

# **Распределенные вычислительные системы**

## **Лекция №10: Безопасность**

Алексей В. Бурдаков, к.т.н.  
burdakov@usa.net

# План лекций

№	Дата	Тема
1	04.09	Вводная лекция
2	11.09	Эволюция распределенных технологий
3	18.09	Принципы ПО среднего слоя
4	25.09	Стандарты OMG, CORBA и ORB
5	02.10	Пример приложения на CORBA
6	09.10	SOM, Java/RMI
7	16.10	Определение местонахождения
	23.10	Перенос
8	30.10	Долговременное хранение + <u>Рейтинг по Л.1-7</u>
9 /		
10	06.11	Долговременное хранение

# План лекций (продолжение)

---

№	Дата	Тема
11	13.11	-
12	20.11	-
13	27.11	Распределенные объектные транзакции
14	04.12	-
15	11.12	Безопасность
16	18.12	Лекция / Рейтинг №2
17	25.12	Зачет

# План лекции

---

- Виды атак
- Шифрование
- Высокоуровневые сервисы
- Сервисы безопасности в объектно-ориентированном среднем слое ПО

# Проблема

---

- Все больше жизненно важной и секретной информации обрабатывается распределенными вычислительными системами
- Безопасность: защита данных, хранимой и передаваемой между распределенными компонентами от неавторизованного доступа
- Безопасность является нефункциональным требованием, которое не может быть добавлено в виде отдельного компонента, а должно быть встроено во все компоненты

# Почему распределенные системы небезопасны?

---

- Распределенные компоненты полагаются на компоненты, которые посылаются через сеть
- Публичные сети небезопасны!
- Является ли клиентский компонент безопасным?
- Является ли клиентский компонент тем, за кого он себя выдает?
- Являются ли пользователи компонентов, осуществляющих вызовы теми за кого они себя выдают?

# Необходимость защиты

---

- Потеря доверия к системе: приведенное выше может привести к потере доверия к системе
- Иски по возмещению ущерба: в случае если данные не были защищены надлежащим образом, это может привести к возникновению исков на возмещение ущерба
- Разглашение конфиденциальных данных: данные, сохраняемые на компьютере могут быть конфиденциальными (медицинские или кадровые данные)

# Последствия небезопасности

---

- Конфиденциальные данные могут быть украдены, как то:
  - Корпоративные планы
  - Проекты новых продуктов
  - Медицинские или финансовые записи
- Данные могут быть подменены, например:
  - Финансовые записи будут выглядеть лучше чем то, что они из себя представляют на самом деле
  - Результаты тестов (на наркотики, и т.п.)
  - Подмена результатов экзаменов

# Угрозы

---

- Категоризация атак, которые могут быть предприняты
- Четыре основные идеи:
  - Утечка (leakage): информация похищается из системы
  - Подмена (tampering): информация подменяется в системе
  - Кража ресурсов (resource stealing): нелегальное использование ресурсов
  - Вандализм (vandalism): нарушение корректной работы системы
- Используется для определения того от чего система защищена

# Методы атак

---

- Подслушивание (eavesdropping): получение копий сообщений без обладания права на то
- Маскировка (masquerading): использование идентификатора другого принципала (лица) без права на то
- Подмена сообщений (tampering): перехват и подмена сообщений
- Повтор (replaying): перехват, сохранение и последующая повторная посылка сообщений

# Проведение атак

---

- Инициирование атак требует доступа к системе:
  - Иницируется законными пользователями
  - Запускается после получения паролей известных пользователей
- Некоторые пути проведения атак:
  - Вирусы
  - Черви
  - Троянские кони

# План лекции

---

- Виды атак
- Шифрование
- Высокоуровневые сервисы
- Сервисы безопасности в объектно-ориентированном среднем слое ПО

# Введение

---

- Криптография: шифрует данные, содержащиеся в сообщении, так что они могут быть прочитаны только желаемыми получателями
- Римляне использовали шифрование в коммуникации между военными подразделениями
- Зная об алгоритме шифрования можно попробовать использовать метод грубой силы:  
необходимо использовать все варианты до тех пор пока сообщение не будет окончательно декодировано
- Компьютеры выполняют эту работу достаточно хорошо
- Для того чтобы оставаться защищенными, современные методы защиты должны быть вычислительно сложными

# Терминология криптографии

---

- Исходное сообщение: сообщение до кодирования
- Закодированное сообщение: сообщение после кодирования
- Ключ: информация, необходимая для преобразования из исходного в закодированное
- Функция: алгоритм шифрования или дешифрования, использованные совместно с ключом для кодирования или декодирования сообщения
- Сервис распределения ключей: доверенный сервер с набором ключей

# Шифрование

---

- Шифрование данных предотвращает неавторизованный доступ и модификацию данных (например, предотвращает подслушивание и подмену)
- Если дешифрование может быть выполнено только с помощью совпадающего ключа, то это может быть использовано для доказательства идентификации запрашивающего (предотвращает маскировку)
- Точно также может использоваться для того, чтобы обеспечить уверенность в том, что только авторизованный пользователь может использовать информацию
- Два основных способа: секретные и открытые ключи

# Секретные ключи

---

- Один ключ используется для шифрования и дешифрования данных
- Функции шифрования и дешифрования часто используются те же самые
- Защищенность никак не снижается при публикации функций, так как защищенность обеспечивается закрытыми ключами

