



STScI | SPACE TELESCOPE
SCIENCE INSTITUTE

EXPANDING THE FRONTIERS OF SPACE ASTRONOMY

ADQL versus Firewalls

Tom Donaldson

IVOA Interop – Northern Fall 2022



Normal TAP Queries – Gaia Search in Topcat

Table Access Protocol (TAP) Query

Select Service Use Service Resume Job Running Jobs

Metadata

Find:

Name Descrip Or

- gaiadr3.frame_rotator_so
- gaiadr3.gaia_cr3_xm
- gaiadr3.gaia_source**
- gaiadr3.gaia_source_lite
- gaiadr3.gaia_source_simu
- gaiadr3.gaia universe mc

Name	Type	Unit	Indexed	Description
random_index	long		<input type="checkbox"/>	
ref_epoch	double	yr	<input type="checkbox"/>	Refer
ra	double	deg	<input checked="" type="checkbox"/>	Right
ra_error	float	mas	<input type="checkbox"/>	Stan
dec	double	deg	<input checked="" type="checkbox"/>	Dec
dec_error	float	mas	<input type="checkbox"/>	Stan
parallax	double	mas	<input checked="" type="checkbox"/>	Para

Service Capabilities

Query Language: ADQL-2.0 Max Rows: 3000000 (default) Uploads: 100Mb

ADQL Text

Mode: Synchronous

```
SELECT *, DISTANCE(
  POINT(81.28, -69.78),
  POINT(ra, dec)) AS ang_sep
FROM gaiadr3.gaia_source
WHERE 1 = CONTAINS(
  POINT(81.28, -69.78),
  CIRCLE(ra, dec, 5./60.))
AND phot_g_mean_mag < 20.5
AND parallax IS NOT NULL
ORDER BY ang_sep ASC
```

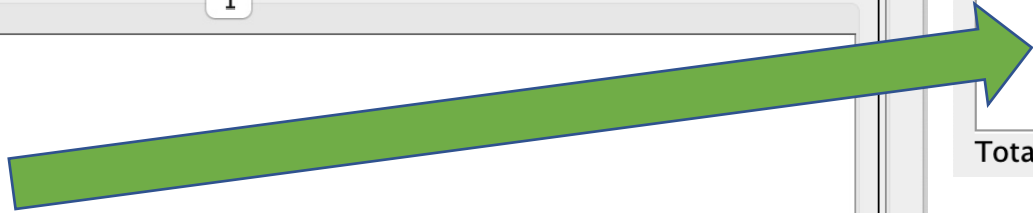
Examples Info

TOPCAT(2): Table Browser

Table Browser for 2: TAP_21_gaiadr3.gaia_source

	ra	ra_error	dec	dec_error	parallax
2	81.28128	0.076791	-69.77952	0.074025	0.02074
3	81.28169	0.604513	-69.78071	0.67206	-0.41268
4	81.28265	0.279567	-69.77973	0.299293	-1.29676
5	81.27835	0.409955	-69.78089	0.471944	0.11124
6	81.27675	1.13079	-69.78064	0.687543	0.14228
7	81.28162	0.125515	-69.78132	0.121687	-0.03004
8	81.28406	0.296852	-69.77963	0.380322	-0.10883
9	81.28382	1.01635	-69.78065	0.99436	-2.33634
10	81.28446	0.691272	-69.78022	0.568826	-0.59392
11	81.27632	0.129207	-69.77868	0.126258	-0.08673
12	81.28025	0.230923	-69.77808	0.214535	-0.27333
13	81.27871	0.235981	-69.77806	0.270248	-0.02248
14	81.28465	0.659052	-69.7787	0.331732	0.45203
15	81.27744	0.073194	-69.77924	0.070771	0.09072

Total: 21,739 Visible: 21,739 Selected: 0





Normal TAP Queries – RegTAP via PyVO

```
[4]: allwise_image_services = vo.regsearch(servicetype='image', keywords=['allwise'])  
allwise_image_services.to_table(['ivoid', 'short_name', 'res_title'])
```



[4]: *Table length=1*

ivoid	short_name	res_title
object	object	object
ivo://irsa.ipac/wise/images/allwise/l3a	AllWISE L3a	AllWISE Atlas (L3a) Coadd Images



But then...

