## zu Aufgabe 2.1

Zeigen Sie: Die direkte AES-256-Blockverschlüsselung bietet für die Plaintextmenge $\{0,1\}^{128}$ sichere Einmalverschlüsselung im Sinne von RoR-OTCPA, wenn man AES-256 als PRP-sicher voraussetzt.

(Erläuterungen siehe Aufgabe 2.2!)

Wir gehen wir gewohnt umgekehrt vor und zeigen: Ist die direkte Blockverschlüsselung RoR-OTCPA-*unsicher*, so ist AES-256 PRP-*unsicher.* Ersteres heißt: Es gibt einen erfolgreichen RoR-OTCPA-Angreifer $\boldsymbol{A}$ gegen das in den Erläuterungen zu Aufgabe 2.2 beschriebene Verschlüsselungsschema $(\mathcal{K},\mathcal{E},\mathcal{D})$ mit $\mathcal{M}=\{0,1\}^{128}$. Letzteres heißt: Es gibt einen erfolgreichen PRP-Angreifer $\boldsymbol{B}$ gegen $E_K$ ($K \in \{0,1\}^{256}$), die Blockverschlüsselung von AES-256. Wir sind also fertig, wenn wir aus einem irgendwie gegebenen $\boldsymbol{A}$ den Angreifer $\boldsymbol{B}$ so konstruieren können, dass wir zeigen können

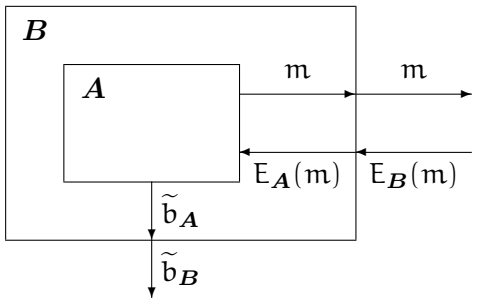
$$\mathrm{Adv}^{\mathsf{PRP}}_{E,\boldsymbol{B}} = \mathrm{Adv}^{\mathsf{RoR\text{-}OTCPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}}$$

(und $\boldsymbol{B}$ sich dabei hinsichtlich der Laufzeit nicht oder nicht wesentlich von $\boldsymbol{A}$ unterscheidet).

Der Angreifer $\boldsymbol{A}$ erwartet eine Umgebung mit einem Real-or-random-Verschlüsselungsorakel; nennen wir es $E_{\boldsymbol{A}}(\cdot)$. Dieses Orakel wird von $\boldsymbol{A}$ nur ein einziges Mal benutzt mit einer Anfrage $m \in \{0,1\}^{128}$. Darauf erwartet $\boldsymbol{A}$ als Orakelantwort entweder eine Verschlüsselung von $m$ (“real”) oder eine Verschlüsselung eines gleichverteilt zufälligen Bitstrings $m_0 \in \{0,1\}^{128}$ (“random”), jeweils vom Algorithmus $\mathcal{E}$ mit einem gleichverteilt zufälligem Schlüssel $K$ verschlüsselt. Diese Fälle des Orakels nennen wir $E_{\boldsymbol{A},1}(\cdot)$ und $E_{\boldsymbol{A},0}(\cdot)$. Schließlich gibt $\boldsymbol{A}$ ein Bit $\widetilde{b}_{\boldsymbol{A}}$ aus. Das Ziel von $\boldsymbol{A}$ ist es zu erraten, ob er es mit dem “Real”-Orakel zu tun hatte (Bitwert 1) oder mit dem “Random”-Orakel (Bitwert 0).

Der zu konstruierende Angreifer $\boldsymbol{B}$ hat ein Orakel $E_{\boldsymbol{B}}(\cdot)$ zu Verfügung, das er mehrmals befragen darf. Dieses Orakel liefert durchgängig entweder Werte der AES-256-Blockverschlüsselung $E_K$ mit einem gleichverteilt zufälligen Schlüssel $K$ oder Werte einer gleichverteilt zufälligen Permutation $\pi$ auf der Menge $\{0,1\}^{128}$. Diese Fälle des Orakels nennen wir $E_{\boldsymbol{B},1}(\cdot)$ und $E_{\boldsymbol{B},0}(\cdot)$. Auch $\boldsymbol{B}$ gibt schließlich ein Bit aus, $\widetilde{b}_{\boldsymbol{B}}$, und hat das Ziel zu erraten, ob er es mit dem Orakel für $E_K$ zu tun hatte (Bitwert 1) oder mit dem Orakel für $\pi$ (Bitwert 0).

