Methods for Accessing a Link Key

Foreword

ellisy

Better Analysis

Since the advent of Simple Secure Pairing (SSP), Bluetooth engineers have faced a new challenge in debugging Bluetooth devices. SSP simplifies the end user's pairing experience while providing greater security for the connection, but it inherently presents obstacles to Bluetooth engineers seeking to decrypt analyzer-captured Bluetooth traffic (of course, these obstacles are intentionally present for anyone seeking to do such decryption).

At the center of the challenge is gaining access to the link key. This document will explore some of the methods available to locate and access this key. The <u>Ellisys Bluetooth Explorer 400 Analyzer</u> is uniquely capable of capturing encrypted Bluetooth traffic without *a priori* knowledge of the link key. Encrypted traffic captured using the Ellisys analyzer can be decrypted at a later time when the link key becomes available, and therefore fully understood by the Bluetooth engineer.

The link key is a 128-bit random number and is a "shared secret" between devices; it is never transmitted over the air. It is created during the pairing process, is stored and kept secret, and is used at every connection for authentication and derivation of the encryption key. Even a portion of the pairing sequence, where the link key is created, is encrypted.

The link key is just one of the parameters used to generate the encryption key. See <u>EEN BT07 - Secure Simple Pairing</u> <u>Explained</u> for details on how SSP works.

Since the link key is never transmitted over the air, the Bluetooth engineer must be aware of the various methods available to access this key. This Expert Note will cover some of these methods. Note that for Bluetooth Smart (Bluetooth Low Energy), things work a bit differently, in that there is in fact an over-the-air transmission of a critical key – see the section on Bluetooth Smart (Low Energy) below.



Key Concepts

Below and right, the Ellisys Bluetooth Explorer software is used to illustrate some key concepts. In the illustration, we see three devices (a phone, headset, and a PC) comprising two connected piconets (one scatternet). The PC is master to the headset, and the headset is dual-role: master to the phone as well as slave to the PC.

The connection between the headset and the phone has been fully decrypted, as the link key has been obtained (as indicated in the Security pane). <u>The connection from the headset to the PC however is still encrypted</u>, because the link key has not yet been obtained.



The link key between the phone and the headset was obtained using a vendor-specific method and is described in the Android stack section in Vendor-specific methods, described later in this document.

Note that the analyzer is actually capturing the encrypted traffic (PC to headset) and displaying this on a single line. This line will expand into decrypted traffic once the key is provided to the software, either during the capture or after it is saved. See the section on Vendor-specific methods later in this document for details on how this (PC to headset) encrypted traffic is decrypted.

| | chucks ph | one headset and | computer.btt* - Ellisys | Bluetooth A | nalyzer | _ 🗖 > |
|--------------------------------------|--|--------------------------|----------------------------------|---------------------------------------|-----------------------------------|------------------------------|
| e View Layout Search Rec | ord Tools Help | | | | | Full screen 🔲 Analysis 📑 Add |
| 🧉 😡 🙀 👫 🕨 Record 🔹 I | 🛛 Stop 🔄 Restart 🐻 Save & Continue 👹 🛔 | 🛛 🖉 Navigate 🕞 🖳 | 🖱 Markers 🔹 🧢 🕒 🏹 Filterin | g: Exclude Backgro | ound 👻 🌏 | |
| Welcome BR/EDR Overview | Low Energy Overview HCI Overview (Serial) | HCI Overview (USB) | Message Log Instant Spectr | um 🛛 🗘 Þ | Instant Piconet | ą |
| ocol: Single selection - All layers | + + + == 🛶 💡 🔎 🖶 🐄 🕒 🚸 🎝 💼 | i 💰 🛍 🕕 💷 🛍 | 🍈 🕥 🍰 兰 38 items kept, 110 | filtered 🛛 😰 🖕 | | |
| Type filter V - Imp,12 | | - | Type filter | ▼ • Type filter | | |
| Time Item | | | Communication | Payloac ^ | | |
| 6.483 426 000 🕀 📲 ACL-U Flo | w Stop | | Master: PC <-> Slave: Headset | No data | | |
| | 1 Traffic (x 1'688, 25.7 s) | (| Master: PC <-> Slave: Headset | 8 bytes | | |
| | adset" 00:24:1C:C1:D7:97 > "Phone" 94:01:C2:63:D7 | 05, responded, 253 ms) | Master: Headset <-> Slave: Phone | 00,000 | | , vals |
| 7.734 052 625 🕀 🔎 SDP Service | Search Attribute Transfer (Hands-Free Audio Gateway | Hands-Free Audio Ga | Master: Headset <-> Slave: Phone | | | 191 |
| | onnect (Channel=Signaling) | | Master: Headset <-> Slave: Phone | | 156 bytesis | 6.57 VBI* PC |
| 7.882 802 000 🗉 🙀 RFCOMM D | LC Parameter Negotiation (Channel=3, Initial Credits=F | .: 7 I: 7) | Master: Headset <-> Slave: Phone | | | Headset |
| 7.900 302 000 🗉 👰 RFCOMM C | onnect (Channel=3) | | Master: Headset <-> Slave: Phone | | 379 54 | |
| 7.905 302 000 🛛 🕀 🙀 RFCOMM M | odem Status (Channel=3, Data Valid=Yes Yes) | | Master: Headset <-> Slave: Phone | | Phone | |
| 8.044 679 625 🛛 🕀 🙀 RFCOMM M | odem Status (Channel=3, Data Valid=Yes Yes) | | Master: Headset <-> Slave: Phone | | | |
| 8.047 802 125 🛛 🕀 🥵 AT HFP Sup | ported Features: AT+BRSF=159\r > \r\n > +BRSF: 871 | /r/n > /r/n > OK/r/n | Master: Headset <-> Slave: Phone | | | |
| | ailable Codecs: AT+BAC=2,1\r > \r\n > OK\r\n | | Master: Headset <-> Slave: Phone | | | |
| | Search Attribute Transaction (Hands-Free: Hands-Fre | | Master: Headset <-> Slave: Phone | | 🚱 H H H H H H 7.734 052 625 | AD: |
| | ator: AT+CIND=?\r > \r\n > +CIND: ("call",(0,1)),("calls | etup",(0-3)),("service", | Master: Headset <-> Slave: Phone | ~ | | |
| < | | | | > | 😪 Instant Piconet 🕙 Summary 🕍 Ins | tant Channels 101 Raw data |
| int Timing | | | | д Х | Details | ą |
| <u> </u> | | | Security | | | × |
| ess 📝 Fill missing fields | | | | | | Manage ECDH Keys |
| dset" Time | Master / Slave | PIN | Link Key | | ACO | IV (ay) |
| E8:2/ 8 6.469 675 000 | "PC" E8:2A:EA:B4:DE:7A "Headset" 00:24:1C:C1:D7:97 | Notapplicable | Missing | | Notapplicable | Not applic |
| ughpu gei 7.777 806 625 ∞ ∞ ∞∞ | "Headset" 00:24:1C:C1:D7:97 "Phone" 94:01:C2:63:D7:05 | Notapplicable | 169B6C41:CDAF9687:D | 1D912EE:EA525A48 | B C3962D4E:40E9D0C8:CE25F1B/ | Not applic |
| | | | | _ | Transaction ID | 0x0000 |
| 5DP Service Search At | | | | | 🖃 🔩 Parameters | |
| 0.02 0.03 0.04 0.05 | 7.683 s | 0.01 0 | 0.02 0.03 0.04 0.0 | 5 0.06 | 🖃 🔩 ServiceSearchPattern | |
| 1 bar | 0.00 0.00 0.00 | 7.70 s | 0.05 0.04 0.0 | · · · · · · · · · · · · · · · · · · · | Service Class 1 | Hands-Free Audio Gateway |
| | | | | | 🖃 🕂 AttributeIDList | |
| nstant Timing 🎝 Instant Audio 🛝 Ins | stant Throughput | | | | Attribute ID | Service Class ID List |

