

# QueryConf – 2019

“Event Driven” Extensions

[atul@polylogyx.com](mailto:atul@polylogyx.com)

# Agenda

- Poly who?
  - What's our deal with osquery? What's our background?
- Evented tables in an extension
  - Why? Why not in core?
- Challenges
  - Volume of data
  - Extension management, deployment, clean up
  - Configuration
- Things we tried, things that worked, things that didn't
- Wishlist for a better world



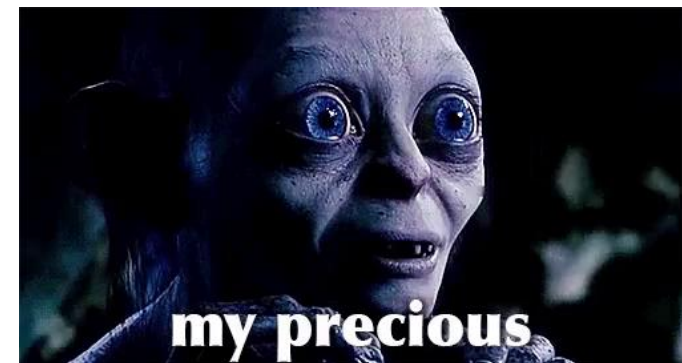
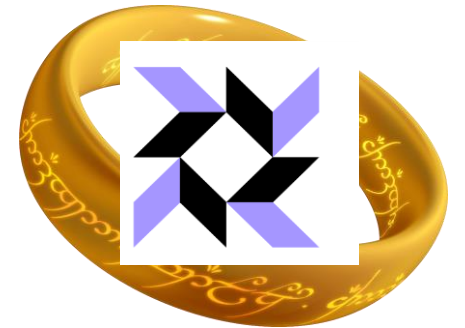
# Background to our story

- Whether Matrix or Security ‘agents are scary’
- Too many of them
  - Agent fatigue, deployment challenges, upgrade challenges, management challenges, redundant functionalities, slow path to innovation
- Yet “Agent” is the dominant form factor for endpoint security
  - Agent-less?? Mmm..maybe for simpler use cases



# Who are we – Our story

- Started by endpoint security folks who were tired of hearing:  
*“Agent’ is a 4-letter word”*
- Our vision – One “agent” to rule them all
  - Build an agent platform with an SDK
  - Let every new tech be a plug-in, Get rid of monolithic agents
  - Set up a market place approach for plug-ins
  - Self contained plug-ins for ease of development, deployment, inter plug-in communication, foster as well as reduce time for innovation, democratization & all the good stuff
- And then ‘osquery’ happened..
  - Let’s build an ‘extension’ – How about adding “real time” events on Windows and make an **EDR** out of it?



# Events..why?

- Security Models are evolving
  - Signature->IoCs-> TTPs e.g. MITRE ATT&CK
  - Properties + Events => Better coverage (no blind spots), Context, Timeliness
  - More muscle for IR/ThreatHunting Query Packs
- Isn't it same as "sysmon + osquery"?
  - Yes and no
  - Sysmon – different agent/provisioning/packaging system, when we are looking for 'The One'
  - Beauty of SQL missing (Can't do 'JOINS' with core tables) – reduces osquery to an event forwarder
  - Dependence on code we don't control
- Can we not do in core?
  - Possibly, but we really wanted to build on extension model
  - Not everything should belong to core or we are back to 'monolithic' agent

# check-with-the-experts time !!

Inbox (206) - atul@polyk... My Drive - Google Drive... \* osquery-extension | Pol... # windows | osquery Slack

Secure | https://osquery.slack.com/messages/C0FHNQ2N6/

**osquery**   
 OpenPlgx

All Unreads   
 All Threads

Channels   
 # alarms   
 # doorman   
 # general   
 # windows

Direct Messages   
 slackbot   
 OpenPlgx (you)

Apps

**#windows**   
 ☆ | 👤 197 | 📎 1 | osquery for Windows has been tested with Windows 8.1, 10, and Server 2012. osquery builds and runs on Windows.   
 Saturday, September 30th

Search

appropriately ?

