



The Curious Case of API Security and Solving the Top 10 API Threats

Gunnar Peterson
Software Security Architect
and CTO, Arctec Group



Our Speakers



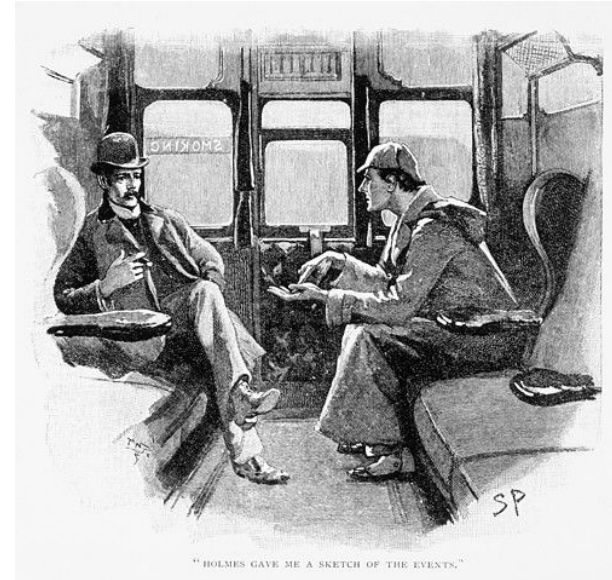
Gunnar Peterson
Managing Principal
Arctec Group
@oneraindrop
<http://1raindrop.typepad.com>



Mark O'Neill
VP of Innovation
Axway
@TheMarkOneill
<http://soatothecloud.com>

The Curious Case of API Security

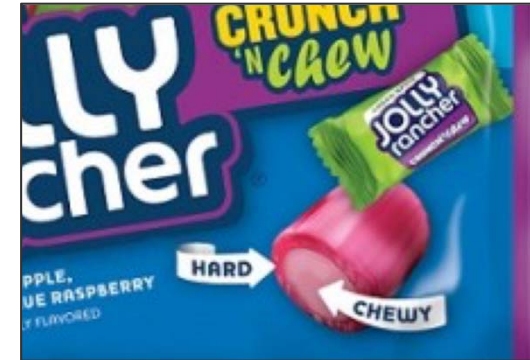
- Our basic process is as follows:
 - Understand the context in which our APIs exist
 - Look for clues of possible vulnerabilities
 - Catalog what tools we use to identify and track vulnerabilities
 - Identify countermeasures that we can use to close out vulnerabilities
 - Provide evidence that can measure the efficacy of the countermeasures



1. The Curious Case of Unprotected APIS

- **Context**

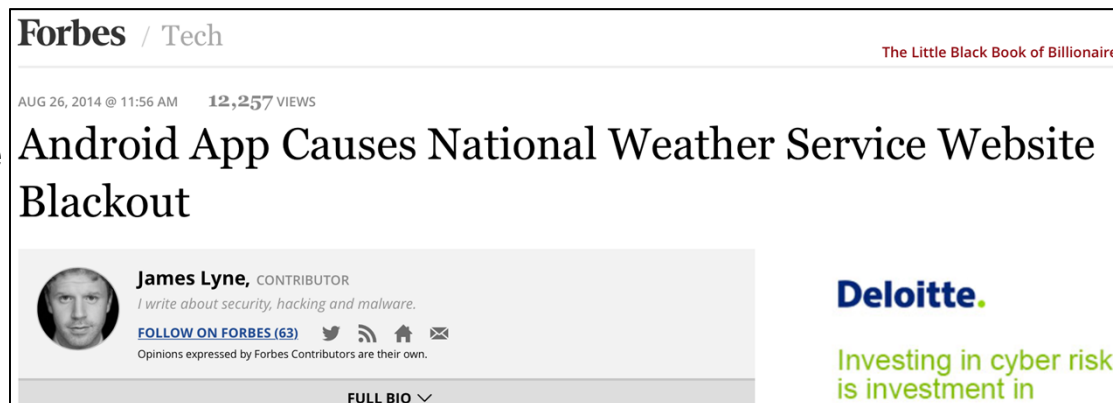
Most enterprise cores are as soft and chewy as the center of a candy bar. That means that once inside an attacker has free reign. Therefore, the API layer is a table pounding, must have security priority.



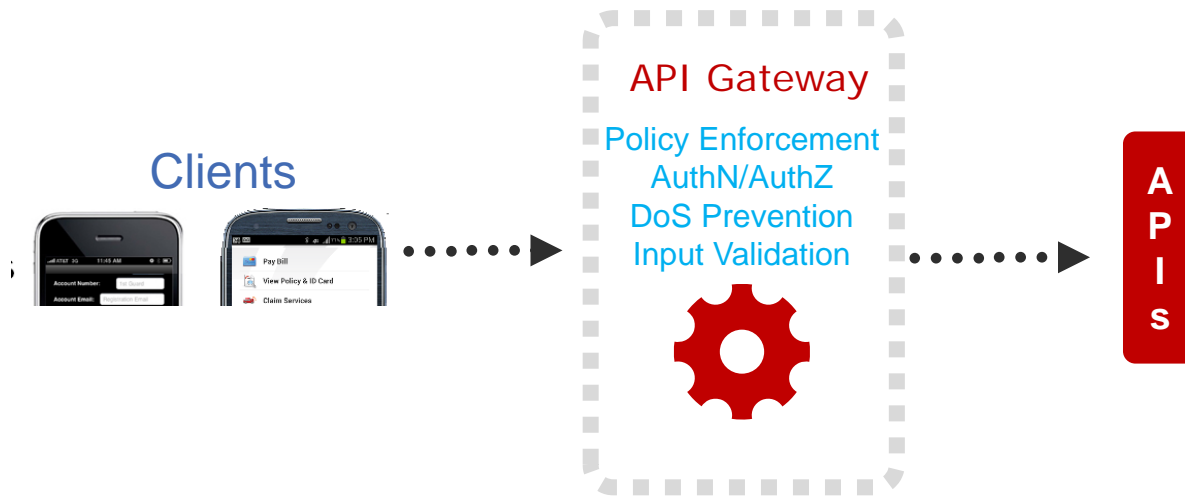
- **Vulnerabilities**

REST, SOAP, and other APIs that make access available to backend systems lack access control, monitoring, and management.

