



splunk>

Go From Dashboards to Applications With Ease

SplunkJS and Splunk Python for Non-Developers

David Veuve | Splunk Security Practice

Dave Herralde | Splunk Security Practice



Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2018 Splunk Inc. All rights reserved.

Agenda

- ▶ Introductions – Who Are We, Who Are You?
- ▶ The App
- ▶ The Meat
 - Running a Search from SplunkJS (Veuve)
 - Tokens for adequate interactivity (Herrald) (Or other content.. Maybe a simple python something?)
 - Modal Dialogs (Veuve)
 - ... (Herrald)
 - Our Top 5 Techniques for Making an App Great (Both)

What Is This Talk About?

- ▶ We have built a lot on top of Splunk
 - ▶ We don't know what we're doing
 - ▶ We've suffered
 - ▶ We don't want you to suffer
 - ▶ We put all our lessons learned in an app, and this talk accompanies the app

Introductions

Who Are We?

David Veuve

- ▶ He wrote the 5th most installed Splunk App: Splunk Security Essentials
- ▶ He also wrote the cult classic Splunk App: Search Activity
- ▶ He's presented 8 talks, 11 times to more than 2800 Splunkers @ .confs
- ▶ He just hit his 10 year anniversary of using Splunk

Dave Herrald

- ▶ More than 8,000 people have used his BOTS Scoring app
- ▶ He's built the most secure Splunk app on record
- ▶ He's presented at X SANS events and is super good at Security
- ▶ Splunk won't stop giving him awards for being awesome at Splunk

Why Are We Here? Because We Believe

- ▶ We believe that Splunk is an awesome development tool
 - ▶ We believe that Splunk dev documentation is for people smarter than us
 - ▶ We believe in maintaining state
 - ▶ We believe in a great application experience
 - ▶ We believe in being friends with people who understand UX design
 - ▶ But above all...
 - ▶ We believe in Copy-Paste Code Samples

Who Are You?

You're A Dashboard/Search Wizard

- ▶ You want to make better things
 - ▶ You've probably copy-pasted code
 - ▶ Maybe you've even written some code yourself!
 - ▶ You're scrappy and want to impress people

