Targeted attacks through ISPs

Denis Legezo
Kaspersky
Senior security researcher

kaspersky

Plan for the next 30 minutes

How to listen encrypted traffic: Malware with really nice technical trick

Why we decided ISPs could be involved in the case

StrongPity: Another one with ISP usage is still active and deliver

Governmental data center in Asia: Way to country-level waterholing

All this started with good old known COMpfun

	Initial infection	Escalation, detection	Reductor RAT
Malware	COMpfun trojan	Reductor dropper-decryptor	Reductor trojan
Process	One of the browsers	Same browser	lsass.exe
Persistence	COM CLSID hijacking	Auxiliary module, N/A	LSA notification package
Net encryption	AES 128	Local module, N/A	AES 128
Host encryption	Configuration data under constant one byte XOR + LZNT1	Reductor in resources under constant one byte XOR + LZNT1	Victims' unique IDs in TLS "client hello" under XOR with changing key

TLS could be a problem for malefactors

Keylogging? May be too loud with current security solutions.

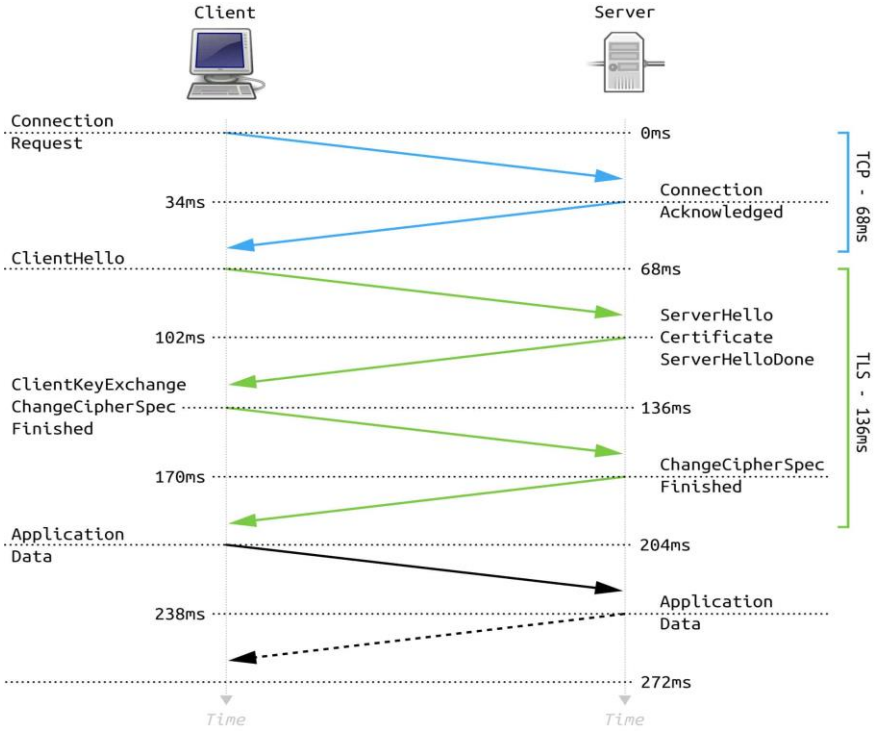
Decrypting? May be not in reasonable time with current TLS.

Certificates pre-installation? Better, could facilitate MITM attack. But what about NAT?

Certificates pre-installation plus marker for packets of interest? Could be next step forward, but how - NDIS proxy? Too loud again

But there is the way to mark TLS session without even single touch of network packets

The idea is to use “client hello” field



Full TLS 1.2 Handshake by FleshGrinder. Licensed under CC0.

Pseudo-random number generator fill the field

nss3.dll	PK11_GenerateRandom()	Call original PRNG function and generate initial XOR key from its result. Change PRNG result: set seventh byte to 1, then save 0x45F2837D, hwid and cert hashes. Encrypt the result and return it instead of the original PRN. It would affect calls to ssl3_SendClientHello() -> ssl3_GetNewRandom(ss->ssl3.hs.client_random);
advapi32.dll	CryptGenRandom()	Spoof these system PRNG functions result in quite similar way with some minor changes;
bcrypt.dll	BCryptGenRandom()	
chrome.dll	PRNG function	Find PRNG function by its binary code template and patch it like all aforementioned;