Example



1. The Curious Case of Unprotected APIs



Countermeasures: Enforce access policy to all APIs through a central chokepoint such as an API Gateway. Implement an API Gateway to:

- mediate and monitor all access requests to the API layer
- enforce API access control policy
- ensure the system does not expose unprotected assets via APIs

2. The Curious Case of Weak Authentication

- **Context**

APIs are designed to be exposed externally, so they cannot trust who is calling them. APIs have to authenticate users to be able to tell friend from foe.

- **Vulnerabilities**

- Unauthenticated access (open APIs)
- Poorly protected secrets and tokens
- Use of password based authentication
- Hard coded secrets
- Lack of replay protection
- Guessable secrets and tokens

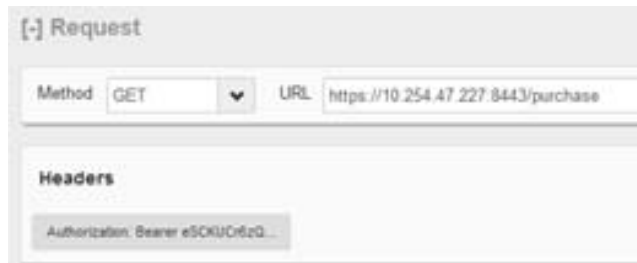


*"There is nothing more
Deceptive than an obvious fact"*
-Sherlock Holmes
Boscombe Valley Mystery

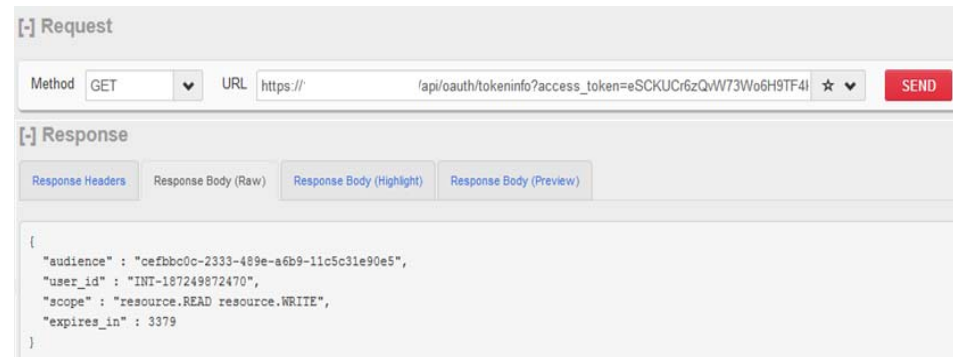
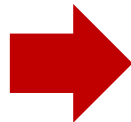
How I Hacked Facebook OAuth To Get Full Permission On Any Facebook Account (Without App "Allow" Interaction)

Posted by Nir Goldshlager at [7:18 AM](#)

2. The Curious Case of Weak Authentication



Token is sent in API call



Token is used to check authorized scopes, expiry, etc.

Countermeasures:

API authentication is best analyzed in two parts. One is responsible to perform logon authentication and mint the API token.

Next is a subtly different case whereby the API layer validates the calling application's token. This is subtly different from initial login because it's the token that is authenticated.

APIs in search of stronger API Authentication approaches should strongly consider SAML and OAuth over TLS as a way to issue and verify API authentication for API consumer applications.

3. The Curious Case of Brute Force

- **Context**

Authentication systems are built on at least one secret that the user knows and the attacker does not. Attackers leverage misplaced trust that secrets cannot be reverse engineered.

- **Vulnerabilities**

Attackers use Brute force replay and retry attacks to impersonate or gain access to a legitimate user's authenticated session.