**thor** 11:37 PM   
 So, while I don't think you could make a *formal* pub/sub that osquery manages, what you *could* do would be to make your extension code behave like a pub/sub, and use the local store as a caching layer, and anytime someone queries against your extension offer up the cached data, which is more or less what osquery pub/subs do

11:38 ☆   
 That being said, I've never heard of or seen someone make a pub/sub inside of an extension. It could be possible but I'm not sure we support that or not, so you'd be in unexplored territory. I'd encourage you to make a generic extension table, and then you can open the DB and insert values into that from within your call back, and then query time you'd just get the values out of the DB. There's quite a few good examples throughout the code base of interfacing with the DB

**theopolis** 11:40 PM   
 Event publishers and subscribers don't work in extensions right now. To date there hasn't been a strong use case to make this work.

**manu** 🍌 11:42 PM   
 hmm, that saves me sometime then 😊 . what i was thinking may be was to have this thing whole pub-sub thing in the extn code, so that i can leverage the existing core features as is and my extension is built independently of the core   
 @thor @theopolis btw can you think of some real use case from extension writer perspective which would require this feature in future may be ?   
 just trying to understand it actually ... why it hasn't been done it this way till now

**theopolis** 11:53 PM   
 I can't think of any from my end, writing a publisher is difficult, they can be high volume, and passing content over the thrift RPC introduces latency + requires serialization of data. If you're publishing/subscribing to abstractions around OS/built-in callbacks, then writing them into osquery core seems the easiest too

**manu** 🍌 11:58 PM   
 @theopolis thanks for the insight, this sounds reasonable in favor of not supporting this feature