# Использование секретных ключей

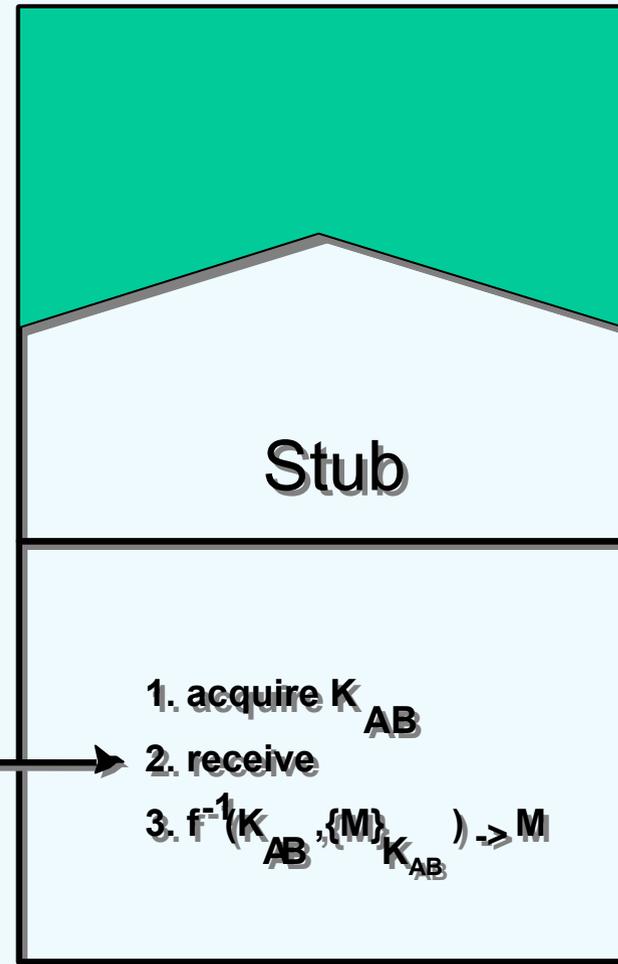
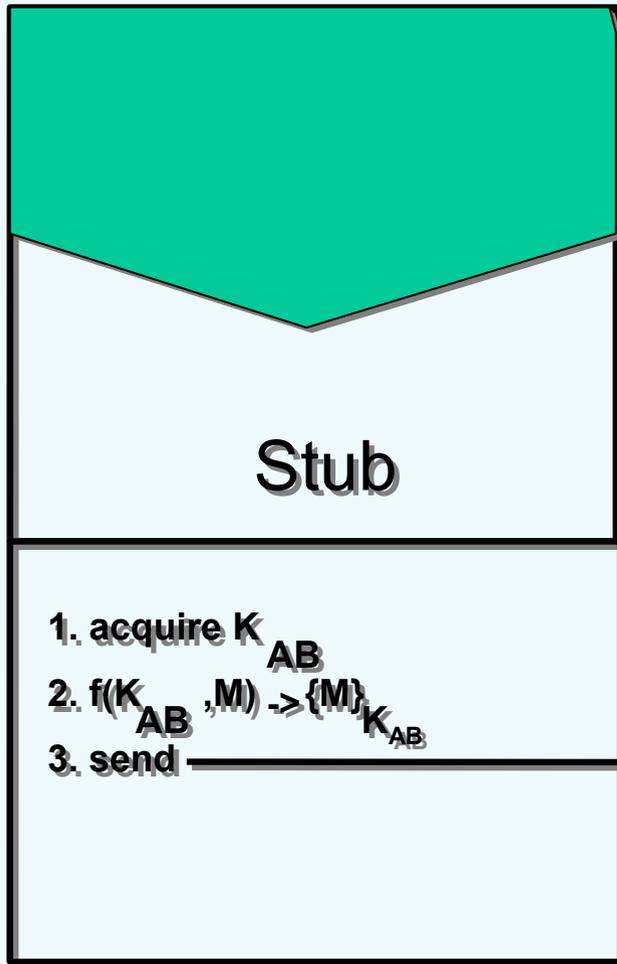
---

- Посылающий и принимающий получают секретные ключи посредством использования доверенных, закрытых не сетевых каналов
- Посылающий шифрует сообщение используя функцию и отсылает его, зная, что только держатель ключа может использовать ее
- Получатель декодирует сообщение и знает, что оно могло быть сгенерировано только отправляющей стороной
- Сообщение может быть перехвачено, однако является бесполезным

# Использование секретных ключей в объектном запросе

Client A

Server B



$\{M\}_{K_{AB}}$

# Открытые ключи

---

- Обеспечивает секретность только в одну сторону
- Генерируются два ключа, где один используется для алгоритма дешифрования (закрытый ключ) и один для шифрования (открытый ключ)
- Генерация закрытого ключа при наличии открытого является вычислительно сложной задачей
- Не требует защищенной передачи ключей