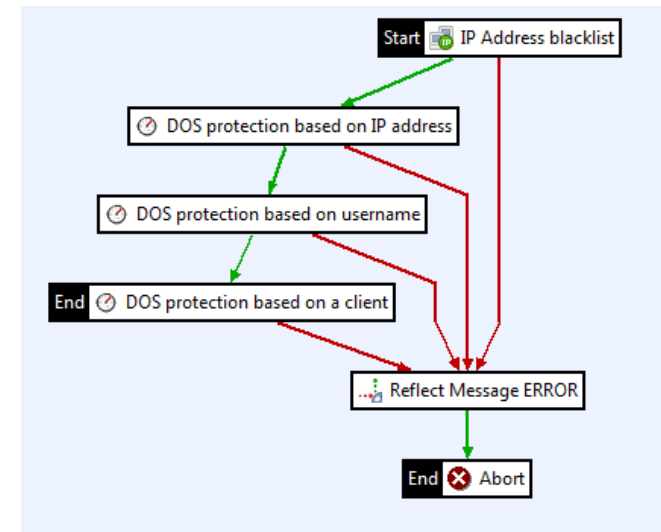
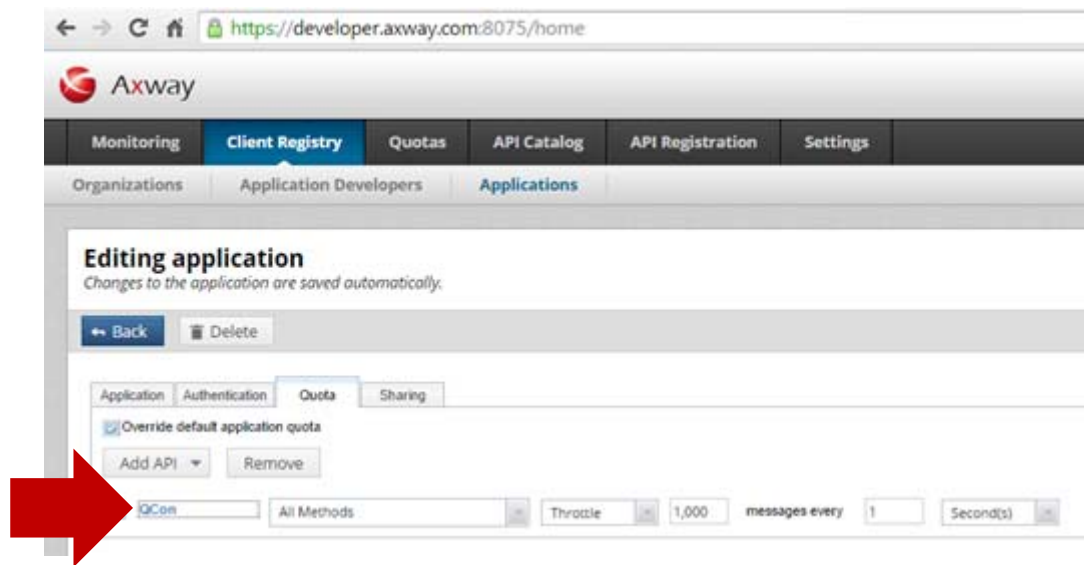


"The world is full of obvious things which nobody by any chance ever observes"

*-Sherlock Holmes
Hound of the Baskervilles*



3. The Curious Case of Brute Force



Countermeasures: Rate limiting services can be delivered via an API Gateway to throttle access requests, detect potential malicious patterns like brute force. This should be done both at a network level and at an application and user level.

4. The Curious Case of Injection

- **Context**

Since APIs are the Gateway to the enterprise core, they often miss attacks that slip right through their grasp. That is because the target of the injection attack is often the backend database (as in SQL Injection) or a Directory service (as in LDAP Injection) or ERP system or even HVAC. The attackers can leverage those resources directly and/or use them as launch pads to further attacks.

- **Vulnerabilities**

The results here are many, but the two most well known are SQL Injection and Cross Site Scripting.

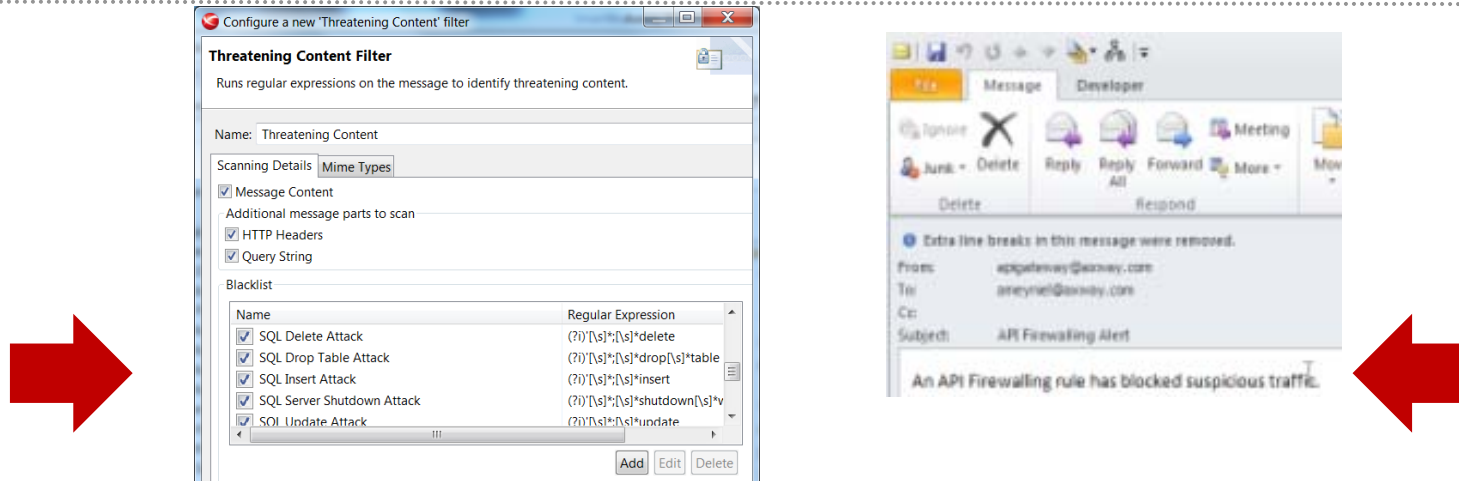


*"I never guess. It is a shocking habit,
—destructive to the logical faculty"*
-Sherlock Holmes
The Sign of Four

"You should proceed under the assumption that every Drupal 7 website was compromised unless updated or patched before Oct 15th, 11pm UTC, **that is 7 hours after the announcement.**" –

Drupal SA <https://www.drupal.org/PSA-2014-003>

4. The Curious Case of Injection



Countermeasures: Safe Input Output Handling is the key countermeasure. For input handling this is generally a mix of data sanitization and escaping that removes or overwrite control characters; and input validation that examines a known good and/or known bad list of words, characters and data types and blocks access based on regular expression failures.