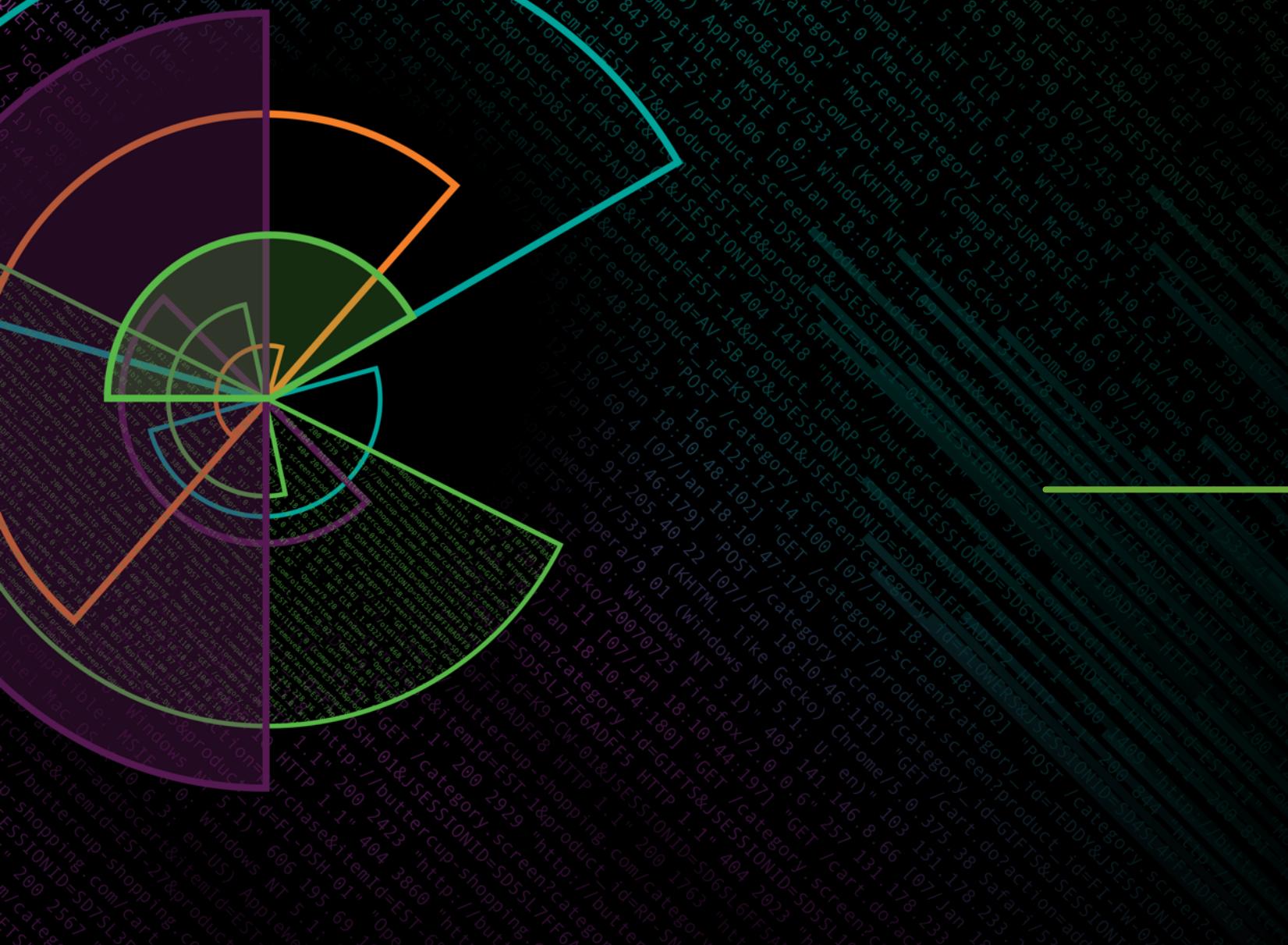
You're a Software Engineer

- ▶ You know how to code
 - ▶ You know how to read docs
 - ▶ You want to save yourself a *ton* of energy with a quick start

Goal

- ▶ You should understand the essentials of doing Splunk Development with SplunkJS and Python
 - ▶ You should understand how to leverage the app we are providing
 - ▶ You should feel comfortable that you'll be able to do great things

The App



Get The App

<https://splunkbase.splunk.com/app/4104/>

30+ Examples

Basics

If you're building out your first real Splunk app, walk through each of these examples, as they'll essential keys to success.

Including a JS file ★★★★★ (1 rating) Hello World, SplunkJS Style	Running a Search from Javascript ★★★★★ (4 ratings) The foundation of all SplunkJS, running a search and then outputting the results into an HTML element. Also includes examples generating a SimpleXML Viz from a Javascript search, and a Javascript Output from a SimpleXML search.	Setting and Reading Tokens ★★★★★ (1 rating) Either right after or right before you learn to run a search from Javascript, you'll want to be able to view and set tokens. Easy!	Find Missing Tokens ★★★★★ (1 rating) When you have advanced dashboards with all kinds of tokens, sometimes they're not set for whatever reason, and then everything mysteriously doesn't work. Figure out what tokens are missing, and what searches depend on them, with ease.	Reading JSON Files from appserver/static ★★★★★ (1 rating) This is built into some of these core concepts, but why not make it explicit!	Dynamically Updating Search String ★★★★★ (1 rating) Dynamically set the search string for any existing search managers.	Indexing Events from Javascript ★★★★★ (1 rating) It's possible you may with to ingest data via Javascript. Consider log events to show what happened, audit events to show what the user did, or even small JSON data sources you download from the internet!
---	---	---	--	--	--	--

Intermediate

These examples are too complicated, but also aren't always required. Familiarity with these tricks will help you be successful with certain endeavors.

