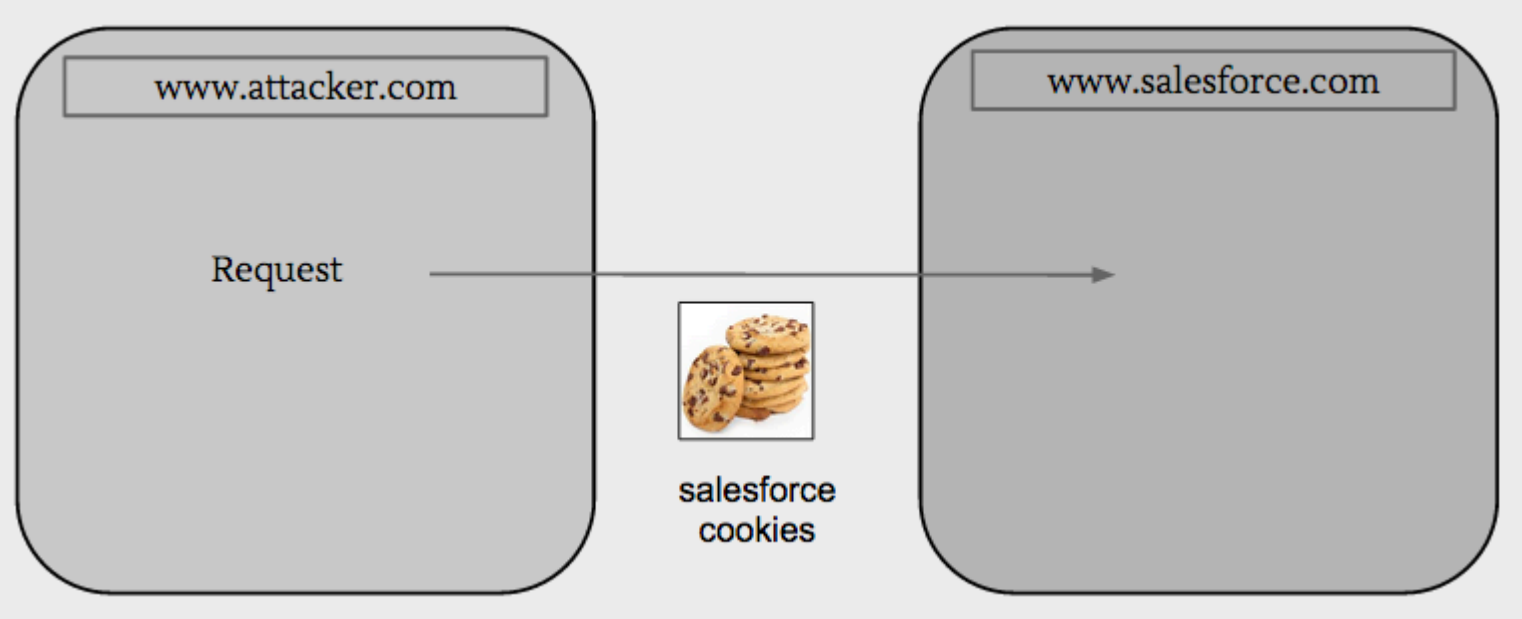
Authentication issues

- When calling APIs from the UI directly via AJAX callouts, make sure to generate an access token for the API endpoint, instead of using the session cookie directly.
- Generating a custom API token gives you better control in terms of being able to define the access scope of the token and maintaining least privilege.
- Using the session cookie directly would lead to disabling HTTPONLY on your session cookie.