For output handling, the system must encode the data in a way that ensures that user controllable data is not propagated through the system in an executable format. This means encoding output data as JSON or HTML or other depending on how the client is set up to consume it.

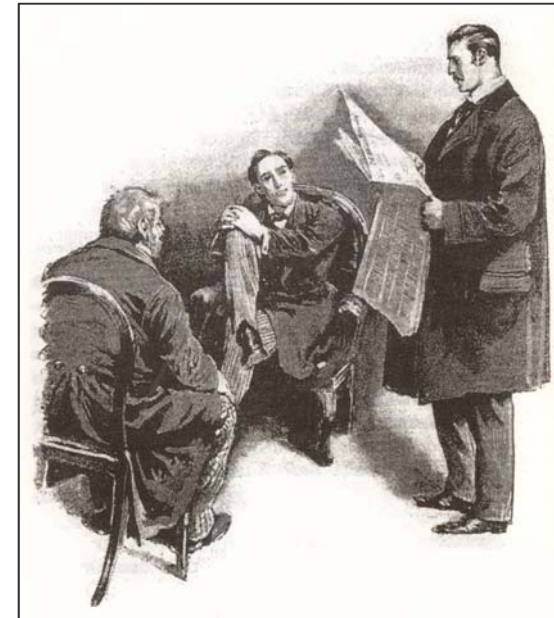
5. The Curious Case of Lateral Movement

- **Context**

Attackers are adept at manipulating security tokens and adapting them to find ways to move laterally across a system.

- **Vulnerabilities**

Direct Object Reference, Cross Site Request Forgery and Open Redirects are three examples of vulnerabilities where lateral movement is used as a vector to by pass authorization.



"As a rule, the more bizarre a thing is the less mysterious it proves to be."
-Sherlock Holmes
The Red-Headed League

5. The Curious Case of Lateral Movement

OAuth Scopes

Add scope ▼ Remove

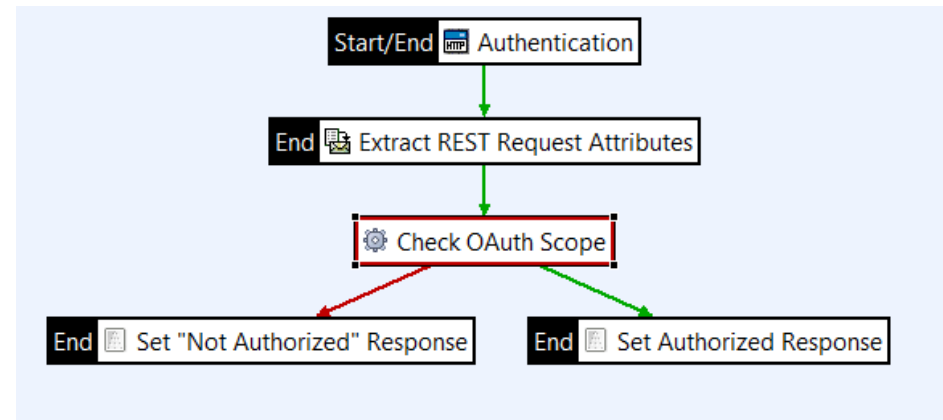
SCOPE	DEFAULT
<input type="checkbox"/> resource.READ	<input checked="" type="checkbox"/> ON
<input type="checkbox"/> resource.WRITE	<input checked="" type="checkbox"/> ON

Managing authorizations

Use the paging arrows to navigate your list of authorizations.

Filter: authorization Manage selected ▼ Refresh

	Applications	Owner	Scopes
<input type="checkbox"/>	Client App	sampleuser	openid,resource.WRITE



Countermeasures: API Gateways should enforce a strict token scoping and validation policy that limits what's access to each API caller. The API server should validate the inbound request against the token scope. The Gateway should dispatch any request to a list of approved services.

6. The Curious Case of Session Promiscuousness

- **Context**

Session tokens like cookies, one time use URLs, SAML tokens, and OAuth tokens are the main (and often only) method for the API server to try to know who is calling it.

- **Vulnerabilities**

If these tokens become corrupted, are replayed or spoofed its difficult to impossible for API servers to distinguish valid access from malice.



"I had," he said, "come to an entirely erroneous conclusion, my dear Watson, how dangerous it always is to reason from insufficient data"

