Федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»

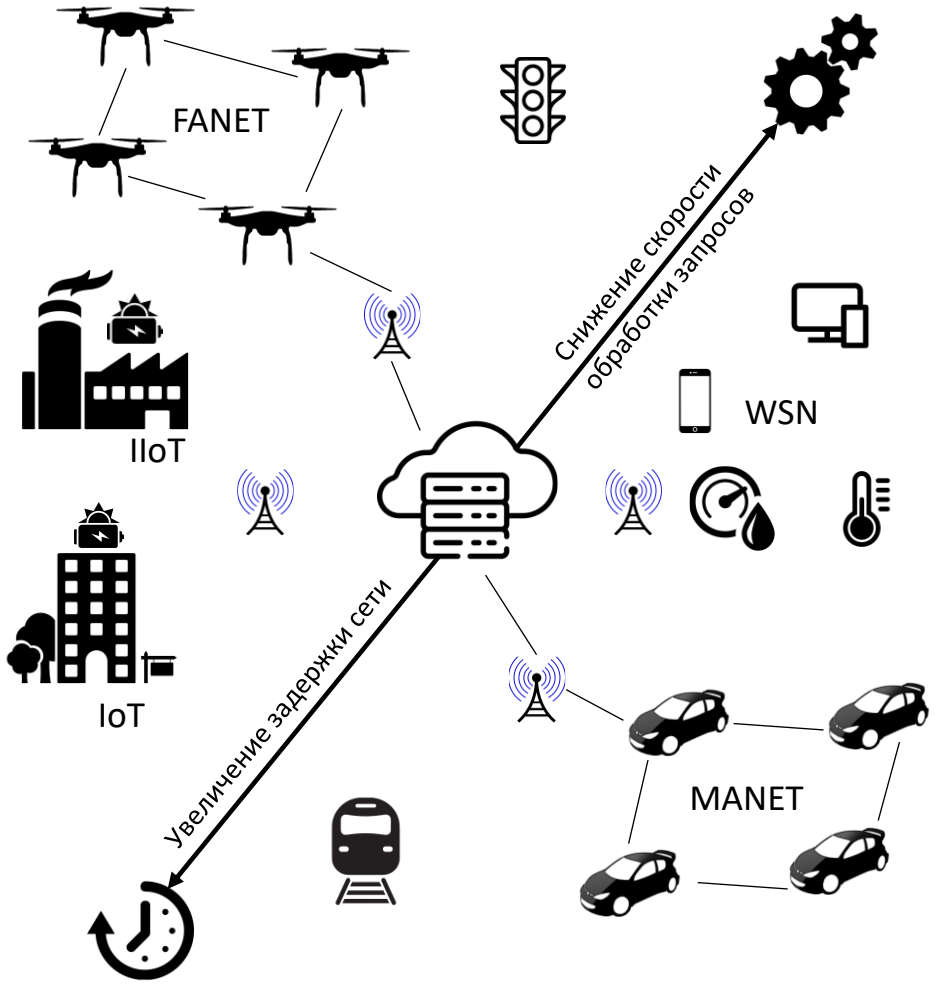
АУТЕНТИФИКАЦИЯ УСТРОЙСТВ НИЗКОРЕСУРСНЫХ КИБЕРФИЗИЧЕСКИХ СИСТЕМ В ГРАНИЧНОЙ ВЫЧИСЛИТЕЛЬНОЙ АРХИТЕКТУРЕ

Шкоркина Елена Николаевна,

Александрова Елена Борисовна

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-37-90110.