Standard methods

The following methods work independently of the Bluetooth radio or stack manufacturer used. However, these methods may require low-level access to the Bluetooth hardware and are therefore not always applicable with off-the-shelf products.



PIN-code-based pairing

LMP pairing, also known as PIN-code-based pairing or sometimes called Legacy Pairing, is not at all secure, but as with SSP, it also uses/creates a link key that must be known in order to decrypt the traffic. The Ellisys BEX400 analyzer software can automatically decipher the PIN code and deduce the link key, just by looking at the captured pairing traffic. PIN-code-based pairing is not used for Bluetooth 2.1 devices and higher (SSP-enabled devices), *unless these devices are pairing to a Bluetooth 2.0 or earlier device*. This means you can use the Ellisys analyzer to fully decrypt traffic from a default SSP-enabled device if you pair it with a Bluetooth 2.0 or earlier device, thus forcing the SSP-enabled device to revert to (crackable) PIN-code-based pairing.

Below is an example of a Bluetooth 2.1 device (or higher) pairing with a Bluetooth 2.0 device (or lower), where the pairing reverts to PIN-code pairing (as the Bluetooth 2.0 device has no concept of SSP). The BEX400 software has deciphered the Pin Code of 0000 and calculated the Link Key, with subsequent decryption of the traffic taking place automatically.

Note that if the user had entered the wrong PIN code, the Ellisys software would have advised of this error and indicated the correct PIN code to be entered. Easy.



SSP Debug Mode

SSP Debug Mode is a special mode used by SSP-enabled devices where a well-known Private/Public key pair is used instead of the normal Private/Public key pair. If at least one of the two devices is in SSP Debug Mode, the link key will be deduced automatically by the BEX400 analyzer, which recognizes the well-known Public key transmitted over-the-air. Again, only one of the two devices must be in Debug Mode, not both. If one of the two devices is in Debug Mode, then the analyzer will know its SSP Private Key and will thus be capable of computing the link key with the same algorithms used by each device.

© Ellisys 2014. Information contained herein is for illustrative purposes and is not intended in any way to be used as a design reference. Readers should refer to the latest technical specifications for specific design guidance. Ellisys – Chemin Du Grand Puits 38 – CH1217 Meyrin – Geneva +41 22 77 777 89 (USA +1 866-724 -9185) www.ellisys.com.



Note that some manufacturers have made the choice to not accept an SSP pairing with other devices that are in Debug Mode when their device is not in Debug Mode. This is a non-standard behavior which unfortunately diminishes the usefulness of Debug Mode, since in such cases both devices have to be put in Debug Mode.

SSP Private Key Injection

As explained above, SSP Debug Mode is just a way of using a known SSP Private Key. Even when not using SSP Debug Mode, if the Private Key is known then it can simply be provided to the protocol analyzer software by the user. By knowing the Private Key, the analyzer software is capable of automatically determining the link key whenever a pairing involving the known Private Key is detected. This provides great usability, since it avoids having to ever enter the link key manually.

This approach also provides an added level of security to the SSP Debug Mode, as only the analyzer configured with the specific SSP Private Key is capable of determining the link key; other sniffers in the area will not be able to decrypt the traffic.

This feature (entry of the Private Key) is provided by the Ellisys analyzer software using the "Manage SSP Key" feature, located in the Security pane, illustrated below.

| | Elliptic | c Curve Diffie-Hellman (ECDH) Key Pair Man | agement | × |
|--------------|---|---|--|-------------|
| Private Key | 985DEA8C6DDC430CEA8699C00349AFFDD33918 | SFCEE8A9C62 | re | verse |
| Public Key X | C7ABA18E:DF02E5AC:769DCF8F:26820E7B:F8 | BDAC81A: 32741816 | | |
| Public Key Y | 90EB4702:9C74C1C3:722D776C:67663896:ES | DCC585:AE6B746C | | |
| | | | | <u>A</u> dd |
| Private | | Public X | Public Y | |
| 985DEA8C:6 | DDC430C:EA8699C0:0349AFFD:D33918FC:EE8A9C62 | C7ABA18E:DF02E5AC:769DCF8F:26820E7B:F8DAC81A:32741816 | 90EB4702:9C74C1C3:722D776C:67663896:E9DCC585:AE6B7 | 46C |
| < | | | | > |
| | | | | <u>O</u> K |

HCI Capture

While the link key is not transmitted over the air, the good news is that it is transmitted between the radio chip and the host, over the host controller interface (HCI) and is accessible over this medium. Even better, the Ellisys analyzer can capture HCI traffic concurrently with the air traffic, hence, the analyzer has access to the link key this way.