Querying REST API from Javascript ★★★★★ (1 rating) How can you directly query elements of Splunk's REST API from Javascript (without launching a search with the <code>I rest</code> search command, which would be silly and we would never use that in published apps on Splunkbase.. 😊).	Using kvstore Collections ★★★★★ (1 rating) Reading the Splunk kvstore directly from SplunkJS, or adding entries.	Authenticated Custom Search Commands ★★★★★ (1 rating) A basic search command that will run Splunk searches on your behalf, or hit the Splunk REST API.	Authenticated Scripted Input ★★★★★ (2 ratings) A scripted input that runs as an authenticated user to accomplish periodic tasks	Creating Zip Files with third party Javascript Libraries ★★★★★ (1 rating) This isn't really SplunkJS, but it's fun! And it gives us an opportunity to show how we can use third party libraries in our Javascript.	Instantiating SplunkJS Service Object ★★★★★ (1 rating) You should never need to do this in reality, however some of Splunk's docs talk about using SplunkJS from other websites which requires a service object. This will instantiate one within a Splunk Dashboard.	Comparing Streaming SDK Methods ★★★★★ (1 rating) For streaming search commands, there are two primary methods for implementation. One uses the Python SDK, and one uses a new Chunked Encoding library. We walk through these.
Combining JSON Files from kvstore ★★★★★ (1 rating) If you have a static configuration file that you want to override with local settings, it may be easiest to load your static JSON and then pull custom entries from the kvstore. Here's an example of doing that.	Using localStorage ★★★★★ (1 rating) Do you need to really, really easily maintain state and are okay with it being limited to a single browser window? Then you'll "love" localStorage.	Automatically Running Javascript on Every Page ★★★★★ (1 rating) Sometimes you need to run a script on every page, every time. Or similarly for stylesheets. Fortunately, Splunk makes that easy with <code>dashboard.css</code> and <code>dashboard.js</code> .	Hiding Admin Functions in Help Menu ★★★★★ (1 rating) Don't let your drive for a simple user experience prevent you from building easy admin functionality. Just hide it in the help menu!	Stored Credentials ★★★★★ (1 rating) Do you need to store a username and password, but don't want to hardcode it unencrypted on the local file system? Stored credentials are here to solve that problem for you!	Tooltips and Popovers ★★★★★ (1 rating) Want to embed helpful descriptions? Fortunately Splunk makes it fairly easy to do so with Bootstrap's tooltip and popover.	

Advanced

Advanced examples shouldn't be tackled until you feel very confident with your development skills. Documentation is.. sparse.. and it will cause you a lot of hassles when you're just getting started. That said, they showcase some great things that Splunk is capable of.

Posting to HEC via Javascript	Creating Modal Dialogs	Editing .conf Files from Javascript	Javascript Dashboards	Dynamically Adding Panels	Javascript App Status
---	--	---	---------------------------------------	---	---------------------------------------

- Guidance
- Basic
- Intermediate
- Advanced

In Each Example...

The screenshot shows a card from the Splunk Examples library. At the top, it says "Creating Modal Dialogs" with a 5-star rating and 1 rating. Below that is a brief description: "Modal Dialogs allow you to warn users about problems, get input from users, and more. Javascript natively has ugly alert capabilities.. why not replace them with pretty modal alerts." There are three main sections: "Description", "Where We've Used This", and "Files Involved". Under "Description", it says: "Click any of the boxes below and you'll get the gist. This code uses a file called Modal.js that was (so far as we know) originally written by the Splunk MLTK team for their Showcase UI, then customized by [David Veuve](#), then perfected by [Scott Haskell](#) and now available for you. Modal.js, as its comments indicate though, is just a wrapper around Bootstrap.js's Modal implementation. But we like it!" Under "Where We've Used This", it mentions Splunk Security Essentials and REST storage/passwords Manager for Splunk. Under "Files Involved", there is a list of files: default/data/ui/views/ex05-creating-modal-dialog.xml, appserver/static/ex05-creating-modal-dialog/ex05-creating-modal-dialog.js, and appserver/static/ex05-creating-modal-dialog/Modal.js. Below this, there are four examples: "Modal One - Simple" (Launch Modal), "Modal Two - With an Action" (Launch Modal), "Modal Three - With two Actions" (Launch Modal), and "Modal Four - With a Splunk Object" (Launch Modal). At the bottom, there are tabs for "Source Code" (selected) showing ex05-creating-modal-dialog.xml and ex05-creating-modal-dialog/ex05-creating-modal-dialog.js, and a code snippet:

```

1. <dashboard script="ex05-creating-modal-dialog/ex05-creating-modal-dialog.js">
2.   <label>Creating Modal Dialogs</label>
3.   <description>Modal Dialogs allow you to warn users about problems, get input from users, and more. Javascript natively has ugly alert capabilities.. why not replace them with pretty modal alerts.
4.   </description>

```

- ▶ The JS / CSS
- ▶ Description
- ▶ Where we've used it
- ▶ Doc links
- ▶ Rating
- ▶ Working examples you can interact with
- ▶ (A few) detailed "how would you learn this" guides

What's In The App?