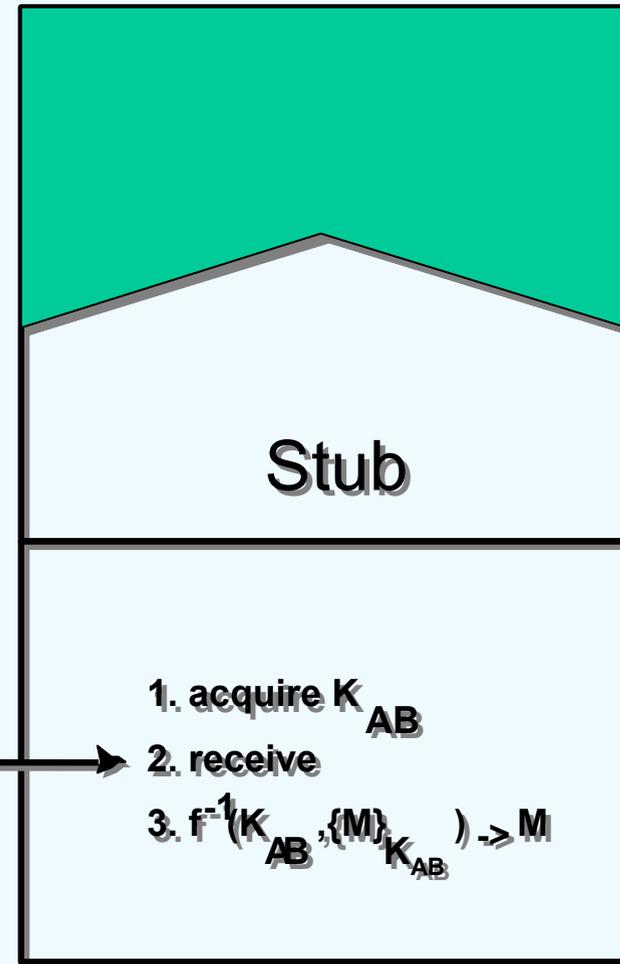
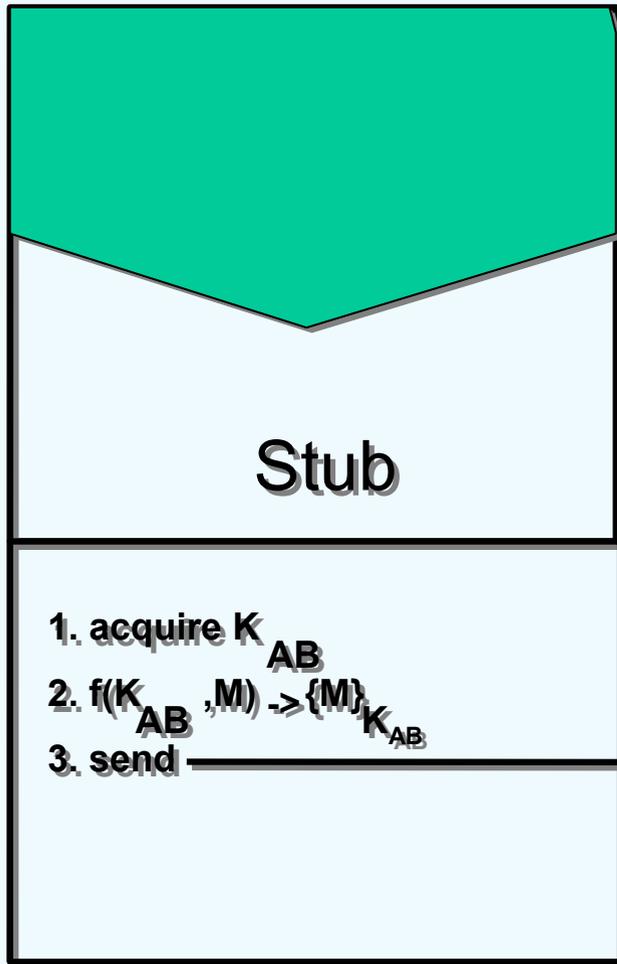
# Использование открытых ключей

- Получатель генерирует пару ключей
- Открытый ключ публикуется доверенным сервисом
- Отправитель получает открытый ключ и использует его для кодирования сообщения
- Получатель декодирует сообщение
- Ответы могут быть закодированы с использованием открытых ключей от доверенных сервисов распространения
- Сообщение может быть перехвачено, однако не представляет никакого смысла

# Использование открытых ключей в объектном запросе

Client A

Server B



$\{M\}_{K_{AB}}$

# Проблема: каким образом распространять ключи?

- Для распределенных объектных систем распространение без использования сетевых соединений является не практичным
- Распределение ключей является проблемой как для секретных, так и для открытых ключей:
- Секретные ключи: очевидно
- Открытые ключи: каким образом мы можем убедиться, что принципал, передающий нам ключ является именно тем, за кого себя выдает?
- Использование доверительного сервиса распространения ключей и защищенного протокола распространения ключей