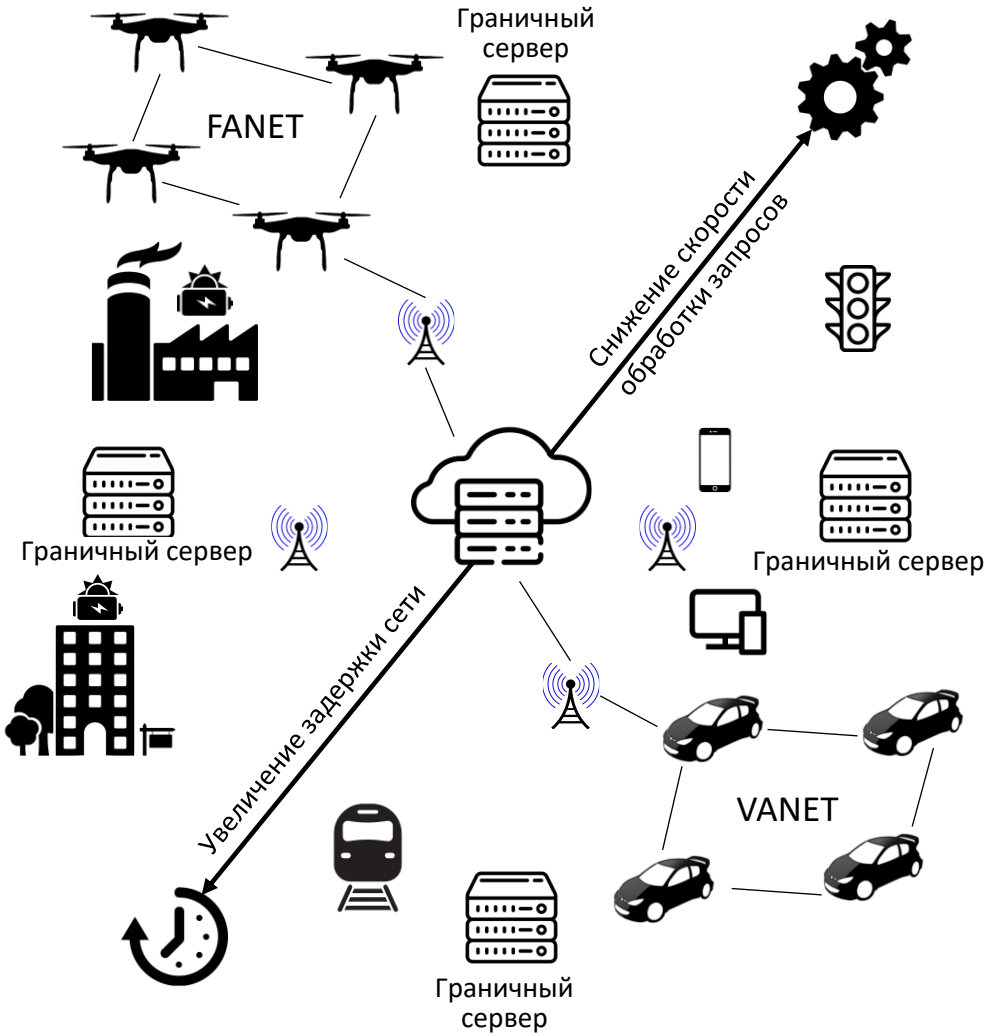
НИЗКОРЕСУРСНЫЕ КИБЕРФИЗИЧЕСКИЕ СИСТЕМЫ



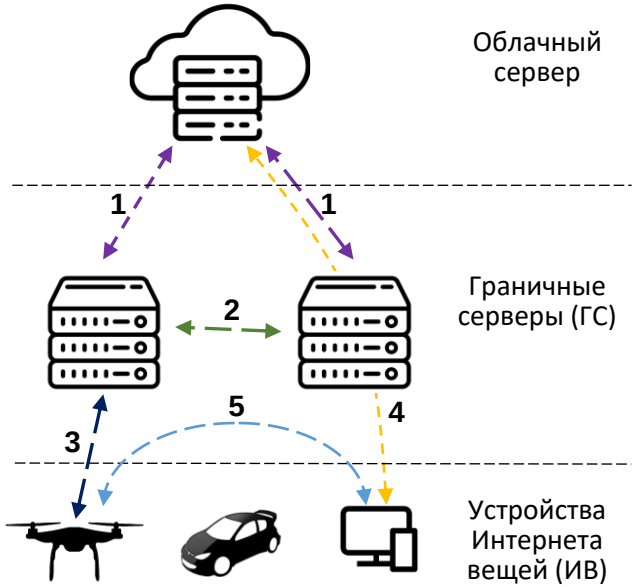
Тесное взаимодействие физических и информационных процессов ⟷ Преимущественно высокая динамичность структуры сети ⟷ Использование разнородных аппаратных платформ и криптографических протоколов

