2023-04-19

Created: 2023-04-29 22:45:11

Chats

- Papers.SGX.0419.md
  - `[07:40]` What is difference between Sealing Identity and Identity of the Sealing Authority? `(Papers.SGX.0419.md)`
  - `[07:43]` Is Sealing Authority a enclave builder? `(Papers.SGX.0419.md)`
  - `[07:54]` Who can write MRENCLAVE and MRSIGNER? `(Papers.SGX.0419.md)`
  - `[07:57]` Only hardware can write the both registers. Right? `(Papers.SGX.0419.md)`
  - `[07:58]` What does SIGSTRUCT contain? please list them. `(Papers.SGX.0419.md)`
  - `[09:07]` What is Sealing Identity? `(Papers.SGX.0419.md)`
  - `[09:09]` What is Sealing Authority? `(Papers.SGX.0419.md)`
  - `[09:12]` Is Sealing Auhority a software or a hardware? `(Papers.SGX.0419.md)`
  - `[09:13]` Is Sealing Authority a software or a hardware? `(Papers.SGX.0419.md)`
  - `[09:16]` Who does prepare SIGSTRUCT? `(Papers.SGX.0419.md)`
  - `[09:18]` Coud I say that SIGSTRUCT is Sealing Identity? `(Papers.SGX.0419.md)`
  - `[09:27]` What is the value of Sealing Identity in the paper? `(Papers.SGX.0419.md)`
  - `[09:29]` What is the value of Sealing Identity in the policy “Sealing to the Enclave Identity” in the paper? `(Papers.SGX.0419.md)`
  - `[09:40]` Who produces attestation assertion? `(Papers.SGX.0419.md)`
  - `[09:45]` What is attestation key? `(Papers.SGX.0419.md)`
  - `[09:46]` How to generate attestation key? `(Papers.SGX.0419.md)`
  - `[09:47]` Is there only one attestation key in a device? `(Papers.SGX.0419.md)`
  - `[09:49]` Where is attestation key sotred in Intel SGX arch? `(Papers.SGX.0419.md)`
  - `[10:27]` What does REPORT contain? Please list them? `(Papers.SGX.0419.md)`
  - `[14:18]` What is MRENCLAVE? `(Papers.SGX.0419.md)`
  - `[14:19]` Who generates MRENCLAVE? `(Papers.SGX.0419.md)`
  - `[14:27]` Who creates an enclave? `(Papers.SGX.0419.md)`
  - `[14:28]` So an untrusted software can also create an enclave. Right? `(Papers.SGX.0419.md)`
  - `[14:30]` How to get the MRENCLAVE value of an enclave? `(Papers.SGX.0419.md)`
  - `[14:37]` I have enclave A and enclave B. How does enclave A know the enclave B before A wants to get MRENCLAVE value of B? `(Papers.SGX.0419.md)`
- SGX.0418.md
  - `[09:50]` Where is attestation key sotred in Intel SGX arch? `(SGX.0418.md)`
  - `[09:52]` What is attestation key? `(SGX.0418.md)`
  - `[09:53]` Is there only one attestation key in a device? `(SGX.0418.md)`

Q & A

Papers.SGX.0419.md

Question: `[07:40]` What is difference between Sealing Identity and Identity of the Sealing Authority? `(Papers.SGX.0419.md)`

Answer:

The Sealing Identity and the Identity of the Sealing Authority are related but not the same.

Sealing Identity refers to the policy used when invoking the EGETKEY instruction in Intel® SGX to determine which enclaves can access a sealing key. The Sealing Identity is based on the enclave's MRSIGNER value and the enclave's version. This policy allows for data migration between different versions of enclaves signed by the same Sealing Authority.

On the other hand, the Identity of the Sealing Authority is represented by the MRSIGNER value, which reflects the key or identity of the authority that signed the enclave's certificate. The Sealing Authority is responsible for signing enclaves and determining which enclaves can access the same seal key, enabling them to transparently access data that was sealed by other enclaves under their authority.

In summary, the Sealing Identity is a policy based on the MRSIGNER and enclave version, while the Identity of the Sealing Authority is represented by the MRSIGNER value itself.