You make a small change, like

- add an ORDER BY
- come into the office for the first time
- just try the same thing a month later

Query Language: ADQL-2.0 Max Rows: 3000000 (default) Uploads: 100Mb

ADQL Text

Mode: Synchronous

```
SELECT *, DISTANCE(
  POINT(81.28, -69.78),
  POINT(ra, dec)) AS ang_sep
FROM gaiadr3.gaia_source
WHERE 1 = CONTAINS(
  POINT(81.28, -69.78),
  CIRCLE(ra, dec, 5./60.))
AND phot_g_mean_mag < 20.5
AND parallax IS NOT NULL
ORDER BY ang_sep ASC
```

Load Error

Error loading TAP_22_rr.resource,rr.capability,rr.interface,rr.resource,rr.resource,rr.res_subject.
Connection reset

OK Show Details

```
allwise_image_services = vo.regsearch(servicetype='image', keywords=['allwise_image_services.to_table()]['ivoid', 'short_name', 'res_title']
```



```
-----  
ConnectionResetError Traceback (most recent call last)  
File ~/opt/anaconda3/envs/navo-env/lib/python3.9/site-packages/urllib3/conn  
ConnectionPool.urlopen(self, method, url, body, headers, retries, redirec  
, pool_timeout, release_conn, chunked, body_pos, **response_kw)  
702 # Make the request on the httplib connection object.  
--> 703 httplib_response = self._make_request(  
704     conn,  
705     method,  
706     url,  
707     timeout=timeout_obj,  
708     body=body,  
709     headers=headers,  
710     chunked=chunked,  
711 )
```



What Happened?

Is the service down?

Is my internet access having problems?

But if I remove ORDER BY, it works.

Some queries work, some don't. Hmm



Wait! We've seen this before. Sort of...

A few years ago

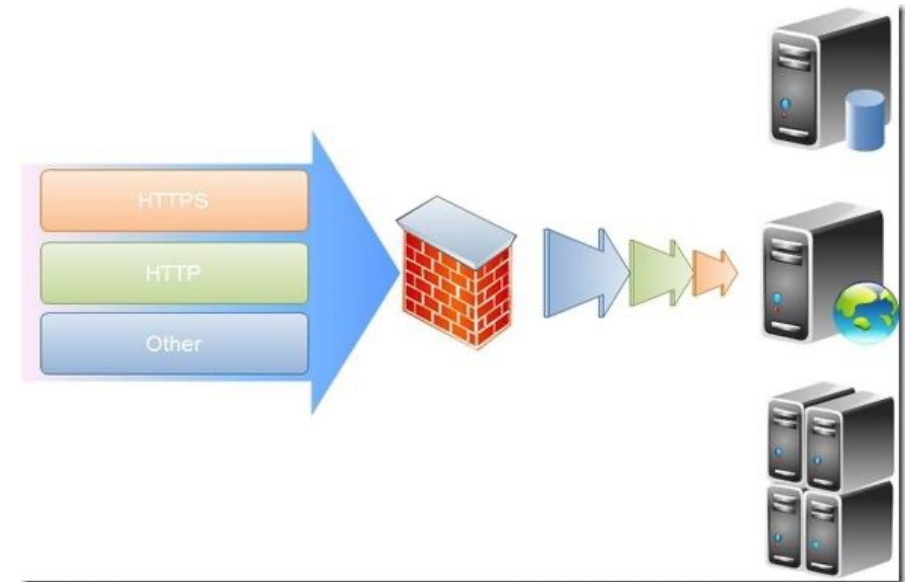
- We noticed that remote clients for MAST TAP services had similar errors
- But the same queries on-site worked fine.

After working with network security team

- Web Application Firewall (WAF) was blocking those queries because they looked like SQL Injection
- Solved: Add exceptions to WAF rules to not check for SQL injection for MAST TAP endpoints

But my recent errors are different

- I'm on-site at STScI
- Some failed queries are to external services like the GAVO RegTAP service.
- Fails on internal network, guest network, and eduroam
 - Need to go home or use my iPhone



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)