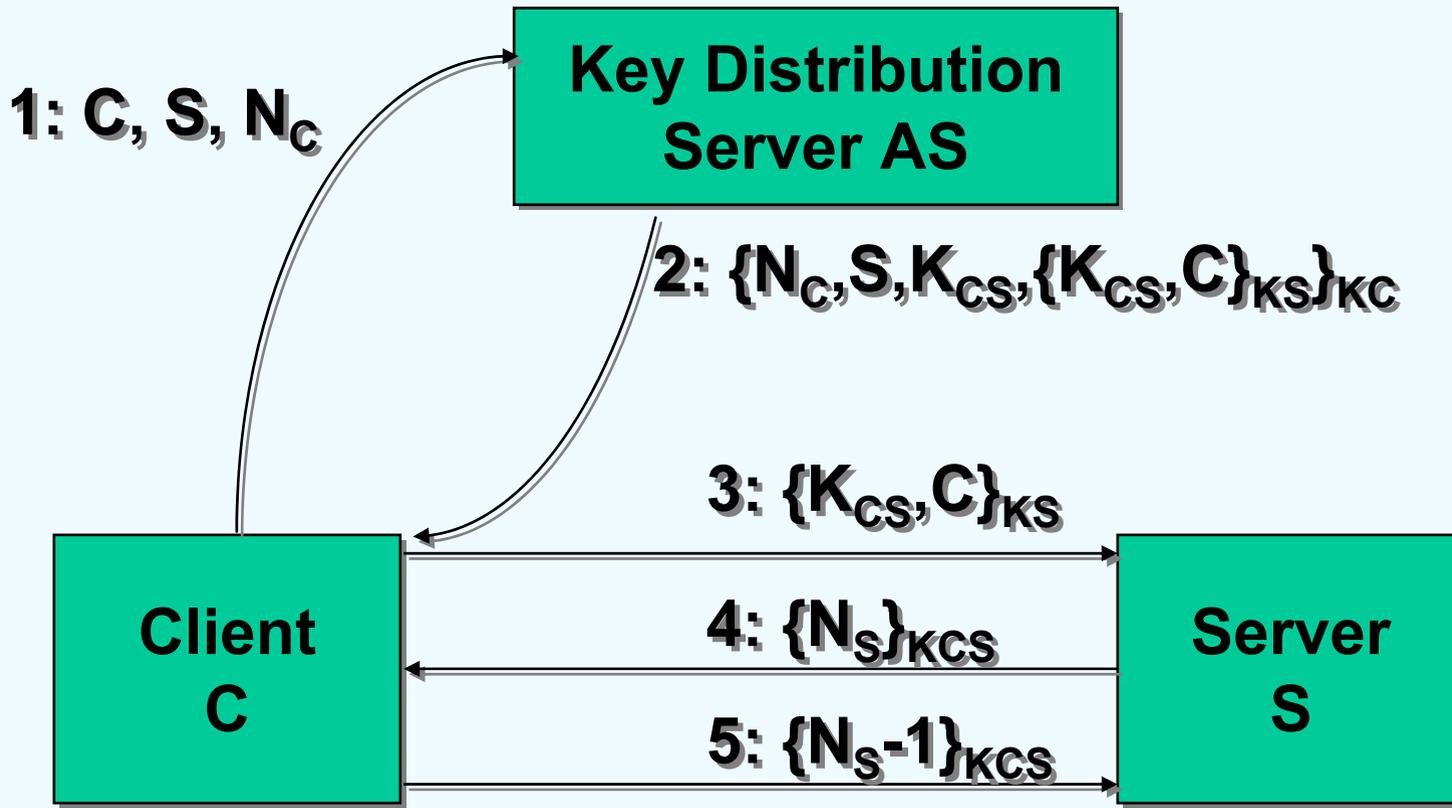
# Протокол Нидхэма/Шредера

---

- Обеспечивает безопасный способ для пар компонентов получить ключи для последующего взаимодействия
- Основанный на аутентификационных серверах:
- Содержит ключ и имя для каждого компонента
- Может генерировать ключи для взаимодействия точка-точка
- Секретные ключи используются для взаимодействия с серверами

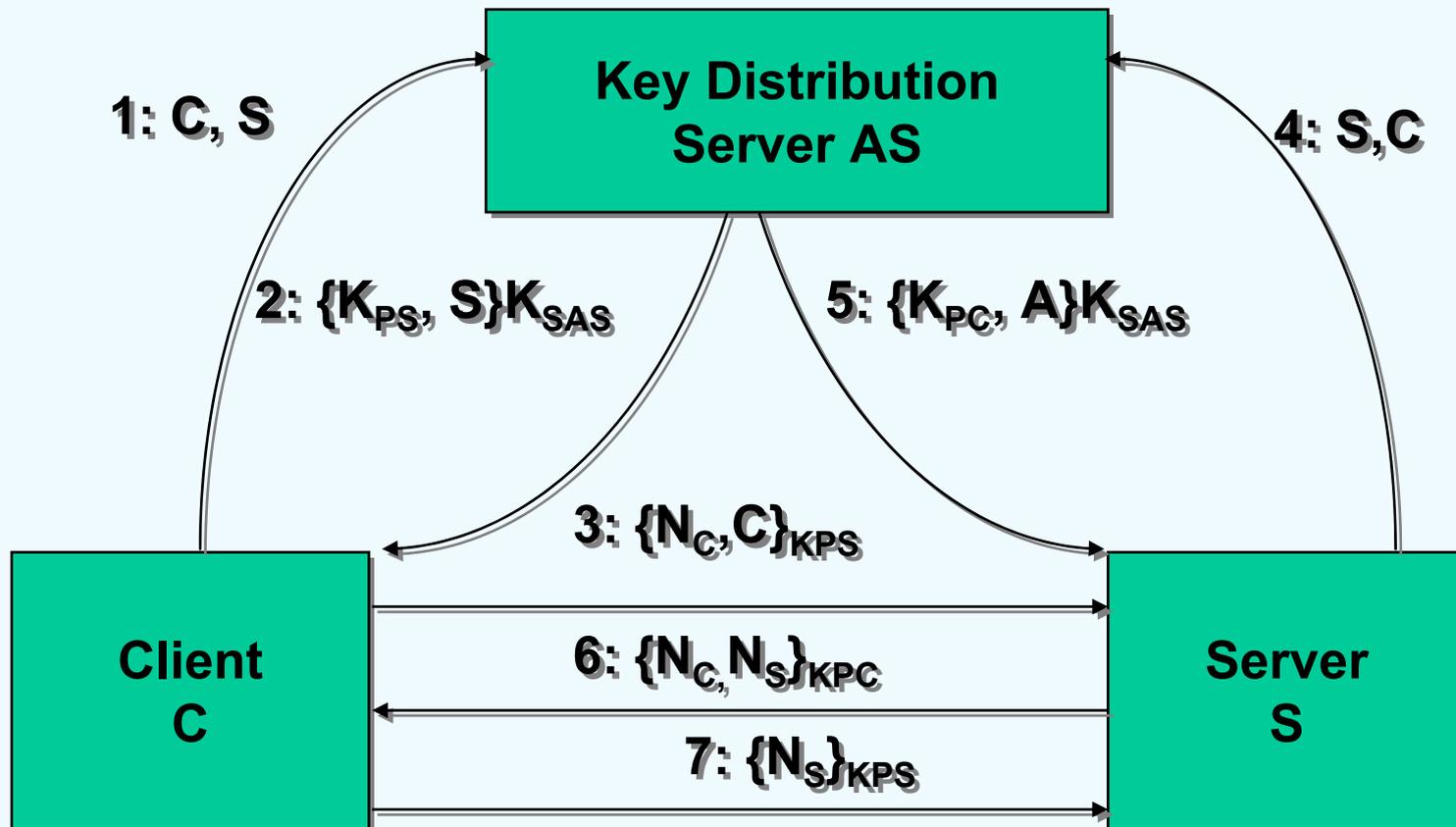
# Протокол Нидхэма/Шредера

- Секретный ключ:



# Протокол Нидхэма/Шредера

- Открытый ключ:



# Протокол Secure Socket Layer (SSL)

- Защищенный транспортный протокол между Веб-клиентом и Веб-сервером
- Также используется для объектно-ориентированного ПО среднего слоя
- Основан на технологии RSA для открытых ключей
- Клиент генерирует секретный ключ сессии
- Клиент использует открытый ключ сервера для шифрования ключа сессии и передает его на сервер
- Ключ сессии используется для шифрования и взаимодействия между клиентом и сервером

# План лекции

---

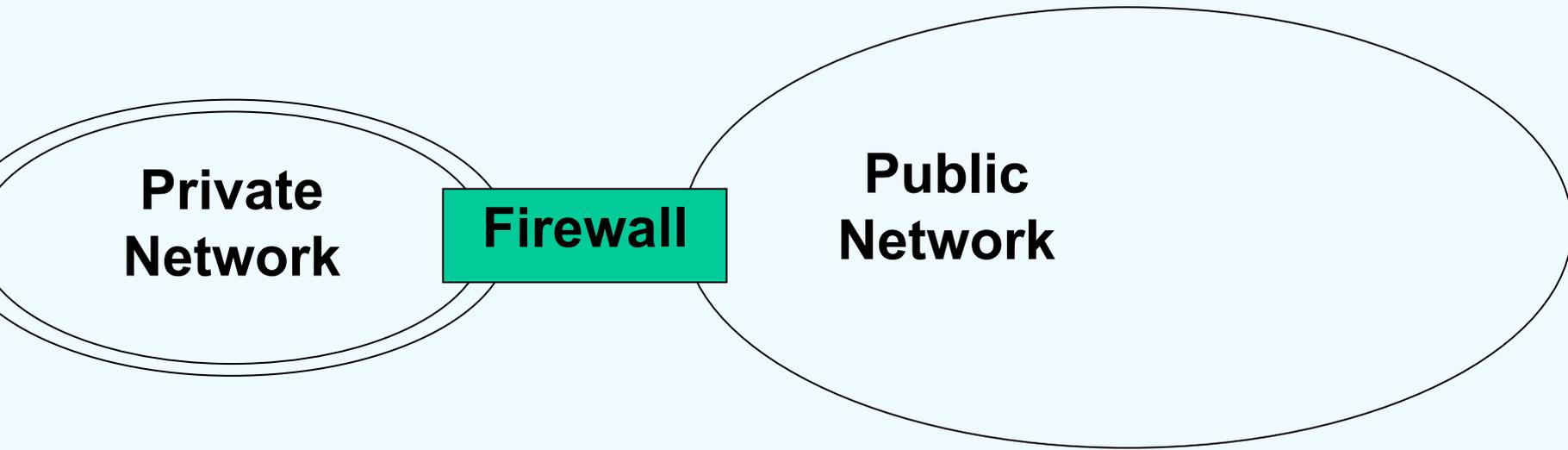
- Виды атак
- Шифрование
- Высокоуровневые сервисы
- Сервисы безопасности в объектно-ориентированном среднем слое ПО

# Что необходимо для осуществления безопасных вызовов?

- Разделение общедоступных и частных сетей с помощью межсетевых экранов (firewalls)
- Установка безопасного взаимодействия между клиентом и сервером (аутентификация)
- Решение о том может ли принципал осуществить данную операцию (контроль доступа)
- Обеспечение неопровержимости доступа к операции (аудит)
- Защита запросов и ответов от подслушивания в процессе передачи (шифрование)
- Неопровержимость доставки

# Межсетевые экраны

---



# Межсетевые экраны

---

- Межсетевые экраны – шлюзы, которые контролируют поток сообщений между частными и общими сетями
- Межсетевые экраны между распределенными объектами должны понимать кодировку объектных запросов
- Межсетевые экраны должны быть интегрированы в механизмы шифрования, используемые для защищенной транспортировки

# Что такое аутентификация?

---

- Аутентификация: доказательство того, что аутентифицируемый является именно тем, за кого себя выдает
- В централизованных системах: проверка пароля во время установления сессии
- В распределенных системах:
- Использование аутентификационного сервера
- Основан на возможности шифрования/дешифрования сообщений (протокол Нидхэма/Шредера)