Question: `[07:43]` Is Sealing Authority a enclave builder? `(Papers.SGX.0419.md)`

Answer:

Yes, the Sealing Authority can be considered as the enclave builder or the entity responsible for creating and signing enclaves. The Sealing Authority has the necessary cryptographic keys to sign the enclave's certificate, which in turn establishes trust in the enclave's identity and authenticity. The MRSIGNER value reflects the identity of the Sealing Authority, and it is used in sealing policies to determine which enclaves can access the same sealing keys, enabling them to transparently access data that was sealed by other enclaves under the authority's control.

Question: `[07:54]` Who can write MRENCLAVE and MRSIGNER? `(Papers.SGX.0419.md)`

Answer:

MRENCLAVE and MRSIGNER are values that are generated and populated by the Intel SGX hardware during the enclave creation process.

1. MRENCLAVE: This value is a measurement (hash) of the enclave's code and data. It is generated by the hardware when the enclave is being built and loaded into memory. The MRENCLAVE value is a result of hashing the enclave's content during the build process, and it represents the identity of the specific enclave. No one writes MRENCLAVE directly; it is produced by the hardware as a part of the enclave creation process.
2. MRSIGNER: This value is a measurement (hash) of the public key of the Sealing Authority, which is responsible for signing the enclave's certificate. MRSIGNER is populated by the hardware when the enclave is loaded into memory and is derived from the enclave's signature. The Sealing Authority generates the key pair used for signing the enclave, and during the enclave loading process, the hardware calculates the MRSIGNER value from the public key in the enclave's signature. This value represents the identity of the Sealing Authority.

In summary, MRENCLAVE and MRSIGNER are not written by any individual or entity directly; instead, they are generated and populated by the Intel SGX hardware during the enclave creation and loading process.

Question: `[07:57]` Only hardware can write the both registers. Right? `(Papers.SGX.0419.md)`

Answer:

Yes, that's correct. Only the Intel SGX hardware can write both MRENCLAVE and MRSIGNER registers. These values are generated and populated by the hardware during the enclave creation and loading process, ensuring that they are secure and trustworthy.

Question: `[07:58]` What does SIGSTRUCT contain? please list them. `(Papers.SGX.0419.md)`

Answer:

SIGSTRUCT is a data structure in Intel SGX that contains the enclave's signature, which is generated by the enclave author or builder during the enclave signing process. The SIGSTRUCT contains the following fields:

1. Header: A constant value identifying the SIGSTRUCT.

2. Vendor: Indicates whether the enclave is a debug or production enclave.
3. Date: The date when the enclave was created.
4. Header2: A constant value representing the second part of the header.
5. SWDefined: Software-defined field, reserved for future use.
6. Reserved1: Reserved field for padding.
7. Modulus: The RSA public key modulus (n) of the enclave signer.
8. Exponent: The RSA public key exponent (e) of the enclave signer.
9. Signature: The RSA signature of the enclave, generated using the enclave signer's private key.
10. MiscSelect: A set of flags indicating which additional information should be included in the enclave measurement.
11. MiscMask: A mask for the MiscSelect field.
12. Reserved2: Reserved field for padding.
13. Attributes: Flags describing various enclave properties.
14. AttributeMask: A mask for the Attributes field.
15. EnclaveHash: The MRENCLAVE value, which is the hash of the enclave's initial state.
16. Reserved3: Reserved field for padding.
17. ISVProdID: The Product ID of the enclave.
18. ISVSVN: The Security Version Number (SVN) of the enclave.
19. Reserved4: Reserved field for padding.
20. Q1, Q2: Two values used in the RSA signature generation process.

These fields are used by the Intel SGX hardware to verify the integrity and authenticity of the enclave during the enclave loading process.

Question: `[09:07]` What is Sealing Identity? `(Papers.SGX.0419.md)`

Answer:

The Sealing Identity is a set of criteria in Intel SGX that determines which enclaves can access a specific Sealing Key. These criteria help control the accessibility of sensitive data to future versions of an enclave.

When an enclave needs to store sensitive data securely outside the enclave, it uses the Sealing Key to encrypt and integrity-protect the data. The Sealing Identity specifies the conditions under which a later instantiation of an enclave can access this Sealing Key and decrypt the sealed data.

Intel SGX supports two policies for Sealing Identity:

1. Sealing to the Enclave Identity: This policy bases the Sealing Key on the MRENCLAVE value (the measurement of the enclave's initial state). The key is available to any instance of the exact enclave, and future software versions cannot access the secrets of the original enclave.