Access to HCI is easily available on prototypes or developer kits (SDKs) used during development stages, but such access will likely not be available on most off-the-shelf devices. One exception is USB HCI, such as that used by Bluetooth

dongles. For example, using a PC with an attached USB dongle is an easy way to get access to the HCI. This is done by attaching the dongle to the BEX400 analyzer at the STD-A port (under the antenna), then connecting a USB Micro-B to USB STD-A cable from the adjacent Micro-AB receptacle on the analyzer to the PC. The analyzer software will capture the air traffic concurrently with the USB HCI traffic, and will extract the link key automatically in order to decrypt the air traffic.



Note that HCl access is required on one of the two devices only, not both, as the link key is the same for both devices.



Below is a typical setup involving USB HCI capture from a Bluetooth dongle. The analyzer captures both the air traffic as well as the HCI traffic, concurrently. Note that in normal circumstances, the dongle is extended away from the antenna using a 6-inch USB STD-A to STD-A extender cable (as otherwise the dongle may present an overly strong signal when situated very close to the antenna).



In the picture below, the BEX400 analyzer has captured USB HCI concurrently with BR/EDR traffic, with the link key being automatically extracted from the HCI capture and used to decrypt the BR/EDR traffic. A similar approach can be used to capture other forms of HCI, such as UART or SPI, using a special cable at the rear of the BEX400 analyzer.

| | , | wireless usb uart spe | ectrum.2btt.btt* - Ellisys Bluetoot | th Analyzer | | _ 🗖 🛛 🗡 |
|---|--|------------------------------|---|-----------------------------|-------------------------------------|----------------------|
| File View Layout Search R | Record Tools Help | | | 🗎 🖬 Fu | ll screen 🔲 tt 🗔 Analysis 🗔 expt nt | : 📑 Add 🔶 |
| 🔁 💕 🛃 🔀 👫 🕨 Record 👻 | 🕶 🖩 Stop 📓 Restart 🗟 Save & Continue | set ese Navigate - | 🖳 🛅 Markers (7) 🔹 📣 🗛 🍓 Filtering: Exe | clude Background 🔹 🧒 | | |
| Welcome BR/EDR Overview | Low Energy Overview 📘 Instant 4 | HCI Overview (USB) | X | 4 Þ | Instant Piconet | 4 X |
| Protocol: Single selection - All layers | s 🗕 🔶 💡 🔎 🖶 🙀 🧐 | Protocol: Single selection | 구 🛛 All layers 🗕 🖉 📥 💡 🔎 🖶 🎘 🖱 |) * 9 🎝 📾 👶 🛍 😫 🖕 | | |
| Type filter Y • Type filter | | Type filter | | v ▼ Type filte | | |
| Time Item | | ^ Item | | Link ^ | | |
| 82.712 938 625 🗉 🚑 SDP Servi | vice Search Attribute Transaction (L2CAP: OBEX | 🗉 🚑 HCI User Confir | mation Request (C0:E4:22:FE: 17:1F, Pass=097175) | | | |
| 82.778 565 625 🛛 🕀 🙀 RFCOMM | I Connect (Channel=Signaling) | 🕀 🛶 HCI Simple Pairi | ing Complete (Success, C0:E4:22:FE:17:1F) | | | |
| | rice Search Attribute Transaction (PnP Informat | | tification (C0:E4:22:FE:17:1F, Key=83359E53:AE76 | | USB Biongle | |
| 7 | IDLC Parameter Negotiation (Channel=1, Initia | VF | ction Encryption (Connection=0x000B, Encryption=O | n (BR/EDR EO, LE AES-CCM)) | <u>.</u> | |
| | isconnection (Src=0x0041, Dst=0x0040) | | ion (Src=0x0041, PSM=SDP > Dst=0x0040) | | | |
| | I Connect (Channel=1) onnection (Src=0x0043, PSM=SDP > Dst=0x00 | | ion (Src=0x0041, PSM=RFCOMM > Dst=0x0042) • (Dst=0x0041, MTLI=1'017 > Src=0x0042, MTLI=1'0 | v 7) | Notebook | |
| Control 200 Control | onnection (Src=uxuues, PSM=SDP > DSt=uxuu | < | P 0.81=0x0041, MITT=10177 Str=0x0042, MITE=10 | > | | |
| Instant Timing | | | · · · · · · · · · · · · · · · · · · · | ÷ х | | |
| 🐑 🔍 🔳 🚡 - origin: 82.08 s | s • span: 2.02 s | ▼ Display ▼ Logic inputs | s 🕞 | | | |
| | | N | philantheantaantheanth | Managan . ^ | | |
| "USB Dongle" C0:E4:22: | 5 | | | | 🛸 I H H H H 82.430 336 53 | 3 AD: ① |
| "Notebook" 00:02:76:1E: | | | | | 😪 Instant Pic 🕙 Sum 🕍 Instan | t Cha 101 Baw |
| | | | | | Detaile | |
| Inqu | | | Security | | × | isplay - |
| Throe Fill missing fields | | | | | Manage ECDH Keys | Value |
| sco. Time | Master / Slave | PIN | Link Key | ACO | IV | |
| Seria 👫 71.378 405 500 | "Notebook" 00:02:76:1E:10:E6 | Notapplicable | 33359E53:AE76153F:C0F69E08:C6231E8D | 82A353E1:3C8F30C1:2 | Not applic | Link Key Notificatio |
| Link | "USB Dongle" C0:E4:22:FE:17:1F | | | | | Link Key Nouncado |
| Link IN | | | | | SD_ADDR | C0:E4:22:FE:17:1F |
| | | | | | 💊 Link Key | 83359E53:AE7615 |
| JSB HCI | 1 | | | | 🧼 Кеу Туре | Authenticated Com |
| JSB 2.0 | | | | | | |
| ← 🖁 HCI Link Key Notificati | <u>L</u> | | 83.340 s | | | |
| 82.00 s | 0.30 0.40 0.50 0.60 0.70 | 0.80 0.90 83.00 s | 0.20 0.30 0.40 0.50 0.60 0.70 | 0 0.80 0.90 0.10 84.00 s | | |
| Zoom bar 🛃 Instant Timing 🎝 Instant Audio 📥 I | Instant Throughput | | | • | < | |
| Ready | | | | | | 4.0.5372 |
| cauy | | | | | | 4.0.3572 |

© Ellisys 2014. Information contained herein is for illustrative purposes and is not intended in any way to be used as a design reference. Readers should refer to the latest technical specifications for specific design guidance. Ellisys – Chemin Du Grand Puits 38 – CH1217 Meyrin – Geneva +41 22 77 777 89 (USA +1 866-724 -9185) www.ellisys.com.



