Inspector Gadget

Automated Extraction of Proprietary Gadgets from Malware Binaries

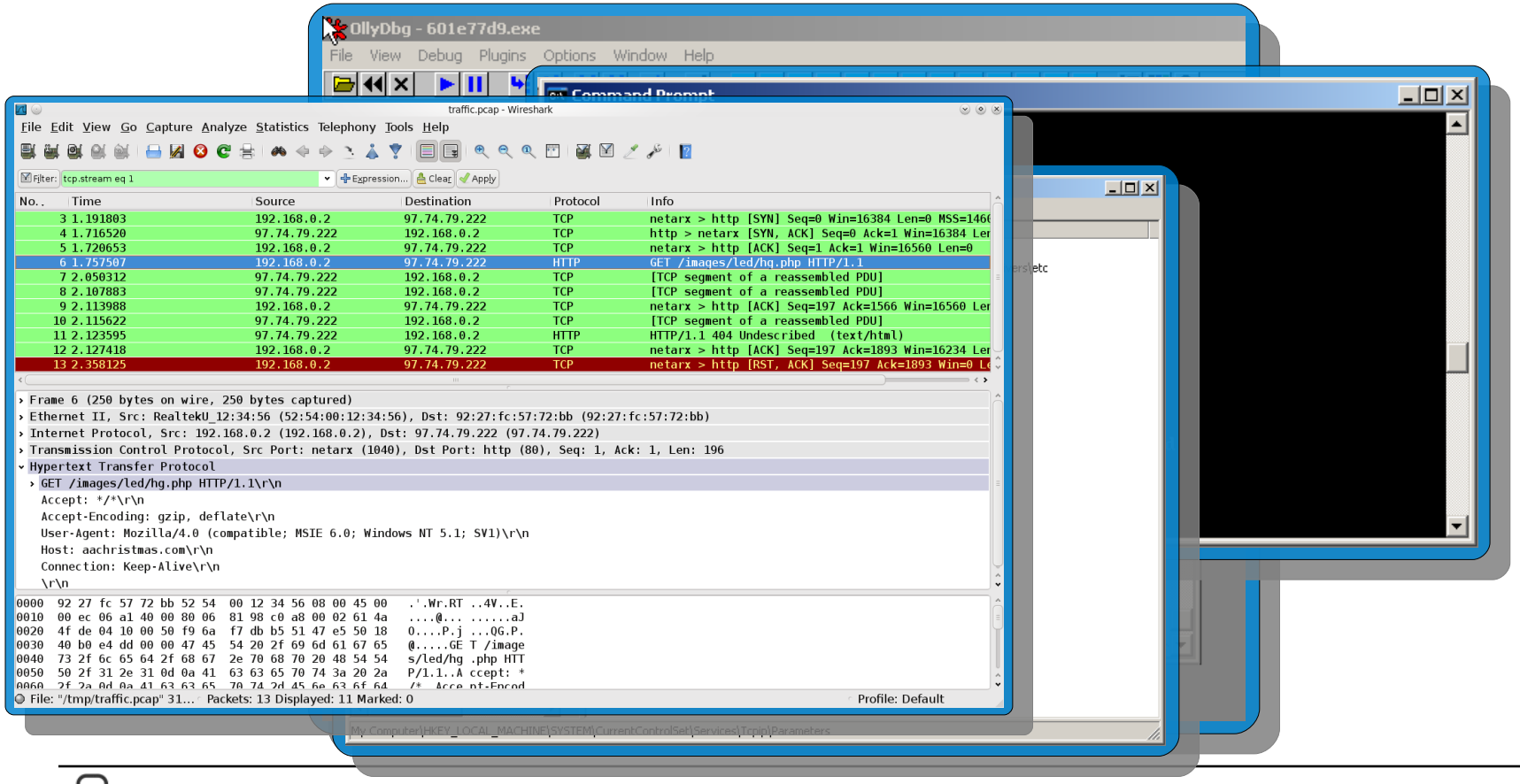
Clemens KOLBITSCH *Thorsten HOLZ*
Engin KIRDA *Christopher KRUEGEL*

ck@iseclab.org

Int. Secure Systems Lab
Vienna University of Technology, Institute Eurecom Sophia Antipolis, UC Santa Barbara