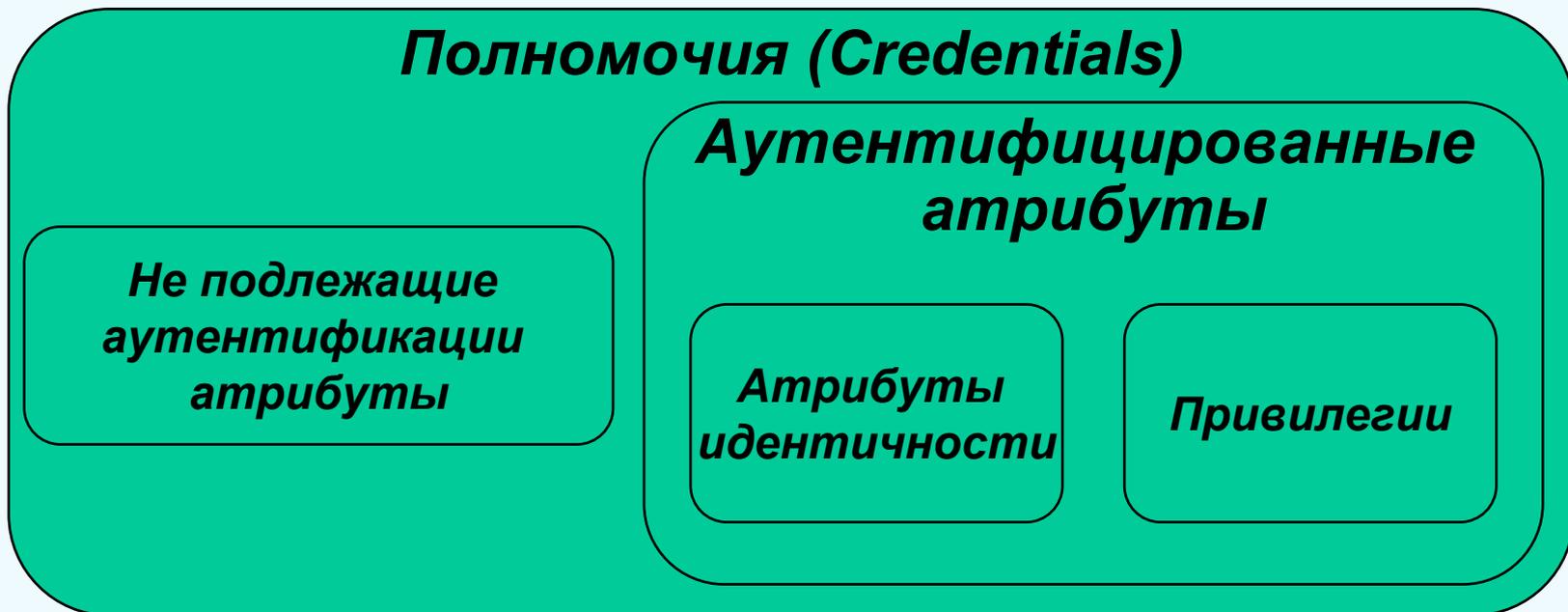
# Принципалы

---

- Человек или системный компонент, который зарегистрирован и является аутентичным по отношению к распределенной системе
- Принципалы обладают идентификаторами, используемыми для:
  - Доказательства неопровержимости действий, осуществленных принципалами
  - Получения доступа к защищенным компонентам
  - Идентификации источников сообщений
  - Идентификации тех, кто должен оплатить предоставление сервиса

# Полномочия

- Информация, которой обладает системе о принципале:



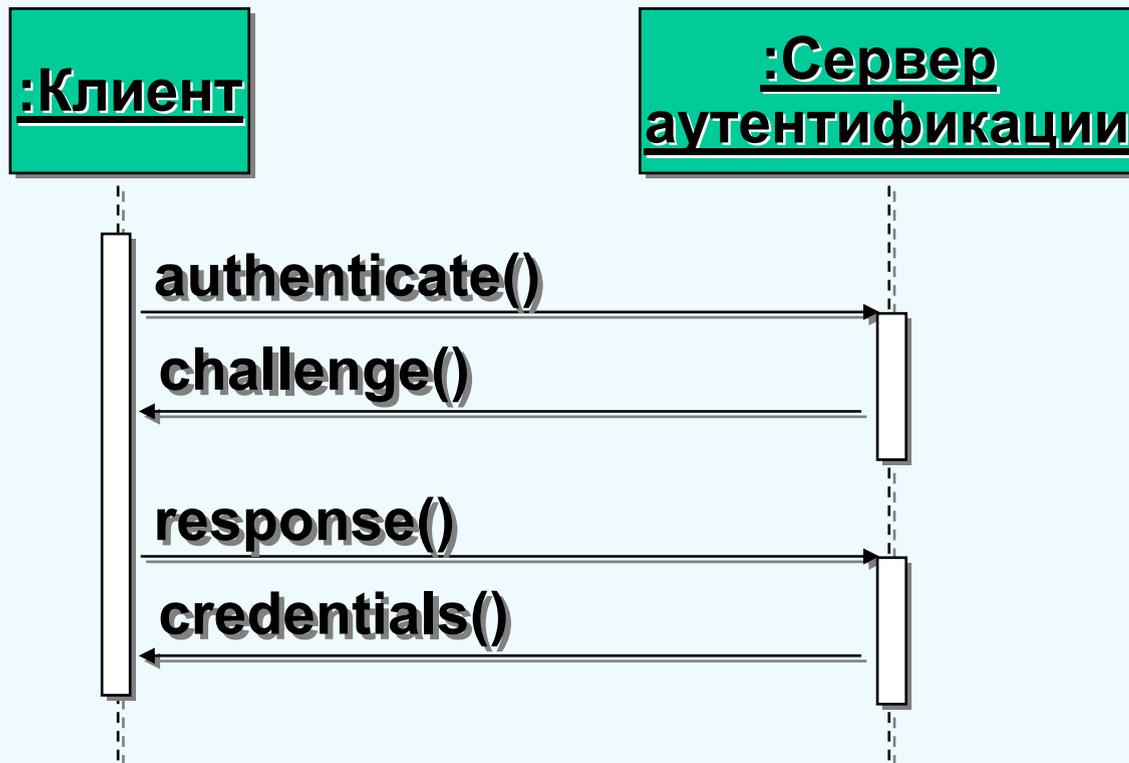
# Аутентификация

---

- Установка доверительных отношений в обоих компонентах
  - Клиент аутентифицирует сервера
  - Сервер аутентифицирует клиента
- Полномочия клиента доступны серверу
- Установка контекста безопасности, используемого для защиты запросов, используемых для защиты запросов и ответов в транзите (например, распределенные закрытые ключи)