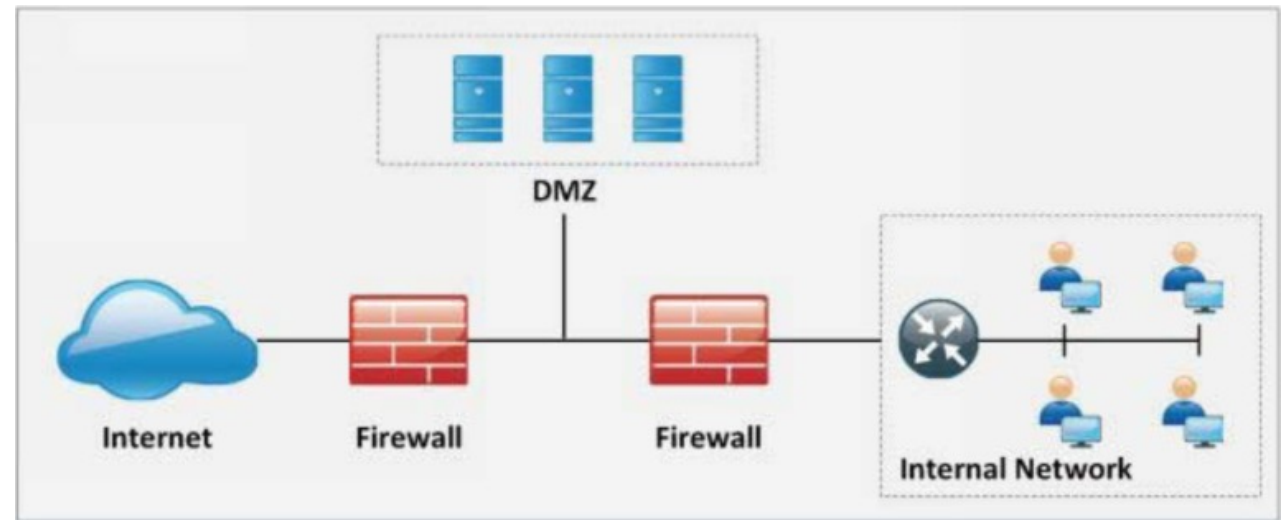
What Changed?

Could a firewall be blocking outbound traffic? Traffic within STScI?

- Yes

Now more partitions and firewalls within the STScI network

- Not just the WAF. Also have perimeter firewall and user firewalls.
 - All have more advanced threat detection built in
- Traffic is scanned for SQL injection in all directions



[This Photo](#) by Unknown Author is licensed under [CC BY](#)

Aside about SQL Injection...



What is SQL Injection and Why Do We Care?

What is SQL Injection?

- User input anticipates how the SQL query will be constructed; breaks out to do something else.

```
var user_input = Request.getParam("obs_id"); // ["myobsid"]
var query = "SELECT * FROM ObsCore WHERE obs_id = " + user_input + ";";
var results = Sql.execute(query);
```

- User inputs like these could break out of intended behavior:
 - 1 or 1=1 → All rows returned
 - 1; DROP TABLE ObsCore → Drops the ObsCore table

Why do we care?

- Worst case successful attack is quite damaging and embarrassing
- Injection is #3 on the [OWASP* top 10](#) list of web app security risks
- STScI is probed pretty much continuously, mostly by bots

(*) OWASP is the Open Web Application Security Project, "an open community dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted."



SQL Injection Mitigation

See the [OWASP Cheat Sheet](#)

→ Use "prepared statements" instead of building a string for SQL to execute.

```
String[] params = ADQLParser.getParams();           // ["myobsid"]
String query = ADQLParser.getQueryTemplate();      // "SELECT * FROM ObsCore WHERE obs_id = ? ";

PreparedStatement pstmt = connection.prepareStatement( query );
for (int i=1; i<=params.length; i++) {
    pstmt.setValue( 1, params[i]);
}
ResultSet results = pstmt.executeQuery( );
```

- Challenging with ADQL's dynamic queries.
 - E.g., not possible to use prepared statement with ORDER BY value.
- **Very** tempting to pass large sections of user input directly to DB.



SQL Injection Mitigation (2)

- Don't give the web service write access to the DB.
 - Helps for attacks that try to modify the DB.
 - Not always practical or possible

- Sanitize user input
 - Very difficult to catch everything, especially with ADQL that allows most SQL syntax.

- Use firewalls to block requests that look like SQL injection.



So what do we do about it?

For inbound requests, added firewall exceptions for the MAST TAP endpoints.

But there are > 120 registered TAP endpoints

- Can we add exceptions for all of them?
 - Even so, sometimes want to access unregistered services
- How many is too many?
 - How would we maintain the exceptions when registrations change?
 - Could too many exceptions affect firewall performance?
- Regex can be used, but what patterns to search for?
 - “LANG=ADQL” is required, so can look for it in GET params or POST data
 - “/sync” or “/async” must be at the end of the URL (prior to GET params)



Consider changes to TAP standard?

Assume STScI adds firewall exceptions allowing inbound and outbound TAP requests.

- Great for STScI TAP services!
- Great for TAP users based at STScI!

But what if universities and research institutions become more security conscious?

- New firewalls could cause astronomers' TAP queries to fail mysteriously.
- Situation may just get worse as more automation is adopted and inspection becomes more sophisticated.

Is there anything the IVOA could or should do?

- Use https for TAP services?
 - Temporary measure only. STScI doesn't unpack https for inspection now, but expects to soon.
- Support (or require) some kind of encoding of the ADQL?
 - Actually a tactic used by malicious actors to bypass some detection
- New syntax?
 - That's a big change, but would move us further away from SQL injection, real and imagined.