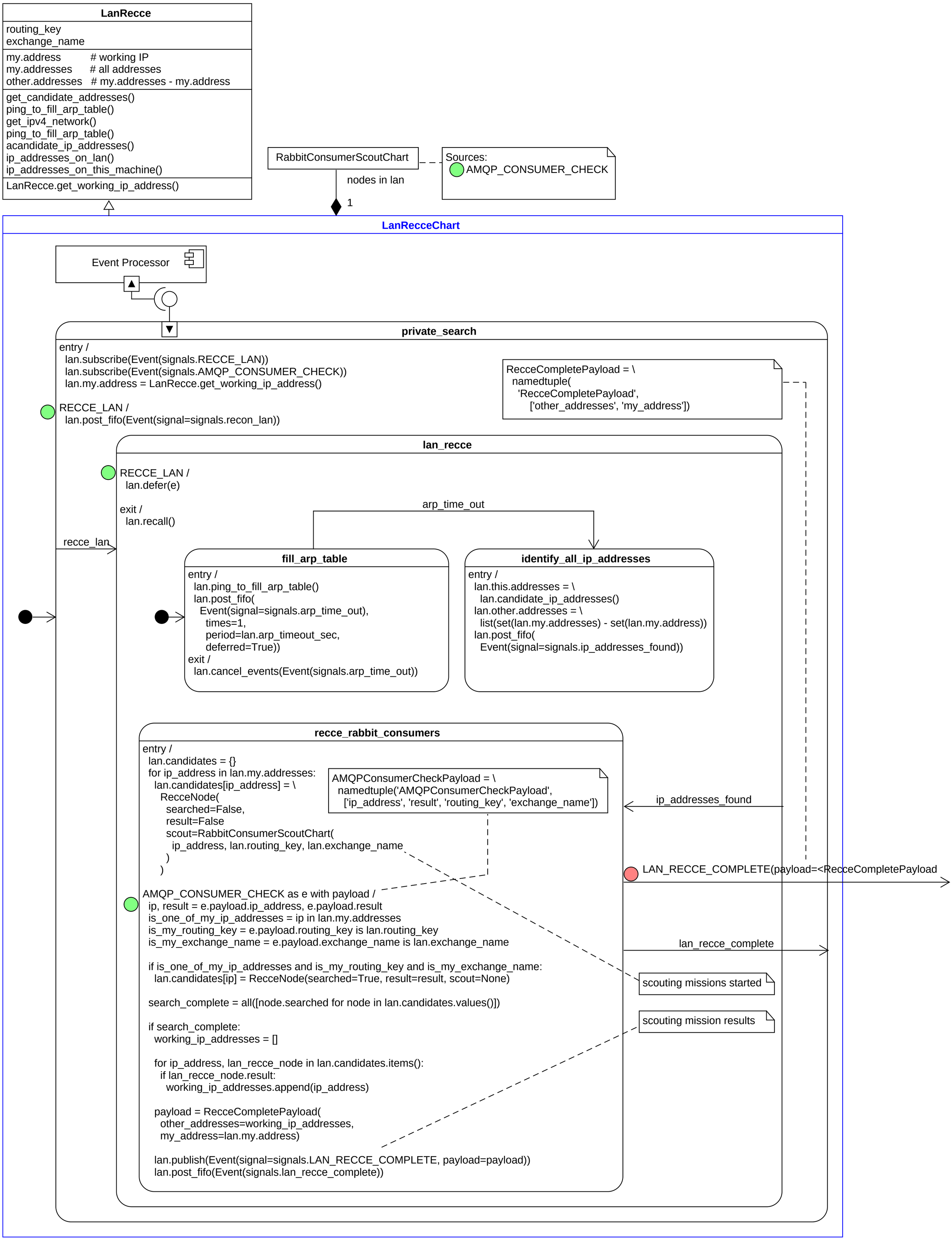
LanRecce
routing_key
exchange_name
my.address # working IP
my.addresses # all addresses
other.addresses # my.addresses - my.address
get_candidate_addresses()
ping_to_fill_arp_table()
get_ipv4_network()
ping_to_fill_arp_table()
acandidate_ip_addresses()
ip_addresses_on_lan()
ip_addresses_on_this_machine()
LanRecce.get_working_ip_address()
RabbitConsumerScoutChart
Sources:
AMQP_CONSUMER_CHECK
nodes in lan
1
LanRecceChart
Event Processor
private_search
entry /
lan.subscribe(Event(signals.RECCE_LAN))
lan.subscribe(Event(signals.AMQP_CONSUMER_CHECK))
lan.my.address = LanRecce.get_working_ip_address()
RECCE_LAN /
lan.post_fifo(Event(signal=signals.recon_lan))
RecceCompletePayload = \
namedtuple(
'RecceCompletePayload',
['other_addresses', 'my_address'])
lan_recce
RECCE_LAN /
lan.defer(e)
exit /
lan.recall()
arp_time_out
recce_lan
fill_arp_table
entry /
lan.ping_to_fill_arp_table()
lan.post_fifo(
Event(signal=signals.arp_time_out),
times=1,
period=lan.arp_timeout_sec,
deferred=True))
exit /
lan.cancel_events(Event(signals.arp_time_out))
identify_all_ip_addresses
entry /
lan.this.addresses = \
lan.candidate_ip_addresses()
lan.other.addresses = \
list(set(lan.my.addresses) - set(lan.my.address))
lan.post_fifo(
Event(signal=signals.ip_addresses_found))
recce_rabbit_consumers
entry /
lan.candidates = {}
for ip_address in lan.my.addresses:
lan.candidates[ip_address] = \
RecceNode(
searched=False,
result=False
scout=RabbitConsumerScoutChart(
ip_address, lan.routing_key, lan.exchange_name
)
)
AMQPConsumerCheckPayload = \
namedtuple('AMQPConsumerCheckPayload',
['ip_address', 'result', 'routing_key', 'exchange_name'])
ip_addresses_found
LAN_RECCE_COMPLETE(payload=<RecceCompletePayload
AMQP_CONSUMER_CHECK as e with payload /
ip, result = e.payload.ip_address, e.payload.result
is_one_of_my_ip_addresses = ip in lan.my.addresses
is_my_routing_key = e.payload.routing_key is lan.routing_key
is_my_exchange_name = e.payload.exchange_name is lan.exchange_name
if is_one_of_my_ip_addresses and is_my_routing_key and is_my_exchange_name:
lan.candidates[ip] = RecceNode(searched=True, result=result, scout=None)
search_complete = all([node.searched for node in lan.candidates.values()])
if search_complete:
working_ip_addresses = []
for ip_address, lan_recce_node in lan.candidates.items():
if lan_recce_node.result:
working_ip_addresses.append(ip_address)
payload = RecceCompletePayload(
other_addresses=working_ip_addresses,
my_address=lan.my.address)
lan.publish(Event(signal=signals.LAN_RECCE_COMPLETE, payload=payload))
lan.post_fifo(Event(signals.lan_recce_complete))
lan_recce_complete
scouting missions started
scouting mission results