Основная цель защиты киберфизической системы – обеспечение непрерывности процесса управления в условиях дестабилизирующих воздействий

ПЕРЕХОД ОТ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ К ГРАНИЧНЫМ



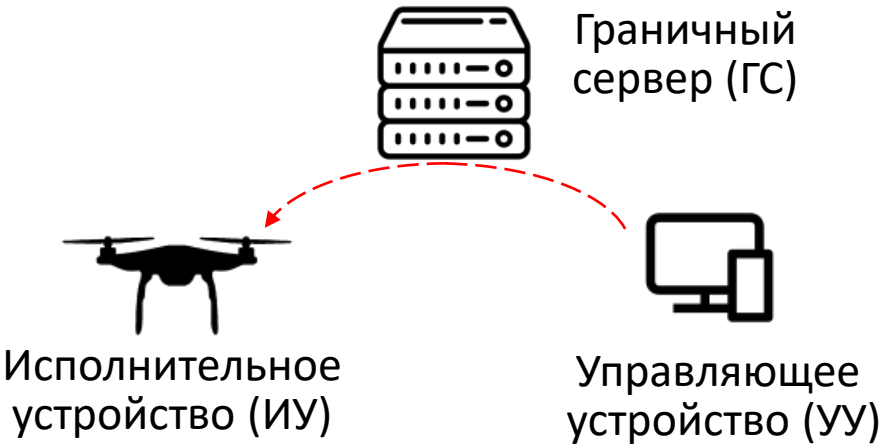
Облачные вычисления	Граничные вычисления
❖ Обработка больших данных	❖ Обработка больших данных
❖ Невозможность работы в режиме реального времени	❖ Принятие решений в режиме реального времени
❖ Физическое ограничение скорости передачи данных	❖ Повышение скорости передачи данных
	❖ Делегирование вычислений
	❖ Повышение безопасности подсетей связанных устройств
	❖ Сочетание с 5G
	❖ Необходимость адаптации существующих, либо разработки новых протоколов взаимодействия



1. Граничный сервер ↔ Облачный сервер
2. Граничный сервер ↔ Граничный сервер
3. Устройство ИВ ↔ Граничный сервер
4. Устройство ИВ $\overset{ГС}{\leftrightarrow}$ Облачный сервер
5. Устройство ИВ $\overset{ГС}{\leftrightarrow}$ Устройство ИВ

ИСПОЛЬЗУЕМЫЕ В РАЗРАБОТАННЫХ ПРОТОКОЛАХ ОБОЗНАЧЕНИЯ

Обозначение	Описание
$Sign_{ES}$	Сформированное доверенной стороной значение подписи уникальных данных ID_{ES}, идентифицирующих граничный сервер в сети
$r,\quad q$	Случайные значения, генерируемые граничным сервером
K_{auth}	Ключ аутентификации УУ на ИУ, используемый в качестве ключа алгоритма симметричного шифрования
$Pk_{Dev_1}^{ENC}$ / $Sk_{Dev_1}^{ENC}$	Открытый / закрытый ключ шифрования УУ
$Pk_{Dev_1}^{Sign}$ / $Sk_{Dev_1}^{Sign}$	Открытый / закрытый ключ подписи УУ
E / D	Асимметричный алгоритм шифрования
E_{auth}^{sym} / D_{auth}^{sym}	Симметричный алгоритм аутентифицированного шифрования
E^{sym} / D^{sym}	Симметричный алгоритм шифрования
$Sign(Sk, M)$	Алгоритм формирования подписи данных M на ключе Sk
$Verify(Sign, M)$	Алгоритм проверки цифровой подписи $Sign$ данных M