As of this slide being built... conf is still weeks away!

Guidance

- ▶ Setting Up Your Development Environment
 - ▶ Powerful Third Party and jQuery Plug-ins
 - ▶ Splunk Style Guidelines (and Icons and such)
 - ▶ Logging and Debugging

Basic

- ▶ Running JS
 - ▶ Running a Search from SplunkJS
 - ▶ Setting and Reading Tokens
 - ▶ Find Missing Tokens
 - ▶ Reading JSON Files from appserver/static
 - ▶ Dynamically Updating Search String
 - ▶ Indexing Events from Javascript
 - ▶ Automatically running Javascript on Every Page

Intermediate

- ▶ Querying REST API from Javascript
 - ▶ Using kvstore Collections
 - ▶ Authenticated Custom Search Commands
 - ▶ Authenticated Scripted Input
 - ▶ Creating Zip Files with Third Party Javascript Libraries
 - ▶ Instantiating SplunkJS Service Objects
 - ▶ Comparing Streaming SDK Methods
 - ▶ Combining JSON files with kvstore
 - ▶ Using localStorage
 - ▶ Hiding Admin Functions in Help Menu
 - ▶ Stored Credentials
 - ▶ Tooltips and Popovers

Advanced

- ▶ Posting to HEC via Javascript
 - ▶ Creating Modal Dialogs
 - ▶ Editing .conf Files from Javascript
 - ▶ Javascript Diag
 - ▶ Dynamically Adding Panels
 - ▶ Javascript App Setup

Running A Search from SplunkJS

Code Example

First The Basics - SplunkJS

Require.js

- ▶ Require.js allows you to import modules – the equivalent of:
 - Perl: use HTTP::Simple;
 - Python: import json
 - Etc
 - ▶ You take the code that you would normally write, and you just stick it in a require function:

```
require(["jquery", "splunkjs/ready!"], // run once view loaded
function($, Ready) {
    $("body").append("<p>Hello World!</p>");
}
)
```

Search Manager

- ▶ Search Managers let you .. Manage searches. Run, edit, etc.

```
require(["splunkjs/mvc/searchmanager"], function(SearchManager) {
```

// New Search

```
var sm = new SearchManager({ [...] },
```

```
{ tokens: true, tokenNamespace: "submitted" } );
```

// Load Search

```
var other_sm = splunkjs.mvc.Components.getInstance("search2")
```

// Delete Search

```
splunkjs.mvc.Components.revokeInstance("search2")
```

```
// Test if Search Exists
```

```
typeof splunkjs.mvc.Components.getInstance("search2") == "object"
```

۷۰

There Are Many Useful Classes

Here are a list of classes, and how the typical name they're loaded with

- ▶ "jquery" -> \$
- ▶ "underscore" -> _
- ▶ "splunkjs/mvc/dropdownview" -> DropdownView
- ▶ "splunkjs/mvc/textinputview" -> TextInputView
- ▶ "splunkjs/mvc/chartview" -> ChartView
- ▶ "splunkjs/mvc/tableview" -> TableView
- ▶ "bootstrap.tooltip" -> N/A (put it at the end, no object required)

- ▶ How to discover these? Easiest – put the element on an SimpleXML Dashboard and convert to HTML

Discovering Objects

- ▶ Every instantiated object is stored in:
`splunkjs.mvc.Components.attributes`
 - ▶ Find this in the Javascript Console:

Elements Console Sources Network Performance Memory Application Security Audits

top Filter Default levels Group similar

```
> splunkjs.mvc.Components.attributes
< ▾ {default: child, submitted: child, env: child, url: child, header: child, ...} ⓘ
  ▶ content1: child {_removed: false, id: "content1", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ content2: child {_removed: false, id: "content2", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ dashboard1: child {_removed: false, id: "dashboard1", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ default: child {cid: "c2", attributes: {...}, _changing: false, _previousAttributes: {...}, changed: {...}, ...}
  ▶ element1: constructor {_removed: false, id: "element1", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element1-header-11477progressbar-11536: child {_removed: false, id: "element1-header-11477progressbar-11536", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element2: constructor {_removed: false, id: "element2", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element2-header-11874progressbar-11933: child {_removed: false, id: "element2-header-11874progressbar-11933", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element3: constructor {_removed: false, id: "element3", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element3-header-12289progressbar-12348: child {_removed: false, id: "element3-header-12289progressbar-12348", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element4: constructor {_removed: false, id: "element4", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element4-header-12704progressbar-12763: child {_removed: false, id: "element4-header-12704progressbar-12763", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element5: constructor {_removed: false, id: "elements5", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element5-header-13119progressbar-13178: child {_removed: false, id: "element5-header-13119progressbar-13178", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element6: constructor {_removed: false, id: "element6", autoId: false, options: {...}, viewOptions: {...}, ...}
  ▶ element6-header-13544progressbar-13603: child {_removed: false, id: "element6-header-13544progressbar-13603", autoId: false, options: {...}, viewOptions: {...}, ...}
```

Loading Custom JavaScript

- ▶ Add a script tag to your dashboard tag:

```
<dashboard script="myscript.js">
```

[...]

</dashboard>

- ▶ `myscript.js` will automatically load from:

`$SPLUNK_HOME/etc/apps/myApp/appserver/static/myscript.js`

- Any other resources you need can also be stored in that folder, or in subfolders.

- You can verify the file is there by going to:

`http://mySplunk:8000/static/app/myApp/myscript.js`

- .. Which adds language automatically:

<http://mySplunk:8000/en-US/static/app/myApp/myscript.js>

Okay, enough with the basics
Let's run a search!

Demo Plan

- ▶ If you're reading this slide, you're looking at the PDF copy. That means you're missing out!
- ▶ What are you missing out on? Well, a 4 minute demo. And in that demo I will do a:
 - Quick SA-jsforall App Demo
 - Create a new Simple XML Dashboard with an HTML element and an empty div
 - Add a Script Tag
 - Create a new Javascript file
 - Copy-paste the code example from ex01 that will output the first event
 - Save, debug refresh and _bump
 - (Discuss how you don't have to do that)
 - (Discuss how splunkd sometimes doesn't _bump on app updates because splunkd can be obnoxious)

Herrald Example 1 (Tokens? Python Basics?)