# Аутентификация

- Использование протокола вызова-отклика и его использование для аутентификации



# Контроль доступа

---

- Базовый вопрос, решаемый контролем доступа: может ли операция, запрашиваемая принципалом, быть выполнена
- Основывается на аутентификации: подразумевается, что принципал и его полномочия были проверены
- Две формы:
  - Политики доступа к вызовы объектов
  - Политики доступа к прикладным объектам

# Политики доступа к прикладным объектам

---

- В предыдущем случае контроль доступа прозрачен и для сервера и для клиента
- В данном случае клиент и/или сервер реализует контроль доступа самостоятельно
- Политики контроля доступа:
  - Могут принимать во внимание данные, к которым предполагается осуществить доступ
  - Могут принимать во внимание семантику запрошенных параметров

# Атрибуты привилегий контроля доступа

- Атрибуты привилегий принципала для контроля доступа включают:
  - Идентификатор принципала
  - Роли (относящиеся к функциям принципала)
  - Группы (относящиеся к организационной структуре, в которую внедрен принципал)
  - Допуск
  - Возможности серверного объекта, которые принципал может использовать
  - Другие ...

# Атрибуты контроля сервера

---

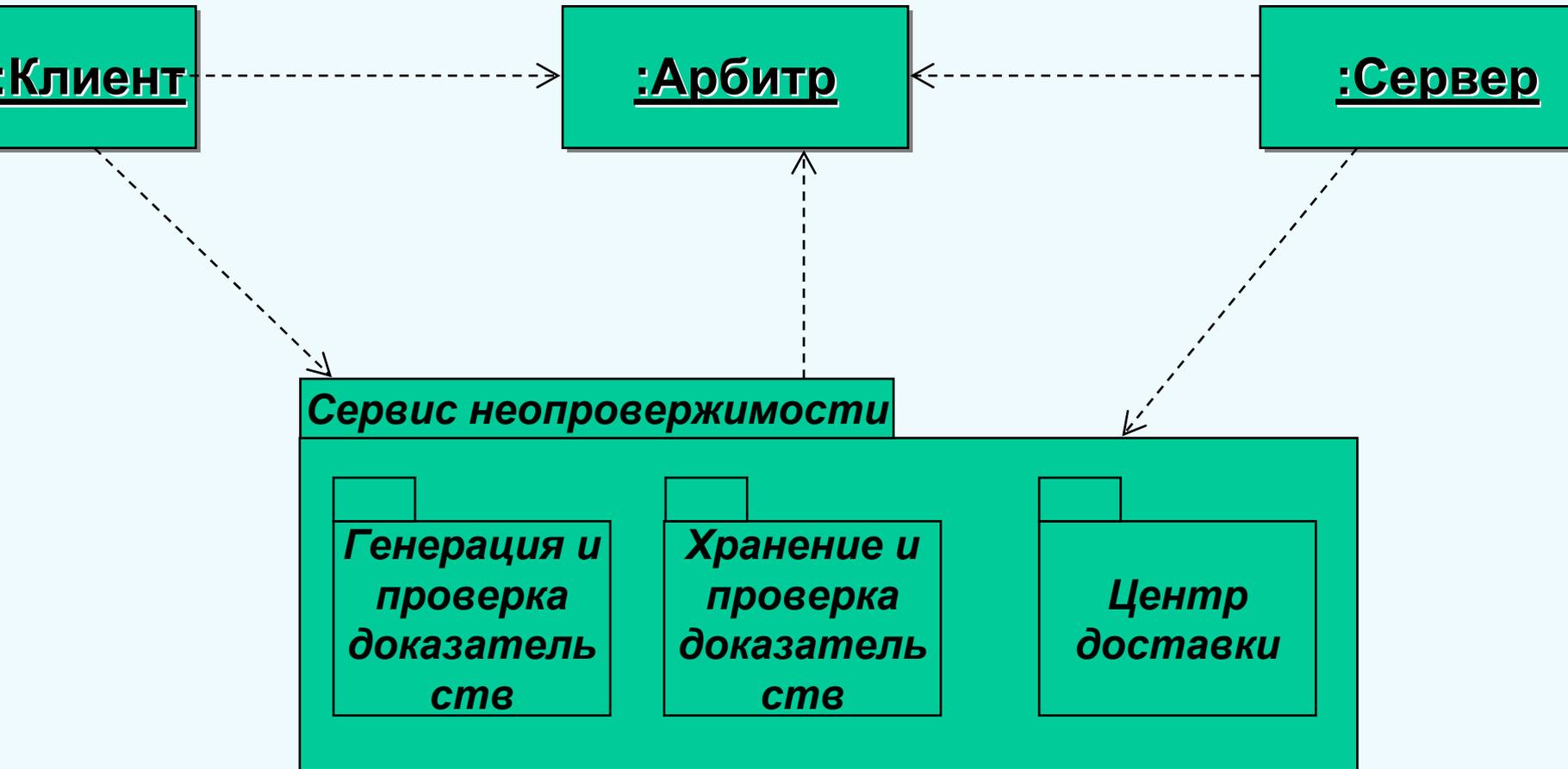
- Списки контроля доступа (ACL) идентифицируют пользователей по:
  - Имени
  - Атрибутам привилегий
- Информация по схемам, основанным на метках
- Атрибуты контроля доступа часто разделяются группами операций и даже группами объектов

# Неопровержимость

---

- Предоставляет неопровержимые доказательства о выполненных операциях принципалами
- Неопровержимое доказательство о событии/действии
- Используется как доказательство в случае отрицания выполнения операции
- Пример: электронная коммерция

# Сервис неопровержимости



# Компоненты доказательств

---

- Зависит от политики неопровержимости
- Примеры:
  - Типы действий и событий
  - Дата, полученная от доверенного сервиса
  - Параметры, относящиеся к действиям и событиям
  - Доказательство источника параметров