They analyzed Firefox source code and Chrome binary

To patch browsers' PRNG functions in memory and add unique user IDs into TLS handshake developers have to analyze Firefox and Chrome code

```
static SECStatus
ssl3_GetNewRandom(SSL3Random random)
{
    SECStatus rv;

    rv = PK11_GenerateRandom(random, SSL3_RANDOM_LENGTH);
    if (rv != SECSuccess) {
        ssl_MapLowLevelError(SSL_ERROR_GENERATE_RANDOM_FAILURE);
    }
    return rv;
}
```

```
/* Generate a new random if this is the first attempt. */
if (type == client_hello_initial) {
    rv = ssl3_GetNewRandom(ss->ssl3.hs.client_random);
    if (rv != SECSuccess) {
        goto loser; /* err set by GetNewRandom. */
    }
}

if (ss->vrange.max >= SSL_LIBRARY_VERSION_TLS_1_3) {
    rv = tls13_SetupClientHello(ss, type);
    if (rv != SECSuccess) {
        goto loser;
    }
}
```

Quite silent certs installed and traffic marked

```
struct client_hello_system_fingerprint {
      DWORD initial_xor_key; // First four bytes generated by original system
PRNG function
      DWORD predefined_const; // Set to 0x45F2837D
      DWORD cert_hash; // Reductor's digital certificates hash
      DWORD hwid_hash // Target's hardware hash
};
```