Wir lassen unser $\boldsymbol{B}$ genau das tun, was der gegebene Angreifer $\boldsymbol{A}$ getan hätte; den Bitstring $m$ aus der Anfrage $E_{\boldsymbol{A}}(m)$ reichen wir einfach an das Orakel $E_{\boldsymbol{B}}(\cdot)$ durch und dessen Antwort als Orakelantwort unverändert zurück an $\boldsymbol{A}$; das Bit $\widetilde{b}_{\boldsymbol{A}}$ geben wir schließlich aus als $\widetilde{b}_{\boldsymbol{B}}$. Der beschriebene Angreifer $\boldsymbol{B}$ benutzt (wie $\boldsymbol{A}$) sein Orakel nur ein einziges Mal, obwohl er nach den Regeln des PRP-Angriffsspiels mehrfach darauf zugreifen könnte. Diese Konstruktion kann man so darstellen:




Hat $\boldsymbol{B}$ es hier bei $\mathsf{E}_{\boldsymbol{B}}(\cdot)$ mit einem Orakel für $\mathsf{E}_K(\cdot)$ zu tun ($K \in_\$ \mathcal{K}$), so ergibt sich für den Angreifer $\boldsymbol{A}$, der quasi innerhalb von $\boldsymbol{B}$ abläuft, exakt die Situation mit dem "Real"-Orakel für $\mathcal{E}_K(\cdot)$, denn $\mathcal{E}_K$ ist direkt über $\mathsf{E}_K$ definiert. Deshalb gilt

$$\Pr_{K\in_\$\mathcal{K}}\left[\boldsymbol{B}^{\mathsf{E}_{\boldsymbol{B}},1(\cdot)} \Rightarrow 1\right] = \Pr_{K\xleftarrow{\$}\mathcal{K}}\left[\boldsymbol{A}^{\mathsf{E}_{\boldsymbol{A}},1(\cdot)} \Rightarrow 1\right].$$

(Wir schreiben einerseits $K \in_\$ \mathcal{K}$, andererseits $K \xleftarrow{\$} \mathcal{K}$, weil wir die Bezeichnung $\mathcal{K}$ einerseits für eine Menge, andererseits für einen probabilistischen Algorithmus verwenden: Gemeint ist beide Male die gleiche Verteilung, nämlich eine Gleichverteilung auf $\{0,1\}^{256}$.)

Hat $\boldsymbol{B}$ es bei $\mathsf{E}_{\boldsymbol{B}}(\cdot)$ mit einem Orakel für $\pi(\cdot)$ zu tun ($\pi \in_\$ \mathrm{Perm}(\{0,1\}^{128})$, so wird die (einzige!) Orakelanfrage $m$ von $\boldsymbol{B}$ beantwortet mit einem gleichverteilt zufälligen $\pi(m) \in \{0,1\}^{128}$: Da es nur eine Anfrage gibt, unterliegt dieser Funktionswert keinen stochastischen Abhängigkeiten. Also ergibt sich auch für $\boldsymbol{A}$ eine Situation mit einer gleichverteilt zufälligen Orakelantwort. Das aber ist genau das gleiche wie ein "Random"-Orakel für das hier betrachtete Verschlüsselungsschema, denn die Blockverschlüsselung $\mathsf{E}_K$ ist (für jedes $K$) eine Permutation, und eine Gleichverteilung von $m_0 \in \{0,1\}^{128}$ bedeutet deshalb gerade eine Gleichverteilung (ebenfalls auf $\{0,1\}^{128}$) für $\mathsf{E}_K(m_0)$ und hier also gemäß Definition des Verschlüsselungsalgorithmus auch für $\mathcal{E}_K(m_0)$. Somit gilt

$$\Pr_{K\in_\$\mathcal{K}}\left[\boldsymbol{B}^{\mathsf{E}_{\boldsymbol{B}},0(\cdot)} \Rightarrow 1\right] = \Pr_{K\xleftarrow{\$}\mathcal{K}}\left[\boldsymbol{A}^{\mathsf{E}_{\boldsymbol{A}},0(\cdot)} \Rightarrow 1\right]$$