# Общие типы доказательств

---

- Доказательство создания сообщения
  - Защищает от ложного отрицания факта создания сообщения
- Доказательство получения сообщения
  - Защищает от ложного отрицания получения сообщения

# Аудит безопасности

---

- Помогает проводить аудит попыток или действительных нарушений защиты
- Записывая детали событий, относящихся к вопросам безопасности
  - Записывая детали событий в журнальный файл
  - Генерируя сигналы нарушения безопасности
  - Принимая другие меры
- Два уровня безопасности
  - Системный
  - Прикладной

# Политики аудита безопасности

---

- Потенциально большое число записей может быть записано
- Политики аудита безопасности определяют набор событий, которые должны быть записаны и являются критичными для каждой конкретной обстановки
- Политики системного аудита записывают все события, относящиеся к безопасности, даже со стороны приложений без специальных функций обеспечения безопасности

# План лекции

---

- Виды атак
- Шифрование
- Высокоуровневые сервисы
- Сервисы безопасности в объектно-ориентированном среднем слое ПО

# Сервисы безопасности

---

- Сервисы безопасности CORBA обеспечивают поддержку и реализуют принципы, обсужденные выше
- Два уровня
  - Уровень 1: Безопасность является обязательной для соответствия и определяет требования по реализации аутентификации, контроль доступа и аудит
  - Уровень 2: Безопасность является необязательной и касается только неопровержимости

# Сервисы безопасности

- Полномочия – список атрибутов привилегий, предоставленных принципалу

```
interface Credentials {
    Credentials copy();
    void destroy();
    ...
    boolean set_privileges(
        in boolean force_commit,
        in AttributeList requested_privileges,
        out AttributeList actual_privileges);
    AttributeList get_attributes(
        in AttributeTypeList attributes);
    boolean is_valid (out UtcT expiry_time);
    boolean refresh();
};
```

# Сервисы безопасности

- Интерфейс аутентификации, используемый сессией начинает проверять полномочия принципала

```
interface PrincipalAuthenticator {  
    AuthenticationStatus authenticate(  
        in AuthenticationMethod method,  
        in SecurityName security_name,  
        in Opaque auth_data,  
        in AttributeList privileges,  
        out Credentials creds,  
        out Opaque continuation_data,  
        out Opaque auth_specific_data);  
    ...  
}
```

# Сервисы безопасности

---

- Аутентифицированные полномочия принципала становятся доступными серверному объекту через интерфейс Current, предоставляемый как часть интерфейса ORB

```
interface Current {  
    CredentialList get_credentials(  
        in CredentialType cred_type);  
    ...  
}
```

# Сервис контроля доступа CORBA

---

- Права доступа CORBA имеют три составляющих:
  - Полномочия
  - Политики доступа
  - Права доступа
- Политики доступа сопоставляют права доступа с привилегиями
- Права доступа определяют привилегии, которые принципалы должны держать для того чтобы иметь возможность запроса операций

# Контроль доступа CORBA

- Основан на интерфейсе принятия решений

```
typedef sequence <Credentials> CredentialsList;  
interface AccessDecision {  
    boolean access_allowed(  
        in CredentialsList cred_list,  
        in Object target,  
        in CORBA::Identifier operation_name,  
        in CORBA::Identifier target_interface_name);  
};
```

# Аудит CORBA

- Интерфейсы определяют политики аудита
- Политики определяют:
- Какие события должны быть журналированы
- Где должны быть записаны события

```
interface AuditPolicy : CORBA::Policy {  
    void set_audit_selectors (  
        in CORBA::InterfaceDef object_type,  
        in AuditEventTypeIdList events,  
        in SelectorValueList selectors);  
  
    ...  
  
    void set_audit_channel (  
        in AuditChannelId audit_channel_id);  
};
```

# Неопровержимость CORBA

```
interface NRCredentials : Credentials {
    void generate_token(in Opaque input_buffer,
                       in EvidenceType generate_evidence_type,
                       in boolean include_data_in_token,
                       in boolean generate_request,
                       in RequestFeatures request_features,
                       in boolean input_buffer_complete,
                       out Opaque nr_token,
                       out Opaque evidence_check);
    NRVerificationResult verify_evidence(
        in Opaque input_token_buffer,
        in Opaque evidence_check,
        in boolean form_complete_evidence,
        in boolean token_buffer_complete,
        out Opaque output_token,
        out Opaque data_included_in_token,
        out boolean evidence_is_complete,
        out boolean trusted_time_used,
        out TimeT complete_evidence_before,
        out TimeT complete_evidence_after);
    ...
};
```

# Ключевые вопросы

---

- Сетевое взаимодействие наследственно небезопасно
- Шифрование может быть использовано для предотвращения подслушивания и подделки сообщений
- Ключи шифрования распространяются доверенными сервисами по протоколу распространения ключей, например, через протокол Нидхэма/Шредера
- Шифрование также используется для аутентификации

# Ключевые вопросы

---

- Контроль доступа основан на аутентификации и определяет имеет ли принципал права на запрос выполнения операции
- Неопровержение используется для генерации неопровержимых доказательств того, что принципал запросил выполнение операции
- Аудит является пассивной мерой безопасностью, которая записывает события в соответствии с политикой безопасности

# Литература / Internet

## ИСТОЧНИКИ

- В. Эммерих *Конструирование распределенных объектов*. - М.:Мир. - 2002.
- Vinoksi S. *CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments*, IEEE'96.
- Schmidt D.C. And Vinoski S. *Object Adapters: Concepts and Technology*, SIGS C++ Report, Vol. 9, No. 11, Nov-Dec 1997.
- Ю.А. Григорьев, А.Д. Плутенко. *Жизненный цикл проектирования распределенных баз данных*. - Благовещенск. - 1999.
- [www.omg.org](http://www.omg.org)