Motivation

- Analysis of malicious code is challenging

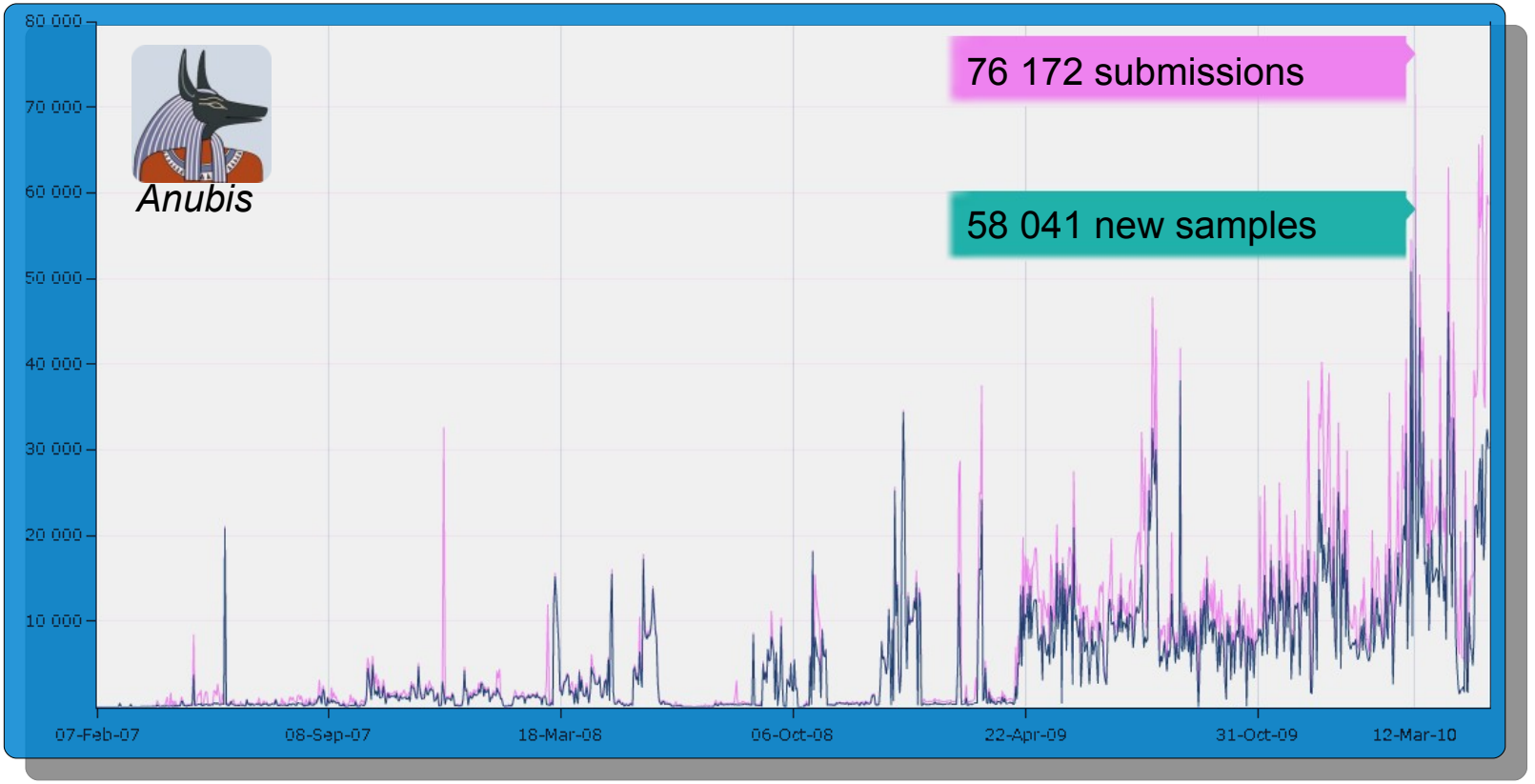


Motivation

- Analysis of malicious code is challenging
- Looking at the inner workings of every samples has become infeasible
  - … due to various *obfuscation* techniques
  - … due to *analysis resistance* (e.g., *anti-debugging techniques)*
  - … due to the huge *number* of malware *families / variants*

Motivation

- Analysis of malicious code is challenging

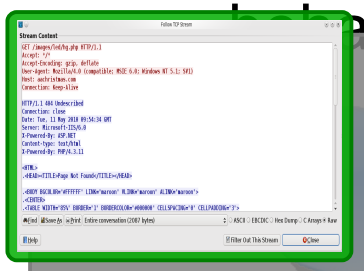


Motivation

- Analysis of malicious code is challenging
- Looking at the inner workings of every samples has become infeasible
  - … due to various *obfuscation* techniques
  - … due to *analysis resistance* (e.g., *anti-debugging techniques)*
  - … due to the huge *number* of malware *families / variants*
- Results of dynamic analys is cluttered by other behavior sample is capable of

Motivation

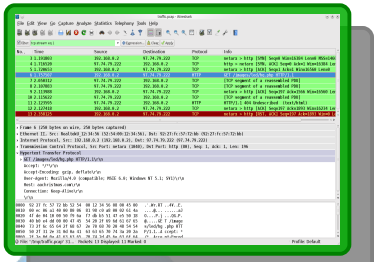
- Results of dynamic analys is cluttered by other behavior sample is capable of



binary update

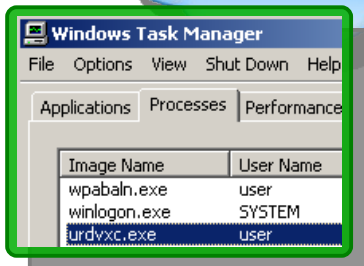
armzasn.net
kevflnwroo.com
dzqbpieiy.info
rqkixea.biz
komug.net
vhiax.org

C & C location



component installation

C & C communication



target selection

spam templating

```
220 mx.google.com
250-google
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-STARTTLS
```

Motivation

- Results of dynamic analysis cluttered by other behavior sample is capable of
- Dynamic analysis is very resource consuming...
- … and only provides temporary snapshot
  - malicious behavior might dependent on
    - analysis date & time
    - analysis environment (e.g., username, host OS, …)
    - availability of remote resources (e.g., C&C hosts)
  - needs to be repeatedly performed on single sample
    - at different points in time
    - preferably on different systems
    - even more time/resource consuming

Motivation – Inspector

Wouldn't it be cool if we were able to extract a single behavior into a standalone component and use this to re-invoke the behavior?

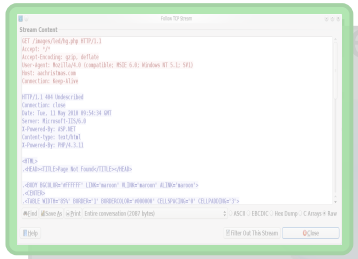
- Removes clutter from analysis results
- Independent of other malicious activity
  - can be executed without virtual environment
- Easy to replay in a different situation such as
  - point in time
  - operating system

Motivating Example

- *Conficker* Domain Generation Algorithm (DGA)
  - decides which remote host to contact for C&C
  - domain depends on *current time*
  - current time is fetched from a remote host (e.g., msn)

Motivating Example

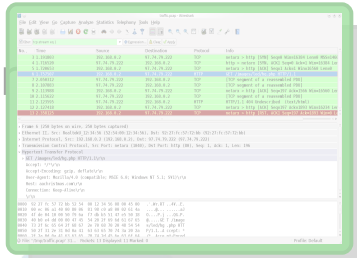
- *Conficker* Domain Generation Algorithm (DGA)

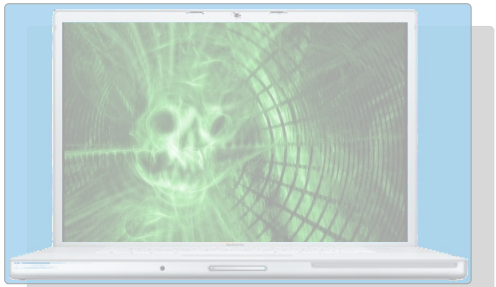


binary update

armzasn.net
kevflnwroo.com
dzqbpieiy.info
rqkixea.biz
komug.net
vhiax.org

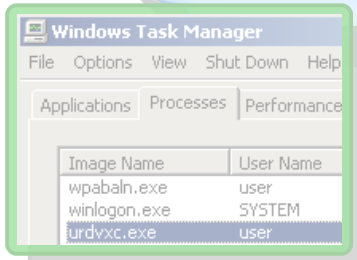
C & C location





component installation

C & C communication



target selection

spam templating

```
220 mx.google.com
250-google
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-STARTTLS
```