Zusammen ergibt sich nach den Definitionen des jeweiligen Vorteils:

$$\begin{aligned}\mathrm{Adv}^{\mathsf{PRP}}_{\mathsf{E},\boldsymbol{B}} &= \Pr_{K\in_\$\mathcal{K}}\left[\boldsymbol{B}^{\mathsf{E}_{\boldsymbol{B}},1(\cdot)} \Rightarrow 1\right] - \Pr_{K\in_\$\mathcal{K}}\left[\boldsymbol{B}^{\mathsf{E}_{\boldsymbol{B}},1(\cdot)} \Rightarrow 1\right] \\ &= \Pr_{K\xleftarrow{\$}\mathcal{K}}\left[\boldsymbol{A}^{\mathsf{E}_{\boldsymbol{A}},1(\cdot)} \Rightarrow 1\right] - \Pr_{K\xleftarrow{\$}\mathcal{K}}\left[\boldsymbol{A}^{\mathsf{E}_{\boldsymbol{A}},1(\cdot)} \Rightarrow 1\right] \\ &= \mathrm{Adv}^{\mathsf{RoR\text{-}OTCPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}}\end{aligned}$$

Q.e.d.!

## zu Aufgabe 2.2

Zeigen Sie: Die direkte blockweise AES-256-Verschlüsselung für eine Plaintextmenge $\{0,1\}^{n\cdot 128}$ mit $n \geq 2$ ist nicht sicher im Sinne von RoR-OTCPA.

*Erläuterungen:* Hier (und in Aufgabe 2.1) geht es um das folgende Verschlüsselungsschema $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ für die Plaintextmenge $\mathcal{M} = \{0,1\}^{n\cdot 128}$ (mit $n = 1$ für Aufgabe 2.1):

- $\mathcal{K}$ erzeugt eine Gleichverteilung auf $\{0,1\}^{256}$.
- Der Verschlüsselungsalgorithmus berechnet $\mathcal{E}_K(m)$ für $m \in \mathcal{M}$ als

  $$E_K(m_1) \,\|\, E_K(m_2) \,\|\, \ldots \,\|\, E_K(m_n),$$

  wobei $m = m_1 \,\|\, m_2 \,\|\, \ldots \,\|\, m_n$ mit $m_i \in \{0,1\}^{128}$.
- Der Entschlüsselungsalgorithmus berechnet $\mathcal{D}_K(c)$ für $c \in \{0,1\}^{n\cdot 128}$ als

  $$D_K(c_1) \,\|\, D_K(c_2) \,\|\, \ldots \,\|\, D_K(c_n),$$

  wobei $c = c_1 \,\|\, c_2 \,\|\, \ldots \,\|\, c_n$ mit $c_i \in \{0,1\}^{128}$. Ist $c \notin \{0,1\}^{n\cdot 128}$, so dass sich keine solche Zerlegung vornehmen lässt, lautet das Resultat $\perp$.

Dabei bezeichnen $E_K$ und $D_K$ die Blockverschlüsselung bzw. Blockentschlüsselung mit AES-256 unter einem Schlüssel $K$:

$$E_K, D_K \colon \{0,1\}^{128} \to \{0,1\}^{128} \quad \text{für } K \in \{0,1\}^{256}.$$

Das hier betrachtete Verschlüsselungsschema auf Basis einer beliebigen Blockchiffre nennt sich "Electronic Codebook Mode" (ECB).

Wir haben einen RoR-OTCPA-Angreifer zu beschreiben, der mit praktikablem Aufwand einen Vorteil erreichen kann, welcher nicht nur verschwindend gering ist. Die Existenz eines solchen Angreifers bedeutet nämlich, dass das Verschlüsselungsschema nicht sicher im Sinne von RoR-OTCPA ist. Einen hohen Vorteil erreicht ein Angreifer dann, wenn er recht zuverlässig zwischen den Fällen "real" und "random" eines Real-or-random-Verschlüsselungsorakels unterscheiden kann und sein Ausgabebit entsprechend wählt.

Die Angriffsidee ist ähnlich wie bei Aufgabe 1.4. (Dort ging es um mehrfache Verschlüsselung, hier um die mehrfache direkte Verwendung des gleichen $E_K$ beim Verschlüsseln nur eines Plaintexts.) Wie die Beschreibung des Verschlüsselungsalgorithmus zeigt, ergibt das Verschlüsseln etwa eines Plaintext $m_1 \,\|\, m_2$ (wobei $m_1, m_2 \in \{0,1\}^{128}$) einen Ciphertext $E_K(m_1) \,\|\, E_K(m_2)$ und das Verschlüsseln eines Plaintext $m_1 \,\|\, m_1$ demnach einen Ciphertext $E_K(m_1) \,\|\, E_K(m_1)$. Eine bestimmte Struktur des Plaintexts bleibt also beim Verschlüssen erhalten: Wie der Plaintext weist auch der resultierende Ciphertext zwei übereinstimmende Blöcke auf. Bei einem "Random"-Verschlüsselungsorakel wären solchen Übereinstimmungen dagegen nur zufällig zu erwarten.