+ Message #windows

Type here to search

12:26 PM 11/7/2017

# PolyLogyx Extension – “Build-It” Time

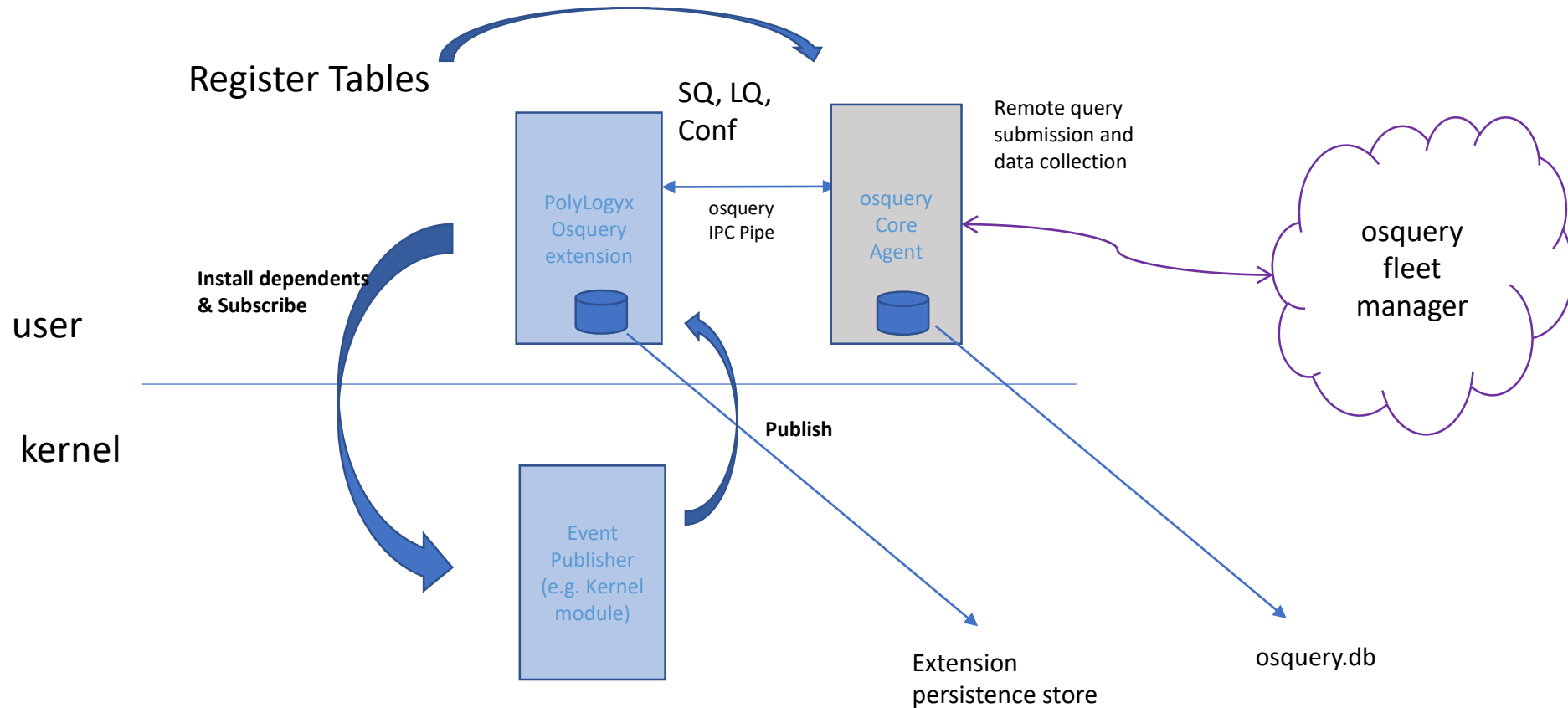
## Goals

- Extend Windows Osquery with tables off real time events
  - Files, process, registry, socket, dns...(20+ tables today)
- The entire extension to be packaged as a single binary
  - Carry all the dependents within
- Install/uninstall should be clean
- Follow existing osquery flags/config/provisioning mechanisms
  - --disable\_events, --events\_expiry, --events\_max
- Single agent model
- Undiluted SQL syntax
- Should work with the community version of osquery and its configs
  - Not just a white-labeled opaque set

## Constraints

- No osquery managed Pub-Sub in extension
  - Create our own
- No feasible access to osquery’s persistence (e.g. osquery.db)
  - Create our own DB management
- No extension specific config schema
- No support for ‘streaming’. Have to make it work with diff-based scheduled queries
- Potentially high volume of data over thrift IPC
- No shutdown or config-refresh notification in extension

# My Beautiful 'architecture' diagram





# osquery with events is good, but what about 'user focus'?

- No good if (RAM usage, CPU spikes, Queries-not-returning)
  - Event Volumes, diffing engine, poorly crafted queries, event flags...(recall all the 'pitfalls')
- If you don't get it right & Windows being so 'chatty'.