Code Example

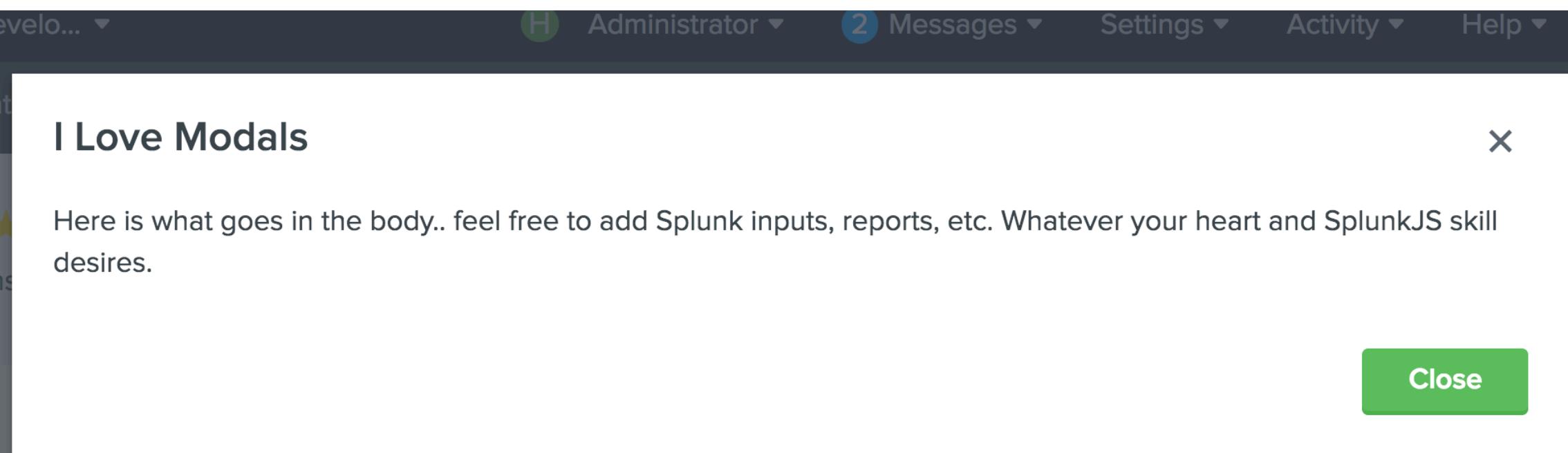
Creating Modal Dialogs

Code Example

First The Theory

What is a Modal?

- ▶ A modal is fully known as a Modal Dialog. It's a box that pops up on the screen and confirms, asks for details, etc.
- ▶ In other words, it's one of these:



at the gist. This code uses a file called Modal.js that was (so far as we know) originally written by the Splunk MLTK team for their Showcase UI and now available for you. Modal is, as its comments indicate though, is just a wrapper around Bootstrap's Modal implementation. Scott Haskell and now available for you. Modal is, as its comments indicate though, is just a wrapper around Bootstrap's Modal implementation.

Modals Come From Bootstrap

- ▶ Bootstrap.js ships with Splunk, and it does all the really heavy lifting here
 - ▶ We *do* need to create a new object for the modal, though.
 - ▶ We'll create a new file (well, copy-paste from the app) and load via require

Modals have actions

- ▶ You can do things on modal hide (start of hide action), on modal hidden (end of hide action), etc.
 - ▶ You can also add buttons to modals.

Let's Do It!

Demo Plan

- ▶ If you're looking at this slide, you're viewing the PDF copy of the slides. That means you're missing out on the demo again!
 - ▶ In this demo, I will start with a SimpleXML dashboard to add a new item into a threat intel dashboard – a text form input, and then a <button> that calls "addElement()"
 - Have a comment which includes the search string that should be run
 - Have a panel that includes the current # of elements and the last element added
 - ▶ Copy-paste example from the app to create a confirmation modal
 - ▶ Copy-paste example from the app to run a search if the user clicks Yes

Herrald Example

Code Example



Top 5 Elements for a Great App



#5 – Use Modals

- ▶ Modals are really easy to use
- ▶ They provide a more polished user experience and more interactivity

- ▶ You can even include Splunk dashboard elements in them (dropdowns, charts, etc.)
 - Trick: you may need to add a setTimeout or add them after the modal finishes rendering

#4 - Herrald

#3 – Maintain State

- ▶ An application remembers what you've done before
- ▶ An application does not repeatedly prompt you for the same information
- ▶ There are several great methods for maintaining state:
 - Locally in the browser via localStorage
 - By accessing the kvstore
 - Even storing details in a macro or lookup

#2 - Herrald

#1 – Get Design Help

- ▶ If you aren't good at design, ask someone who is.
- ▶ If you don't know someone, buy some time on fiverr.com, upwork.com, etc.
 - \$300 spent on outsourced design can net you a promotion

Splunk > App: Splunk Security Essentials

Use Cases

Welcome to the Use Case Overview in Splunk Security Essentials. This app provides generic search builders for doing time series analysis and first time analysis, which you can apply to any data you have in Splunk, for any use case you might desire. To help illustrate how this works, and also provide you with easy out of the box analytics you can use today, the app also includes many pre-built reports based on Common Information Model data, or anonymized demo data from Splunk Inc. or volunteer customers. There are also several normal Splunk searches that customers have used for Anomaly Detection, and that you will find in many UBA products in the marketplace.