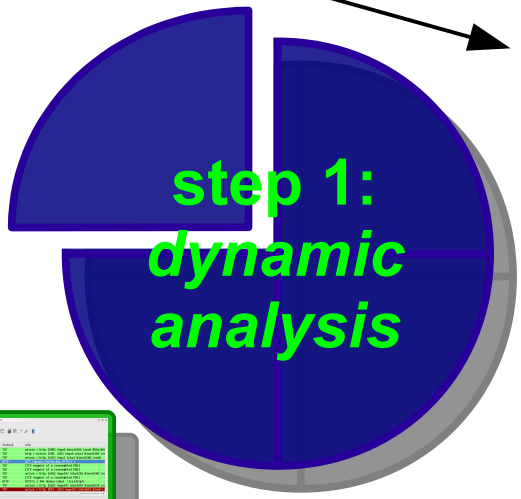
Outline

- Motivation
  - dynamic analysis reveals limited, temporary behavior
- Behavior analysis & extraction
  - storing identified behavior into *gadget*
- Behavior re-invocation
  - gadget player
  - gadget inversion
- So... again, why...?
  - benefit recap
- Gadget examples

Behavior Analysis and Extraction

Extraction Overview

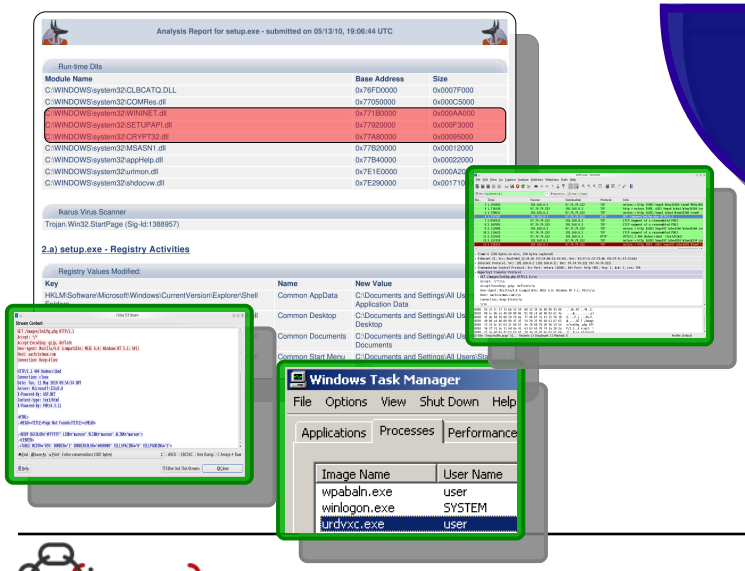
Extraction Overview



control flow & instructions

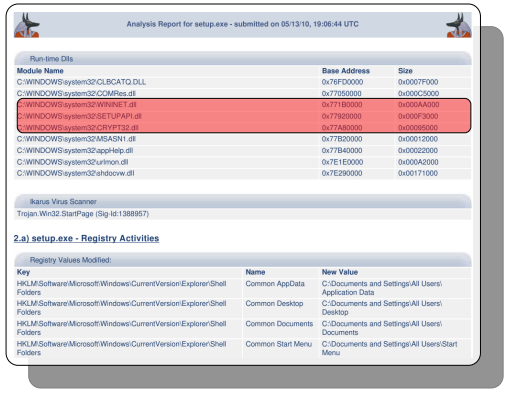
API taint dependencies

memory accesses

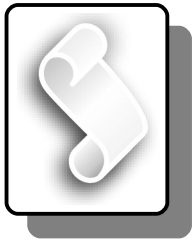


Find interesting behavior that is to be extracted.
Example: *Hm, to which domain am I connecting here??*

Extraction Overview



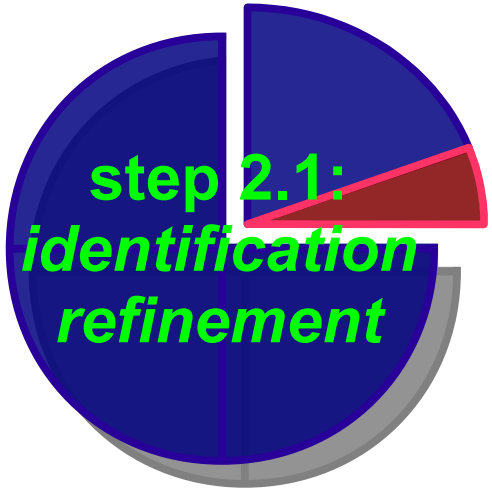
step 2:
behavior identification

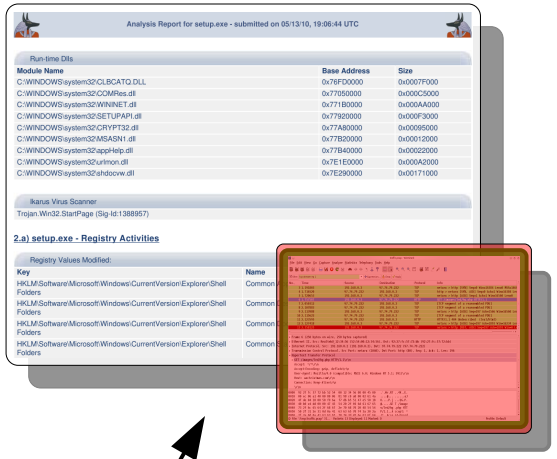


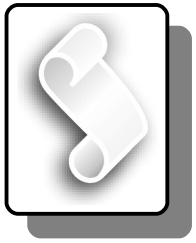
control flow & instructions

Map selected behavior to analyzed process & thread, API accesses and control flow.
outcome: API call / flow position