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            fa:9b:b7:53:21:86:97:bd:ed:1a:8c:85:59:fb:f6:94
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C = EN, CN = GeoTrust Rsa CA, O = GeoTrust Rsa CA
        Validity
            Not Before: Oct 23 22:56:10 2011 GMT
            Not After : Nov 17 22:56:10 2031 GMT
        Subject: C = EN, CN = GeoTrust Rsa CA, O = GeoTrust Rsa CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:d1:02:fa:c5:94:71:f2:45:4e:80:b9:ee:08:61:
                    ed:6b:c6:2c:3a:df:c7:99:48:a7:4c:ab:64:31:22:
```

“UAC is useless” and compfun[.]net

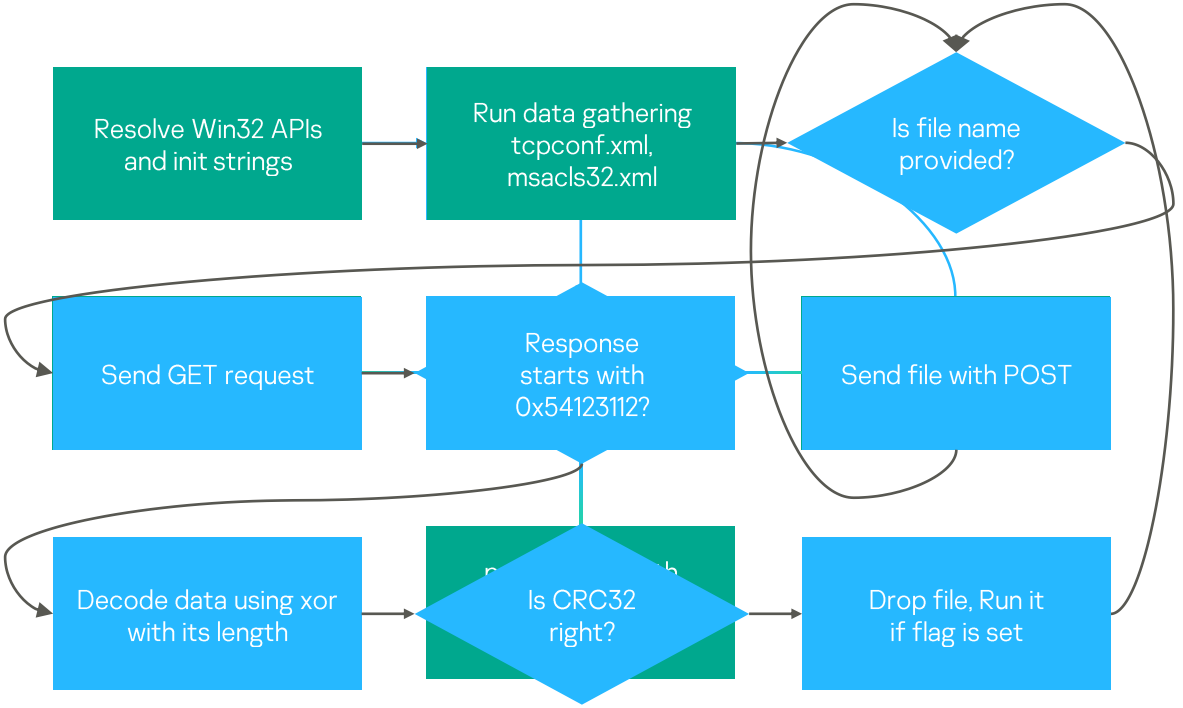
```
func_result = p_advapi32_addr->CreateProcessWithLogonW(
                uac_user_wide,
                is_domain_wide,
                useless_password_wide,
                LOGON_NETCREDENTIALS_ONLY,
                app_name,
                0i64,
                0,
                0i64,
                0i64,
                &startup_info,
                &process_info);
```

Something in the way

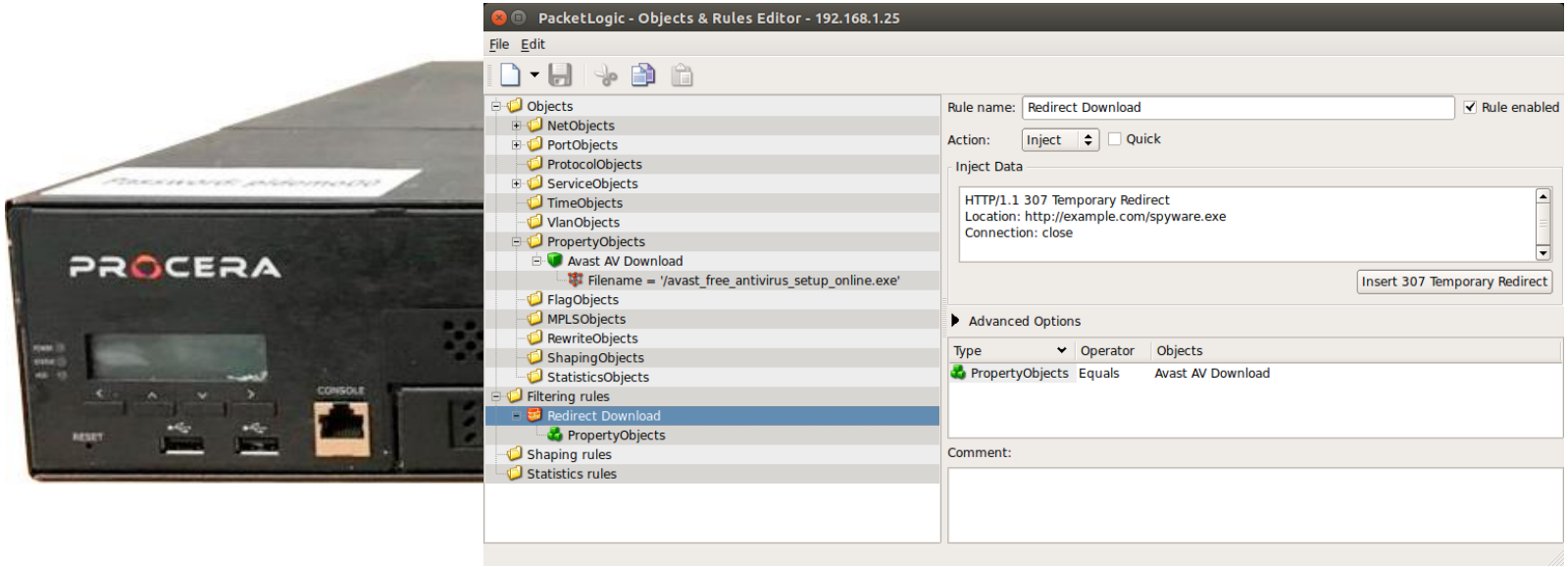
In July our telemetry show new URLs and that time installers were available on the warez web-site. Available and uninfected

