# Nexthop selection enhancements

## Add support for next hop selection strategy plugins

**John Rushford and Robert Butts, Comcast**

# Nexthop Selection Strategies.

- Added to ATS master in the fall of 2019, merged to ATS 9.0
- Strategies are named and defined in strategies.yaml
- Provides the same functionality as parent selection and is configured for use per remap in remap.config using **@strategy=strategy-name**
- Wrapper functions in HttpTransact choose between parent selection and the new next hop selection based upon the remap rule configuration.
- Eliminates parent selection table searches in the state machine, strategies are associated with the matched remap.
- Adds improved debug logging.
- Large number of unit-tests.
- Can replace parent selection and parent.config entirely.
- See ATS 2019 Fall Summit Presentation, NextHop Selection strategies for more detail.

# Can we now build Next Hop Strategy plugins

## Load a DSO with a class implementing NextHopSelectionStrategy

- Investigate using the NextHopSelectionFactory or a remap to load a DSO with a C++ class that implements a NextHopSelectionStrategy.
- The plugin uses a C API function such as **TSRemapNewInstance()** to create the NextHopSelectionStrategy instance from the loaded DSO and makes it available for use with the matched remap.
- All this is doable but requires some changes to the NextHopSelectionStrategy C++ interface.
- Requires the addition of some TSAPI functions and making the TSAPI available to the NextHopSelectionStrategy.

# NextHopSelectionStrategy C++ interface.

## ATS 9.0 Interface - needs refactor for plugin use case

```
class NextHopSelectionStrategy {
    virtual void findNextHop(const uint64_t sm_id, ParentResult& result, RequestData& rdata,
          const uint64_t fail_threshold,  const uint64_t retry_time, time_t now = 0) = 0

    void markNextHopDown(const uint64_t sm_id, ParentResult& result, const uint64_t fail_threshold,
          const uint64_t retry_time, time_t now = 0)

    void markNextHopUp(const uint64_t sm_id, ParentResult& result, const uint64_t fail_threshold,
          const uint64_t retry_time, time_t now = 0

    bool nextHopExists(const uint64_t sm_id)
}
```

# New NextHopSelection Strategy interface

## Refactored and simplified PR6782

```
class NextHopSelectionStrategy {
    virtual void findNextHop(TSHttpTxn txnp, void *ih = nullptr, time_t now = 0) = 0

    void markNextHop(TSHttpTxn txnp, const char *hostname, const int port, const NHCmd status, void *ih = nullptr,
          const time_t now = 0)

    bool nextHopExists(TSHttpTxn txnp, void *ih = nullptr)

    virtual bool responseIsRetryable(const unsigned int current_retry_attempts, HTTPStatus response_code)
}
```

- This new interface passed the SM object to the next hop functions to allow them to use the sandboxed TSAPI function calls
- The core NextHopConsistentHash and NextHopRoundRobin classes use reinterpret_cast back to HttpSM * and function as before.
- Simplifies the logic in the HttpTransact wrapper functions used to wrap parent selection and next hop selection.
- **PR 6782** changes the interface, simplifies the HttpTransact wrapper functions, and fixes some NextHopSelectionStrategy bugs found while testing. The PR is under review and we hope to have it merged and back ported to ATS 9.0

# Experimental next hop selection plugin

## Work in progress - Rob Butts

- Experimental NextHopSelectionStrategy consistent hash plugin.
- Plugin implements a ‘C’ **TSRemapInitStrategy()** function that is used to create an instance of a C++ object that implements NextHopSelectionStrategy along with its configuration when the DSO is loaded by remap.
- The object is registered with the NextHopStrategyFactory by it’s assigned name in remap.config. name.
- map http://example.com/ http://example.com/ @plugin=nexthop_strategy_consistenthash.so @pparam=/opt/trafficserver/etc/trafficserver/strategy-plugin-0.yaml @strategy=nexthop_strategy_consistenthash.so
- When remap.config is parsed and @strategy=“dso_filename.so” the shared pointer to that object is set on the remap rule and used by HttpTransact thru the findParent(), nextParent(), markParent() wrapper functions.

# Experimental Nexthop plugin adds new API functions

## Work in progress - Rob Butts

```
bool TSHostnameIsSelf(const char* hostname);

// get current host status.
bool TSHostStatusGet(const char *hostname, TSHostStatus* status, unsigned int *reason);

// update host status.
void TSHostStatusSet(const char *hostname, TSHostStatus status, const unsigned int down_time,
        const unsigned int reason);

// retrieve the ParentResult struct for use by the plugin for updating.
struct TSParentResult* TSHttpTxnParentResultGet(TSHttpTxn txnp);
```

# Additional investigation

## Nexthop plugins

- Add additional Transaction Hooks as needed with HttpSM and HttpTransact changes.
- Run next hop callback functions on a continuation using new hooks.