

Quilt: A System for Distributed Temporal Queries of Security Relevant Heterogeneous Data

Timothy Shimeall, Ph.D. (tjs@cert.org)
George M. Jones (gmj@cert.org)

January 14, 2014



Copyright 2014 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

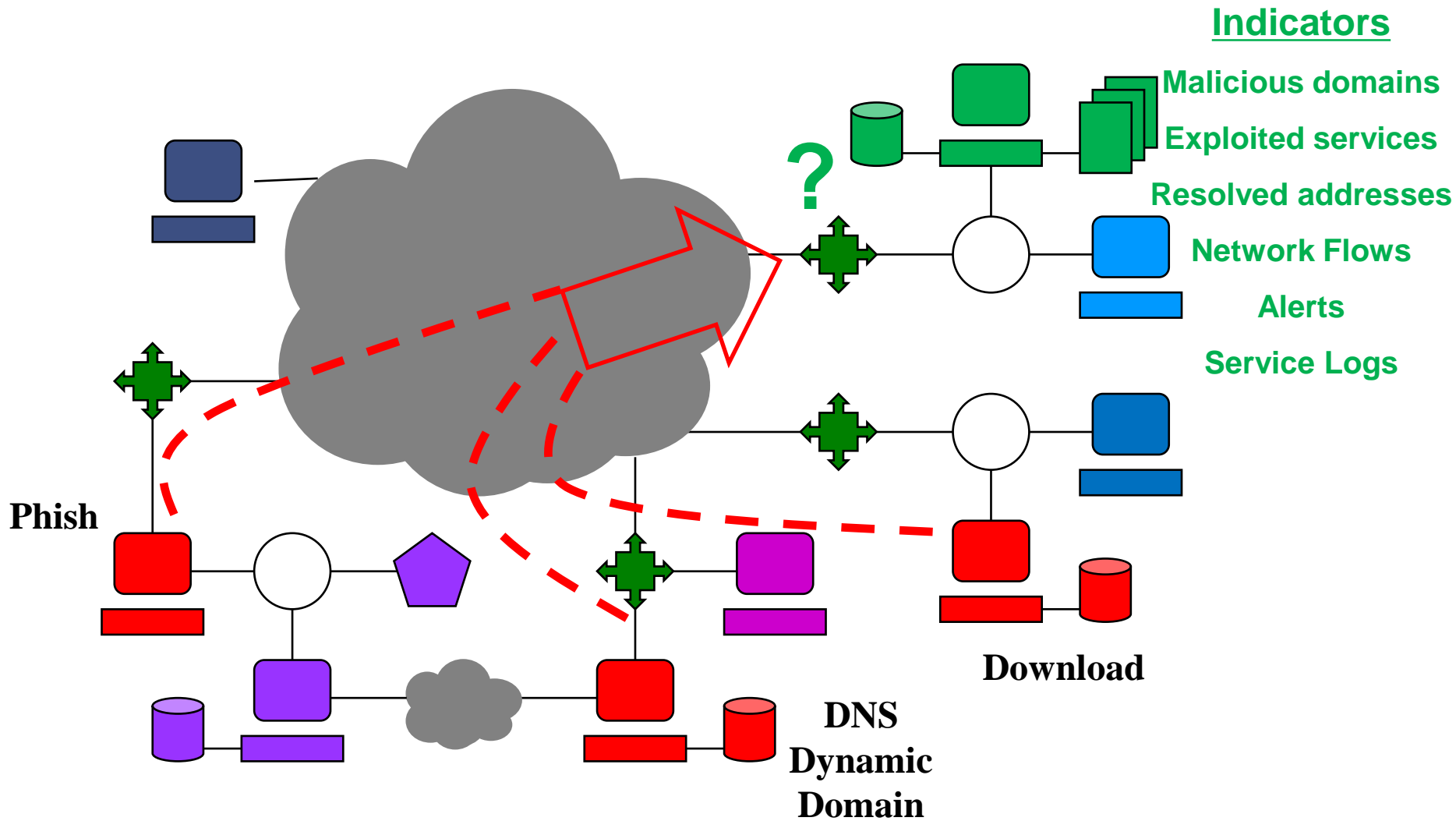
This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon®, CERT® and FloCon® are registered marks of Carnegie Mellon University.

DM-0000849



Problem: Querying multi-sorted network data



Project Introduction

Provide a basis for queries across data sources of differing kinds, locations, and content to retrieve security-relevant data