Ellisys HCI Injection API

The Bluetooth Explorer 400 software application supports an HCl Injection API, selectable in the Available Analyzers dialog in the Record menu. The Injection API uses UDP (User Datagram Protocol) to push HCl traffic to the Bluetooth Explorer 400 software, which is displayed live in the HCl Overview (Injection), *just as if this traffic was being captured* on one of the standard HCl capture ports on the analyzer.

Link keys captured with this method are stored by the analyzer software and associated with the device connection, so that any subsequent captures of air traffic between the devices is automatically decrypted.

| HCI Overview (Injection) Message Log BR/EDR Overview | Lov |
|--|-----|
| otocol: Single selection 🗸 🗚 ll layers 🗧 🥔 🍲 💡 🔎 🖶 | × 🕘 |
| Available analyzers | × |
| Please select an analyzer: | |
| Ellisys Injection API | |
| | |
| | |
| | |
| Use this analyzer by default OK Cance | |
| | |

The User Manual for the Bluetooth Explorer 400 provides a link to download explanatory documents and sample code.

Ellisys Remote Control API

The Ellisys Remote Control API can be used to programmatically inject link keys (obtained from any source) into the Ellisys BEX400 software application. For instance, it is possible to build a Remote Control API client that monitors Windows Registries for changes and pushes new link keys into the analysis application.

The link provided below contains an introductory guide, the plug-in DLL, and sample code written in C#, using Microsoft Visual Studio 2010. Compatibility is Visual Studio 2005 or higher.

http://www.ellisys.com/better_analysis/bex400a_remote_api.zip

Vendor-specific methods

The BEX400 analyzer captures all Bluetooth traffic in the vicinity, even encrypted traffic. The analyzer's software allows the user to manually input a link key, either during capture, or even on a saved capture, in order to decrypt the traffic. When standard link key access methods (such as HCI capture or injected HCI) are not applicable, it may still be possible to extract the link key from devices with vendor-specific methods, and then enter the link key into the analyzer's Security pane for decryption.

In these cases, it is required to have access to special features of the radio chip, the operating system, the SDK, or the host stack. For example, access to a Windows registry, where a link key may be stored, or to developer modes available on some mobile phones.

By no means is the list below complete - it is a work in progress. Ellisys encourages readers to submit suggestions on methods to find link keys across various devices, stacks, etc. for inclusion in this document. See information at the end of this document on how to contact Ellisys. Ellisys also strongly suggests checking with your device manufacturer before attempting any changes to your system software.

For developers, most Bluetooth stack providers or chip developers will include an API or an SDK that provides access to the system registers that likely will include an option to read Bluetooth link keys. Contact your supplier as needed for details.

Windows Bluetooth stack

The link key for Windows-based systems (using the Windows stack) is often accessible from the Windows registry. In many cases, this is located in the BTHPORT folder as shown below, associated with the applicable Bluetooth address (i.e., the Bluetooth device to which the system is connected). Note that this folder may require administrative permissions for access, and that such permissions may be changed in order to gain access (right-click the folder for this option).



HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\BTHPORT\Parameters\Keys

In our case here (at right), the computer (a WIN8 PC) shows a registry directory in BTHPORT/Parameters labeled "e82aeab4de7a" which is the BD ADDR of the PC.

Below, we see the contents of this directory, with "00241cc1d797" shown as the BD ADDR of the connected device, the headset in this case. The data associated with the headset is the link key. We simply copy this link key and paste it into the Ellisys analyzer application (in the Security pane), and as shown below, the traffic between these devices is immediately decrypted.



| Name | Туре | Data |
|--------------------|------------|---|
| 赴 (Default) | REG_SZ | (value not set) |
| 殿 00241cc1d797 | REG_BINARY | 58 48 f3 de ff d5 9c 2b a9 e4 8b 53 9c 3c 1e 61 |

Below, we see the analyzer has now decrypted the traffic (filtered to show only the AT Protocol). Note the Communication column, which lists the two communicating devices and their BD ADDRs.

| | | | Security | | × |
|------------|--------------------------------|--|---|----------------------------|------------------|
| 1 | Fill missing fields | | | | Manage ECDH Keys |
| | Time | Master / Slave | PIN Link Key | ACO | IV |
| 86 | 6.469 675 000 6.582 176 250 | "Comp/PC" E8:2A:EA:B4:DE:7A "Headset" 00:24:1C:C1:D7:97 | Notapplicable 611E3C9C:538BE4A9:2B9CD5FF:DEF34858 | 99E7C83A:3D2B872A:D1351D76 | Not applic 🗸 |
| 101 011 | Raw data 👌 Securit | γ | | | |

| BR/EDR Overview Low Energy Overview | HCI Ove | erview (Serial) | HCI Overview (USB) Message Log 📃 Instant Spectrum | 4 |
|--|---------|-----------------|---|---------------|
| Protocol: Single selection 🗸 🛛 All layers 🛛 🗲 📾 🥧 💡 | 🔎 🗄 🎘 | 🖰 🚸 🎝 🖮 | 💰 🗊 🌐 🖆 🎬 🚭 💑 👑 16 items displayed 👘 Search | - 64 |
| Type filter Y • | Тур 🝸 🕶 | Type filte 🝸 🗸 | Type filter Y • | Type filter |
| Item | Status | Payload | Communication | Destinatior ' |
| 🕀 🍕 AT HFP Supported Features: AT+BRSF=159\r > \r\n > | OK | (| Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| 🕀 🅰 AT MT Indicator: AT+CIND=?\r > \r\n > +CIND: ("ser | OK | | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24: IC:C1:D7:97 | 0x0041 (RF |
| | OK | | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| 🗄 🍕 AT MT EventReporting: AT+CMER=3,0,0,1 \r > \r \n > | OK | | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| 🕀 🅰 AT String: \r\n | OK | 2 bytes (0D 0A) | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| 🕀 🅰 AT String: +BSIR: 0\r\n | OK | 10 bytes (2B 4 | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| ■ 🧟 AT Exchange: AT+CLIP=1\r > \r\n > OK\r\n | OK | | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |
| 🕀 🅰 AT Exchange: AT+NREC=0\r > \r\n > OK\r\n | ОК | | Master: "Comp/PC" E8:2A:EA:B4:DE:7A <-> Slave: "Headset" 00:24:1C:C1:D7:97 | 0x0041 (RF |