Unser RoR-OTCPA-Angreifer $\boldsymbol{A}$ kann also wie folgt vorgehen: Er sendet einen Plaintext $m_1 \,\|\, m_1$ mit $m_1 \in \{0,1\}^{128}$ an sein Verschlüsselungsorakel. Die Orakelantwort $c$ wird beim zu betrachtenden Verschlüsselungsschema stets (wie der Plaintext) die Länge 256 haben. Diese Antwort zerlegt der Angreifer in der Form $c = c_1 \,\|\, c_2$ mit $|c_1| = |c_2| = 128$. Ist nun $c_1 = c_2$, so gibt $\boldsymbol{A}$ das Bit 1 aus (und vermutet damit, es mit dem "Real"-Verschlüsselungsorakel zu tun gehabt zu haben); andernfalls gibt $\boldsymbol{A}$ das Bit 0 aus ("Random"-Verschlüsselungsorakel).

Wenn $E_1(\cdot)$ das "Real"-Verschlüsselungsorakel bezeichnet und $E_0(\cdot)$ das "Random"-Verschlüsselungsorakel, kann man leicht nachprüfen, dass gilt

$$\Pr_{K \xleftarrow{\$} \mathcal{K}}\left[\boldsymbol{A}^{E_1(\cdot)} \Rightarrow 1\right] = 1$$

und

$$\Pr_{K \xleftarrow{\$} \mathcal{K}}\left[\boldsymbol{A}^{E_0(\cdot)} \Rightarrow 1\right] = \frac{1}{2^{128}}$$

(letzteres, weil $c_1$ und $c_2$ im "Random"-Fall unabhängig gleichverteilte Elemente von $\{0,1\}^{128}$ sind und deshalb mit Wahrscheinlichkeit $1/2^{128}$ das Ereignis $c_1 = c_2$ eintritt). Damit folgt

$$\mathrm{Adv}^{\text{RoR-OTCPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}} = \Pr_{K \xleftarrow{\$} \mathcal{K}}\left[\boldsymbol{A}^{E_1(\cdot)} \Rightarrow 1\right] - \Pr_{K \xleftarrow{\$} \mathcal{K}}\left[\boldsymbol{A}^{E_0(\cdot)} \Rightarrow 1\right] = 1 - \frac{1}{2^{128}},$$

der Angreifer erreicht also einen sehr großen Vorteil.

## zu Aufgabe 2.3

> Beschreiben Sie den Counter Mode einer Blockchiffre wie auf Vorlesungsfolie 4.21 skizziert als Verschlüsselungsverfahren $(\mathcal{K},\mathcal{E},\mathcal{D})$ mit einer Plaintextmenge $\mathcal{M}$, ausgehend von einer Blockchiffre $E_K$ mit einer Schlüsselmenge $\mathcal{K}$ und einer geraden Blocklänge $\ell$. Der Schlüsselgenerierungsalgorithmus $\mathcal{K}$ erzeugt einfach eine Gleichverteilung auf der gleichnamigen Menge $\mathcal{K}$. Aber wie würden Sie $\mathcal{M}$ festlegen? Wie sollten die Algorithmen $\mathcal{E}$ zur Verschlüsselung und $\mathcal{D}$ zur Entschlüsselung aussehen?
>
> (Hinweis: Bei einigen Details kann man sich hier mehr oder weniger willkürlich entscheiden.)

Es sei $\mathcal{M} = \bigcup_{0 \le n \le 2^{\ell/2}\cdot\ell} \{0,1\}^n$. Diese **Plaintextmenge** lässt jeden Bitstring einer Länge von 0 bis $2^{\ell/2} \cdot \ell$ als Plaintext zu: Soviele Bits können wir ohne Zählerüberlauf als Präfix von

$$E_K(iv \,\|\, 00\ldots0000) \,\|\, E_K(iv \,\|\, 00\ldots0001) \,\|\, E_K(iv \,\|\, 00\ldots0010) \,\|\, \ldots$$

erhalten, wenn $iv$ und der Zähler jeweils $\ell/2$ Bits umfassen. (Jeder Block von $E_K(iv \,\|\, 00\ldots00)$ bis $E_K(iv \,\|\, 11\ldots11)$ trägt $\ell$ Bits bei, und es gibt $2^{\ell/2}$ solche Blöcke.)