2. Sealing to the Sealing Authority Identity (Sealing Identity): This policy bases the Sealing Key on the MRSIGNER value (the measurement of the enclave's signing authority) and the enclave's version. It allows newer enclaves signed by the same Sealing Authority to access data stored by previous versions, enabling data migration between enclave versions.

In summary, the Sealing Identity determines which enclaves can access the Sealing Key based on the selected policy, either by restricting access to the same enclave version or allowing access to newer enclave versions signed by the same Sealing Authority.

Question: `[09:09]` What is Sealing Authority? `(Papers.SGX.0419.md)`

Answer:

The Sealing Authority is an entity responsible for signing and endorsing enclaves in the context of Intel SGX. It is typically the enclave developer or an organization that signs and vouches for the integrity and authenticity of the enclave software. The Sealing Authority signs enclaves using its private key, and the corresponding public key is embedded within the enclave to verify its authenticity.

When sealing data to a Sealing Authority Identity (Sealing Identity), the Sealing Key is based on the MRSIGNER value (the measurement of the enclave's signing authority) and the enclave's version. This policy allows newer enclaves signed by the same Sealing Authority to access data stored by previous versions, enabling data migration between enclave versions and ensuring that only trusted and authenticated enclaves can access sensitive data.

Question: `[09:12]` Is Sealing Auhority a software or a hardware? `(Papers.SGX.0419.md)`

Answer:

The Sealing Authority is neither software nor hardware. It is an entity, such as an enclave developer or an organization, responsible for signing and endorsing enclaves in the context of Intel SGX. The Sealing Authority signs enclaves using its private key, and the corresponding public key is embedded within the enclave to verify its authenticity.

Question: `[09:13]` Is Sealing Authority a software or a hardware? `(Papers.SGX.0419.md)`

Answer:

I apologize for any confusion in my previous response. The Sealing Authority is not a specific piece of software or hardware; it refers to the entity responsible for signing enclaves. This entity could be an individual developer, a company, or an organization. They use software tools and cryptographic techniques to sign enclaves, ensuring their authenticity and integrity.

Question: `[09:16]` Who does prepare SIGSTRUCT? `(Papers.SGX.0419.md)`

Answer:

SIGSTRUCT is prepared by the enclave developer or the entity responsible for creating the enclave. This entity, sometimes called the Sealing Authority, generates the SIGSTRUCT as part of the enclave building process. They sign the enclave with their private key, which is then verified by the Intel SGX hardware using the public key (MRSIGNER) to ensure the enclave's authenticity and integrity.

Question: `[09:18]` Coud I say that SIGSTRUCT is Sealing Identity? `(Papers.SGX.0419.md)`

Answer:

No, SIGSTRUCT and Sealing Identity are not the same thing. SIGSTRUCT is a data structure that contains the enclave's measurement (MRENCLAVE), the signer's measurement (MRSIGNER), and other metadata related to the enclave, including the signature of the enclave created by the Sealing Authority.

On the other hand, Sealing Identity is a concept that refers to the criteria or policy used to determine which enclaves may access the sealing key generated by the EGETKEY instruction. It is either based on the enclave's identity (MRENCLAVE) or the identity of the Sealing Authority (MRSIGNER).

Question: `[09:27]` What is the value of Sealing Identity in the paper? `(Papers.SGX.0419.md)`

Answer:

In the paper, the Sealing Identity refers to the criteria or policy used to determine which enclaves may access the sealing key generated by the EGETKEY instruction. Intel® SGX supports two policies for Seal Keys:

1. Sealing to the Enclave Identity: This policy is based on the enclave's MRENCLAVE. Any change that impacts the enclave's measurement will yield a different key. This results in a different key for each enclave, providing full isolation between enclaves.
2. Sealing to the Sealing Identity: This policy is based on the enclave's MRSIGNER and the enclave's version. MRSIGNER reflects the key/identity of the Sealing Authority that signed the enclave's certificate. This allows offline migration of sealed data between enclave versions.

The actual value of the Sealing Identity will depend on which policy is chosen and the specific enclave measurements

involved.