Many Windows systems will use a third-party Bluetooth stack instead of the Microsoft stack, such as Toshiba, Widcomm, etc. Registry access to these third-party stacks may not be quite as obvious, so one approach is to force the system to revert from the third-party stack to the Windows stack. The user should set a system restore point, do an uninstall of the third-party stack, and then reboot. The system should then default to the Microsoft stack.

In some cases, browsing online forums may be useful. The URL below is one example:

http://superuser.com/questions/229930/finding-bluetooth-link-key-in-win7-to-double-pair-a-device-on-dualbootcomputer



Android stack

Later versions of the Android operating system provide a Developer Mode, which can be enabled through the *Settings/About Phone* menu by depressing the *Build Number* selection several times repeatedly (typically seven times). Consult your phone's User Manual to be sure you have Developer Mode available and for specific instructions.

The Developer Mode includes a variety of features including an option to enable a log file containing HCI traffic. This file is in BT Snoop format. This file, which will contain link keys from pairing events, can be easily imported to the BEX400 analyzer software. The analyzer software will store these link keys and will associate these with the devices using them, so subsequent air captures of the connected devices will be automatically decrypted.

The pictures below depict an abbreviated sequence to enable the Developer Mode, and the resulting BT Snoop HCI log file. The file can be accessed over the phone's USB connection or it can be shared in a variety of ways, such as by E-Mail.



The screenshot below shows the BEX400 Import menu, with BT Snoop HCI selected for import. The BT Snoop file created by the phone is referenced as the location from which to do the import.

| Import | | × |
|---|--------------------------|---|
| Import Please choose the input file and format. | Concession of the second | ^ |
| Import format: | | |
| BTERSon HCI Ellisys USB 2.0 HCI Capture Ellisys USB 2.0 HCI Capture | Preview | |
| Import data from: | | |
| C: Users \chuck/Desktop \btsnoop_hci.log | Browse | |
| < <u>B</u> ack <u>N</u> ext > | Cancel | ¥ |



Below, the resulting capture after the phone's BT Snoop log has been imported by the BEX400 application. This shows the link key established between the Phone and the Headset. This key will be stored by the analyzer software and used to decrypt traffic on captures of subsequent connections between the phone and headset.

| | Untitled* - Ellisys Blu | uetooth Analyzer | | _ <u> </u> |
|--|---|---------------------------------------|---------------------------------------|--------------------------------------|
| ile View Layout Search Record Tools Help | | | 🗧 Full screen 🔳 | tt 🖪 Analysis 🗐 expt nt 📑 Add / |
| 🚹 🧀 🙀 👫 🕨 Record 👻 🗉 Stop 💷 Restart 💷 Save & Cont | inue 🚰 🎆 🚚 Navigate 🕞 🖳 Markers (1 |) 🝷 🚛 📭 🙀 Filtering: Exclude Backgrou | nd 👻 🌏 | |
| HCI Overview (Injection) A Welcome BR/EDR Overview Low E | nergy Overview HCI Overview (Serial) HCI Over | erview (USB) Message Log 🔚 In 4 🕨 | Details | 4 |
| õtosol: Single selection 🖬 🗚 layers 🔶 🥒 🖕 🚱 📲 | - · · · · · · · · · · · · · · · · · · · | 1'223 items displayed | 🛛 🗱 All fields 🛛 🚔 Show in overvi | ew Display - 🔁 Search |
| Type filter | | Type filter | Name | Value |
| Item | Communication | Link Key | * HCI Simple Pairing Complet | (Success 00:24:10:(1:D7:07) |
| HCI Create Connection (00:24:1C:C1:D7:97, Allow Switch=Yes) > Con | Master: Host (Unknown BD_ADDR) <-> Slave: Phone | | × net simple raining complet | e (Success, 00.24.1C.CI.D7.57) |
| ← HCI Max Slots Change (Connection=0x000C, Slots=5) | Master: Host (Unknown BD_ADDR) <-> Slave: Phone | | 😑 🛶 HCI Packet | |
| HCI Read Clock Offset (Connection=0x000C) > 0x00145FC | Master: Host (Unknown BD_ADDR) <-> Slave: Phone | | Event Code | Simple Pairing Complete |
| L2CAP Information (Extended Features Supported) | Master: Phone <-> Slave: Headset | | Arameters | Simple Pairing Complete |
| HCI Read Remote Version Information (Connection=0x000C) > Version=3 | | | Status | Success |
| HCI Read Remote Supported Features (Connection=0x000C) > 41 Fe | | | Ø BDADDR | 00:24:1C:C1:D7:97 |
| HCI Authentication Requested (Connection=0x000C) > 283545 > Key=1. | | 16986C41:CDAF9687:D1D912EE:EA525A4 | × | |
| HCI Write Link Policy Settings (Connection=0x0000C, Settings=Role Sw | | | HCI Link Key Notification (0 | 0:24:1C:C1:D7:97, Key=169B6C41:CDAF9 |
| HCI Write Link Supervision Timeout (Connection=0x000C, Timeout=5 s) | | \smallsetminus | | |
| HCI Read Remote Extended Features (Connection=0x000C, Page=1) > . | | | 😑 🕂 HCI Packet | |
| Kernet Connection Packet Type (Connection=0x000C) | Master: Host (Unknown BD_ADDR) <-> Slave: Phone | | Event Code | Link Key Notification |
| Her entrige connection = 0x0000C, Settings=Role Sw | | | 📃 🕂 Parameters | |
| | Master: Host (Unknown BD_ADDR) <-> Slave: Phone | | Ø BD_ADDR | 00:24:1C:C1:D7:97 |
| | Master: Phone <-> Slave: Headset | | 💡 Link Key | 169B6C41:CDAF9687:D1D912EE:EA |
| | Master: Phone <-> Slave: Headset | | 🧳 Кеу Туре | Unauthenticated Combination Key (P |
| L2CAP Configure (Dst=0x0041, MT0=256 > Stc=0x0040) L2CAP Configure (Dst=0x0040, MTU=64 > Stc=0x0041) | Master: Phone <-> Slave: Headset | | | |
| | | | A HCI Authentication Complete | te (Success, Connection=0x000C) |
| B/P SDP Service Search Attribute Transfer (PnP Information: PnP Informat | Master: Phone <-> Slave: Headset | ~ | 🗄 🕂 HCI Packet | |
| < | | > | 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | |
| tant Timing | | Р Х | Event Code Arrow Parameters | Authentication Complete |
| 🖑 🔍 🔲 🚡 - origin: 70,059.33 ms | Display Logic inputs | | A Status | Success |
| a congin: 10,059.55 ms + span: 5.20 ms | | | < | SUCCESS |
| (Injected) | <u>é</u> | ^ | Details 📢 Instant Piconet 🤮 | Summary |
| OUT | * | | | |
| | - | - | Raw data | 7 |
| IN | | | Data type: | |
| | | | 0 1 2 3 | 4 5 6 7 8 012345678 |
| HCI Auth | | | 0x0000: 18 17 97 D7 C | |
| 9.20 9.40 9.60 9.80 0.20 0.40 0.60 70.060.00 ms | 0.80 1.00 1.20 1.40 1.60 | 1.80 2.20 2.40 2.€ 70,062.00 ms | 0x0009: <mark>5A 52 EA EE 1</mark> | 2 D9 D1 87 96 ZR |
| om bar | | × | | |
| Instant Timing 🧦 Instant Audio 📥 Instant Throughput | | | 101 Raw data 👌 Security | |
| ady | | | | 4.0.53 |