ПРОТОКОЛ АУТЕНТИФИКАЦИИ УПРАВЛЯЮЩЕГО УСТРОЙСТВА С РАСПРЕДЕЛЕНИЕМ КЛЮЧЕЙ ЗАЩИЩЕННОГО УПРАВЛЕНИЯ В АРХИТЕКТУРЕ ГРАНИЧНЫХ ВЫЧИСЛЕНИЙ

Этап инициализации:

1. Получение ГС от доверенной стороны значения подписи $Sign_{ES}$ уникальных данных ID_{ES} ГС
2. Распределение ключа K_{auth} аутентификации УУ на ИУ

Распределение ключей защищенного управления:

$$K_1 = Sign_{ES} \oplus r,\ K_2 = Sign_{ES} \oplus q,\ Pk_{Dev_1}^{Sign}$$

Этап аутентификации:

1. УУ $\rightarrow$ ГС: $<Pk_{Dev_1}^{ENC}, Pk_{Dev_1}^{Sign}>$
2. ГС $\rightarrow$ УУ: $E_{Pk_{Dev_1}^{ENC}}(Sign_{ES} \oplus r \| r)$
3. УУ: $D_{Sk_{Dev_1}^{ENC}}(E_{Pk_{Dev_1}^{ENC}}(Sign_{ES} \oplus r \| r))$
4. УУ: $Sign' = (Sign_{ES} \oplus r) \oplus r$
5. УУ: $Verify(Sign', ID_{ES})$
6. УУ $\rightarrow$ ГС: $E_{auth}^{sym}(Sign_{ES} \oplus r, K_{auth})$
7. ГС: $D_{auth}^{sym}\left(Sign_{ES} \oplus r, E_{auth}^{sym}(Sign_{ES} \oplus r, K_{auth})\right)$
8. ГС $\rightarrow$ ИУ: $E_{K_{auth}}^{sym}(Sign_{ES} \oplus q \| q) \| Pk_{Dev_1}^{Sign}$
9. ИУ: $Sign'' = (Sign_{ES} \oplus q) \oplus q$
10. ИУ: $D_{K_{auth}}^{sym}\left(E_{K_{auth}}^{sym}(Sign_{ES} \oplus q \| q)\right)$
11. ИУ: $Verify(Sign'', ID_{ES})$

Упрощенная аутентификация: *если от ГС к УУ уже распределён ключ K_1 (аутентификация на ИУ1 уже выполнялась), ГС начинает процесс аутентификации на ИУ2 с шага 6 с использованием имеющегося K_1*

Безопасность:

- Защита от атак подмены сервера (шаги 5 и 11)
- Защита от атак по известному открытому тексту (шаги 2 и 8)
- Защита от атак истощения энергоресурсов ИУ (шаг 6)

Гибкость: определены только типы используемых криптографических алгоритмов

Обозначение	Тип криптографического алгоритма
E / D	Асимметричный алгоритм шифрования
E^{sym} / D^{sym}	Симметричный алгоритм шифрования
E_{auth}^{sym} / D_{auth}^{sym}	Аутентифицированное шифрование
$Sign()$ / $Verify()$	Формирование / проверка цифровой подписи

ПРОТОКОЛ ЗАЩИЩЕННОГО УПРАВЛЕНИЯ. КРИПТОГРАФИЧЕСКИЕ НАБОРЫ

Защищенное управление:

1. УУ: $Sign_{M_{com}} = Sign(Pk_{Dev_1}^{Sign}, M_{com})$
2. УУ → ГС: $E_{auth}^{sym}(K_1, M_{com} \| Sign_{M_{com}})$
3. ГС: $D_{auth}^{sym}\left(K_1,\ E_{auth}^{sym}(K_1, M_{com} \| Sign_{M_{com}})\right)$
4. ГС → ИУ: $E_{K_2}^{sym}\left(M_{com} \| Sign_{M_{com}}\right)$
5. ИУ: $M_{com} \| Sign'_{M_{com}} = D_{K_2}^{sym}\left(E_{K_2}^{sym}\left(M_{com} \| Sign_{M_{com}}\right)\right)$
6. ИУ: $Verify(Sign'_{M_{com}}, M_{com})$

Использование протокола инкапсуляции CRYSTALS-Kyber:

1. УУ → ГС: $< Pk_{Dev_1}^{Exp}, Pk_{Dev_1}^{Sign} >$
2. ГС: $< c, K > \leftarrow Encaps\left(Pk_{Dev_1}^{Exp}\right)$
3. ГС: $E = E_K^{Sym}(Sign_{ES} \oplus r \| r)$
4. ГС → УУ: $< c, E >$
5. УУ: $< K > \leftarrow Decaps\left(Pk_{Dev_1}^{Exp}, c\right)$
6. УУ: $Sign_{ES} \oplus r \leftarrow E_K^{Sym}(E)$

Обозначение	**Криптографические наборы классической модели**		**Криптографические наборы постквантовой модели**	
E / D ($Encaps()$/ $Decaps()$)	Схема шифрования Эль-Гамаля на эллиптических кривых	Механизм согласования ключей VKO_GOSTR3410_2012, блочный шифр «Магма»	CRYSTALS-Kyber	
E^{sym} / D^{sym}	PRESENT / AES-128 (режим счетчика CTR)	Блочный шифр «Магма» в режиме гаммирования	AES-256 (режим счетчика CTR)	Блочный шифр «Кузнечик» в режиме гаммирования
E_{auth}^{sym} / D_{auth}^{sym}	Режим GCM блочного шифра AES-128	Режим MGM блочного шифра «Магма»	Режим GCM блочного шифра AES-256	Режим MGM блочного шифра «Кузнечик»
$Sign$ () / $Verify()$	Протокол подписи ECDSA	Протокол подписи ГОСТ Р 34.10–2018	CRYSTALS-Dilithium	

РАСПРЕДЕЛЕНИЕ ВЫЧИСЛИТЕЛЬНОЙ НАГРУЗКИ

Полная аутентификация

Упрощенная аутентификация

Защищенное управление

ИУ	ГС	УУ
	Зашифрование $E()$	
		Расшифрование $D()$
		Проверка подписи $Verify()$
		Зашифрование $E_{auth}^{sym}()$
	Расшифрование $D_{auth}^{sym}()$	
	Зашифрование $E^{sym}()$	
Расшифрование $D^{sym}()$		
Проверка подписи $Verify()$		
		Формирование подписи $Sign()$
		Зашифрование $E_{auth}^{sym}()$
	Расшифрование $D_{auth}^{sym}()$	
	Зашифрование $E^{sym}()$	
Расшифрование $D^{sym}()$		
Проверка подписи $Verify()$		

ЗАКЛЮЧЕНИЕ

1. Обработка информации с помощью граничных вычислений позволяет функционировать информационным системам в режиме реального времени за счет физической близости серверов к устройствам.

2. Разработанные для данной архитектуры протоколы позволяют обеспечивать аутентификацию с распределением ключей и защищенное управление в модели взаимодействия двух устройств Интернета вещей через граничный сервер.

3. Приведенные рекомендации по практическому применению данных протоколов включают в себя криптографические наборы двух вычислительных моделей (классической и постквантовой).

ПОЛИТЕХ
Санкт-Петербургский политехнический университет Петра Великого

Институт кибербезопасности и защиты информации

Спасибо за внимание!