Question: `[09:29]` What is the value of Sealing Identity in the policy “Sealing to the Enclave Identity” in the paper? `(Papers.SGX.0419.md)`

Answer:

In the policy “Sealing to the Enclave Identity,” the value of the Sealing Identity is based on the enclave’s MRENCLAVE. MRENCLAVE is a unique value that represents the measurement of the enclave’s code and data. Any change that impacts the enclave’s measurement will yield a different MRENCLAVE value. Thus, the Sealing Identity in this case would be the MRENCLAVE value of the enclave.

Question: `[09:40]` Who produces attestation assertion? `(Papers.SGX.0419.md)`

Answer:

In the context of Intel SGX, the attestation assertion is produced by the enclave software running on the hardware. The enclave generates a data structure called a quote, which contains the attestation information, including the measurement of the enclave (MRENCLAVE) and the enclave’s identity. The quote is then typically sent to a remote party (verifier) who evaluates the attestation assertion to determine if they trust the enclave and its execution environment. In some cases, an additional step is involved where the quote is sent to the Intel Attestation Service, which returns a signed attestation report that the verifier can then use to verify the enclave’s authenticity.

Question: `[09:45]` What is attestation key? `(Papers.SGX.0419.md)`

Answer:

An attestation key is a cryptographic key used in the process of remote attestation. Remote attestation is a security mechanism that allows a party to verify the integrity and authenticity of a software running in a secure execution environment, such as an Intel SGX enclave.

The attestation key is used to sign the attestation data, which typically includes measurements of the software (e.g., a hash of the software code), the identity of the software, and other relevant information. By signing the attestation data with the attestation key, the software proves that it is running in a secure environment and has not been tampered with. The verifier, who receives the signed attestation data, can then use the corresponding public key to validate the signature and establish trust in the remote software.

Question: `[09:46]` How to generate attestation key? `(Papers.SGX.0419.md)`

Answer:

Generating an attestation key typically involves the following steps:

1. Choose a cryptographic algorithm: Select a suitable cryptographic algorithm for the attestation process. Common choices include RSA, ECDSA (Elliptic Curve Digital Signature Algorithm), or EdDSA (Edwards-curve Digital Signature Algorithm), depending on your security requirements and performance considerations.
2. Generate a key pair: Use a cryptographically secure random number generator to create a private and public key pair for the chosen algorithm. The private key is used for signing the attestation data, while the public key is used by the verifier to validate the signature.
3. Secure the private key: It’s crucial to protect the private attestation key from unauthorized access or tampering. In a secure execution environment like Intel SGX, the private key can be generated and stored within the enclave, ensuring that it remains confidential and tamper-resistant. If the key is generated outside the enclave, make sure to securely store it in a hardware security module (HSM) or other secure storage.
4. Distribute the public key: Share the public key with the verifier or other trusted entities that need to validate the attestation data. It’s essential to ensure the authenticity and integrity of the public key during transmission, which can be achieved using a secure channel, digital signatures, or public key infrastructure (PKI).

5. Perform remote attestation: Use the private attestation key to sign the attestation data, which typically includes measurements of the software, its identity, and other relevant information. The verifier, who receives the signed attestation data, can then use the public key to validate the signature and establish trust in the remote software.

Remember that the specific steps and requirements for generating an attestation key may vary depending on the secure execution environment, the cryptographic algorithm, and the implementation of the remote attestation protocol.

Question: `[09:47]` Is there only one attestation key in a device? `(Papers.SGX.0419.md)`

Answer:

There isn't necessarily just one attestation key in a device. The number of attestation keys can vary depending on the device's architecture, the security requirements, and the use case.

In some cases, there may be a single attestation key for the entire device, which is used for all remote attestation purposes. This approach can simplify key management but may not provide fine-grained control over the different attestation scenarios.

In other cases, there may be multiple attestation keys within a device, each corresponding to a specific security domain, application, or purpose. For example, Intel SGX enclaves can have unique attestation keys, allowing for independent remote attestation of each enclave.

Using multiple attestation keys can provide more flexibility and fine-grained control over attestation and trust relationships. However, it may also introduce more complexity in key management and distribution.

In general, the choice of using one or multiple attestation keys will depend on factors such as the device's security requirements, the complexity of the system, and the specific use cases that need to be supported.