Authorization issues - CSRF

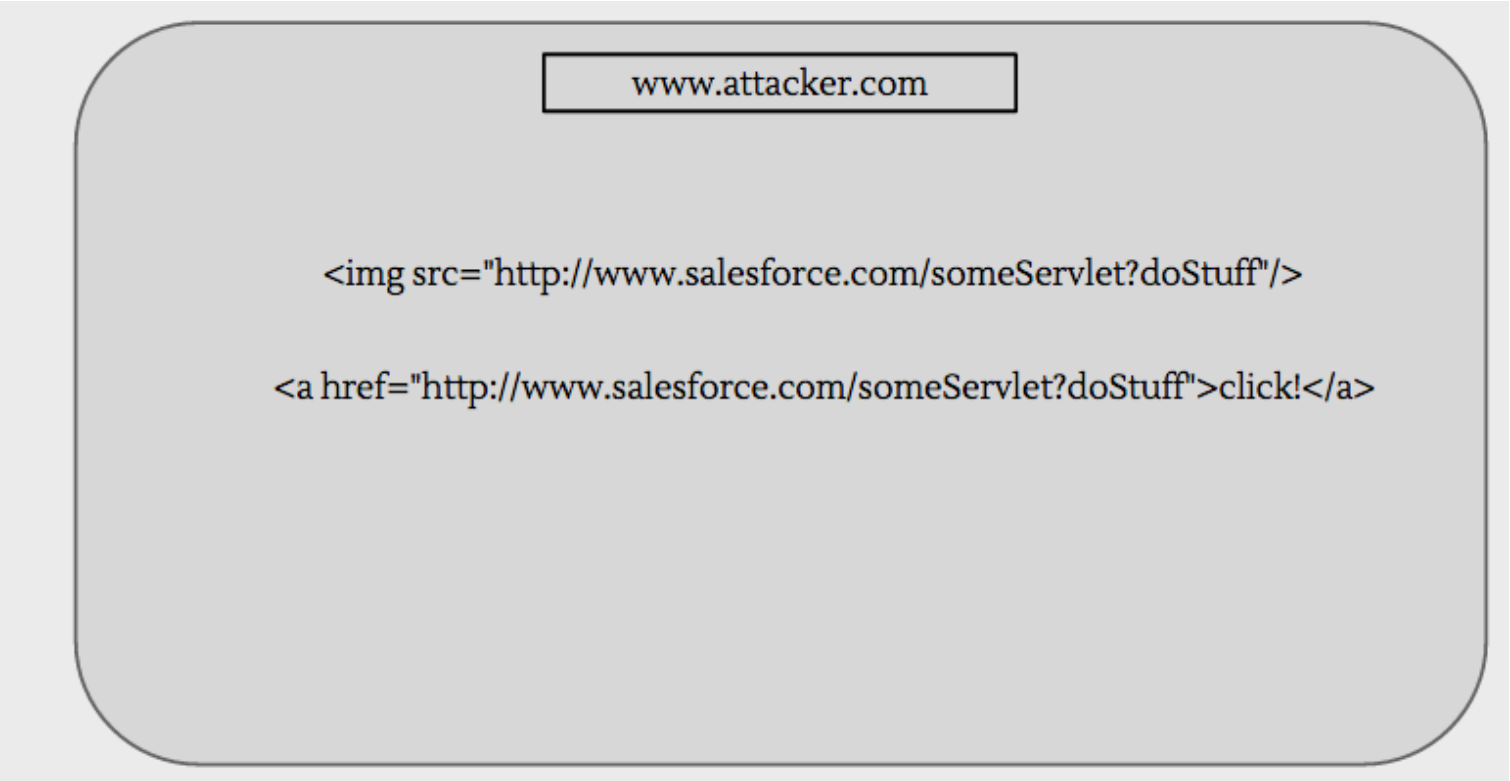
- Cookies are sent by the browser for all requests, including cross-domain requests.

- What this means is every time a request is made to salesforce.com from your browser, it would automatically include the cookies set for that domain.



Authorization issues - CSRF

- An attacker can thus force a user to make legitimate requests with attacker supplied parameters to any site via something like this:



Authorization issues - CSRF

- For API No page load, so no CSRF token load. Instead, using an 'Auth' header
  - CSRF Protection via custom headers since headers aren't auto-tracked & sent by browsers (unlike cookies)
  - Make sure to reject request when no Auth header present, don’t fallback to Cookie in that case. (Defeats the whole purpose).