Extraction Overview



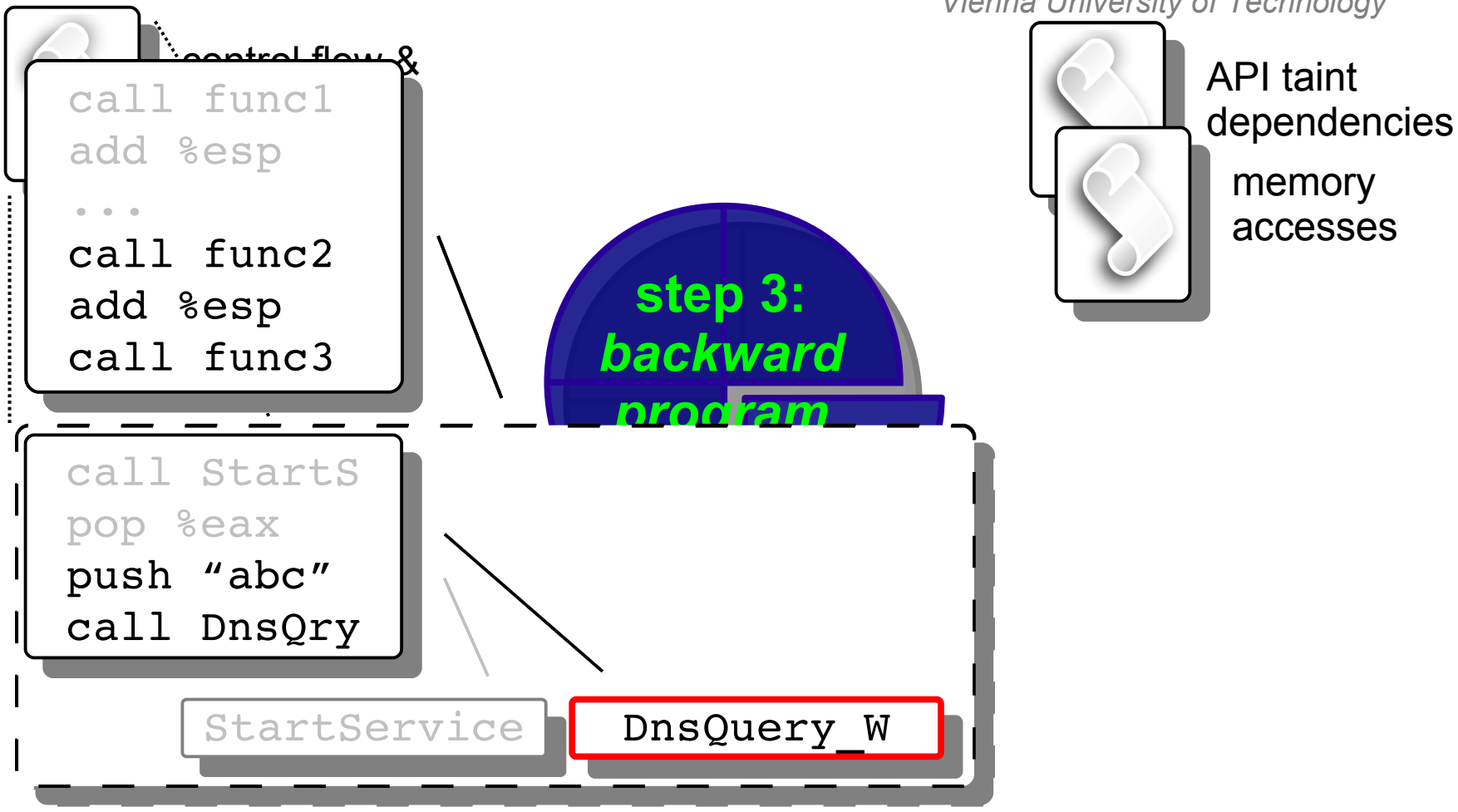




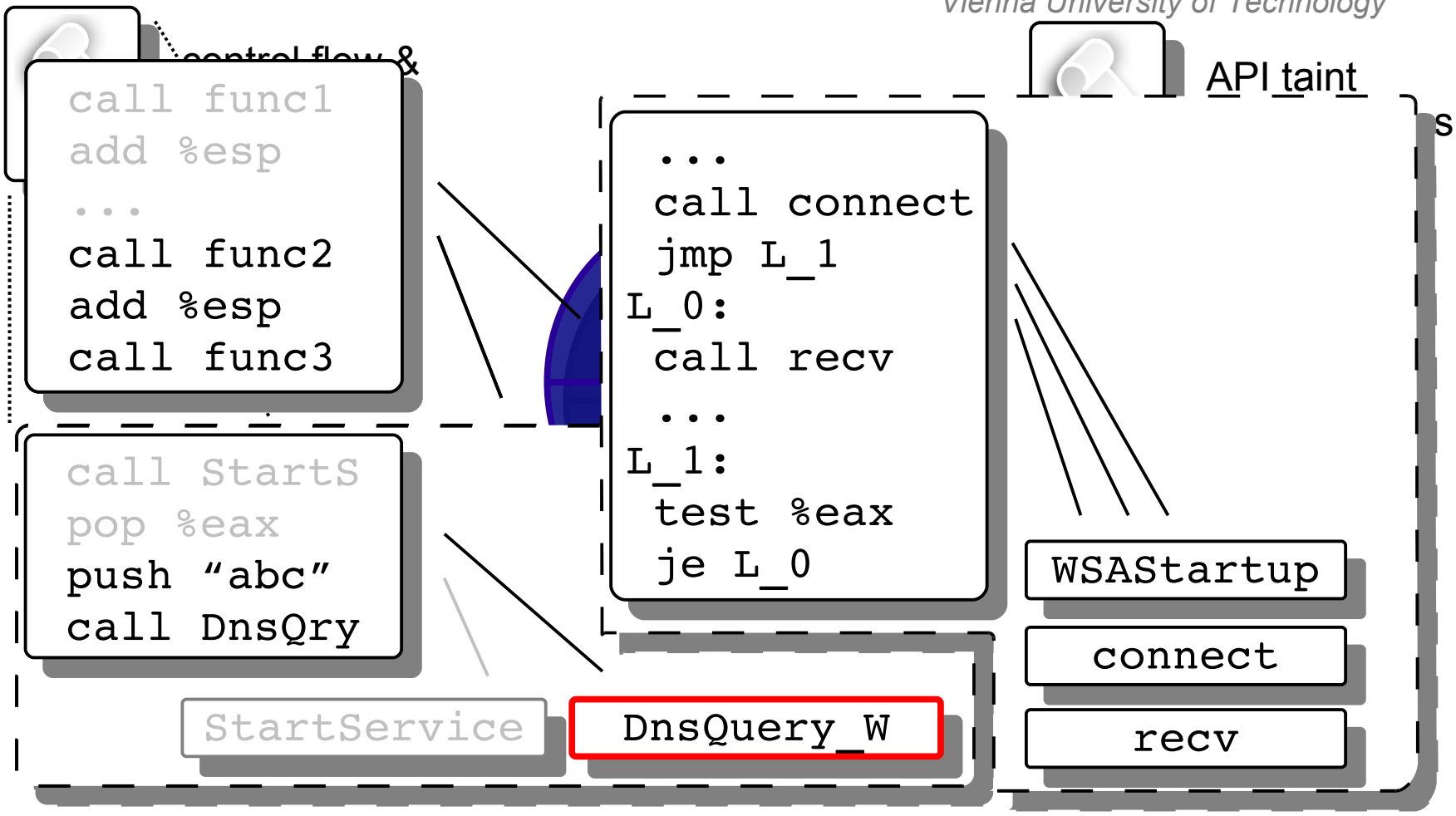
control flow &
instructions

Find and suggest data manipulating instructions after chosen API call. Possibly refine chosen position to include the data processing.