*-Sherlock Holmes
The Adventure of the
Speckled Band*

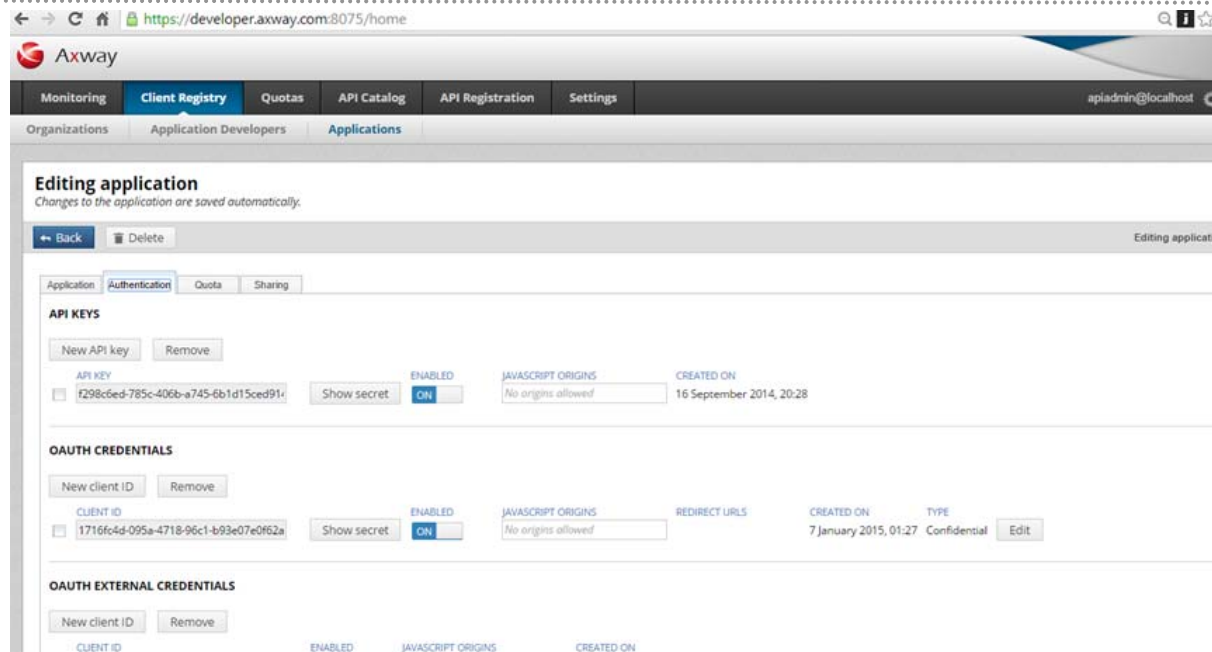
Google closes hole in Single Sign-On service

Google has fixed an implementation flaw in the single sign-on service that powers Google Apps following a warning from researchers that remote attackers can exploit a hole to access Google accounts. The vulnerability, described in this white paper (



By Ryan Naraine for Zero Day | September 10, 2008 -- 09:23 GMT (02:23 PDT) | Topic: [Google](#)

6. The Curious Case of Session Promiscuousness



Countermeasures: Ensure token protection schemes are in place that sign and hash tokens when they are issued. The API Gateway must authenticate the signature and verify the hash to ensure the request is from an authorized source, has not been tampered with.

To ensure the tokens are fresh, a one time use code (nonce) and/or timestamp should be issued and verified.

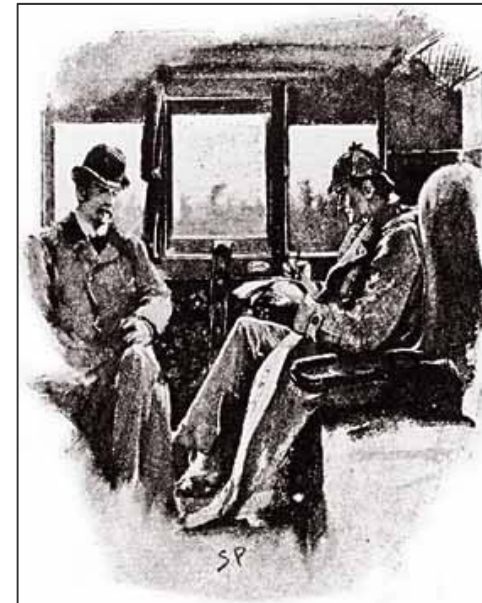
7. The Curious Case of the Invisible Attacker

- **Context**

Access control divides the system into known good and known bad states. These partitions are useful for defining and enforcing authorized access, but they do not hold up in all cases when deliberate malice is involved.

- **Vulnerabilities**

Attackers inject false messages into log files, find events that are not tracked, and/or tamper with log messages.



"You know my method. It is founded upon the observation of trifles."

*-Sherlock Holmes
Boscombe Valley Mystery*



7. The Curious Case of the Invisible Attacker

Axway API Gateway

Dashboard Monitoring **Traffic** Logs Events Messaging Settings admin

HTTP (1) Websocket JMS File Transfer Directory Performance

Back Transaction: Id-cb472c5604008e406549e46f Collapse All Download

FILTER EXECUTION PATH

Filter	Status	Transaction Audit Message	Execution Time (ms)	Time
Security				
Detect Threatening Content	⚠	Message contains known attacks	15	Oct 25, 2015, 03:08:59.503
Call 'Return HTTP Error 403: Access Denied (Forbidden) {1445742384922}'	✓		0	Oct 25, 2015, 03:08:59.519
Return HTTP Error 403: Access Denied (Forbidden)				
Make the "Forbidden" Message	✓		0	Oct 25, 2015, 03:08:59.519

REQUEST FROM CLIENT 127.0.0.1 (127.0.0.1) AND RESPONSE FROM API GATEWAY

TRACE

POST /api/service HTTP/1.1

Countermeasures: Network only logging won't cut it. Logging and monitoring must be done at an application level. Application sensors should be deployed at boundary crossing layers like the API Gateway. These sensors should record access, exception, malicious and related events.

8. The Curious Case of Broken TLS/SSL

- **Context**

There is no other security protocol as widely used as SSL and TLS. But this does not mean that its always deployed correctly.

- **Vulnerabilities**

BEAST and Poodle are two recent examples of SSL/TLS weakness. In addition certificate naming, chain validation, and protocol issues can open up Man in the Middle, information disclosure, and broken authentication vulnerabilities.