2019-07-11 06:49:33	http://dl1.sarzamindownload.com/sdlftpuser/91/09/01/Windows.8.Activator_CMD.exe
2019-07-11 06:49:33	http://dl1.sarzamindownload.com/sdlftpuser/91/09/01/Windows.8.Activator_Blakeymort_4.0.1.5.exe
2019-07-11 06:49:33	http://dl1.sarzamindownload.com/sdlftpuser/91/09/01/Windows.8.Activator_Offline_Build_121105.exe

StrongPity still active in September

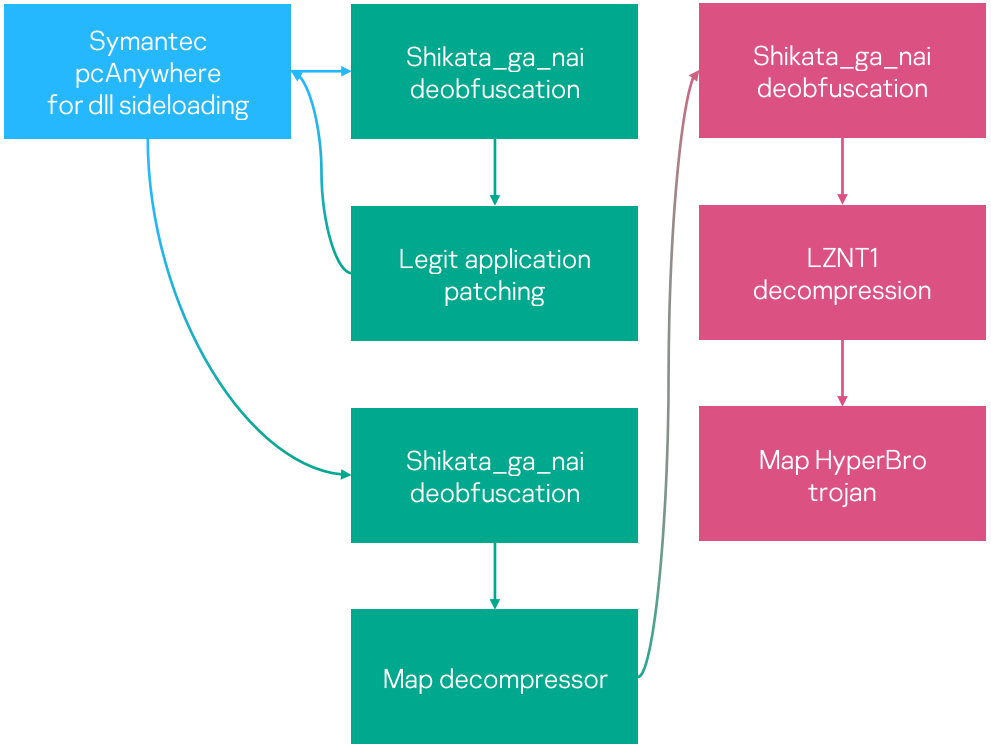


“Server” side of StrongPity



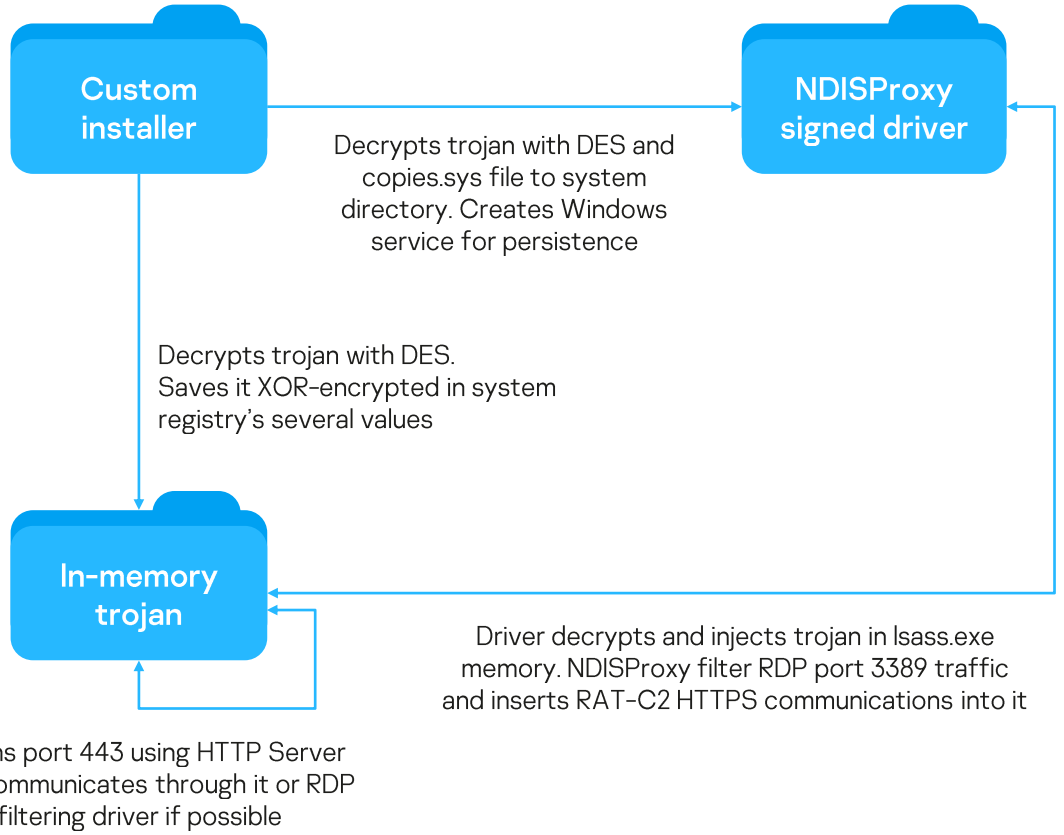
Illustrations and experiment by
citizenlab.ca

Governmental data center as perfect target



Anti-detection stages. Different colors show the three dropped modules: legit app (blue), and decompressor with the Trojan embedded (red).

BTW, these guys choose NDIS way to gather traffic



Takeaways

Indirect attack: You don't have to be the final target to suffer from malware

Sometime they are real engineers: Malicious tasks are typical, but methods vary greatly

Sometime they have leverage: Access to network channel most probably means ISP involvement

Sometime they just know the point to hit: Country level data center as initial infection point was quite an idea

Thank you!

If you are looking at this last slide, you are already a hero!

kaspersky