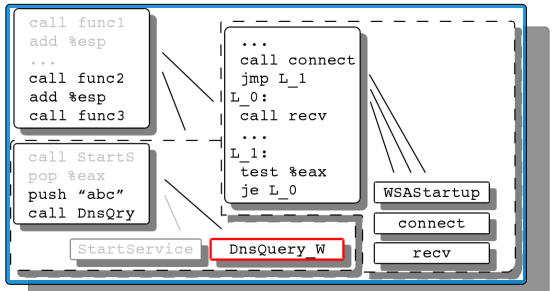
Extraction Overview

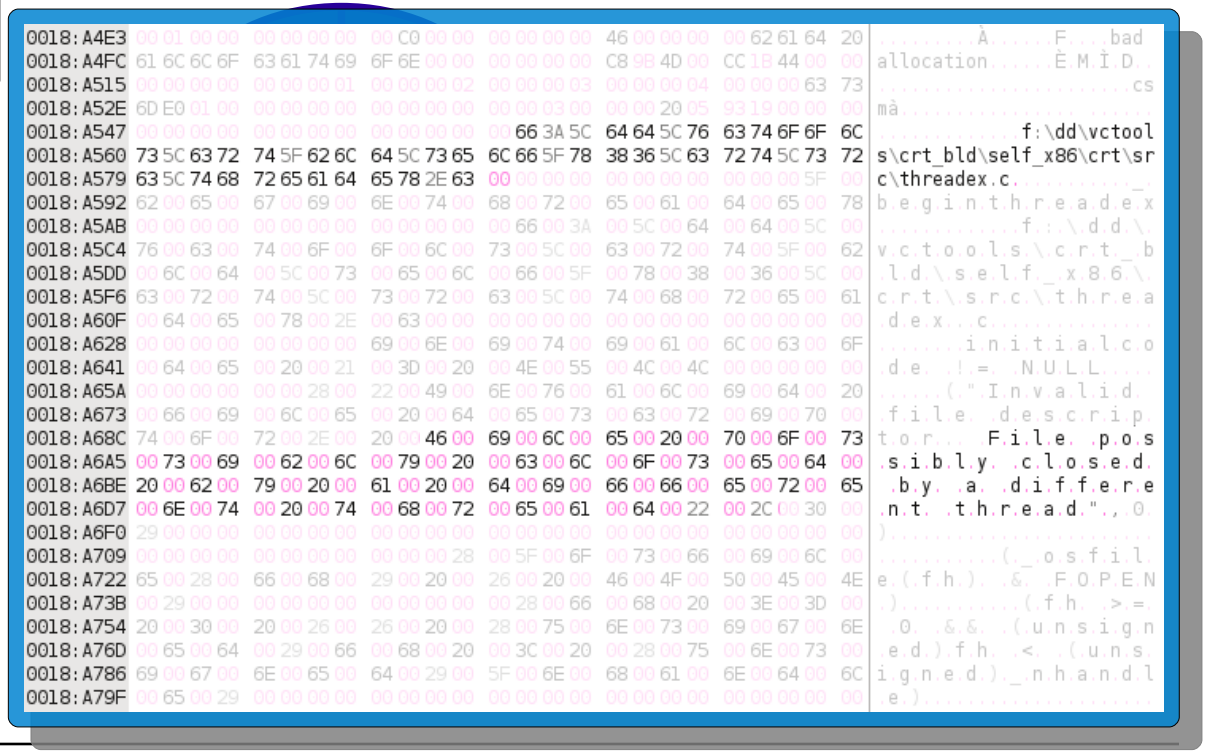


Extraction Overview

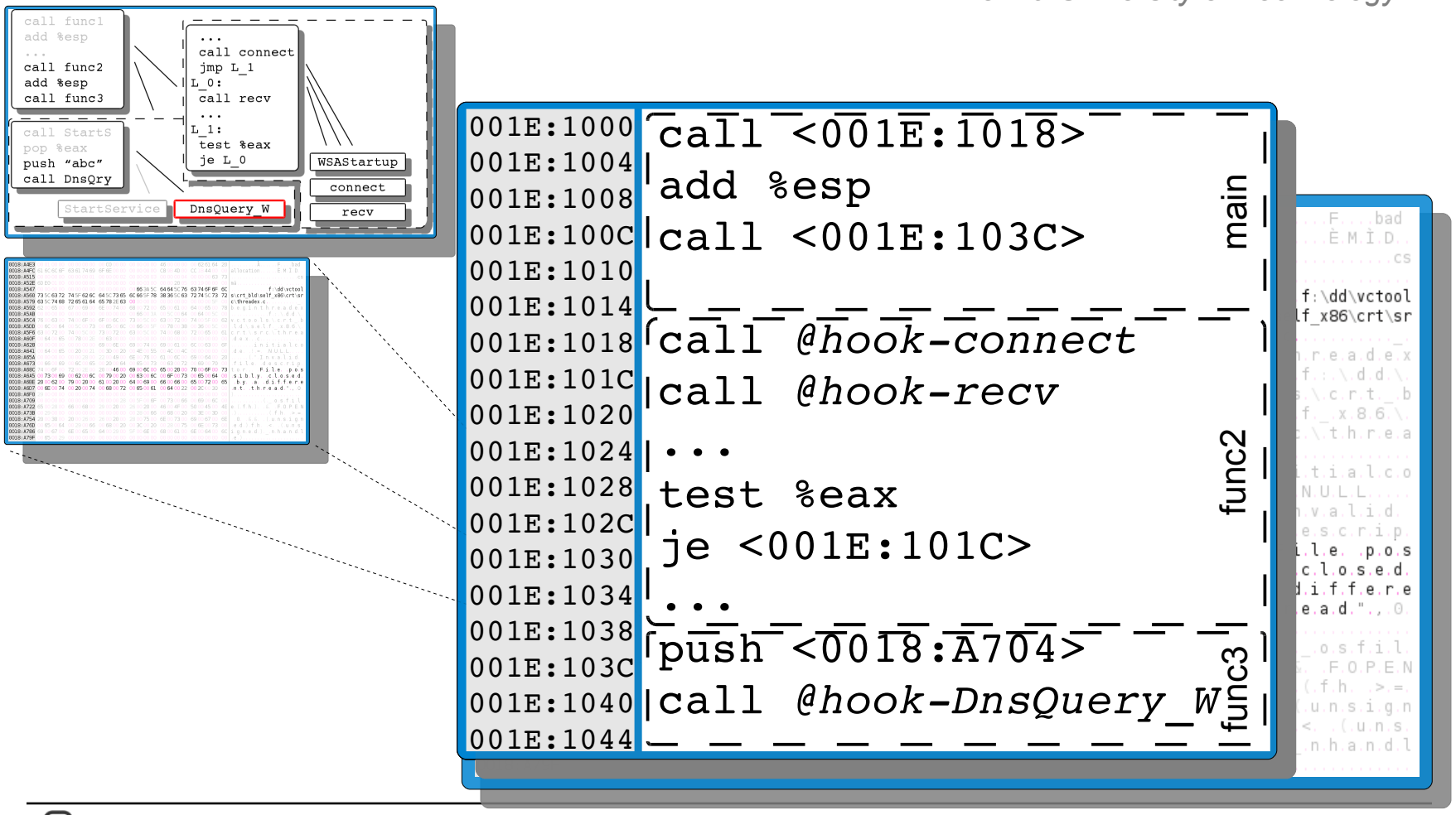


Extraction Overview

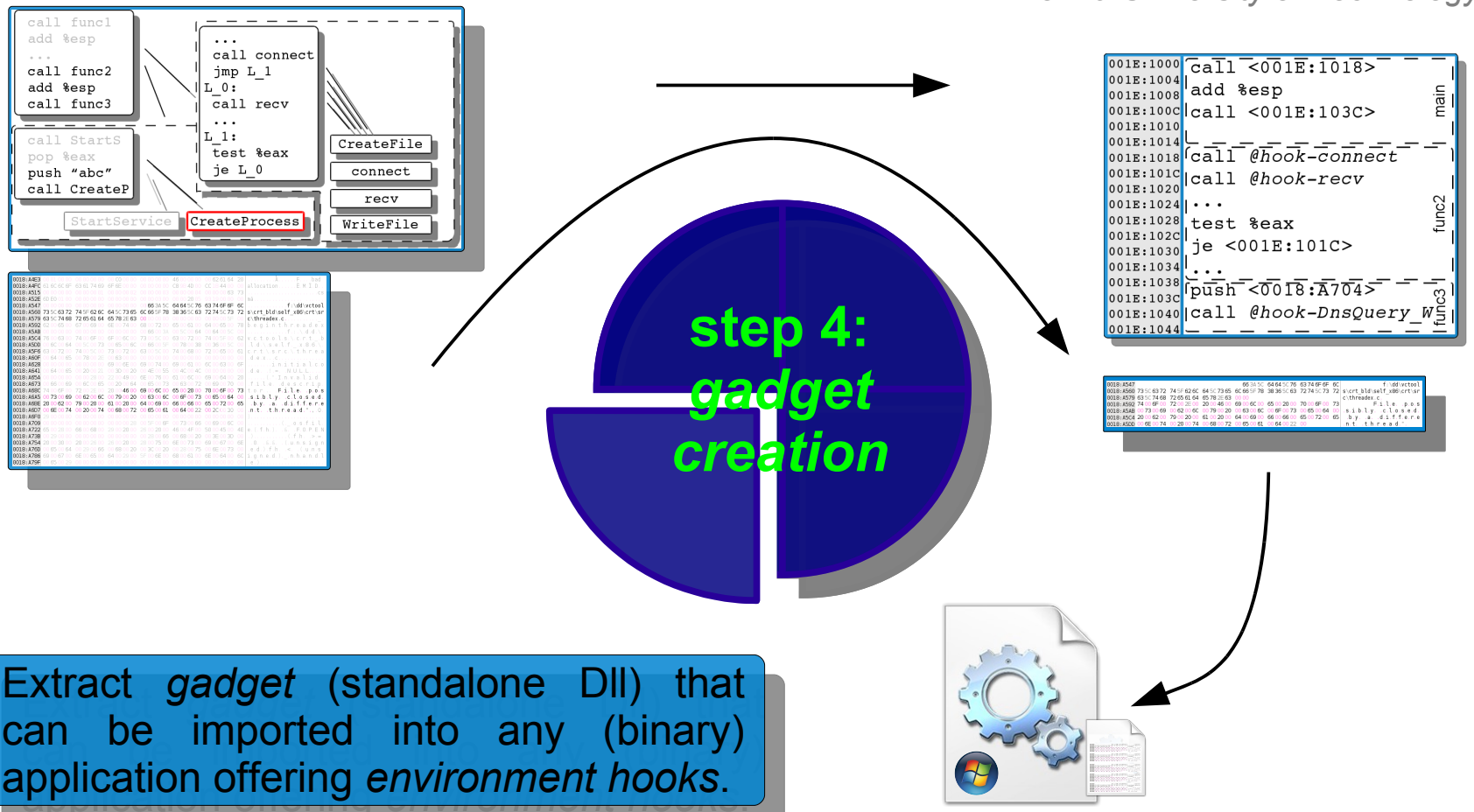




Extraction Overview



Extraction Overview



Extract *gadget* (standalone Dll) that can be imported into any (binary) application offering *environment hooks*.

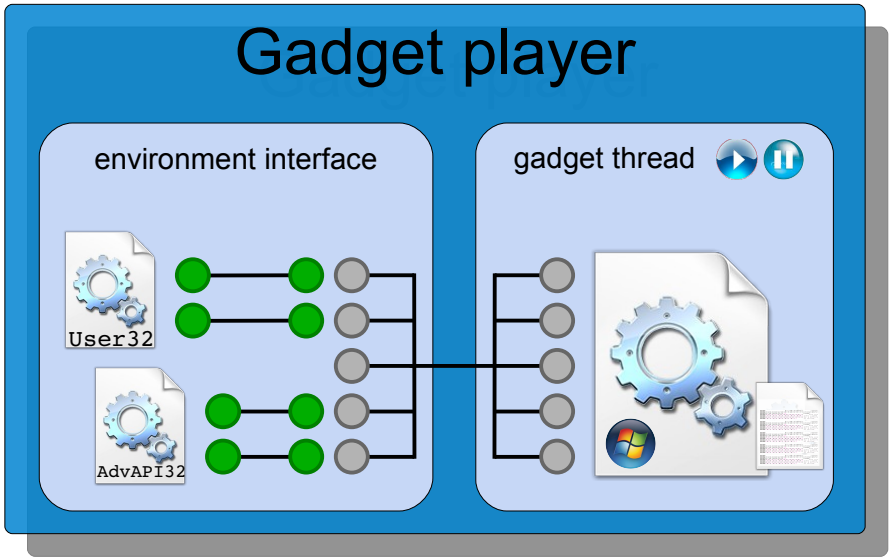
Gadget Replay

Gadget Player

- As library, the gadget can be reused in many areas
  - statically linked into the application
  - dynamically loadable
- … but, application must confine gadget execution
  - handle crashes (e.g., possible, invalid memory accesses)
  - one possibility: code emulation
  - here: separate, monitored thread with signal handling
- … and mediate accesses to the *host OS*
  - gadgets are guaranteed to contain no calls to system or API functionality directly
  - each access is done through *environment hooks*