To get started, click on the name of any use case to view the demo data. Once you understand what the analytic is doing, you can try looking at the live data or accelerated data views. The app will try to help guide you toward making sure you have the right data in Splunk to do the analysis, but remember you can always click Open in Search to explore it on your own. Once you've got the search running how you want, you can schedule that alert to run regularly, and feed the results into Splunk User Behavior Analytics (UBA), Splunk ES's Risk Framework, or any other upstream ticketing system.

Each use case also includes the expected alert volume – for "low" you can expect the alert to fire rarely, probably only every few weeks if that, whereas high volume alerts are likely to fire multiple times per day and should be sent into some upstream processing such as Splunk ES Risk, or Splunk UBA.

To make the examples easy to follow, they are organized into Security Domain, and several are showcased as highlights at the top. Select a Security Domain you're interested in (or just select All Examples) below.

Use Case Filters (Hide)

Security Domains	Data Sources	Alert Volume	Last Updated
<input checked="" type="radio"/> All <input type="radio"/> All Host Logs (2/2) <input type="radio"/> Data (1/12) <input type="radio"/> Endpoint (20/20) <input type="radio"/> Network (9/9) <input type="radio"/> Threat (3/3)	<input type="radio"/> Electronic Medical Record System (1/1) <input type="radio"/> Firewall (3/3) <input type="radio"/> Other (4/4) <input type="radio"/> Netflow (3/3) <input type="radio"/> Print Server Logs (1/1) <input type="radio"/> Salesforce Event Log File (6/6) <input type="radio"/> Splunk Notable Events (1/1) <input type="radio"/> Source Code Repository Logs (3/3) <input type="radio"/> Windows Security Logs (24/24) <input type="radio"/> Windows System Logs (1/15)	<input checked="" type="radio"/> All <input type="radio"/> Very Low (4/4) <input type="radio"/> Low (20/20) <input type="radio"/> Medium (17/17) <input type="radio"/> High (3/3) <input type="radio"/> Very High (1/1)	<input checked="" type="radio"/> All <input type="radio"/> 0.0 (42/42) <input type="radio"/> 1.0 (5/5) <input type="radio"/> 1.2.0 (5/5) <input type="radio"/> 1.4.0 (6/6)

Highlights

Authentication Against a New Domain Controller

A common indicator for lateral movement is when a user starts logging into new domain controllers.

Alert Volume: Medium

Examples:

- Demo Data
- Live Data

Concentration of Hacker Tools by Filename

It's uncommon to see filenames associated with attacker tools in rapid succession on an endpoint. The first time, it's probably fine. The fourth or fifth file used should be suspicious. ([MITRE CAR Reference](#))

Alert Volume: Low

Examples:

- Demo Data
- Live Data

SSE 1.0

splunk > enterprise App: Splunk Security... ▾

Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find

Introduction Security Content ▾ Security Data Journey Data Source Check Documentation ▾ Advanced ▾ Splunk Security Essentials

Export ▾ ...

Security Content

▶ **i** How can you map this content to Splunk's Security Journey, and make your environment more secure?

Filter Examples

Journey Stage 1 - Collection (20 matches) ▾ **Security Use Case** All ▾ **Data Sources** All ▾ **Recommended** Yes (20 matches) ▾

Stage 1: Collection You have the data onboard, what do you do first?

> Access to In-Scope Resources

Visibility into who is accessing in-scope resources is key to your GDPR efforts. Splunk allows easy analysis of that information.

Recommended

Searches Included

Web Proxy

> Access to In-Scope Unencrypted Resources

Unencrypted communications leaves you vulnerable to a data breach -- when users access PII data, ensure that all connections are encrypted.

Recommended

Searches Included

Web Proxy

> Authentication Against a New Domain Controller

A common indicator for lateral movement is when a user starts logging into new domain controllers.

Recommended

Searches Included

Windows Security

> Basic Brute Force Detection

Uses a simple threshold for Windows Security Logs to alert if there are a large number of failed logins, and at least one successful login from the same source.

Recommended

Searches Included

Windows Security

SSE 2.0

splunk > .conf18

Key Takeaways

Download the app!

<https://splunkbase.com/app/4104>

Making more interactive
and powerful UIs is
easy

If you find content like this useful, please rate us in the app so that Splunk provides more content like this.

Thank You

Don't forget to rate this session
in the .conf18 mobile app