Authorization using OAuth

- Tokens should be least privilege
  - OAuth tokens are based on a specific scope, API access tokens should be generated appropriately for desired scope.
  - Make sure to securely store OAuth refresh tokens on the external app
  - Follow industry best practices for secure storage on the platform you are using to build your application.
  - Don’t log access tokens on both client and server side.

Authorization – HTTP methods

- Make sure to whitelist allowable HTTP methods.
- Not all methods are valid for all endpoints.
- Make sure to define authorization correctly for the different HTTP verbs on the same endpoint.
  - For example: A user might be allowed to GET a resource, but not DELETE it.

Transport Security

Security Expectations of HTTP

- None
- Anyone on the network can eavesdrop traffic
- Anyone on the network can modify content
- Anyone on the network can divert traffic

Secure Sockets Layer

- A user visiting a site over HTTP has no assurance that the user is interacting with the legitimate site
- SSL allows a server to authenticate itself to a client and vice versa
- In addition to that, it also ensures that user communication cannot be intercepted by a malicious third party on the network.

Secure Sockets Layer

- Make sure that web servers have up-to-date SSL configurations.
- Supporting weaker ciphers and older protocol versions leads to compromising the security guarantees of SSL.
- A good tool to test server configurations:

https://www.ssllabs.com/ssltest/index.html

DEMO – ssllabs

Interacting with the Salesforce API

- All standard APIs enforce user CRUD/FLS restrictions.
- Custom web services written in Apex run in System context.
- Make sure to add CRUD, FLS and Sharing checks when building custom web services.
- Be sure to enable API access only for profiles that necessarily need it.

Interacting with the Salesforce API

- API access to an org allows a user to bypass presentation layer controls (Visualforce pages, standard page layouts) and comes with some unique situations to be aware of.
- For cases where sharing is enforced via Apex and VF would be unprotected when accessed via API.
- Some objects do not allow CRUD/FLS/Sharing to be configured (For eg: Custom settings in local namespace)
- API access allow easy mass extraction of data. So be careful while providing API access.

Summary

- Handle authentication credentials carefully.
- Make sure to not use Cookie header for Authorization to prevent CSRF.
- Always perform proper input validation on all user supplied input.
- Make sure to keep data secure in transit by using well configured SSL endpoints.

Thank You

Secure Development Sessions

Secure Coding: Field-level Security, CRUD, and Sharing
Monday, October 13 @ 11:00 a.m. - 11:40 a.m.

Secure Coding: Storing Secrets in Your Salesforce Instance
Monday, October 13 @ 2:00 p.m. - 2:40 p.m.

Building Secure Mobile Apps
Monday, October 13 @ 5:00 p.m. - 5:40 p.m.

Protect Your Data Against Malicious Scripts
Tuesday, October 14 @ 11:00 a.m. - 11:40 a.m.

Secure Coding: External App Integration
Wednesday, October 15 @ 9:00 a.m. - 9:40 a.m.

Secure Coding: SSL, SOAP, and REST
Thursday, October 16 @ 10:30 a.m. - 11:10 a.m.

dreamforce

Announcements:

Force.com Code Scanner now supports Salesforce1 and JavaScript! Try it here: http://bit.ly/SF1Scanner

Chimera Web App Scanner alpha nominations are open. Partners apply at: http://bit.ly/SFChimera

Live security office hours are available in the Partner Zone.

salesforce

Additional Resources

- Secure Coding Guidelines - https://developer.salesforce.com/page/Secure_Coding_Storing_Secrets
- Salesforce StackExchange - http://salesforce.stackexchange.com/questions/tagged/security
- Developer.Salesforce.com Security Forum - https://developer.salesforce.com/forums (full link hidden)
- Salesforce Security resources - https://developer.salesforce.com/page/Security
- Security Office Hours (Partners) - http://security.force.com/security/contact/ohours