Gadget Player

- Host OS accesses mediation: *environment hooks*
  - every system / API call is redirected to the gadget player (using a multiplexor function)
  - player has the possibility to *sanitize* and/or *manipulate* call parameters
  - if player decides to allow the API invocation, call and parameters are *forwarded* to the actual implementation (e.g., inside a Windows library)



Gadget Inversion

- Player can use gadget as *transformation oracle*
  - *input* is transformed into *output*, depending on algorithm implemented by gadget
  - example: sample reads local data, obfuscates, and transmits to remote host

Gadget Inversion

- Player can use gadget as *transformation oracle*
  - *input* is transformed into *output*, depending on algorithm implemented by gadget
  - example: sample reads local data, obfuscates, and transmits to remote host
- In many scenarios, the inverse algorithm would be interesting, however
  - we capture obfuscated traffic and want the plain-text data that has was transmitted

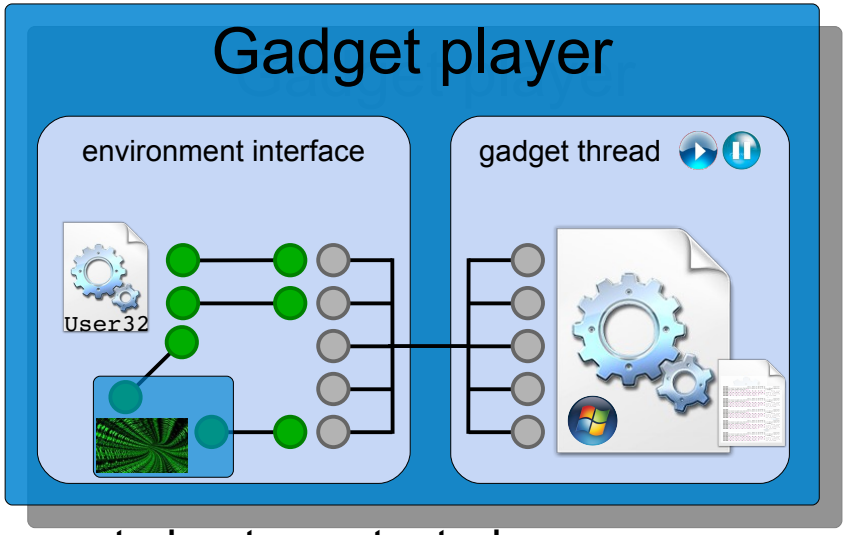
Gadget Inversion

- Player can use gadget as *transformation oracle*
  - *input* is transformed into *output*, depending on algorithm implemented by gadget
  - example: sample reads local data, obfuscates, and transmits to remote host
- In many scenarios, the inverse algorithm would be interesting, however
  - we capture obfuscated traffic and want the plain-text data that has was transmitted
- In the paper, we present basic inversion capabilities:
  - *detect* (taint) *dependencies* between bytes in input and output
  - apply guided *brute-force* heuristics to invert algorithm contained in the gadget

So … again, why...?

Gadget Benefits

- … so why not simply execute it in a VM (over and over again)?
  - *sleep timeouts:* can be eliminated during gadget extraction
  - *fast & lightweight analysis:*
    - no virtual environment, snapshot restoring
    - *we ran our analysis on Linux (under Wine)!*
  - *precise, uncluttered behavior observation*
  - *advanced monitoring:* the player has access to the gadget's heap and stack regions!
  - *environment tampering:* all requests go through a single interface: tamper with date or time, registry settings, hostOS, remote hosts contacted, ...