Question: `[09:49]` Where is attestation key sotred in Intel SGX arch? `(Papers.SGX.0419.md)`

Answer:

In Intel SGX, the attestation key is not directly stored within the enclave. Instead, the enclave uses the Intel Enhanced Privacy ID (EPID) provisioning and attestation services. The EPID private key is securely stored within the platform and is not accessible to the software, including the enclave code itself.

During the remote attestation process, an enclave will generate a quote, which contains information about the enclave's identity, such as MRENCLAVE and MRSIGNER, along with a measurement of the enclave's initial state. The quote is signed with the EPID private key, which is handled by the hardware and the SGX-enabled processor's quoting enclave.

The Intel SGX architecture ensures that the attestation key, in this case, the EPID private key, is securely managed and protected by the hardware, making it difficult for an attacker to compromise the key or forge an attestation.

Question: `[10:27]` What does REPORT contain? Please list them? `(Papers.SGX.0419.md)`

Answer:

A REPORT in Intel SGX is a data structure that contains information about an enclave, attesting its identity and integrity. The main components of a REPORT include:

1. MRENCLAVE: A hash representing the enclave's identity and code. It is generated during enclave creation and depends on the enclave's content and initial state.
2. MRSIGNER: A hash of the public key of the enclave's signer, which represents the identity of the Sealing Authority or enclave developer.
3. ISVPRODID: The product ID assigned by the enclave developer.
4. ISVSVN: The security version number assigned by the enclave developer, representing the security-related version or patch level of the enclave.
5. REPORTDATA: A user-defined data field, which can be used to store additional information, such as a challenge-response for the remote attestation process.
6. ATTRIBUTES: Flags indicating properties of the enclave, such as its access permissions, execution modes, or other relevant characteristics.

7. CPU_SVN: A value representing the security version number of the CPU and microcode.
8. MiscSelect: A value specifying the set of miscellaneous properties to include in the measurement.
9. Timestamp: The time at which the REPORT was generated by the CPU.

The REPORT is used as a part of the local and remote attestation processes to prove the identity, integrity, and authenticity of an enclave running on an SGX-enabled platform.

Question: `[14:18]` What is MRENCLAVE? `(Papers.SGX.0419.md)`

Answer:

MRENCLAVE is a hash value that represents the identity and code of an enclave in Intel SGX (Software Guard Extensions). It is generated during the enclave's creation process and is based on the enclave's content, initial state, and memory layout.

The MRENCLAVE value is calculated as a hash over the enclave's code and data, including its initial register values, memory layout, and other attributes. This hash uniquely identifies the enclave, as even a small change in the code or data would result in a different MRENCLAVE value.

MRENCLAVE is used in various security mechanisms, such as attestation, sealing, and key management, to ensure that the enclave's integrity and authenticity are maintained. It is included in the REPORT data structure, which is used for local and remote attestation processes to prove the identity, integrity, and authenticity of an enclave running on an SGX-enabled platform.

Question: `[14:19]` Who generates MRENCLAVE? `(Papers.SGX.0419.md)`

Answer:

MRENCLAVE is generated by the Intel SGX hardware during the enclave creation process. As the enclave is loaded into memory and initialized, the hardware computes the MRENCLAVE value by hashing the enclave's content, initial state, and memory layout. This ensures that the MRENCLAVE value uniquely represents the identity and integrity of the specific enclave.

Question: `[14:27]` Who creates an enclave? `(Papers.SGX.0419.md)`

Answer:

An enclave is created by the software running on a system that utilizes Intel SGX technology. The application, or a part of it, that wants to use SGX enclaves for secure execution of sensitive code and data initiates the enclave creation process. During this process, the enclave's code, data, and initial state are loaded into the enclave's protected memory region, and the hardware generates the MRENCLAVE value to uniquely represent the enclave.

Question: `[14:28]` So an untrusted software can also create an enclave. Right? `(Papers.SGX.0419.md)`

Answer:

Yes, even untrusted software can create an enclave. The primary purpose of Intel SGX is to allow secure execution of sensitive code and data within an enclave, even if the environment outside the enclave is not trusted. The untrusted software can create an enclave, but it cannot access or tamper with the code and data inside the enclave. Once the enclave is created and initialized, only the trusted code inside the enclave can execute and manipulate the data securely.