For information on the Android Developer SDK, consult the link below: <u>http://developer.android.com/sdk/index.html</u>



CSR Persistent Store

For developers working with CSR devices and tools, there is good news: the Ellisys BEX400 Explorer application natively supports the format of CSR's PSTool utility. The keys are stored in configuration data 42 to 49:



The hex data is directly supported by the Ellisys software and can be provided in the Link Key field of the Security view as is, without further conversion. The Ellisys software will recognize CSR format and will convert it automatically.

Nokia Mobile Phones

Some Nokia models can be put in Debug Mode by typing *#2873#

Linux/BlueZ Bluetooth stack

BlueZ is official Linux Bluetooth protocol stack. This stack is also used on many Android platforms (prior to Android 4.2). See <u>www.bluez.org</u> for information on utilities that may be useful for accessing link keys.

The link below has information on the BlueZ Utils application, which features an HCI tool that may also be useful. https://www.linux.com/news/hardware/peripherals/44623-working-with-bluetooth-connecting-to-all-those-cool-devices

/var/lib/Bluetooth/[BD_ADDR]/linkkeys



Widcomm Bluetooth stack

Note that Widcomm was acquired by Broadcom. Widcomm was the first Bluetooth stack used by Windows. Similar to the Windows stack section above, try the location in the registry below for Widcomm.

HKEY_CURRENT_USER\Software\Widcomm\BTConfig\LinkKeys

Bluetooth Smart (Low Energy)

Security features for Bluetooth Smart (per specification version 4.0) differ from those used by BR/EDR. A Bluetooth Smart pairing procedure results in the generation of a long term key (LTK), which is derived using a process that varies significantly from the process used by BR/EDR that results in generation of a link key. Bluetooth Smart devices are architecturally simpler than BR/EDR devices as they are intended for simpler tasks, so there is generally less processing and storage capabilities on these devices, and in fact a simpler and less secure approach to privacy.

See the section below titled Secure Connections for Bluetooth Smart for information on future changes involving security for Bluetooth Smart.

With Bluetooth Smart, one device determines the LTK, then sends this to another device using temporary encryption based on a short term key (STK). One benefit to this simpler approach is faster connections and reconnections, a primary design goal of Bluetooth Smart and a major difference to the slower-connecting BR/EDR. A drawback is a lack of eavesdropping protection.

So, with Bluetooth Smart, we're using "key transport" (the LTK is transmitted to the other device) instead of "key agreement" (the link key is determined symmetrically by both devices).

Importantly for BEX400 users, the hunt for a key is not really needed, as the analyzer is able to crack the security for Bluetooth Smart. Future evolutions of Bluetooth Smart will make this much more difficult, but the BEX400 architecture will accommodate these changes easily (see Secure Connections for Bluetooth Smart, below).



The screenshot below depicts the LTK being sent from a key fob to a dongle. Note that STK-enabled encryption is already in place at the time this packet is sent.