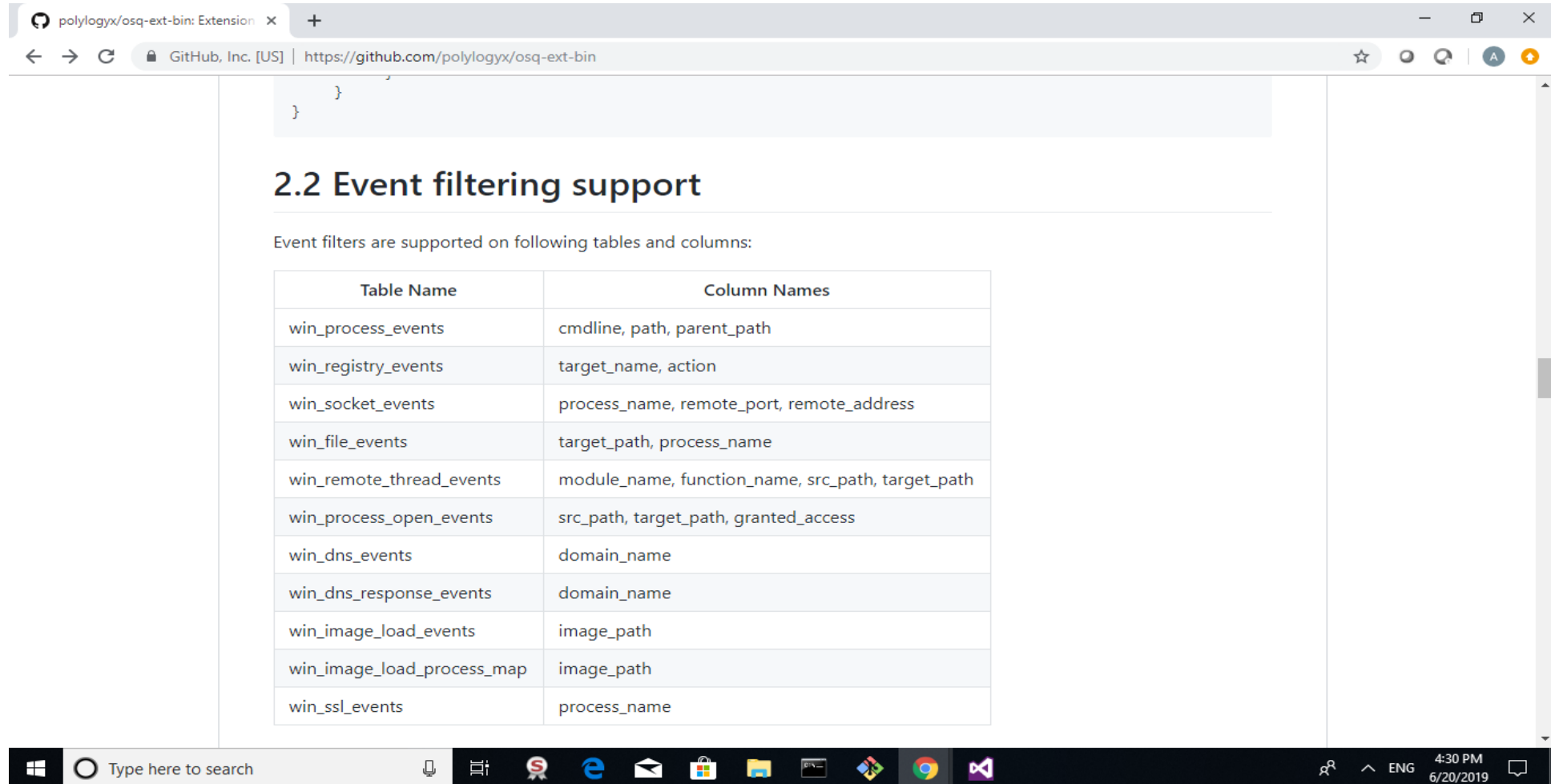
# Agent Performance with events

- Type of persistent store
  - File, DB (RocksDB, SQLite..), Event Log
  - Depending on the type/volume of events, read/write contention, query schedules
- Managing event lifecycle
  - Osquery events follow the flags `--disable_events`, `--events_expiry`, `--events_max`
  - Support those or use custom flags
- Configuration for event filters
  - Extend the config
  - Keep filters as close as possible to event publication
  - Support refreshing the config in extension (in a thread or a config plugin)

```
{  
  schedule {  
  }  
  filters {  
    include {  
    };  
    exclude {  
    };  
  }  
}
```

```
auto status = osquery::Registry::call("config", { { "action", "genConfig" } }, response);
```

# Filters in PolyLogyx Extension



polylogyx/osq-ext-bin: Extension x +

GitHub, Inc. [US] | https://github.com/polylogyx/osq-ext-bin

```
}  
}
```

## 2.2 Event filtering support

Event filters are supported on following tables and columns:

Table Name	Column Names
win_process_events	cmdline, path, parent_path
win_registry_events	target_name, action
win_socket_events	process_name, remote_port, remote_address
win_file_events	target_path, process_name
win_remote_thread_events	module_name, function_name, src_path, target_path
win_process_open_events	src_path, target_path, granted_access
win_dns_events	domain_name
win_dns_response_events	domain_name
win_image_load_events	image_path
win_image_load_process_map	image_path
win_ssl_events	process_name

Type here to search

4:30 PM 6/20/2019

# Query with evented tables

- Example query: `““select * from win_file_events where md5= '<>'””;`
- Returning 'events' from extended tables in response to a query
  - *virtual void generator(RowYield& yield, QueryContext& context) – Not available to external plugins*
  - `virtual osquery::QueryData <Table>::generate(osquery::QueryContext& Request)`
    - Redundant events processing, diffing penalties, IPC limitationsQueryContext
- QueryContext
  - Shove the SQL constrains in extension (Virtual SQL tables)
  - Can't implement the 'differential' logic in extension – Bummer ☹️
- Directly impacted by the count of events
  - Use filters wherever possible

# Handling shutdown notifications

- Clean up Pub/Sub infrastructure
  - Unsubscribe to notifications
  - Clean up any dropped components
- Earlier versions – SIGUSR1 mapped to SIGILL
  - `waitForShutdown()` – No working on Windows
  - `-#define SIGUSR1 SIGILL /+#define SIGUSR1 SIGTERM`
- Newer versions - Outside monitoring service
  - OS notification on `osqueryd` service shutdown
  - Trigger cleanup

# Wishlist

- Rule book for extensions
  - Avoid namespace conflicts
- Expand QueryContext
- Provision for streaming of events
- Shutdown notification in extension
- Extensions aware fleet management
  - Room for extensions managements
- Pub/Sub across extension
  - Common event bus
  - Common event schema
- Community/commercial extension ecosystem



[atul@polylogyx.com](mailto:atul@polylogyx.com)

Slack: openplgx

twitter: @polylogyx

Github: <https://github.com/polylogyx>