Question: `[14:30]` How to get the MRENCLAVE value of an enclave? `(Papers.SGX.0419.md)`

Answer:

To obtain the MRENCLAVE value of an enclave, you can use the EREPORT instruction provided by Intel SGX. The EREPORT instruction generates a REPORT structure, which includes the MRENCLAVE value of the target enclave. When you call EREPORT, you need to provide the target enclave's targetinfo structure as an input. The targetinfo structure includes

the MRENCLAVE value of the target enclave. Once the report is generated, you can extract the MRENCLAVE value from it.

Please note that obtaining the MRENCLAVE value typically involves communication between enclaves or between an enclave and a remote party during the attestation process. In this context, one enclave (reporting enclave) calls the EREPORT instruction to create a report for another enclave (target enclave). The target enclave can then verify the report's authenticity and extract the MRENCLAVE value.

Question: `[14:37]` I have enclave A and enclave B. How does enclave A know the enclave B before A wants to get MRENCLAVE value of B? `(Papers.SGX.0419.md)`

Answer:

Before enclave A can get the MRENCLAVE value of enclave B, it needs some knowledge about enclave B. Typically, this knowledge is shared through some out-of-band mechanism or by the application that hosts both enclaves. Here's a high-level overview of the process:

1. The application creates enclave A and enclave B.
2. The application shares some information about enclave B with enclave A. This could be done by passing the targetinfo structure of enclave B to enclave A, which contains the MRENCLAVE value of enclave B.
3. Enclave A uses the shared information about enclave B to call the EREPORT instruction and generate a report targeting enclave B.
4. The report generated by enclave A is passed to enclave B, either by the application or through some inter-enclave communication mechanism.
5. Enclave B verifies the report's authenticity using the EGETKEY instruction to ensure it comes from enclave A. If the report is valid, enclave B can extract the MRENCLAVE value of enclave A from the report.

Keep in mind that the specific implementation details and communication methods may vary depending on the application's design and security requirements. The above steps are a high-level description of the process.

SGX.0418.md

Question: `[09:50]` Where is attestation key sotred in Intel SGX arch? `(SGX.0418.md)`

Answer:

In the Intel SGX architecture, the attestation key is not stored directly on the platform. Instead, the hardware-protected enclave generates an ephemeral attestation key pair during the attestation process.

The private attestation key is created and kept within the enclave, and it is not accessible by any other software or hardware outside the enclave. The public part of the attestation key pair is included in the attestation report, which is then sent to a remote party for verification.

The attestation process also involves the use of Intel's Enhanced Privacy ID (EPID), which is a group signature scheme that provides anonymity and revocability. The EPID private key is embedded in the processor during manufacturing and is securely stored inside the processor. The corresponding EPID group public key is maintained by Intel, allowing verification of EPID signatures.

To summarize, the attestation keys in Intel SGX are stored as follows:

1. Ephemeral attestation key pair: Generated within the enclave during the attestation process. The private key is kept inside the enclave, while the public key is included in the attestation report.
2. EPID private key: Embedded in the processor during manufacturing and securely stored inside the processor.

Question: `[09:52]` What is attestation key? `(SGX.0418.md)`

Answer:

An attestation key is a cryptographic key used in the process of attestation to prove the authenticity and integrity of a system, software, or hardware component. The attestation key is typically an asymmetric key pair, consisting of a private key and a

public key.

In the context of secure enclaves or trusted execution environments like Intel SGX, the attestation key is used to sign data, such as measurements or claims about the secure environment, to provide a proof that the data is genuine and has not been tampered with. The private attestation key is kept securely within the enclave, while the public key is used by a verifier or remote party to validate the signature and ensure the integrity and authenticity of the data.

Attestation keys play a critical role in establishing trust in systems, as they provide a strong cryptographic guarantee that the information being attested to is genuine and originates from a trusted source.

Question: `[09:53]` Is there only one attestation key in a device? `(SGX.0418.md)`

Answer:

In a device that supports hardware-based attestation, there can be multiple attestation keys depending on the specific technology being used and the device's intended purposes. For example, with Intel SGX, a device can have multiple enclaves, each with its own set of keys for secure operations.

In the case of TPM (Trusted Platform Module), a single device can have multiple attestation keys, each with a different purpose. For instance, there may be separate keys for platform configuration, software integrity, and user identity. These keys can be generated on-demand by the TPM and securely stored within the module.

In summary, a device can have more than one attestation key depending on the specific attestation technology being used and the device's security requirements.