| File Upout Sector Tool Hop Image: Sector Stop Restart (S Sector Continue) Image: Sector Continue) Image: Sector Continue | 3 | | LowEnergySecuri | ity.btt* - Ellisy: | s Bluetooth | Analyzer | | _ 0 | × |
|---|---|---|---|---------------------|-----------------|----------------------------|-------------------------------|------------------------|------------|
| Webser SREER Overview Los facery Overview HCL Overview (Bath) Hessage Log Protect C1 Protocol Multiple statiction - Imate Protocol Imate Protocol Imate Protocol Imate Protocol Protocol Multiple statiction - Imate Protocol | File View Layout Search Red | cord Tools Help | | | | | 🗔 Full screen 🔲 tt 🔲 Analysis | 🖬 expt nt 📑 Add | 7 |
| PortocolMultiple selection • • • • • • • • • • • • • | : 🛅 💕 🛃 🌠 👫 🕨 Record 🔹 | 🖩 Stop 🔄 Restart 🕞 Save & Continue 👹 | 🕷 🚚 Navigate 🕞 🖳 | 🛅 Markers 👻 🚚 | 🖦 🛛 🍓 Filtering | g: Only Dongle, Keyfob 👻 🌏 | \ | | _ |
| Type file V Type file Tammitter Item Tammitter Item Save: Keyfeb 3C:2007846667 Save: Keyfeb 3C:2007846667 Save: Keyfeb 3C:2007846667 Save: Keyfeb 3C:200784667 Save: Keyfeb 3C:200784667 Save: Keyfeb 3C:2008784667 | Welcome BR/EDR Overview | Low Energy Overview HCI Overview (USB) | HCI Overview (Serial) | Message Log | Instant 5 4 🕨 | Instant Piconet | | | д х |
| Transmitter Nem Mater: Toolge 220:000-0974:50 See: Keyfeld 32::027340667 See: Keyfeld 32::027340667 See: Keyfeld 32::027340667 See: Keyfeld 32::027340667 See Transport Specific Key Detruption Information (T=-3457:224::0411383:20728051:251986-0) See: Keyfeld 32::027340667 See: Keyfeld 32::027340667 See: Keyfeld 32::027340667 See Transport Specific Key Detruption Information (T=-3457:224::0411383:20728051:251986-0) See Transport Specific Key Detruption Information (T=-3457:224::0411383:20728051:251986-0) See: Keyfeld 32::0287840667 See Transport Specific Key Detruption Information (T=-3457:224::0411383:20728051:251986-0) See Transport Specific Key Detruption Information (T=-3457:224::0411383:20728051:251986-0) See: Keyfeld 32::0287840667 See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:251986-0) See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:251986-0) See: Keyfeld 32::0287840667 See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:261986-0) See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:261986-0) See: Keyfeld 32::02878406687 See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:261986-0) See Transport Specific Key Detruption Information (T=-3457:226::0411383:20728051:261986-0) Fill missing fields Manage EDDH Key Manage EDDH Key See Transport Specific Key Detruption Info | Protocol: Multiple selection 👻 👄 📥 | 💡 🖶 🦻 224 items kept, 4 filtered | | Search | - 😫 | | | | |
| Master: Tongle* 29:00:00:99FF:55 Bit CDE Brangeton Start (211Y - 040000, 2010 - 04074272.344/FAREE), 11404022.2875 5: 9020 - 040424504 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 Same: Keyfold* 20:20:07:04:06:67 <td>Type filter Y</td> <td>Type filter</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> | Type filter Y | Type filter | | | | | | | |
| Ser: Yerfeb* 21:20:27:20:40:667 SeP Transport Specific Yergeb* 22:20:27:20:40:67 Ser: Yerfeb* 22:20:27:20:40:67 Ser: Yerfeb* 22:20:27:20:40:667 Sep Transport Specific Yergeb* 22:20:27: | Transmitter | Item | | | ^ | | | | |
| See: Yerfol' \$2:0387340667 See: Yerfol' \$2:0387340667 See: Yerfol' \$2:0387340667 See Yerfol' \$2:0387340667 See Yerfol' \$2:0387340667 See Yerfol' \$2:038734067 < | Master: "Dongle" 29:CD:00:99:FF:56 | LLCP Encryption Start (EDIV=0x0000, SKD) | n=0x074C7C3A41FAFBE8, I | /m=0x0C23B7B5 > SKD | s=0xA245DAE | | | | |
| Sive: Yeyfob' 3C:20:87:84:06:67 | Slave: "Keyfob" 3C:2D:B7:84:06:67 | 🖃 🔑 SMP Transport Specific Key Distribution 💓 | =3A57C2AC:40411828:2BC | 726D5:C2B198E4> ED | IV=0x53DB) | | Keyfob | | |
| Sile:: Ykeyfeb* 3C:2087840667 | Slave: "Keyfob" 3C:2D:B7:84:06:67 | | C2AC:40411828:28C726D5:0 | C2B198E4) | | 📦 H H H H H4 H 442.270 37 | 18 875 AD: 👁 | | |
| Bise: Fx8/100 Stude: Torogle Stude: Tororogle Stude: Torogle <td></td> <td></td> <td></td> <td></td> <td></td> <td>Summary</td> <td>Instant Channels</td> <td></td> <td></td> | | | | | | Summary | Instant Channels | | |
| Slave: Ykeyfob' 3C:2D:87:84:06:67 SearCh Value Value Slave: Ykeyfob' 3C:2D:87:84:06:67 Start/Complete LE-U Packet, LILD L2CAP Start Fragment / Complete Slave: Ykeyfob' 3C:2D:87:84:06:67 Start/Komplete LE-U Packet, Start Notification Packet (0xFF+4: 0.3) L2CAP Start Fragment / Complete Slave: Ykeyfob' 3C:2D:87:84:06:67 Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Ykeyfob' 3C:2D:87:84:06:67 Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Ykeyfob' 3C:2D:87:84:06:67 Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) The Master / Slave PIN Link Key ACO V Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) The Master / Slave PIN Link Key ACO V Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0.3) Start Notification Packet (0xFF+4: 0 | | | | | | | | | |
| Master: Dongle* 28:00 Master: Dongle* 28 | | - | | | | | | | φ× |
| Silve: Y. Keyfob '3C:2D:87:84:06:67 iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii | | | | | | ➤ All fields | / Display 👻 📄 | Search | - |
| Skret: Keyfob 3C:20:87:84:06:67 B SMP Master Identification (EDIV=0x5006, FL038854) LID LID </td <td></td> <td>-</td> <td>and the second se</td> <td></td> <td></td> <td>Name</td> <td>Value</td> <td></td> <td>^</td> | | - | and the second se | | | Name | Value | | ^ |
| Slave: Yeyfob' 3C:20:87:8406:67 ATT Notification Packet (0xFFF4: 03) Security SN SN Payload Data Length SS Payload Data Length SS Data Clocket Payload Data Length Start/Conceleta LE-U Start/Conceleta LE-U Start/Conceleta | | | | 0954) | | 🧼 LLID | L2CAP Start | Fragment / Complete | |
| Security SN 0 Fill missing fields Manage ECDH Key 25 Tme Master / Slave PN Link Key ACO IV Security ACO IV Data 25 Data 21 bytes 21 bytes 21 bytes Security ACO IV Data 21 bytes Werens Vicyfob 30:201:87:84:06:67 Data Works SF665668:0E1CDDEBF; 400258A:FC3D94A4 Not applicable Not applicable Not applicable Werens Werens Start/Complets LE:U Basic, 17 bytes Data 21 bytes Scoleco Start/Complets LE:U Start/Complets LE:U Start/Complets LE:U Start/Complets LE:U Werens Scoleco O 0,12 3 4 5 6 7 8 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 7 7 8 4 9 A B C O 12 3 4 5 | | | D, Raid-OXEASEDODI TICOS | 55547 | | | - | | |
| Fill missing fields Manage ECDH Keys 25 Time Master / Slave PIN Link Key ACO IV Link Key ACO IV Data Data Data Data Start / Slave PIN Link Key ACO IV Data | | | | | | | | | |
| If ill missing fields Manage ECDH Keys | | Security | / | | | | • | | |
| Time Master / Slave PIN Link Key ACO IV Description Descrint <thdescription< th=""> <th< td=""><td>Fill missing fields</td><td></td><td>/</td><td>N</td><td>lanage ECDH Ke</td><td>VC</td><td></td><td></td><td></td></th<></thdescription<> | Fill missing fields | | / | N | lanage ECDH Ke | VC | | | |
| | Time Master / Slave | PIN Link Key | | ACO | TV | | | (SMP) | |
| ¹ · · · · · · · · · · · · · · · · · · · | | | D25BA:FC3D94A4 | | | | | | |
| Wereness Concide Encryption information "Keyfob" 30:20:09:91-6 Encryption information 3457C2AC:40411828:26C72605:C2B19864 "Dongle" Encryption information 3457C2AC:40411828:26C72605:C2B19864 "Dongle" Encryption information 1 Start/Conclete LE-U Information 1 Start/Conclete LE-U Encryption information 1 Start/Conclete LE-U Information 1 Start/Conclete LE-U Information 1 Start/Conclete LE-U Information Information Start/Conclete LE-U Information Information Start/Conclete LE-U Information Information Start/Conclete LE-U Information Information Information Start/Conclete LE-U Information Information Information Start/Conclete LE-U Information Information Information Information Start/Conclete LE-U Information Information Information Information Start/Conclete LE-U Information Information Information Information Start/Conclete LE-U Information <td></td> <td></td> <td></td> <td></td> <td></td> <td>Y SMP Frame</td> <td></td> <td></td> <td></td> | | | | | | Y SMP Frame | | | |
| "Keyfob" 3C:20:B7:8406 UCcode | wireless | | | | | | | | |
| Torongle* 29 CD.00.939 F Energy Conceptes LE-U Same data Image: Conceptes LE-U Image: Concentes LE-U Image: Conceptes LE-U | "Keyfob" 3C:2D:B7:84:06 | - / | | | | | | | 054 |
| Throughput Raw data Raw data Image: Construction of the construction o | Empty LE-L | JStart/Complete LE-U | | | | | 3A5/C2AC:4 | 0411828:2BC726D5:C2B19 | |
| Instant Timing Instant | | | | | _ | | | | - |
| Scole200 Data type Packet Raw Data • Search • \$\$\$ \$\$\$ \$\$\$ \$\$\$\$ \$\$\$\$ \$\$\$\$\$\$\$\$\$\$\$\$\$\$\$ | | 295 B/s | | | | | | | 4 X |
| 1000000000000000000000000000000000000 | | | | | | Data type: Packet Raw Data | • | | • |
| | | | | | | | | | |
| SMP Encryption Infor. 442,270,220 ms 0.05 0.15 0.20 0.45 | | | | | | | | | |
| s 442,270.00 d/s 0.10 '0.15 '0.20 '0.25 '0.30 '0.35 '0.40 '0.45 '0.45 '0.40 '0.45 '0.45 '0.40 '0.45 | SMP Encryption Infor. | 442 270 220 mc | | | | | 10 10 00 07 04 21 01 3 | | · |
| Zeen bar her historie für Stant Audo Arstant Throughput | 5 0.05 0.10 0.15 0.20 | | 0.55 0.60 0.65 | 0.70 0.75 0.4 | 80 0.85 0.3 | | | | |
| | | 442,27 | 0.50 ms | | | | | | |
| Ready 4.0.5372 | 🛤 Instant Timing 🎝 Instant Audio 📥 In | istant Throughput | | | | | | | |
| | Ready | | | | | | | 4.0 | 0.5372 |