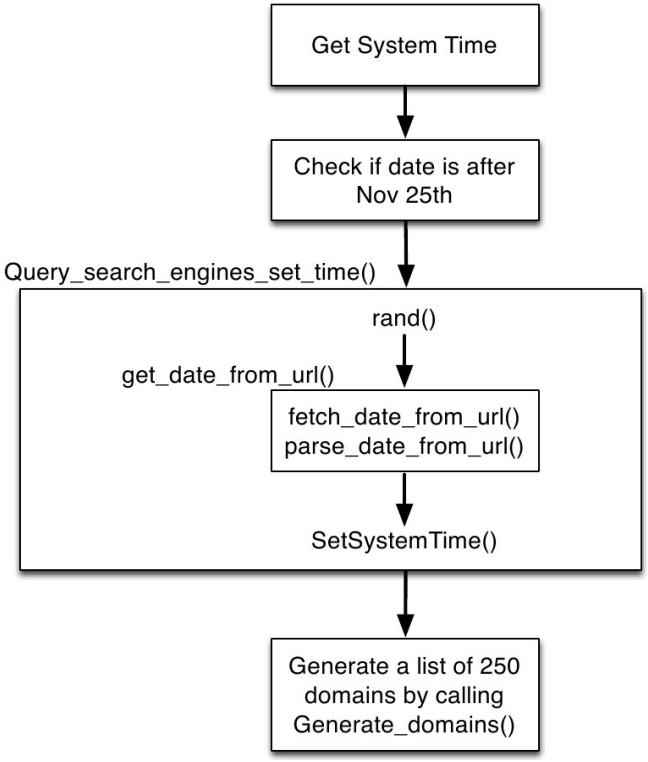


Gadget examples

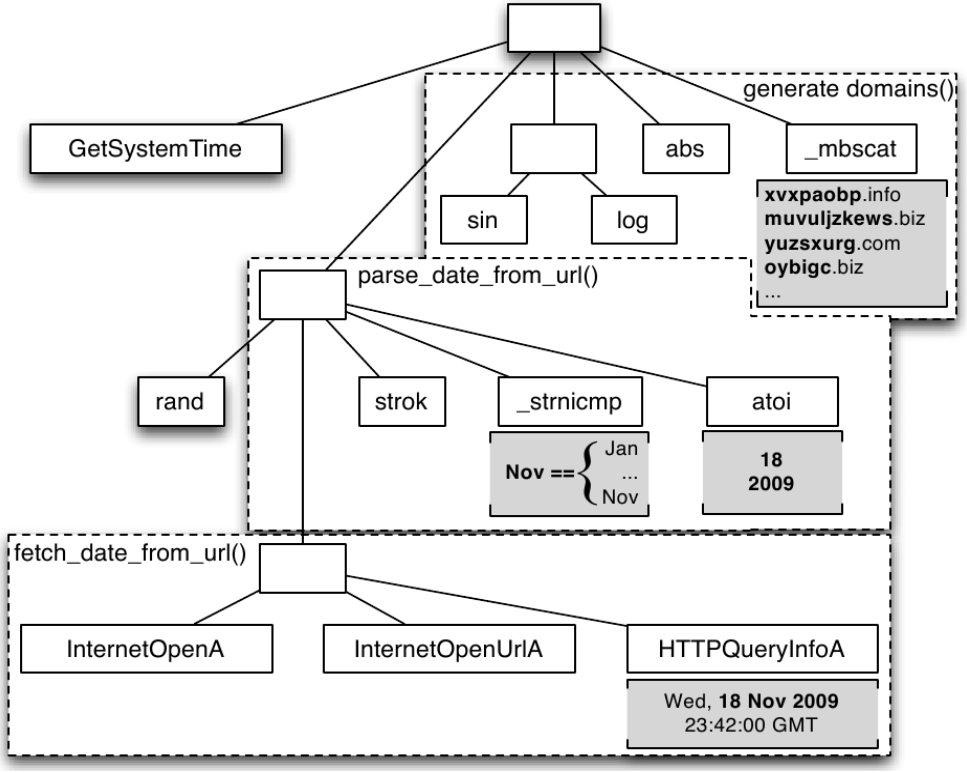
Conficker DGA

- *Conficker* generates (pseudo) random domain names upon startup
- Current time, fetched from remote site (e.g., msn.com), controls domain generation randomization seed
- Randomly selected domain name is used for contacting C&C host
- Gadget:
  - start extraction (slicing) from invocation of `DnsQuery_W`
  - extracts complete *Domain Generation Algorithm* (DGA)
  - see *one domain* on query invocation
  - find *all domains* on gadget heap
- Possible environment tampering:
  - manipulate remote site's reply to change DGA input (i.e., date for which domains are generated)

Conficker DGA



P. Porras et al. “A Foray into Conficker's Logic and Rendezvous Points



Inspector Gadget control flow *“debug output”*

Cutwail Spam Templating

- *Cutwail* (mass-mailer) generates spam Emails from *templates* downloaded from remote C&C hosts
- Communication employs proprietary encryption algorithm
- Template is not stored on file system
  - content decrypted and handled solely in memory
- Gadget:
  - inspect download behavior
  - start extraction after download is complete
  - *Inspector* suggests to automatically refine extraction starting-point to *end-of-decryption*
  - extract complete template download & decryption algorithm

Cutwail Spam Templating


```
"{_FIRSTNAME} {_LASTNAME}" <{MAIL_FROM}>

Hello my new friend, I search a good man at other
country...\n For me it to communicate for the first
time with the person from other country, by
Internet.\nAnd it
...

{nReceived}
Message-ID: <{DIGIT[10]}.{SYMBOL[8]} {DIGIT[6]} @{nHOST}>
From:        {TAGMAILFROM}
To:         <{MAIL_TO}>
Subject:     {SUBJECT}
Date:        {DATE}
MIME-Version: 1.0
Content-Type: multipart/mixed;
boundary="----=_NextPart_000_0006_{_nOutlook_Boundary}"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express {_nOutlookExpress_4}
```

Cutwail Spam Templating


```
configver          194
addr               91.206.231.230
port               25
knockdelay         60
mxrecvtimeout      120
mxconntimeout      120
maxtrybadfrom      1
maxtryconn         5
```

```
nHOST
{nChar[5-15]}.{nChar[5-15]}
    .{LET:ru,org,com,va,net,biz,
          info,tv,ua,su}

nReceived
Received: from [{nIP}]
          (helo={nHOST})
          {N[1]}by {BOT_HOST}
          ...
```

```
FIRSTNAME          Christi
                   Lea
                   Staci
                   Jodie
                   Summer
                   Katharine
                   ...

LASTNAME           Schafer
                   Stacy
                   Grayson
                   Ham
                   Landers
                   Mims
                   Parham
                   Pritchett
                   ...

name               Lusia R., Texas
                   Lusia R., New York
                   Lusia R., Chicago
                   Lusia R., Colorado
                   Lusia R., Boston
                   Lusia R., Washington
                   Lusia R., Las Vegas
                   Lusia R., Bellevue WA
                   Lusia R., San Diego
                   Amelia B., Chicago
                   ...
```

URLZone Config Update

- *URLZone* (BHO banking-trojan) sniffs and manipulates user interaction with banking web-site
- Steals credentials and hides previous (malicious) transactions from user
- Remote configuration through encrypted configuration files
  - domains to attack
  - URLs to inspect
  - form content to modify
- Gadget:
  - extract complete template download & decrypt algorithm
  - similar to *Cutwail* gadget

URLZone Config Update


```
======================POST======================
[ITBEGINBLOCKHOOK]
ITHOST=|banking.postbank.de|End
ITPAGE=|/app/login.d*|End
ITMETHOD=|2|End
ITIFINIT=|%DISP%|End
ITREQMATH=|jsOn=*&accountNumber=*&pinNumber=*|End
```

```
----------------- STATA -----------------
ITINJHOST=|my.hypovereinsbank.de|End
ITINJPAGE=|/*?view=/*|End
...
ITINJSTART=|Aktueller Kontosaldo</label>[*]
           <p class="right">|End
ITINJEND=|</p>|End
ITINJCODE=||End
ITINJPASTE=|%HYPOBAL%+%AMOUNT%-%TRUEAMOUNT%|End
ITINJPASTEMN=|<span
class="negative-balance">%HYPOBAL%+%AMOUNT%-
           %TRUEAMOUNT%</span><span
class="negative-balance">EUR</span>|End
```

Summary

- Dynamic analysis is resource consuming, results are cluttered and limited to temporary snapshots of malicious behavior
- *Inspector* allows to automatically extract behavior into standalone *gadgets*
- Gadgets can be reused in many scenarios and
  - enhance information extraction
  - simplify repeated analysis of behavior
- Evaluation shows that extraction is applicable to real world, malicious programs

Thanks for listening!