from idioxit import jid
from abusehelper.core import bot

class ForwardBot(bot.XMPPBot):
    # Define two parameters (in addition to the default XMPPBot ones)
    room_src = bot.Param("the source room")
    room_dst = bot.Param("the destination room")

@threado.stream
def main(inner, self):
    # Join the XMPP network using credentials given from the command line
    conn = yield self.xmpp_connect()
    # Join the XMPP rooms
    src =
    dst =
    self.xmpp
    yield src | self.room_filter(own_jid) | dst | threado.dev_null()

@threado.stream
def room_filter(inner, self, own_jid):
    while True:
        # Receive one XML element from the pipe input
        element = yield inner
        # Prevent endless feedback loops
        sender = jid.JID(element.get_attr("from")).get_jid()
        if sender == own_jid:
            continue
        # Forward the body elements
        for body in element.getElementsByTagName("body")...
AH fits the bill for automating handling of bulk incident data, but...

1. We need a feedback loop (email) which tracks history and provides easy statistics.
and we do not want to run disparate systems.
Solution:

Make AH a part of the RTIR workflow
class RtirBot(CollectorBot):
    rt_url = bot.Param("URL of RT instance")
    rt_user = bot.Param("RT user with rights to create and modify tickets")
    rt_password = bot.Param("RT user's password")
    rt_use_ah_template = bot.Param("Use AH templates for investigation content generation")
    rt_write_interval = bot.IntParam("write interval", default=60)

    def __init__(self, *args, **keys):
        CollectorBot.__init__(self, *args, **keys)
        self.rtirs = dict()

        # Create base_logger
        self.log = self.create_logger(self.rinis)

        try:
            self.log.info("Connected to RT")
        except:
            self.log.info("Failed connecting to RT")
        raise

        # Create RT Workflow automator
        self.rt_wf = rt.RtirWorkflow(self.rt_client)

    def write_to_rt_ticket(self, room, events):
        rtir_params = self.rtirs.get(room, None)
        if not rtir_params or not events:
            return
class RtClient
class RtTicket
class RtTicketTemplate
class RtirWorkflow
class RtirBot

RT REST API magic
Basic functions for manipulating RT ticket data
Template selection and population with IR data

Logic for linking and/or creating tickets
Bastardized WikiBot :)

The RT REST API
http://example.com/REST/1.0/

REST/1.0/ticket/new

Pass as “content” POST-parameter:

id: ticket/new
Queue: <queue name>
Requestor: <requestor email address>
Subject: <subject>
Cc: <...>
AdminCc: <...>
Owner: <...>
Status: <...>
Priority: <...>
InitialPriority: <...>
FinalPriority: <...>
TimeEstimated: <...>
Starts: <...>
Due: <...>
Text: <The ticket content>

CF.<CustomFieldName>:
<CustomFieldValue>

CF.{} is short for CustomField, aka RT-magic.
Take events and create a corresponding RT Ticket (Incident Report) for each one

Take the created IRs, match IP addresses from them to open incidents and...
link to an existing, open incident according to “best match” or...

create a new incident and link to that ticket.
After creating a new batch of IRs, take the ticket contents and generate/update an Investigation and resolve the batch (IRs).

If a team member updates an Investigation manually, further IRs will be linked, but Investigations will not be auto-updated.
While meant to be fairly easy to deploy, RtirBot does have a few prerequisites.

Expects tickets to have at least the following ‘keys’:

time
ip
Preferrably also type and type_detail, which will allow for auto-selection of a specific template.

Edit your sanitizers to add these keys to your data if they do not exist.
Templates exist in the contrib/rtirbot/templates/directory.

```
Template directory

  drwxr-xr-x  2 root staff 4096 Jul 29 16:41 ./
  drwxr-xr-x  3 root staff 4096 Aug 16 20:24 ../
  lrwxrwxrwx  1 root staff   4 Jul 28 10:45 cbl -> spam
  lrwxrwxrwx  1 root staff   3 Jul 29  9:21 ddos -> dos
  -rw-r--r--  1 cert staff  516 Jul 27 23:06 default
  -rw-r--r--  1 cert staff  531 Jul 29  9:21 dos
  -rw-r--r--  1 cert staff  591 Jul 29 16:41 drone
  -rw-r--r--  1 cert staff  523 Jul 27 23:06 malware
  lrwxrwxrwx  1 root staff  14 Jul 28 09:53 malware_anserin -> malware_torpig
  -rw-r--r--  1 cert staff  766 Jul 28 00:03 malware_conficker
  lrwxrwxrwx  1 root staff  17 Jul 28 09:53 malware_downadup -> malware_conficker
  lrwxrwxrwx  1 root staff  17 Jul 28 09:53 malware_downup -> malware_conficker
  lrwxrwxrwx  1 root staff  17 Jul 28 09:53 malware_kido -> malware_conficker
  lrwxrwxrwx  1 root staff  15 Jul 29 09:20 malware_mbr -> malware_mebroot
  lrwxrwxrwx  1 root staff  15 Jul 29 09:20 malware_mbroot -> malware_mebroot
  -rw-r--r--  1 cert staff 1291 Jul 29 09:19 malware_mebroot
  lrwxrwxrwx  1 root staff  14 Jul 28 09:52 malware_sinomal -> malware_torpig
  -rw-r--r--  1 cert staff  853 Jul 27 23:19 malware_torpig
  -rw-r--r--  1 cert staff  842 Jul 28 10:45 spam
  lrwxrwxrwx  1 root staff   4 Jul 28 10:45 spam_cbl -> spam

sylvande@d2:/usr/local/lib/python2.6/dist-packages/abusehelper/contrib/rtirbot/templates$```
Default template

Teidän verkossanne olevasta laitteesta on havaittu normaalista poikkeavaa, haitallisksi luokiteltua liikennettä. Tarkempia tietoja havainnosta alla. Kaikki aikaleimat ovat UTC:ta:

IR_DATA_PLACEHOLDER

Mahdollisia lisätietoja voi tarvittaessa tiedustella Funet CERTiltä.

-----------------------------
Tämä tiketti on luotu Abusen AbuseHelper-järjestelmällä.
-----------------------------
default (END)

Replaced with actual incident data

Must add:

{CF}.AbuseHelperCheck
to Tickets in RT-IR.
Carlos
“RTIR-Wizard”
Fuentes
provided a quick fix to
add the CF to RT-IR.

Thanks!

Demo time!
Would you like to know more (about AbuseHelper)?

hg clone https://bitbucket.org/clarifiednetworks/abusehelper

contact@clarifiednetworks.com

Access to collab wiki, jabber, etc
P.S. Keep an eye on

http://fordrop.org/

Johan Berggren from NORDUnet is building a platform for distributed analysis of malware, etc.

So long, and thanks for all the phish!

harri@sylvander.net