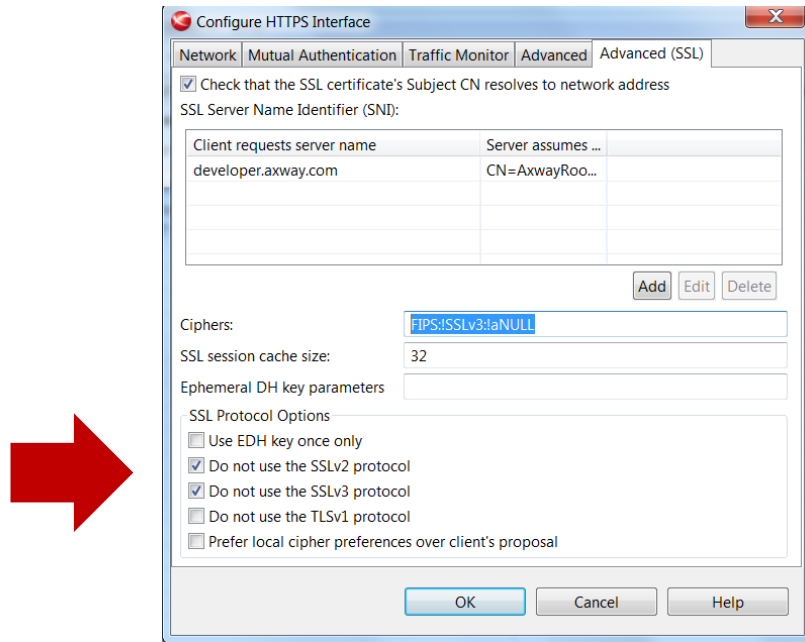


*"Crime is common. Logic is rare.
Therefore, it is upon logic rather than
upon the crime that you should dwell."
-Sherlock Holmes
The Adventure of the
Copper Beeches*

"When Eve and Mallory Fell in Love"

<https://www.dfn-cert.de/dokumente/workshop/2013/FolienSmith.pdf>

8. The Curious Case of Broken TLS/SSL



Countermeasures: SSL should be replaced with TLS. TLS should be upgraded to the highest level your organization can support (TLS 1.2). The provisioning, design, implementation and deployment should be carefully reviewed and tested. The API Gateway can play a role as the central choke point for terminating and validating TLS traffic.

9. The Curious Case of Inversion of Control

- **Context**

APIs are a glue layer. In the old days, APIs were strictly request-response client-server protocols. Not any more. With mobile, HTML5 and other technologies we are seeing many applications where the server pushes data to the client. Since most security protocols are set to trust servers and distrust clients, this turns the security protocols upside down as well.

- **Vulnerabilities**

Clients lack the protection and isolation of a DMZ.



“When you follow two separate chains of thought, Watson, you will find some point of intersection which should approximate to the truth”

-Sherlock Holmes

The Disappearance of Lady Carfax

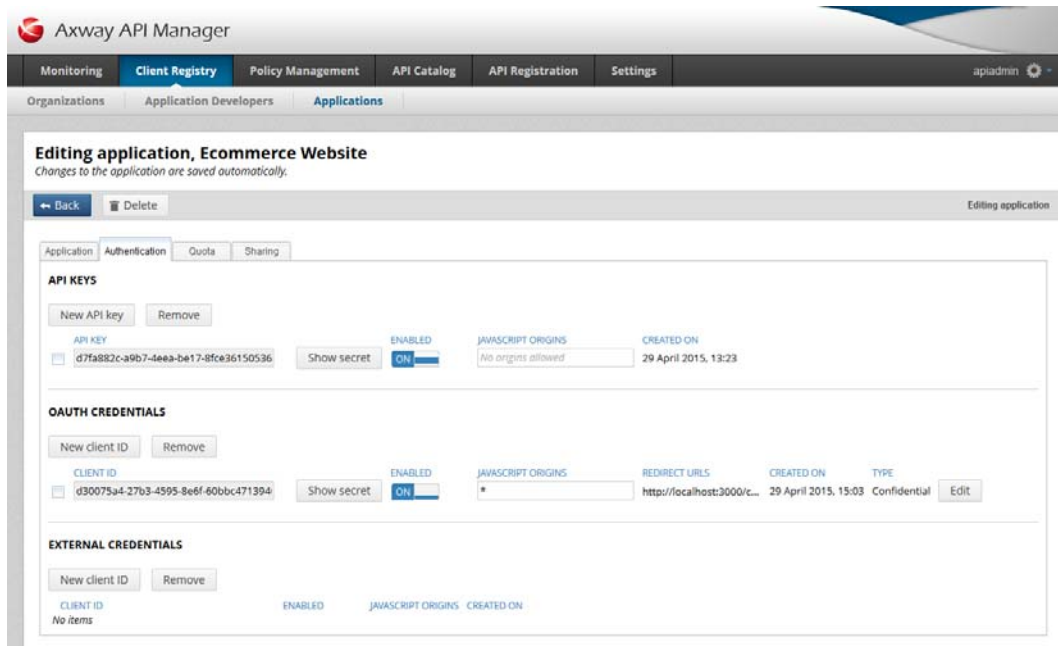
Exploit beamed via NFC to hack Samsung Galaxy S3 (Android 4.0.4)

Using a pair of zero day vulnerabilities, a team of security researchers from U.K.-based MWR Labs hacked into a Samsung Galaxy S3 phone running Android 4.0.4 by beaming an exploit via NFC.