$\mathcal{K}$ als **Schlüsselgenerierungsalgorithmus** erzeuge (wie schon in der Aufgabenstellung vorgegeben) eine Gleichverteilung auf der gleichnamigen Menge $\mathcal{K}$.

Der **Verschlüsselungsalgorithmus** $\mathcal{E}$ arbeite für einen Schlüssel $K$ (von $\mathcal{K}$ erzeugt) und einen Plaintext $m$ (aus $\mathcal{M}$) wie folgt: Zunächst wird randomisiert gewählt $iv \xleftarrow{\$} \{0,1\}^{\ell/2}$. Dann wird ein Bitstring $g_K(iv, |m|)$ bestimmt als Präfix der Länge $|m|$ von

$$E_K(iv \,\|\, 00\ldots0000) \,\|\, E_K(iv \,\|\, 00\ldots0001) \,\|\, \ldots \,\|\, E_K(iv \,\|\, 11\ldots1111)$$

(dafür reicht es, die ersten $\left\lceil \frac{|m|}{\ell} \right\rceil$ Werte $E_K(\ldots)$ zu bestimmen, denn die weiteren gehen in $g_K(iv, |m|)$ gar nicht ein). Als Resultat der Verschlüsselung $\mathcal{E}_K(m)$ ausgegeben wird schließlich

$$c = iv \,\|\, \big(m \oplus g_K(iv, |m|)\big).$$

Es ist wichtig, dass $iv$ als Teil des Ciphertexts erscheint (sei es an dessen Anfang wie hier oder anders, z. B. am Ende)! Damit nämlich eine Entschlüsselung möglich ist, muss

dieser Initialisierungswert weitergegeben werden – und in unserer Formalisierung bekommt der Entschlüsselungsalgorithmus nur den Schlüssel und den Ciphertext als Eingaben, deshalb gehört $iv$ in den Ciphertext. (Im Schlüssel können wir $iv$ nicht unterbringen, denn das Verschlüsselungsschema soll mit *einem* Schlüssel für *mehrere* Verschlüsselungsvorgänge brauchbar sein, die jeweils einen frischen Wert $iv$ verwenden.)

Der **Entschlüsselungsalgorithmus** $\mathcal{D}$ arbeite für einen Schlüssel $K$ und einen Ciphertext $c$ wie folgt: Ist $|c| < \ell/2$ oder $|c| > \ell/2 + 2^{\ell/2} \cdot \ell$, so wird als Resultat der Entschlüsselung $\mathcal{D}_K(c)$ der besondere Wert $\perp$ zurückgegeben, um einen Fehler anzuzeigen. Ansonsten hat $c$ eine gültige Länge und wird zerlegt als $c = iv \,\|\, C$ mit $|iv| = \ell/2$; anschließend wird $g_K(iv, |C|)$ bestimmt als Präfix der Länge $|C|$ von

$$E_K(iv \,\|\, 00\ldots0000) \,\|\, E_K(iv \,\|\, 00\ldots0001) \,\|\, \ldots \,\|\, E_K(iv \,\|\, 11\ldots1111)$$

und als Resultat der Entschlüsselung $\mathcal{D}_K(c)$ der Wert $C \oplus g_K(iv, |C|)$ zurückgegeben.

(Die **Korrektheit** dieses Verschlüsselungsverfahrens ist leicht nachzuprüfen: Für jedes $m \in \mathcal{M}$ gilt stets $\mathcal{D}_K\big(\mathcal{E}_K(m)\big) = m$, denn mit den Benennungen oben verwenden hier die Algorithmen $\mathcal{E}$ und $\mathcal{D}$ die gleichen Werte $iv$ und $g_K(iv, |m|) = g_K(iv, |C|)$ (denn dann ist $|m| = |C|$), und der Wert $C$ im Entschlüsselungsalgorithmus ist gerade $m \oplus g_K(iv, |m|)$ aus dem Verschlüsselungsalgorithmus, so dass sich als Entschlüsselungsresultat

$$C \oplus g_K(iv, |m|) \;=\; m \oplus g_K(iv, |m|) \oplus g_K(iv, |m|) \;=\; m$$

ergibt.)