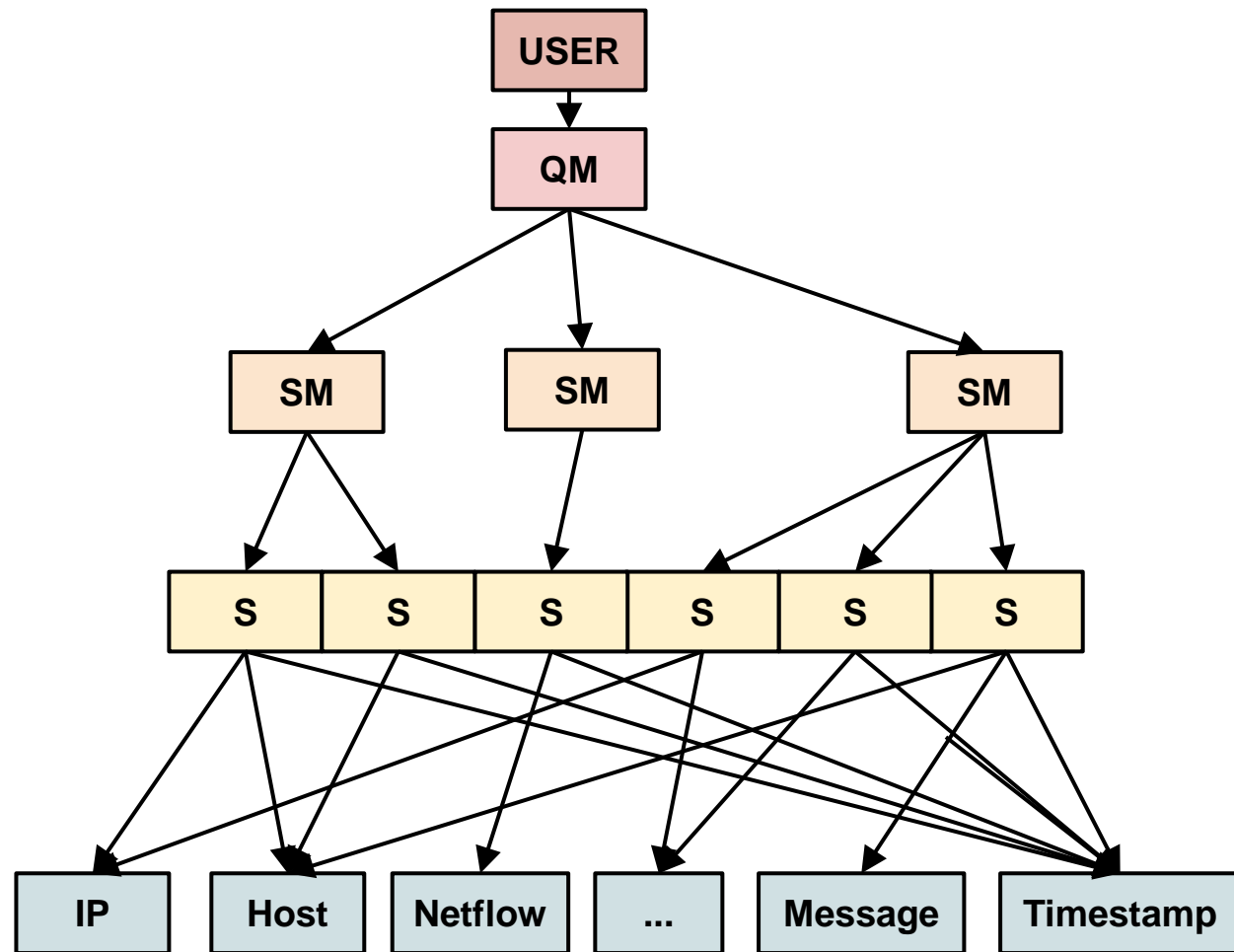
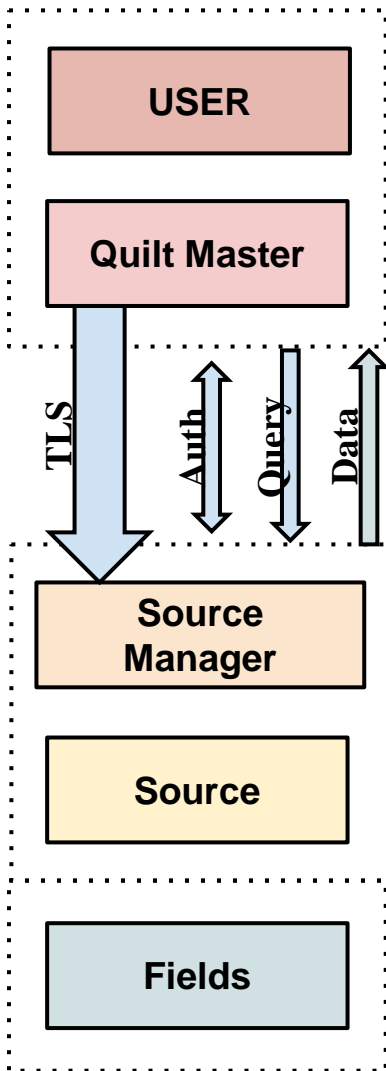
Currently: manual, back-haul & normalize, team caucus (time consuming, repetitious, space and bandwidth consumptive, timeliness)

Quilt Approach:

- 1) Central **query master**, restating query into source-related components queried at data location via co-located **source masters**
 - 2) Source masters return minimal and parameterized results with requery capability for drill-down or data completeness
 - 3) Query master **temporally associates** results
- Flexibility in queries
 - Convenient and simple data exploration model
 - Easy to maintain and expand
 - Safety of data
 - Lightweight



Quilt Architecture



Quilt Architecture Levels(1)

User

- Domain knowledge
- Access rights
- Awareness

Query Master

- User interface
- Restate query for source manager
- Specify filtering of results
- Temporal filtering
(concurrent, sequential, until)
- Results presentation

Source Manager

- Native data retrieval interface
- Present credentials as required
- Results filtering

Source

- Retrieve data
- Preserve native structure
- Preserve access rights
- All results timestamped



Quilt API

User: Standard Query Language (Augmented Python)

Query master – User: Standard Query Library

Query master – Source manager: Smart client that stays on line long enough for authentication, refinement and potential callbacks.

- `package{dict of types needed, dict of filters, dict of sources}`

Source manager – Query master:

- `SMQuery(package, authentication)`

Source manager – Source:

- `Startup(authentication, authCallback())`
- `DBQuery(derived_query, DataCallback())`
- `CloseDown(authentication)`



Summary

Distributed data query engine

Allows for broad range of data

- Network flow
- IDS Logs
- System logs and messages

Intrinsically supports temporal relationships

Minimal data interchange

- Bulk query using native retrieval on site
- Filter retrieved results to isolate needed fields
- Communicate with efficient representations