By [Ryan Naraine](#) for [Zero Day](#) | September 19, 2012 -- 17:59 GMT (10:59 PDT) | Topic: [Security](#)

9. The Curious Case of Inversion of Control



Countermeasures: A full Client side DMZ is impractical, however a client side sandbox with server side restrictions such as server side session management and control flow is possible. Clients should only accept pushes from authorized servers over strongly authenticated and encrypted channels.

10. The Curious Case of Order of Operations

- **Context**

APIs can appear to be a static set of getters and setters, but once they are built into other applications the combinations and permutations can drive unexpected behavior on the enterprise back end.

- **Vulnerabilities**

Old school security folks may call this one Madame TOCTOU for time of check, time of use vulnerability or race conditions.

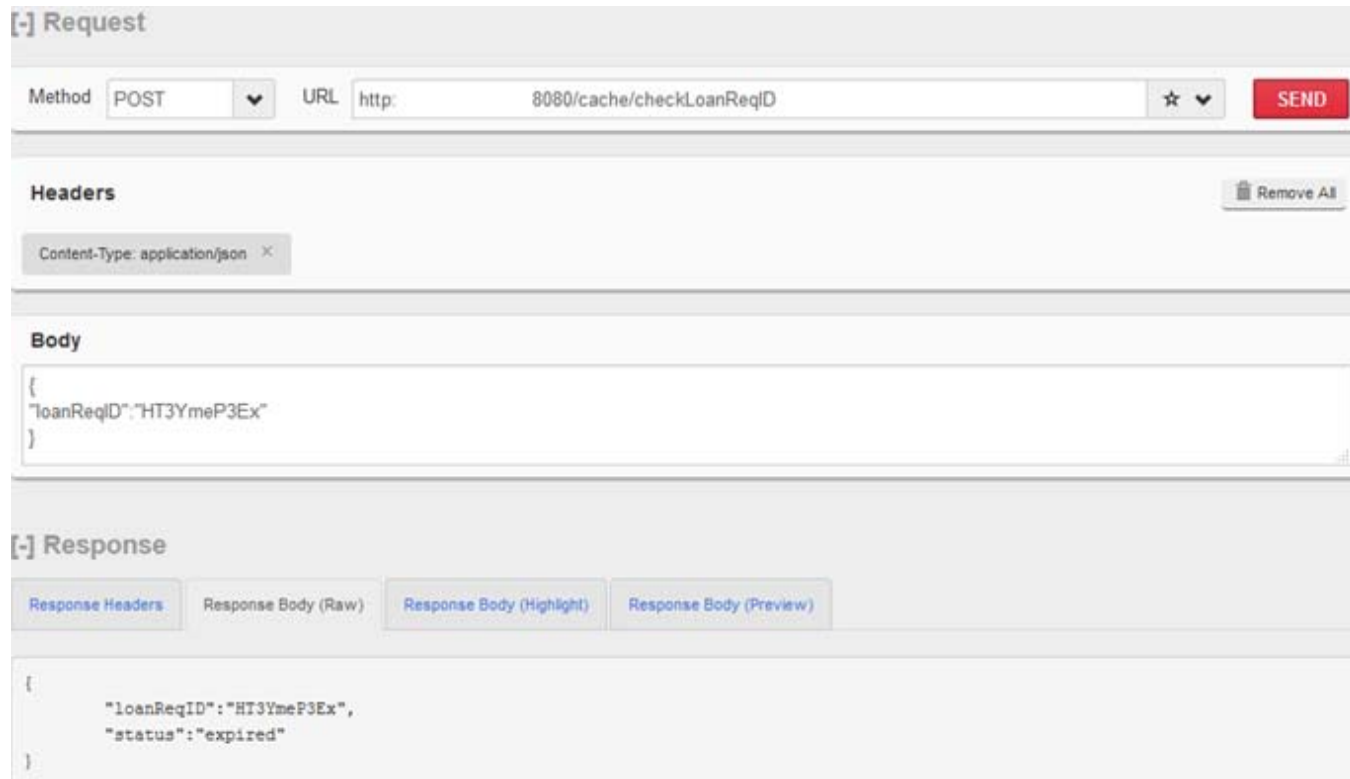


“The principle difficulty in your case,” remarked Holmes in his didactic fashion, “lay in the fact of there being too much evidence. What was vital was overlaid and hidden by what was irrelevant.

Of all the facts which were presented to us we had to pick just those which we deemed to be essential, and then piece them together in their order, so as to reconstruct this very remarkable chain of events.”

-Sherlock Holmes
The Naval Treaty

10. The Curious Case of Order of Operations



The screenshot displays a REST client interface with two main sections: 'Request' and 'Response'.

Request Section:

- Method:** POST
- URL:** http://8080/cache/checkLoanReqID
- Headers:** Content-Type: application/json
- Body:**

```
{
  "loanReqID": "HT3YmeP3Ex"
}
```