Secure Connections for Bluetooth Smart

An upcoming revision to the Bluetooth specification (v4.2) will add various enhancements, including the addition of LE Secure Connections that will greatly improve security for Bluetooth Smart devices. This addition updates the pairing procedures to utilize the P-256 elliptic curve and FIPS-approved algorithms (AES-CMAC and P-256 elliptic curve) in a way that is similar to BR/EDR Secure Connection SSP. This change also includes provisions for a shared secret key. LE legacy pairing (described in the section above) will still be available. Note that for BR/EDR, Secure Connections was added with the 4.1 version of the specification and consisted mainly of a change in the encryption method; the pairing method still being SSP but with longer Private/Public Keys (256 bits instead of 192 bits).



The diagram below describes the expected architecture for LE Secure Connections as well as a comparison to LE legacy pairing and BR/EDR Secure Connections.



As depicted in the diagram above, LE Secure Connections will also allow a key derived over one transport (e.g., BR/EDR) to be used for another transport (e.g., Bluetooth Smart). So, if you have two devices that both support BR/EDR and Bluetooth Smart, and both support Secure Connections on both of these transports, you just have to pair once. In other words, an LTK generated during the LE pairing process may be converted to a BR/EDR link key for use on the BR/EDR transport, and conversely, a BR/EDR link key generated during the BR/EDR SSP pairing process can be converted to an LTK for use on the LE transport.

Feedback needed

Do you know of other ways to access link keys, or do you have feedback on the implementation of the suggestions in this document? Please share your feedback with us at expert@ellisys.com and we will update this document accordingly (we'll attribute your input in the updated document if you wish). This will be a great help to the Bluetooth community!

Other interesting readings

- <u>EEN BT03 Your First Wide-Band Capture</u>
- <u>EEN BT06 Bluetooth Security Truths and Fictions</u>
- <u>EEN BT07 Secure Simple Pairing Explained</u>
- More Ellisys Expert Notes available at: <u>http://www.ellisys.com/technology/expert_notes.php</u>

Rev. A. Updated 2014-10-01