## zu Aufgabe 2.4

Hier betrachten wir die Counter-Mode-Verschlüsselung (Aufgabe 2.3) als Einmalverschlüsselung für bis zu $q \cdot \ell$ Bits. Wir gehen davon aus, dass $q$ recht klein ist. Geben Sie eine Ungleichung an, die ausgehend von der PRP-Sicherheit der verwendeten Blockchiffre die RoR-OTCPA-Sicherheit zeigt.

Bei dieser Aufgabe geht es darum, bekannte Resultate aus der Vorlesung und vom Übungsblatt 1 richtig zusammenzufügen: PRP (Blockchiffre) als PRF ("PRP/PRF Switching Lemma"), PRF zur Konstruktion eines PRG, PRG zur Einmalverschlüsselung.

Wir wollen zeigen: Ist die Blockchiffre PRP-sicher, so ist die Counter-Mode-Verschlüsselung RoR-OTCPA-sicher. Wie gewohnt betrachten wir die Kontraposition hiervon, zeigen zum Beweis nämlich: Ist die Counter-Mode-Verschlüsselung RoR-OTCPA-*unsicher,* so muß bereits die zugrundeliegende Blockchiffre PRP-*unsicher* sein.

Nehmen wir also einen Angreifer $\boldsymbol{A}$ im RoR-OTCPA-Angriffsspiel an. Analog zu Aufgabe 1.2 gibt es dazu einen Angreifer $\boldsymbol{B}$ in einer Variante des PRG-Angriffsspiels mit $iv$ mit

$$\mathrm{Adv}^{\mathsf{PRG}}_{g,\boldsymbol{B}} = \mathrm{Adv}^{\mathsf{RoR\text{-}OTCPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}},$$

wobei $g_K(\cdot,\cdot)$ die in der Lösung zu Aufgabe 2.3 beschriebene Abbildung bezeichnet. (Der $iv$ in einem PRG ist hier eine formale Komplikation, ändert an der Sache aber so gut wie nichts. Die Details überspringen wir an dieser Stelle.) Hieraus können wir wiederum

einen Angreifer $\boldsymbol{C}$ im PRF-Angriffsspiel gegen $\mathsf{E}_K$ konstruieren, der genau wie Angreifer $\boldsymbol{B}$ vorgeht und die Orakelantworten für $\boldsymbol{B}$ entsprechend der Konstruktion von $g_K$ aus seinen eigenen Orakelantworten zusammensetzt; sein Vorteil ist

$$\mathrm{Adv}^{\mathsf{PRF}}_{\mathsf{E},\boldsymbol{C}} = \mathrm{Adv}^{\mathsf{PRG}}_{g,\boldsymbol{B}}$$

(analog zu Folie 3.7), denn eine Interaktion von $\boldsymbol{B}$ mit der echten Funktion $g_K$ entspricht genau eine Interaktion von $\boldsymbol{C}$ mit der echten Funktion $\mathsf{E}_K$, und eine Interaktion mit dem idealen Fall entspricht genau einer Interaktion mit einer zufälligen Funktion. Wir wollen auf den PRP-Vorteil hinaus und setzen deshalb das PRP/PRF Switching Lemma ein (Folie 4.19), das besagt

$$\left|\mathrm{Adv}^{\mathsf{PRP}}_{\mathsf{E},\boldsymbol{C}} - \mathrm{Adv}^{\mathsf{PRF}}_{\mathsf{E},\boldsymbol{C}}\right| \le \frac{q(q-1)}{2^{\ell+1}}$$

und also insbesondere

$$\mathrm{Adv}^{\mathsf{PRF}}_{\mathsf{E},\boldsymbol{C}} - \mathrm{Adv}^{\mathsf{PRP}}_{\mathsf{E},\boldsymbol{C}} \le \frac{q(q-1)}{2^{\ell+1}}.$$

Alles in allem folgt

$$\mathrm{Adv}^{\mathsf{RoR\text{-}OTCPA}}_{(\mathcal{K},\mathcal{E},\mathcal{D}),\boldsymbol{A}} = \mathrm{Adv}^{\mathsf{PRP}}_{\mathsf{E},\boldsymbol{C}} + \frac{q(q-1)}{2^{\ell+1}}.$$

Das ist die gesuchte Ungleichung.