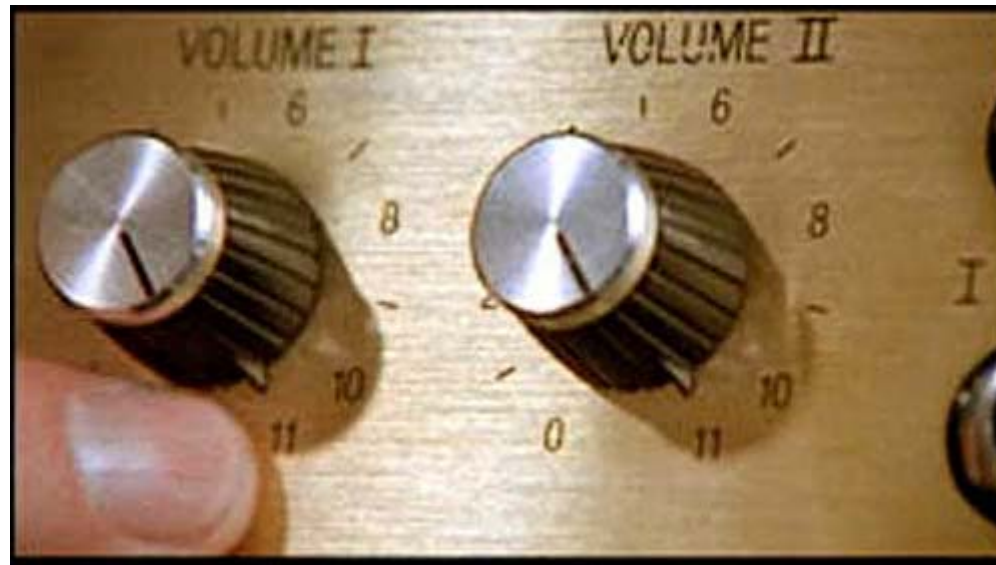
Response Section:

- Response Headers:** (tab selected)
- Response Body (Raw):**

```
{
  "loanReqID": "HT3YmeP3Ex",
  "status": "expired"
}
```

Countermeasures: Granular control on the server side for full session state management is the main key here.

We're taking it to 11...



Source: "This is Spinal Tap" movie

11. Trusted is not Trustworthy

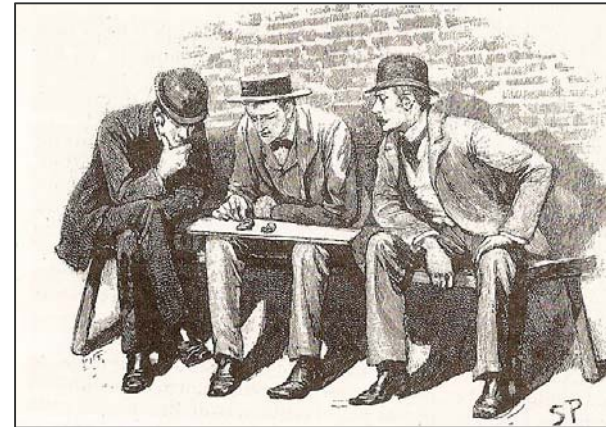
- **Context**

The security architect's most difficult opponent is probably not a malicious attacker. The security architect's most difficult opponent is probably themselves.

Microsoft's John Lambert says it well, "Defenders think in lists, attacker think in graphs. As long as that is true attackers win."

- **Vulnerabilities**

Humans have cognitive biases including overconfidence, blind spots, easily susceptible to seductive details, data, and security conference presentations.



"Always approach a case with an absolutely blank mind. It is always an advantage. Form no theories, just simply observe and draw inferences from your observations."

-Sherlock Holmes

The Adventure of the Cardboard Box



11. Trusted is not Trustworthy

API Transaction	Valid time	Age	Method	Service
b4164a554b487df3e091c6bd	06/05 at 15:27 - 06/05 at 15:27	10 s 98 ms	Access Token Info	OAuth 2.0 Token Info Service
cd104a55953c1b825c1a0368	06/05 at 15:02 - 06/05 at 15:02	10 s 138 ms	Access Token Info	OAuth 2.0 Token Info Service
2e134a55e140fb2d5227bfa	06/05 at 15:12 - 06/05 at 15:12	10 s 143 ms	Access Token Info	OAuth 2.0 Token Info Service
bd144a55b205739c18950107	06/05 at 15:19 - 06/05 at 15:19	10 s 79 ms	getCustomerById	CustomerInfo
1dd549553f13b903072e0e44	06/05 at 10:47 - 06/05 at 10:47	1 s 599 ms	getAll	Technical Forms API
c0164a55c009608773c8d3a	06/05 at 15:27 - 06/05 at 15:27	1 s 75 ms	getAll	CustomerInfo
17124a557c008bd583038814	06/05 at 15:09 - 06/05 at 15:12	2 min 17 s 948 ms	getAll	CustomerInfo
57154a5507070eefa7355d8b	06/05 at 15:24 - 06/05 at 15:27	2 min 55 s 242 ms	getAll	CustomerInfo
f8164a55e74835e5d1791a71	06/05 at 15:26 - 06/05 at 15:28	10 s 162 ms	Access Token Info	OAuth 2.0 Token Info Service

MOST UTILIZED API

Service	Number of calls (today)
OAuth 2.0 Token Info Service	85
CurrentAccountOperations	27
OAuth 2.0 Authorization Service	20
Technical Forms API	15

LESS UTILIZED API

Service	Number of calls (today)
StockQuote	0
SearchCustomerService_TOKENService	0
Cards API	3
ClientInformation	5



Duration	Date/Time	Group
898 ms	07/05/15 11:17:13.966	Internal
972 ms	06/05/15 19:49:21.881	Internal
732 ms	06/05/15 14:08:39.186	Internal
750 ms	06/05/15 14:05:54.397	Internal
763 ms	06/05/15 14:05:00.856	Internal
758 ms	06/05/15 14:04:16.331	Internal

Countermeasures: Don't rely on lists alone. Think of ways how an API Gateway and other security can make it more costly for an attacker to access your enterprise's resource graph.

Questions?

- Gunnar's blog: <http://1raindrop.typepad.com>
- Twitter @oneraindrop
- Mark's blog: <http://soatothecloud.com/>
- Twitter: @TheMarkOneill
- Axway: www.axway.com/api-first

